InterSystems™
IRIS Data Platform

System Administration Guide

Version 2023.3
2024-05-16

InterSystems Corporation   1 Memorial Drive   Cambridge MA 02142   www.intersystems.com

System Administration Guide
InterSystems IRIS Data Platform   Version 2023.3   2024-05-16

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

Table of Contents

List of Tables

1

Using the Management Portal

InterSystems IRIS® data platform enables you to perform system administration and management tasks via a web application, the InterSystems Management Portal.

1.1 Starting the Management Portal

You can start the Management Portal in the following ways:

- On Microsoft Windows platforms, click **Management Portal** on the InterSystems IRIS launcher menu.
- In a browser, access the following URL, using the <baseURL> for your instance:

  ```
  https:<baseURL>/csp/sys/UtilHome.csp
  ```

For information about the Management Portal, see Management Portal Overview.

Depending on the authentication settings for your system, you may have to log in before going to the Portal home page. See Management Portal Login Page for an overview of login requirements. In addition, each functional area requires access to particular resources; see Management Portal Web Application Structure.

Important: *Microsoft Windows 2003 Users Trusted Site Security Setting* — The first time you visit the Management Portal, you may receive a warning about the web site being blocked. When prompted, add the site to the trusted sites zone. InterSystems also recommends you allow session cookies for Portal procedures to function properly.

1.1.1 Management Portal Login Page

Whether or not you must enter a username and password to use the Portal depends on the authentication settings of the Management Portal web application (/csp/sys). There are two conditions:

Unauthenticated Access Only

Neither requires nor accepts a username and password — If the Portal accepts only unauthenticated connections (the default setting for Minimal security installs), you do not require a username and password to access the Portal; you bypass the **Login** page when you use the methods in the previous section to start the Portal. If navigation does bring you to the login page (by clicking **Logout**, for example), you see the following message:

Please click here to log in.

Authenticated Access

Requires a username and password — If your security settings require authentication for the Management Portal web application and you are not already authenticated on the system, the login page displays asking you to enter a **User Name** and **Password**. After entering these fields, click **Login** to display the Management Portal home page.

Note: To change your password, click your name in the Management Portal Header, which displays the change password dialog box.

Important: You can look up the **Authentication allowed** settings on the **Web Applications** page (**System Administration** > **Security** > **Applications** > **Web Applications**) by clicking **Edit** in the /csp/sys application row.

1.1.2 Management Portal Web Application Structure

The main Portal web application is /csp/sys. To provide enforcement of privileges within the Management Portal, the Portal is further split into distinct web applications. The Management Portal itself is not responsible for preventing non-privileged users from performing actions; this is handled by the system API methods that the Portal calls. The Portal does, however, attempt to keep non-privileged users out of restricted pages to prevent `<PROTECT>` errors.

There are four types of user for the Portal, which roughly correspond to predefined resources within InterSystems IRIS:

User Type	Resource	Web Application	Tasks
Security Manager	%Admin_Secure	/csp/sys/sec	View and edit list of users, roles, and other security tasks.
Manager	%Admin_Manage	/csp/sys/mgr	Change system configuration and define backup sets.
Operator	%Admin_Operate	/csp/sys/op	View system status pages and perform backups.
Explorer (public user)	%Development	/csp/sys/exp	View home page, view classes, routines, and globals, and use SQL pages, provided the user has access to the appropriate resources.

1.2 Management Portal Overview

This section describes some common layout elements of Management Portal pages. The following topics are discussed:

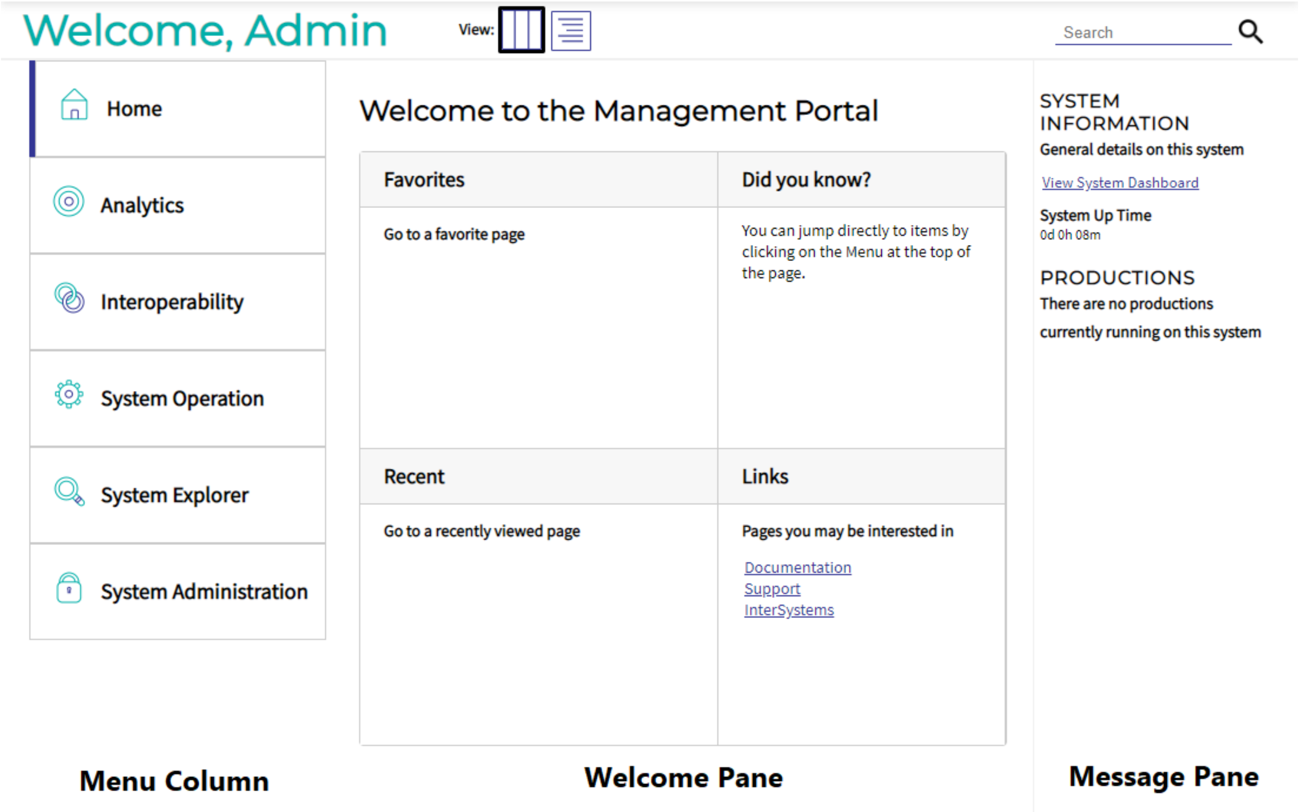
- Management Portal Home Page
- Management Portal Header
- Management Portal Ribbon
- System Overview Information

Note: Anywhere in the Management Portal, moving your cursor over a menu item displays a description of that item.

1.2.1 Management Portal Home Page

The Management Portal home page is titled **Welcome, <user>**. Next to the title, the ribbon contains the following options:

- Two **View** buttons, which let you specify how to display links in the menu column.
- A search bar, located on the right side of the ribbon. When you specify a word and press **Enter**, a list of all pages containing that word is displayed; then, you can click the destination page you want to display without having to navigate through the submenus.



Welcome to the Management Portal

Go to a favorite page

You can jump directly to items by clicking on the Menu at the top of the page.

Go to a recently viewed page

Pages you may be interested in

SYSTEM INFORMATION

General details on this system

System Up Time

0d 0h 08m

PRODUCTIONS

There are no productions currently running on this system

The following sections describe the areas of the home page:

- Management Portal Menu Column
- Management Portal Welcome Pane
- Management Portal Message Pane

1.2.1.1 Management Portal Menu Column

The menu column, located on the left edge of the home page, is the primary method of navigating the Portal.

For detailed information about navigating the Management Portal menus, see Navigating the Management Portal.

For information about specific pages of the Management Portal, see Management Portal Page Reference.

1.2.1.2 Management Portal Welcome Pane

The welcome pane occupies the center of the home page, and includes shortcuts to pages you visit frequently. It contains the following fields:

- **Favorites** — Lists Management Portal pages you have selected as favorites (see Action Pane); you can click each page title to go directly to that page.
- **Recent** — Lists the most recent pages you have displayed since the last time InterSystems IRIS was started.
- **Did you know?** — Displays tips.

- **Links** — Links to pages you might want to visit.

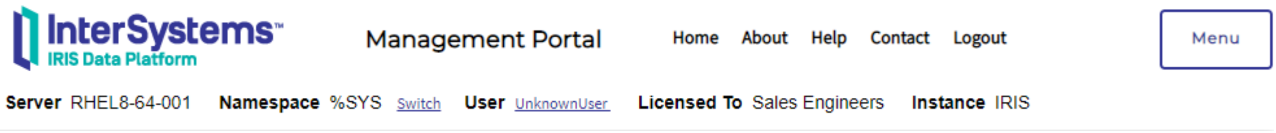
1.2.1.3 Management Portal Message Pane

The message pane, located on the right edge of the home page, displays general system information and provides a link to the system dashboard. For more information about the dashboard, see Monitoring InterSystems IRIS Using the Management Portal.

If the instance is a mirror member, the message pane also displays the mirror it belongs to, its status and member type, and a link to the Mirror Monitor (see Monitoring Mirrors).

1.2.2 Management Portal Header

The header, located at the top of every page in the Management Portal, can be used to navigate the Portal quickly.



Server RHEL8-64-001 **Namespace** %SYS Switch **User** UnknownUser **Licensed To** Sales Engineers **Instance** IRIS

The header contains the following links:

- **Home** — Displays the Management Portal home page.
- **About** — Displays system overview information.
- **Help** — Displays the online documentation (help) for the page/topic you are viewing.
- **Contact** — Displays a contact page for the InterSystems Worldwide Response Center (WRC).
- **Logout** — Logs you out and brings you to the **Login** page of the Management Portal.
- **Menu** — Displays a list of common tasks based on the roles the user holds.

The header also contains useful information, such as:

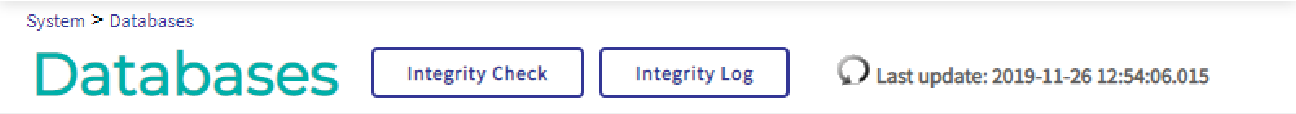
- **Server** — The name of the server running InterSystems IRIS.
- **Namespace** — The name of the namespace currently being used. To work in a different namespace, click the namespace name (for example **%SYS**) and select the desired namespace.
- **User** — The name of the user logged in to the Management Portal. To change the password for the user, click the name.
- **Licensed to** — The customer name that appears in the license key information.
- **Instance** — The name of the InterSystems IRIS instance running on the server.

In addition, you can display a **System Mode** label (for example, **Test System**); for information, see Memory and Startup Settings.

The left side of the Management Portal header displays the name of the product you are using.

1.2.3 Management Portal Ribbon

The ribbon lies directly beneath the header, and displays different contents specific to each page. As an example, the ribbon from the **Databases** page (**System Explorer** > **Databases**) is pictured below:



Typical contents of the ribbon are:

- The title of the Management Portal page being displayed.
- The breadcrumbs to the current page, listed directly above the page title. Each page listed in the path is an active link, which you can use to return to a previously displayed submenu/list. When you have made unsaved changes on a page, an asterisk is appended to the breadcrumbs, such as **System > Configuration > Memory and Startup — (configuration settings)***. You are always prompted for confirmation before navigating away from unsaved changes.

  Note: The breadcrumbs do not list every page in the path, and the pages in the breadcrumbs do not always match those in the navigation menu. You can always navigate to a specific page by clicking **Home** to return to the Management Portal home page and using the search tool, which is described later on in this section.

- Several buttons that allow you to perform operations on the page. For example, the **Databases** page contains the buttons **Integrity Check** and **Integrity Log**.
- A refresh button, with information about when the page was last updated.

1.2.4 System Overview Information

When you click **About** on the header of the Management Portal, a table displays with the following information:

- **Version** — Specific build information for this instance of InterSystems IRIS including platform, build number, and build date.
- **Configuration** — Name and location of the configuration (.cpf) file this instance is using.
- **Database Cache (MB)** — Space allocated for databases.
- **Routine Cache (MB)** — Space allocated for routines.
- **Journal file** — Name and location of current journal file.
- **SuperServer Port** — Port number on which the InterSystems IRIS server is running.
- **Web Server Port** — Port number on which the web server is running.
- **License Server Address/Port** — IP address of the InterSystems IRIS license server and port number on which it is running.
- **Licensed to** — Customer name that appears in the license key information.
- **Cluster support** — Indicates whether or not this instance is part of a cluster.
- **Mirroring** — Indicates whether or not this instance is a member of a mirror.
- **Time System Started** — Date and time this instance of InterSystems IRIS last started.
- **Encryption Key Identifier** — If encryption is activated, the GUID (global unique ID) of the encryption key.
- **NLS Locale** — National Language Support locale. For more information, see the Using System Classes for National Language Support and Using the NLS Pages of the Management Portal.
- **Preferred language for this session** — Drop-down list of languages in which the Management Portal has been localized and can be displayed. You can change the display language by selecting a new one from the drop-down. Initially, the preferred language for the browsing session is the one specified for the browser, or English if the browser language is

not supported; after you select a preferred language in a particular browser, it is used by the Management Portal in that browser even if the browser language is changed.

1.3 Navigating the Management Portal

The Management Portal has two different view modes to choose from for page navigation. You can switch between view modes at any time by clicking the desired **View** button in the Management Portal ribbon. The view modes are:

- Columns view — Displays page in columns; this is the default mode. This view mode allows you to select favorites and assign custom resources.
- List view — Displays page names in drop-down list.

Note: In addition to using views to navigate the Management Portal, you can go directly to a page via the search tool, which is located in the ribbon; for more information, see Management Portal Ribbon.

1.3.1 Columns view

The columns view displays submenus that you use to reach a destination page. If you click on a title/name in the final submenu, the destination page is displayed; if you click inside the box but not on the title/name itself, the Action Pane is displayed on the right.

1.3.1.1 Action Pane

The action pane explains a menu option and lets you perform several actions related to it. To display the action pane, while in Columns view, click inside the box that surrounds an option in the final submenu, but not on the option itself. You can do the following in the action pane:

- Add the destination page to the **Favorites** list on the home page, as described in Management Portal Welcome Pane.
- View the system resource required to load the page and assign a custom resource, as described in Use Custom Resources with the Management Portal.

  Note: If the name of a resource is truncated in the action pane, move the pointer over the name to see it in its entirety.

- Go to the destination page by clicking **Go**.

The action pane is available only in Columns view. When you navigate to a page using list view or the search tool, you cannot add it as a favorite or assign a custom resource.

1.3.2 List View

The list view displays lists that you use to reach your destination page, but does not let you specify a destination page as a favorite, or assign a custom resource. When you select the page you want to display, click **Go**.

1.4 Management Portal Page Reference

The Management Portal is made up of a number of pages divided into the following functional areas:

- Home — Management Portal home page
- Health — Healthcare pages. Only for InterSystems IRIS for Health™ and HealthShare® Health Connect.
- Analytics — InterSystems IRIS® data platform Business Intelligence pages
- Interoperability — Productions pages
- System Operations — System operation pages
- System Explorer — Database management pages
- System Administration — System administration pages

This document provides links to the appropriate section of documentation for each page of the Portal.

1.4.1 Home

The home page contains a number of options for navigating the Management Portal. See Management Portal Home Page for more information.

1.4.2 Health

Healthcare-specific features in InterSystems IRIS for Health and Health Connect are displayed on pages available from the **Health** portion of the Management Portal. You need to install a Foundation namespace before accessing these health pages. For details, see the "Using the Installer Wizard" section of the InterSystems IRIS for Health Installation Guide or Health Connect Installation Guide.

1.4.3 Analytics

The Business Intelligence pages are divided into the categories displayed in the **Analytics** portion of the Management Portal. See Introduction to the Business Intelligence User Interfaces.

1.4.4 Interoperability

The productions pages are divided into the categories displayed in the **Interoperability** portion of the Management Portal. See Finding Information on Menu Items for more information.

1.4.5 System Operations

The system operator pages are divided into the categories displayed in the **System Operations** portion of the Management Portal. The following table displays each category and the relevant documentation source.

Menu Item	Information
System Dashboard	Monitoring InterSystems IRIS Using the Management Portal
Backup	Managing Online Backups
Databases	Maintaining Local Databases
Processes	Controlling InterSystems IRIS Processes
SQL Activity	Monitoring SQL Activity
Locks	Monitoring Locks
Journals	Journaling

Menu Item	Information
Mirror Monitor	Monitoring Mirrors
Task Manager	Using the Task Manager
LDAP Configurations	View LDAP Configurations in the Portal as %Operator.
System Logs	Monitoring Log Files
System Usage	Monitoring System Performance
License Usage	Managing InterSystems IRIS Licensing
Background Tasks	JOB
Diagnostic Reports	Using the InterSystems IRIS Diagnostic Report

1.4.6 System Explorer

The database management pages are divided into the categories displayed in the **System Explorer** portion of the Management Portal. The following table displays each category and the relevant documentation source.

Menu Item	Information
Classes	Classes
SQL	Introduction to InterSystems SQL
Routines	User-Defined Code
Globals	Formal Rules about Globals
Tools	Provides classes that you can use to record, randomize and playback HTTP-based scripts against various applications for the purpose of QA, scalability, and network load testing. See %WebStress in the *InterSystems Class Reference* for more information.

1.4.7 System Administration

The system administrator pages are divided into the categories displayed in the **System Administration** portion of the Management Portal. The following table displays each category and the relevant documentation source.

Menu Item	Information
Configuration	Configuration Pagestable (below)
Security	System Management and Security
Licensing	Managing InterSystems IRIS Licensing
Encryption	Encryption Guide

1.4.7.1 Configuration Pages

The Configuration Pages are a subcategory of the system administration pages.

Menu Item	Information
System Configuration	System Configuration Pages table (below)
Connectivity	Connectivity Pages table (below)
Mirror Settings	Configuring Mirroring
Database Backup	Configuring InterSystems IRIS Backup Settings
SQL and Object Settings	SQL and Object Settings Pages table (below)
Device Settings	Device Settings Pages table (below)
National Language Settings	Configuring National Language Support (NLS)
Additional Settings	Additional Settings Pages table (below)

System Configuration Pages

The System Configuration Pages are a subcategory of the configuration pages.

Menu Item	Information
Memory and Startup	Configuring System Information
Namespaces	Configuring Namespaces
Local Databases	Configuring Databases
Remote Databases	Remote Databases
Journal Settings	Configuring Journal Settings

Connectivity Pages

The Connectivity Pages are a subcategory of the configuration pages.

Menu Item	Information
ECP Settings	Deploying a Distributed Cache Cluster
SQL Gateway Settings	Creating Gateway Connections for External Sources
External Language Servers	[Gateways]

SQL and Object Settings Pages

The SQL and Object Settings pages are a subcategory of the configuration pages.

Menu Item	Information
SQL	[SQL].
Objects	Swizzling a Fieldname Property with %ObjectSelectMode=1
TSQL Compatibility	TSQL Settings
ISQL Compatibility	ISQL Configuration Settings
System DDL Mappings	System Datatypes.
User DDL Mappings	User Datatypes

Device Settings Pages

The Device Settings pages are a subcategory of the configuration pages.

Menu Item	Information
Devices	Devices *Configuration Parameter File Reference*
Device Subtypes	DeviceSubTypes
IO Settings	[IO]
Telnet Settings	[Telnet]

Additional Settings Pages

The Additional Settings pages are a subcategory of the configuration pages.

Menu Item	Information
Compatibility	[Miscellaneous]
Advanced Memory	[config] parameters
Monitor	Monitoring InterSystems IRIS Using SNMP
Source Control	*Integrating InterSystems IRIS with Source Control Systems*.
Startup	[Startup]
Task Manager Email Settings	Configuring Task Manager Email Settings in the “System Administration Guide”.

2

Configuring System Information

InterSystems IRIS® data platform stores system-wide configuration information in a file called the configuration parameter file (CPF). This file is an important tool, as it contains most of the configurable settings for an InterSystems IRIS instance. A default CPF is deployed with every instance and is located in the installation directory. You can modify the CPF directly in a text editor, or indirectly from the Management Portal or the Terminal. On UNIX® and Linux, you can also customize the CPF during deployment by specifying a configuration merge file, which InterSystems IRIS uses to update the default CPF before the instance starts for the first time. For more information about the CPF and its parameters, Introduction to the Configuration Parameter File; for information about the configuration merge feature, see Using Configuration Merge to Deploy and Reconfigure InterSystems IRIS.

There are several startup settings that you must change for any newly installed instance, and others you should review. This page discusses the settings to consider initially.

2.1 Memory and Startup Settings

When you first install InterSystems IRIS, you should review and adjust the memory allocations, among other configuration settings. There are three primary actions you must take in determining the way an InterSystems IRIS instance uses memory, as follows:

- Allocate memory to the database and routine caches, using the Memory and Startup page (**System Administration** > **Configuration** > **System Configuration** > **Memory and Startup**); see Allocating Memory to the Database and Routine Caches.
- Set the maximum memory an InterSystems IRIS process can use; see Setting the Maximum Per-Process Memory.
- Configure the size of the shared memory heap, which determines the memory available to InterSystems IRIS for purposes other than the routine and database caches; see Configuring Shared Memory Heap (gmheap).

Important: When InterSystems IRIS is first installed, database and routine cache memory allocation is set to **Automatically**, under which InterSystems IRIS allocates 25% of total physical memory for the database cache (global buffers).

This setting is not appropriate for production use. For guidelines for allocating memory to an InterSystems IRIS instance's routine and database caches and the shared memory heap, see Configuring InterSystems IRIS Shared Memory.

Other than the memory settings, the Memory and Startup page includes the following:

- **Auto-start on System Boot** — On Windows systems, an InterSystems IRIS instance is configured by default to start automatically when the host system starts. You can change this behavior, so that the instance does not start automatically on system startup, by clearing this check box.

  On UNIX ® systems, InterSystems IRIS does not provide this option and does not restart when the host system restarts. If you want InterSystems IRIS to restart when the host system restarts, use OS-level scripting as appropriate for your environment.

  Note: You may occasionally need to *prevent* IRIS from starting with the host system while the host system is down (for example during restores). This can only be done at the OS level and is dependent on your configuration.

- **Superserver Port Number** — The superserver port is the TCP port used by an InterSystems IRIS instance to accept incoming client requests. When you change this setting (which should be done with caution, as many clients may be configured to connect to the instance using this port), the change will not take effect until you restart the instance.
- **System Mode** — You can enter a label to be displayed in the Management Portal header, or select one from the drop-down list.

Click **Save** to save your modifications to these settings.

Important: Some changes on this page require an InterSystems IRIS restart and some do not. If you modify a setting that requires a restart and save your changes, none of the changes take effect until you restart InterSystems IRIS, even those that by themselves do not require a restart. If a restart is required, the message `Modification saved. You must restart system for the new values to take effect.` displays. After you close the page, the warning message does not appear again to remind you that a restart is required, and it is therefore best to restart the instance immediately.

2.1.1 Allocating Memory to the Database and Routine Caches

The database cache is system memory allocated for buffering data, and is sometimes referred to as the *global buffer pool*. The routines cache is system memory allocated for buffering routines.

When InterSystems IRIS is first installed, configuration of the database cache is set to the **Initial**, under which InterSystems IRIS allocates 25% of total system physical memory for the database cache (global buffers). *This setting is not appropriate for production use*. Before deploying the system for production use or before performing any tests or benchmarking intended to simulate production use, you must manually create an appropriate memory allocation for the database cache using the procedure described here.

By default, the configuration of the routine cache is set to **Automatic**, under which InterSystems IRIS allocates to the routine cache an amount of memory equal to 10% of the database cache for the default 8-kilobyte buffer size, with a minimum of 80 MB and a maximum of 1020 MB. For typical production instances, if the database cache size is appropriately configured, automatic allocation of the routine cache is sufficient; however, the ideal allocation for a given application depends on many factors, and adjustment may be necessary to optimize performance.

For guidelines for allocating memory on an InterSystems IRIS instance, including to the database and routine caches, see Estimating Memory Requirements.

To allocate memory to the database and routine caches:

1. On the Management Portal, navigate to the Memory and Startup page (**System Administration** > **Configuration** > **System Configuration** > **Memory and Startup**).
2. For **Configure Database Cache (Global Buffers)**, select **Specify Amount** and enter the database cache memory allocation, in megabytes.

   If you have enabled database block sizes other than the default 8-kilobyte size using the DBSizesAllowed setting on the **Startup Settings** page (**System Administration** > **Additional Settings** > **Startup**), enter an allocation for each enabled block size; the range for each size is displayed on the page. Both block size and the maximum number of buffers available have implications for performance; to determine how many global buffers InterSystems IRIS will create for a particular block size, divide the allocation for a block size by the block size. For guidelines for selecting the appropriate block sizes for your applications, see Large Block Size Considerations.

Important: If you are configuring a large ECP system, allocate at least 50 MB of 8 KB buffers for ECP control structures in addition to the 8 KB buffers required to serve your 8 KB blocks over ECP. See Increase Data Server Database Caches for ECP Control Structures.

3. For **Configure Routine Cache**, you can accept the default **Automatic** setting to allocate an amount of memory equal to 10% of the database cache for the default 8-kilobyte buffer size, or select **Specify Amount** and enter the routine cache memory allocation, in megabytes.

For more information about specifying the database and routine caches, see the globals and routines parameters in the *Configuration Parameter File Reference*.

2.1.2 Setting the Maximum Per-Process Memory

The **Maximum Per-Process Memory (KB)** setting on the Memory and Startup page specifies the maximum memory that can be allocated to a new process running on this InterSystems IRIS instance. The allowed range is 256 KB to 2147483647 KB; the default is the initial value of the bbsiz parameter. InterSystems recommends setting this equal to -1 (which resolves to the maximum value) for most circumstances.

Note: It is not necessary to reset this value unless you have set it lower than its default. If you receive <STORE> errors, increase the size and restart the process that created the error.

Process private memory, which is used for symbol table allocation and various other memory requirements (for example I/O device access structures and buffers), is allocated in increasing extents as required by the application until the specified maximum is reached. Once private memory is allocated to a process, it is not deallocated until the process exits.

See bbsiz for more information about this setting, including how to change it programmatically. For a detailed discussion of process memory in InterSystems IRIS, see Process Memory in InterSystems Products.

2.1.3 Configuring Shared Memory Heap (gmheap)

The shared memory heap is the memory used by InterSystems IRIS for purposes other than the routine and database caches. This setting is not on the Memory and Startup page. You can configure **gmheap** on the **Advanced Memory** page (**System Administration** > **Configuration** > **Additional Settings** > **Advanced Memory**) or programmatically; for more information, see gmheap in *Configuration Parameter File Reference*.

To see details of used and available memory for **gmheap**, navigate to the Shared Memory Heap Usage page (**System Operation > System Usage**) and click the **Shared Memory Heap Usage** link; for more information, see Shared Memory Heap Usage.

If your application will utilize a large number of SQL queries or if you plan on enabling parallel dejournaling, you will need to allocate additional memory to `gmheap`. For more information, see Shared Memory Considerations and System Requirements for Parallel Dejournaling.

2.2 IPv6 Support

You can enable or disable the use of IPv6 addresses in InterSystems IRIS by navigating to the **Startup Settings** page (**System Administration** > **Configuration** > **Additional Settings** > **Startup**) page; in the **IPv6** row, click **Edit**. Select **IPv6** to enable this option.

Note: This option is visible only if the network to which this InterSystems IRIS instance is connected permits IPv6 addressing.

When IPv6 is enabled, InterSystems IRIS accepts IPv6 addresses, IPv4 addresses, or DNS forms of addressing (host names, with or without domain qualifiers); when IPv6 is disabled, InterSystems IRIS accepts only IPv4 addresses or DNS forms of addressing.

When dotted-decimal IPv4 addresses (for example, 192.29.233.19) are specified, an IPv4 connection is attempted; when colon-separated IPv6 addresses (for example, 2001:fecd:ba23:cd1f:dcb1:1010:9234:4085) are specified, an IPv6 connection is attempted. When a DNS name (for example, mycomputer.myorg.com) is specified, it resolves to an actual IP address: first, it attempts to make an IPv4 connection; then, if an IPv4 connection cannot be made, it attempts an IPv6 connection.

InterSystems IRIS allows Internet addresses to be supplied in DNS, IPv4 and IPv6 formats. For example, localhost, 127.0.0.1, and ::1 are representations of the loopback address in each format, respectively. Detailed information about IPv6 addressing can be found in the following Internet Engineering Task Force documents:

- IP Version 6 Addressing Architecture (RFC 4291)
- Application Aspects of IPv6 Transition (RFC 4038)
- Format for Literal IPv6 Addresses in URL's (RFC 2732)

IPv6 addressing can also be checked and controlled using the **IPv6Format** method of the %SYSTEM.Process class (for the current process) or the **IPv6** method of the Config.Startup class (for the system generally).

Even though an InterSystems IRIS instance may be using an IPv4 network, IPv6 addresses can still be used as input to the various services provided that the IPv6 address supplied has a valid IPv4 equivalent. The loopback address used earlier in this section is such an example; *RFC 4291* describes several more formats. Thus, the various InterSystems services will accept either IPv4 or IPv6 addresses without error as long as the address form given can be validly converted for use on the connected network. So all of these forms (and several more) are acceptable

- localhost (DNS)
- 127.0.0.1 (IPv4)
- ::FFFF:127.0.0.1 (IPv4 mapped IPv6 format)
- 0:0:0:0:0:0:0:1 (full IPV6)
- ::1 (compressed IPv6)

as valid representations of the loopback address.

Generally, when asked for an Internet address that has been supplied to an InterSystems service earlier, InterSystems IRIS does not alter the address format. Addresses supplied in IPv4, or IPv6 format are returned as IPv4 or IPv6, respectively. The only exception is that addresses supplied as host names and translated by the Domain Name Server (DNS) may be returned in whatever form the DNS returns.

Note: InterSystems IRIS does not support the use of wildcard characters or ranges in IPv6 addresses.

3

Configuring Namespaces

This page provides information on configuring namespaces in InterSystems IRIS® data platform. For background information, see Namespaces and Databases. Also see Adding Mappings to a Namespace.

You can configure namespaces on the **Namespace** page of the Management Portal, which you can navigate to by selecting **System Administration** on the home page, then **Configuration**, then **System Configuration**, then **Namespaces**.

The absolute limit to the number of namespaces within a single InterSystems IRIS instance is 2048. The size of the namespaces table is not configurable.

3.1 Rules for Namespace Names

The following rules apply to a namespace name:

- It must be between one and 64 characters long.
- It must start with an alphabetic character or a percent sign (%); the remainder can include alphanumeric characters, dashes, or underscores.

For information on system-supplied namespaces, see System-Supplied Namespaces.

3.2 Create/Modify a Namespace

To create a namespace, click **Create New Namespace** to display the **New Namespace** page, then do the following:

1. Enter a **Name** for the namespace. See Rules for Namespace Names.
2. You can **Copy from** an existing namespace, creating a duplicate of the selected namespace. In this case, all other options will be made unavailable except for the **Web application** check box described in step 6 below.
3. Choose whether the default database for globals is **local** or **remote**.
4. **Select an existing database for Globals** for the default Global mapping of this namespace or click **Create New Database**, which launches either the database wizard or the remote database wizard.
5. Optionally, you can choose whether the default database for routines is **local** or **remote**, then either use the **Select an existing database for Routines** drop-down to choose a database for the default Routine mapping of this namespace, or click **Create New Database**, which launches either the database wizard or the remote database wizard.

6. Select the **Create a default Web application for this namespace** check box if you are creating a web application that accesses this namespace.
7. Optionally, you can clear the Enable namespace for interoperability productions check box. For more information, see Create an Interoperability-Enabled Namespace below.
8. After entering the required information, click **Save** to add your namespace to the configuration.

3.2.1 Create an Interoperability-Enabled Namespace

When you create a namespace on an InterSystems IRIS instance, the **Enable namespace for interoperability productions** check box is displayed at the bottom of the **New Namespace** page and is automatically selected. To create a namespace that is not interoperability-enabled, clear this check box before clicking **Save**.

If you do not clear the check box and create an interoperability-enabled namespace, the system automatically performs additional configuration tasks for the new namespace, as follows:

- If the default globals database for this namespace is an existing database, it upgrades and recompiles some classes in that database.

  CAUTION: If you are also using this database in other namespaces, you might consider this change undesirable. When you create a new namespace in an InterSystems IRIS instance, carefully consider whether it is appropriate for this namespace to reuse an existing database.

- It defines global mappings, routine mappings, and package mappings that make the InterSystems IRIS system classes and data available to the namespace.
- It adds nodes to the *^%SYS* global.
- It creates a web application for the namespace, using the application name required by InterSystems IRIS: `/csp/`*namespace*.

3.2.2 Create a Foundation Namespace for Healthcare Interoperability

To create a *Foundation* namespace and production for healthcare interoperability, use the InterSystems IRIS for Health™ Installer Wizard or the HealthShare® HealthConnect Installer Wizard, rather than the **New Namespace** page. Once you have activated a Foundation namespace in the Installer Wizard, if you later need to modify an option, such as the routine database, you can use the Namespaces page.

3.3 Rename a Namespace or Modify Default Mappings

You can rename a namespace, or change the databases to which your namespace is mapped without restarting InterSystems IRIS, using the following procedure:

1. Go to the **Namespaces** page (**System Administration** > **Configuration** > **System Configuration** > **Namespaces**).
2. On the **Namespaces** page, click the name of namespace you wish to modify.
3. Change or replace the existing name to rename the namespace.

Important: If you are renaming an interoperability-enabled namespace, you must take additional steps to complete the process.

a. Open the InterSystems Terminal from the System Tray.

b. Enter:

do ##class(%Library.EnsembleMgr).EnableNamespace("<NewNamespace>",1)

Where <*NewNamespace*> is the new name of the existing namespace.

c. Go to the **Web Applications** page (**System Administration** > **Security** > **Applications** > **Web Applications**).

d. Find the name of the application that corresponds to the old name of the namespace, and click **Delete**.

e. Click the name of the application that corresponds to the new name of the namespace.

f. Select **Namespace Default Application** and click **Save**.

g. In the Terminal, enter:

```
do ##class(%EnsembleMgr).DisableNamespace("<OldName>",1)
```

Where <*OldName*> is the original name of the namespace that you are renaming.

4. Choose the **Default Database for Globals**, the **Default Database for Routines**, and the **Default Database for Temporary Storage** from the list of defined databases.

   Note: Selecting a database that is configured not to journal globals (that is, the **Journal globals** property is set to **No**) from the **Default Database for Temporary Storage** drop-down list is not the same as selecting IRISTEMP; for more information, see Using Temporary Globals and IRISTEMP.

5. Click **Save**.

Users directly accessing the database at the time of the change may need to log out of and then back into InterSystems IRIS to update their namespace mappings.

3.4 Delete a Namespace

You can delete a namespace, including all mappings associated with it:

1. Navigate to the **Namespaces** page (**System Administration** > **Configuration** > **System Configuration** > **Namespaces**) and click **Delete** in the row of the namespace you want to delete.
2. On the **Delete Namespaces** page, if you want to delete the Web Gateway pages from the physical path, select the check box.
3. To delete the namespace and associated mappings, click **Perform Action Now**.

3.5 Programmatically Configure a Namespace

InterSystems IRIS also provides programmatic tools that are useful for automating namespace configuration. You can use the Config.Namespaces class to create and configure namespaces; see the *InterSystems Class Reference* for more information.

Another method for configuring namespaces is to include the CreateNamespace, ModifyNamespace, or DeleteNamespace operations in conjunction with configuration merge. Configuration merge allows you to customize an InterSystems IRIS instance by applying a declarative merge file, which specifies settings and operations to apply to that instance. For more information about this feature, see Using Configuration Merge to Deploy and Reconfigure InterSystems IRIS.

3.6 See Also

- Namespaces and Databases (which includes information on system-defined namespaces and databases)
- Adding Mappings to a Namespace
- Using Configuration Merge to Deploy and Reconfigure InterSystems IRIS

4

Adding Mappings to a Namespace

As part of configuring a namespace, you can also map globals, routines, and class packages from other databases on the same or different systems.

This allows simple references to data which can exist anywhere and is the primary feature of a namespace. You can map whole globals or pieces of globals; this feature allows data to easily span disks.

Note: Mappings are sorted alphabetically; if subscripts are specified, they are sorted by name and subscript. For more information, see Formal Rules about Globals.

Important: If there is mapped content with the same identifier as local content (such as a package, class, global, or routine name), the mapped content will be visible, rather than the local content. As such, you should be as specific with your mappings as possible, to prevent mapping away from existing content.

Be sure to follow Rules and Guidelines for Identifiers when naming content and creating mappings, to avoid any unwanted conflicts.

4.1 Global Mappings

You can add a mapping for a new global to your namespace at the global and global subscript level that overrides the default database mapping for globals of the namespace:

1. Navigate to the **Namespaces** page (**System Administration** > **Configuration** > **System Configuration** > **Namespaces**) and click **Global Mappings** in the row of the namespace where you want to map the global.
2. From the **Global Mappings** page click **New**.
3. Select the **Global database location** database where the global is located.
4. Enter the **Global name.** You can use the `*` character as part of the global name to specify multiple globals, for example **`ABC*`**.
5. Enter the **Global subscripts to be mapped**. The subscript reference must begin with an open parenthesis. Some examples follow:

```
(1)
("A")
(1):(5)
("A"):("Z")
("B",23,"m"):("E",5)
(BEGIN):("X")
("Y"):(END)
```

Note: When specifying a range (for example, `("A"):("Z")`, the range is “from-to” (not “from-through”) the specified subscripts; that is, the lower end of a defined subscript range is inclusive, while the upper end of the defined subscript range is exclusive. For example, `Name (1):(10)` includes `Name (1)` but does not include `Name (10)`; the exclusive upper range allows you to have a defined upper boundary when working with subscripted ranges, such as `Name ("a"):("b")`, where `Name ("aa")` and `Name ("aaaaa")` are equally valid ranges to precede `Name ("b")`.

You can use the reserved words `BEGIN` and `END` to refer to the first and last possible subscripts; however, you cannot use the asterisk (`*`) wildcard with subscripted globals because global subscripts must be mapped individually.

For more information about subscript-level mapping (SLM) ranges, see Setting Up Global Mappings.

6. Click **Advanced** to display the following:
   a. Select the **Collation**. Collation applies only to new subscript-level mapping globals.
   b. Select the **Lock Database Location**. For more information see Global in the [Map] section of the *Configuration Parameter File Reference*.
7. Click **OK**.

   Note: **>>** displayed in the first column of the new mappings row indicates that you opened the mapping for editing.
8. To save the mappings in the cpf file, click **Save Changes**.

Important: While it is possible to add a mapping that changes the database location of an existing global, this does not actually move the global. As a consequence, the global becomes inaccessible, as it remains in the original database while the namespace expects to find it in the newly mapped database. For a new mapping for an existing global to be successful, you must relocate the global manually, for example using the Terminal or the Management Portal, by creating it on the new database and removing it from the original database.

4.2 Routine Mappings

You can add mappings to your namespace at the routine level that overrides the default database mapping for routines of the namespace:

1. Navigate to the **Namespaces** page (**System Administration** > **Configuration** > **System Configuration** > **Namespaces**) and click **Routine Mappings** in the row of the namespace where you want to map the global.
2. From the **Routine Mappings** page, click **New**.
3. Select the **Routine database location** database where the routine is located.
4. Enter the **Routine name**. The routine does not have to exist when you map it (that is, it can be the name of a routine you plan to create).
5. Click **OK**.

   Note: **>>** displayed in the first column of the new mappings row indicates that you opened the mapping for editing.
6. To save the mappings in the cpf file, click **Save Changes**.

For example, using the preceding Test Namespace Mapping example, if you plan to create a schedule routine (for example, BOSZZairline) in the airports database (in the FlightSchedule namespace) and you want it to be available to users in the TravelAgent namespace, navigate to the **Routine Mappings** page (in the TravelAgent namespace row), then click **New Routine Mapping**. Enter the information as shown in the following **Routine Mapping** dialog box:

Important: When you map one or more routines, be sure to identify all the code and data needed by those routines, and ensure that all that code and data is available in all the target namespaces. The mapped routines could depend on the following items:

- Include files
- Other routines
- Classes
- Tables
- Globals

Use additional routine, package, and global mappings as needed to ensure that these items are available in the target namespaces.

4.3 Package Mappings

You can add a class package mappings which makes all the classes within a package (and all the generated routines for those classes) in a specific database visible to another namespace:

1. Navigate to the **Namespaces** page (**System Administration** > **Configuration** > **System Configuration** > **Namespaces**) and click **Package Mappings** in the row of the namespace where you want to map the package.
2. From the **Package Mappings** page, click **New**.
3. Select the **Package database location** database where the package is located.
4. Select the **Package name**. The package does not have to exist when you map it (that is, it can be the name of a package you plan to create); you can specify a new package name, as follows:
   a. Click **New Package**.
   b. In the **New package name** text box, enter a name.
5. Click **OK**.

   Note: **>>** displayed in the first column of the new mappings row indicates that you opened the mapping for editing.

6. To save the mappings in the cpf file, click **Save Changes**.

See Package Mapping for a description of packages and the procedure for mapping them.

Important: When you map a package, be sure to identify all the code and data needed by the classes in that package, and ensure that all that code and data is available in all the target namespaces. The mapped classes could depend on the following items:

- Include files
- Routines
- Other classes
- Tables
- Globals

Use additional routine, package, and global mappings as needed to ensure that these items are available in the target namespaces.

InterSystems IRIS also provides functionality to make a package available in all namespaces through a single action.

4.4 Mapping to All Namespaces

In addition to mapping globals, routines, and packages to specific namespaces, you can map them to all namespaces. To enable this form of mapping:

1. First, create a namespace named `%ALL`, as described in Create/Modify a Namespace.

   Note: `%ALL` is not visible except for the purposes of mapping data; that is, it is not a real namespace, but a mechanism for mapping data to all namespaces.

2. Then, make the desired mappings in the `%ALL` namespace, as described in the following topics:
   - To map globals, see Global Mappings.
   - To map routines, see Routine Mappings.
   - To map packages, see Package Mappings.

These `%ALL` mappings apply in all namespaces. You cannot create namespace-specific mappings to resources that are mapped in the `%ALL` namespace, as `%ALL` mappings override any namespace-specific mappings to the same resource.

CAUTION: `%ALL` mappings apply to every namespace, including `%SYS`. It is possible to break certain features by creating a mapping that overrides a routine or global the instance relies on.

You should create mappings as narrowly as possible when using `%ALL`. Additionally, closely follow the Rules and Guidelines for Identifiers; in particular, do not create `%ALL` mappings for any globals listed in Global Names to Avoid.

When you create a subscript-level mapping in the `%ALL` namespace, a mapping for the root global is automatically created to `%DEFAULTDB`. The `%DEFAULTDB` variable represents the default database for any given namespace.

4.5 See Also

- Namespaces and Databases

- Configuring Namespaces
- Using Configuration Merge to Deploy and Reconfigure InterSystems IRIS

5

Configuring Local Databases

This page provides information on configuring local databases in InterSystems IRIS® data platform. Note that you can make most database configuration changes dynamically; you can create and delete databases and modify database attributes while the system is running.

5.1 Database Considerations

For background information, see Namespaces and Databases. This section presents additional concepts relevant to system administrators.

CAUTION: On Windows systems, *do not* use file compression on IRIS.DAT database files. (Files are compressed by right-clicking a file or folder in Windows Explorer and selecting **Properties**, then **Advanced**, then **Compress contents to save disk space**; once compressed, a folder name or filename is rendered in blue in Windows Explorer.) If you compress a IRIS.DAT file, the instance to which it belongs will fail to start, with misleading errors.

5.1.1 Database Size and Growth

InterSystems IRIS databases dynamically expand as needed (assuming free space is available), though you can specify a maximum size. A database can grow until it is 32 terabytes if you are using the default 8KB block size.

5.1.2 The Database Cache

Each InterSystems IRIS system maintains a database cache — a local, shared memory buffer used to cache data retrieved from the physical databases. This cache greatly reduces the amount of costly I/O operations required to access data and provides much of the performance benefits of InterSystems IRIS. (For information about allocating the database cache, see Memory and Startup Settings.)

5.1.3 Database Total Limit

The absolute limit on the number of databases that can be configured within a single InterSystems IRIS instance (given sufficient storage space) is 15,998. There are other limitations as follows:

- The directory information for databases cannot exceed 256 KB. This means that if the average length of a database directory name is longer, an instance can have fewer total databases. The following formula describes this relation:

  $maximum\_DBs = 258048 / (avg\_DB\_path\_length + 3)$

For example, if all database directory paths are of the form `c:\InterSystems\IRIS\mgr\DB`*NNNN*`\`, the average length is 33 bytes. Thus, the maximum number of databases is 7,168, calculated as follows: 258048/ (33 + 3) = 7168.

- Mirrored databases count twice toward the absolute limit of 15,998. If all databases on the instance are mirrored, the effective limit is 7,499 databases. This is because InterSystems IRIS creates two database definitions for mirrored databases; one for the directory path (c:\InterSystems\IRIS\mgr\DBNNNN\), and one for the mirror definition (:mirror:MIRRORNAME:MirrorDBName).
- The number of databases that can be in use simultaneously is limited by the operating system's restriction on the number of open files (either per process or system-wide). InterSystems IRIS reserves approximately half of the operating system's open file allocation for its own use and devices.

5.1.4 Database Configuration Considerations

The following are tips to consider to consider when configuring databases:

- InterSystems IRIS provides a seamless option to spread data across multiple physical database (IRIS.DAT) files. Therefore, you can build applications with multiple databases or splitting data by global or subscript-level mappings, as appropriate.
- Keep database sizes within a manageable value based on the infrastructure available for administration tasks such as backup, restore, integrity checks, etc.
- It is recommended that stream globals (if storing streams within IRIS.DAT database files) be global mapped to a separate database, and that the stream database(s) be configured with a large (64 KB) block size.
- Depending on your workload, it may be beneficial to consider alternate (larger) block sizes than the default 8 KB database block size. For general guidelines, see Large Block Size Considerations below.

5.1.5 Large Block Size Considerations

In addition to the 8 KB (default) block size supported by InterSystems IRIS (which is always enabled), you can also enable the following block sizes:

- 16 KB (16384)
- 32 KB (32768)
- 64 KB (65536)

However, you should exercise caution when creating your database to use large block size because using them can impact the performance of the system. Consider the following before enabling and using large block sizes:

- If your application workload consists primarily of sequential inserts or sequential reads/queries, large block sizes may improve performance.
- If your application workload consists primarily of random inserts or random reads/queries, large block sizes may degrade performance. Since larger block sizes result in fewer blocks being cached for a given total size of database cache, to reduce the impact on random database access, you should also consider making more total memory available as database cache.
- For index-type databases, the default block size (8 KB) ensures optimum performance; larger block sizes potentially degrade performance. If you are considering larger block sizes for your data, you should consider mapping index globals to a separate 8 KB block size database.

To create a database that uses block sizes other than the supported blocks, do the following:

1. Enable the block sizes using the setting on the **Startup Settings** page (**System Administration** > **Additional Settings** > **Startup**); see DBSizesAllowed.
2. Configure the database cache for the enabled block size on the **Startup Settings** page (**System Administration** > **Additional Settings** > **Startup**), as described in Memory and Startup Settings.
3. Restart InterSystems IRIS.
4. Create the database as described in Create Local Databases.

5.1.6 Database Compatibility Considerations

As described in the Create a Local Database procedure, you can copy or move an InterSystems IRIS database to an instance other than the one in which it was created by copying or moving its IRIS.DAT file, or temporarily mount a database created in another instance on the same system. You can also restore a backup of a database (see Backup and Restore) to an instance other than its original instance. To avoid data incompatibility, however, the following requirements must be met:

- The target (new) instance must use the same character width (8-bit or Unicode; see Character Width Setting) and the same locale (see Using the NLS Settings Page of the Management Portal) as the instance that created the database.

  The one exception to this requirement is that an 8-bit instance using a locale based on the ISO 8859 Latin-1 character set is compatible with a Unicode instance using the corresponding wide character locale. For example, a database created in an 8-bit instance using the **enu8** locale can be used in a Unicode instance using the **enuw** locale.

- If the source and target instances are on systems of different endianness, the database must be converted to the endianness of the target instance before being used.

  Depending on the platform, multibyte data is stored with either the most significant byte or the least significant byte in the lowest memory address (that is, first): when the most significant byte is stored first, it is referred to as “Big-endian;” when the least significant byte is stored first, it is referred to as “Little-endian.”

  When defining a database using an existing IRIS.DAT created on a system of different endianness, use the cvendian utility (see Using cvendian to Convert Between Big-endian and Little-endian Systems) to convert the database before you use it. When restoring a backup of a database to a system of different endianness than the source system, see Considering Endianness.

5.2 Rules for Database Names

The following rules apply to a database name:

- It must be between one and 64 characters long
- It must start with an alphabetic character or an underscore; the remainder can include alphanumeric characters, dashes, or underscores

For information on system-supplied databases, see System-Supplied Databases.

5.3 Viewing Local Databases

To view the local databases, navigate to the **Local Databases** page (**System Administration** > **Configuration** > **System Configuration** > **Local Databases**).

The **Local Databases** page displays the following information about the databases on your system:

- **Name** — Database name.
- **Mirror** — If the database is mirrored, the name of the mirror; for more information see Add Databases to the Mirror.
- **Directory** — Location of the IRIS.DAT file.
- **Size (MB)** — Size of the database in megabytes.
- **Status** — Specifies whether or not the database is mounted, unmounted, or dismounted; if it is mounted, specifies whether it has read-only or read-write permission. For more information, see "The Local Databases List Information" table in Maintaining Local Databases.
- **Resource Name** — The name of the database resource that governs access to the database; for more information, see Using Resources to Protect Assets.
- **Encrypted** — Specifies whether or not the database is encrypted; for more information, see Encryption Guide.
- **Journal** Specifies whether or not the database is journaled; for more information, see Journaling.

You can use this page to:

- Create a Local Database
- Edit a Local Database's properties
- Relocate a Local Database
- Delete a Local Database

5.4 Creating a Local Database

To create a local database:

1. Navigate to the **Local Databases** page (**System Administration** > **Configuration** > **System Configuration** > **Local Databases**).
2. Click **Create New Database** to open the **Database Wizard**.
3. Enter a database name in the text box. A database name must not already be in use within the InterSystems IRIS instance; also see Rules for Database Names.
4. The first time you create a local database in an InterSystems IRIS instance using a particular browser, you must either
    - Enter the name of the database directory, in which case this directory, containing the IRIS.DAT file, is created in c:\InterSystems\mgr once you confirm it.
    - Click the folder icon to browse to an existing directory, in which case the IRIS.DAT file is created in that directory.

    Thereafter, by default a directory of the same name as the database name you provide, containing the IRIS.DAT file, is created in the same location as the previous database directory. For example, if you first create database db22 in any directory under c:\InterSystems\mgr, when you click **Create New Database** again and enter db33 in the **Enter the name of your database** box, c:\InterSystems\mgr\db33 is automatically filled into the **Database directory** text box. If you change this to c:\InterSystems\db33 and create db33, the base directory c:\InterSystems will be filled in the next time.

    Note: InterSystems does not support the use of symbolic links when configuring database directories.

5. Click **Next** to continue configuring the database. If a IRIS.DAT file already exists in the directory you specified, you are warned of this and can either

- Click **Finish** to use the existing file, in which case all of the databases characteristics are determined by the IRIS.DAT file. You would typically do this when copying or moving a database from another instance, or temporarily mounting a database created in another instance on the same system.
- Click **Back** to specify another directory, then click **Next** again to continue specifying the characteristics of the new database in the next step.

6. In the **Initial Size** text box, type the number of megabytes for your database size (the default is 1 MB).

   Note: You cannot create or edit a database so that its size is larger than the total available disk space. If the size you specify is within 90% of the disk's free space, you are warned and must confirm the action.

7. Select the desired block size from the **Block size for this database will be** drop-down list. By default, all new databases are created with a **Block Size** of 8 KB.

   CAUTION: Do not select block sizes other than 8 KB from the drop-down list unless you have read and understand the guidelines described in Large Block Size Considerations.

8. Select whether or not you want to journal globals in this database from the **Journal globals?** drop-down list. See Journaling.

   Note: If you are configuring the database to store temporary globals, setting the **Journal globals** property to **No** is not the same as storing the temporary globals in IRISTEMP; for more information, see Using Temporary Globals and IRISTEMP.

9. If encryption is activated, you can encrypt this database by selecting **Yes** for **Encrypt Database?**.
10. If the instance is part of a mirror, you can add this database to the mirror by selecting **Yes** for **Mirrored Database?**. See Add Databases to the Mirror for information about creating mirrored databases.
11. From this panel onward, you can click **Next**. to continue configuring the database or Finish to accept the remaining defaults
12. Choose the resource to control access to this database:
    - Default — %DB_%DEFAULT
    - Existing — Choose from a list of existing database resources
    - New — Create a new database resource (the new name defaults to %DB_% *database name*)
13. Click **Next** to view a list of the database attributes.
14. Click **Finish** to add your database.

You are now ready to configure and manage your new database.

Note: To protect you from accidentally corrupting a database, you cannot open or write to an operating system file called IRIS.DAT, even if it is not a mounted database.

5.5 Editing a Local Database’s Properties

To edit a local database’s properties:

1. Navigate to the **Local Databases** page (**System Administration** > **Configuration** > **System Configuration** > **Local Databases**).

2. Click the name of the database.

   The information displayed varies depending on whether or not the database is mirrored. See the sections for:

   - Nonmirrored Local Databases
   - Mirrored Local Databases

5.5.1 Editing Nonmirrored Local Database Properties

If you are editing a nonmirrored database, the system displays the following properties (Create a Local Database describes many of these fields).

- **Name**
- **Directory** (this setting must always reflect the location of the IRIS.DAT database file)
- **Encrypted** (cannot be changed)
- **Mirrored** — Click **Add to Mirror***mirror_name* to add the database to the mirror in which the InterSystems IRIS instance is the primary failover member. (This option is available only when the instance is the primary in a mirror.) See Add an Existing Database to the Mirror.
- **Block Size (Bytes)** (cannot be changed)
- **Size (MB)** — There are three size settings, as follows:
  - Change **Current** to modify the current size of the database.

    Note: You cannot create or edit a database so that its size is larger than the total available disk space. If the size you specify is within 90% of the disk's free space, you are warned and must confirm the action.

  - **Expansion** sets the amount by which to expand the database when required; the default (and recommended) setting of zero (0) indicates 12% of current size or 10 MB, whichever is larger. When using 12% of the current size, the expansion size will not be greater than 1GB.
  - **Maximum** specifies the maximum size to which the database can grow, in megabytes; the default setting of zero (0) indicates no maximum. To modify this setting, you can enter a new number of MB, or you can precede a number by **+** or **-**, for example **+10** or **-20**, to enlarge or reduce the maximum by the specified amount. When you reduce the maximum size of a database, you are warned and must confirm the action. Note, the IRISAUDIT database must be configured to have no maximum size.
- **Resource Name** — Select the resource with which to associate the database. Click the resource icon next to the drop-down to display the Resources page so you can create a resource.
- **New Global** — Specify attributes for new globals.
- **Global Journal State** — Select to enable journaling, clear to disable. See Journaling.
- **Preserve global attributes on delete** — Specify whether a global’s directory entry and attributes should be preserved when it is deleted; attributes include collation, journaling status, and growth pointer. Select to preserve a global’s directory entry and attributes when the global is entirely deleted; clear to remove the directory entry and attributes.
- **Mount Read-Only** — Select to specify that the database be mounted as read-only; clear to specify that it be mounted as read-write.
- **Mount Required at Startup** — Select to indicate that the database must be mounted when InterSystems IRIS starts up; if the database cannot be mounted, InterSystems IRIS does not start. This lets you ensure that journal recovery and transaction rollback can be performed on the database before startup following a crash (as described in Journaling). Clear to let InterSystems IRIS start without first mounting the database.

Note: By default, this setting is selected for required InterSystems IRIS databases (for example, IRISLIB and IRISAUDIT) and cannot be changed. The default is cleared but can be selected for databases that you create, as well as the USER and ENSLIB databases). For additional information about database status and explicitly dismounting and mounting databases, see "The Local Databases List Information" table in Maintaining Local Databases.

- **Stream Location** — Click the **Browse** button to select the directory in which streams associated with this database are stored. By default, the stream location for a local database is a subdirectory named stream in the database **Directory**, which is one of the above fields (for example, *install-dir*\mgr\DB1\stream).

  Note: InterSystems recommends that you use the default location.

5.5.2 Editing Mirrored Local Database Properties

If you are editing a mirrored database, the system displays the following properties (Create a Local Database describes many of these fields).

Note: Journaling is required for a mirrored database, therefore the **Global Journal State** setting does not appear.

- **Name**
- **Mirror Name** — Name by which the database is identified within the mirror; cannot be changed.
- **Directory** (this setting must always reflect the location of the IRIS.DAT database file)
- **Encrypted** (cannot be changed)
- **Stream Location** — Click the **Browse** button to select the directory in which streams associated with this database are stored. By default, the stream location for a local database is a subdirectory named stream in the database **Directory**, which is one of the above fields (for example, *install-dir*\mgr\DB1\stream).

  Note: Like other database-related data that is not contained in the database itself (see Mirror Configuration Guidelines), a mirrored database's file streams are *not* mirrored. (For information about file streams, see Working with Streams.)

  InterSystems recommends that you use the default location.

- **Resource Name** — Select the resource with which to associate the database. Click the resource icon next to the drop-down to display the Resources page so you can create a resource.
- **Block Size (Bytes)** (cannot be changed)
- **Collation** — Among global attributes, only the collation attribute can be changed, for new globals only.
- **Preserve global attributes on delete** — Specify whether a global's directory entry and attributes should be preserved when it is deleted; attributes include collation, journaling status, and growth pointer. Select to preserve a global's directory entry and attributes when the global is entirely deleted; clear to remove the directory entry and attributes.
- **Mount Read-Only** — Select to specify that the database be mounted as read-only; clear to specify that it be mounted as read-write.
- **Mount Required at Startup** — Select to indicate that the database must be mounted when InterSystems IRIS starts up or becomes a mirror primary; if the database cannot be mounted, InterSystems IRIS does not start or become primary. This lets you ensure that journal recovery and transaction rollback can be performed on the database before startup following a crash (as described in Journaling) and that open transactions on the former primary have been rolled back as part of failover. Clear to let InterSystems IRIS start without first mounting the database. For additional information

about the **Mount Required at Startup** setting, see “The Local Databases List Information” table in Maintaining Local Databases.

- **Local Properties** — This area contains three size settings, as follows:
  - Change **Size** to modify the current size of the database.

    Note: You cannot create or edit a database so that its size is larger than the total available disk space. If the size you specify is within 90% of the disk's free space, you are warned and must confirm the action.
  - **Expansion** sets the amount by which to expand the database when required (and assuming free space is available); the default (and recommended) setting of zero (0) indicates 12% of current size or 10 MB, whichever is larger.
  - **Maximum** specifies the maximum size to which the database can grow, in megabytes; the default setting of zero (0) indicates no maximum. To modify this setting, you can enter a new number of MB, or you can precede a number by **+** or **-**, for example **+10** or **-20**, to enlarge or reduce the maximum by the specified amount. When you reduce the maximum size of a database, you are warned and must confirm the action. Note, the IRISAUDIT database must be configured to have no maximum size.

  This area also contains the current, expansion, and maximum size settings for **Other System** — if the current instance is a failover member, this is the other failover member; if the current instance is an async member, this is the first failover member that the async could obtain the information from. For important information about how the properties of a mirrored database on the backup and async mirror members are synchronized with those on the primary, see Mirrored Database Considerations.

5.6 Relocating a Local Database

To move a local database’s IRIS.DAT file to a different directory, do the following:

1. Make note of the current database directory. You can view this information from the **Local Databases** page (**System Administration** > **Configuration** > **System Configuration** > **Local Databases**).
2. Perform a clean shut down of the instance, such as by using the **iris stop** command.
3. Copy the IRIS.DAT file and the stream directory from the current database directory to the desired location.

   Important: If there is an iris.lck file in the database directory, do not attempt to move the database. This means the database is still in use, and attempting to move it could lead to unforeseen problems. Contact the InterSystems Worldwide Response Center (WRC) for assistance.

   Note: InterSystems does not support the use of symbolic links when configuring database directories, and recommends choosing a fully resolved physical path for the new database location.
4. Open the iris.cpf file for the instance in a text editor. This file is usually located in the *installation directory*.
5. Locate the name of the database beneath the `[Databases]` section. Replace the old directory path with the new path, then save iris.cpf.
6. Delete the old database directory.
7. Start the InterSystems IRIS instance, and view the **Local Database** page to confirm that the directory is set to the new location.
8. If the database is mirrored, use **^MIRROR** to remove the original location of the mirrored database. Specifically use the option **Mirror management** > **Remove Mirrored Database**.

Then activate and catch up the mirror.

Important: After you relocate a local database directory, you must also update any systems that access the database remotely (such as ECP application servers). On each remote system, update the remote database directory to the new location, as described in Configuring Remote Databases.

5.7 Deleting a Local Database

To delete a local database, click the **Delete** link in the appropriate row. The Delete Database page displays information about the database you are deleting, and lets you:

- Select the namespaces mapped to this database for deletion. You cannot delete a database if a namespace is mapped to it, so unless you select all of the listed namespaces you cannot delete the database.

  You cannot delete namespaces that are also mapped to other databases. When this is the case, a link is provided to take you to the Namespaces page, where you can modify the database mappings of the namespaces involved. After you delete all mappings to another database, that database will be removed from the list of databases you have to delete.

- You can choose to delete the database’s IRIS.DAT file, *if and only if*:
  - No other databases use this IRIS.DAT file.
  - You have marked all namespaces mapped to the database for deletion.

  If these conditions are not met, you can still delete the database from the current configuration, but the IRIS.DAT file cannot be deleted.

- Confirm that you want to delete the database after reviewing information about it by clicking **Delete Database Now**.

If you cannot or chose not to delete the IRIS.DAT file, the database is still removed from the Databases section of the InterSystems IRIS parameters file and therefore from the list of local databases displayed by the Management Portal.

5.8 Programmatically Configuring a Database

InterSystems IRIS also contains programmatic tools that are useful for automating database configuration. You can use the Config.Databases class to create and configure databases; see the *InterSystems Class Reference* for more information.

You can also configure databases using the ^DATABASE command line utility.

Another method for configuring databases is to include the CreateDatabase, ModifyDatabase, or DeleteDatabase operations in conjunction with configuration merge. Configuration merge allows you to customize an InterSystems IRIS instance by applying a declarative merge file, which specifies settings and operations to apply to that instance. For more information about this feature, see Using Configuration Merge to Deploy and Reconfigure InterSystems IRIS.

5.9 See Also

- Namespaces and Databases
- Configuring Remote Databases

- ^DATABASE
- Using Configuration Merge to Deploy and Reconfigure InterSystems IRIS
- Using cvendian to Convert Between Big-endian and Little-endian Systems
- Maintaining Local Databases

6

Configuring Remote Databases

This page describes how to add and delete remote databases. A *remote database* is a database that is physically located on another server system, as opposed to a *local database* which is physically located on the local server system.

6.1 Adding a Remote Database

You can define a remote database on the local server if the database's host is configured on that server as a distributed cache data server. For instructions for adding a data server, see Configuring an Application Server.

To add a remote database, follow these steps:

1. Navigate to the **Remote Databases** page (**System Administration** > **Configuration** > **System Configuration** > **Remote Databases**) and click **Create Remote Database** to launch the wizard.
2. Select the data server that hosts the database from the **Remote Server** drop down.
3. Choose how you want to specify the remote database directory from the **Remote Directory** radio buttons:
   - **Select databases from a list** lets you choose from a drop-down list of database directories on the remote server. If the remote data server cannot currently be reached, the drop-down list is empty.
   - **Enter your own database specification** lets you enter the database directory directly, but the Portal does not validate your entry.
4. Enter a database name (its name on the local server; it does not need to match its name on the remote server); see Rules for Database Names. You have defined a remote database.
5. You can optionally specify the directory in which streams associated with this database are stored. By default, the stream location for a remote database is the InterSystems IRIS `Temp` directory (*install-dir*`\mgr\Temp`).

   Note: InterSystems recommends that you use the default location.
6. Click **Save** to configure the remote database.

You can click the **Edit** link for a remote database at any time to modify the remote database fields.

6.2 Deleting a Remote Database

To delete a remote database, click the **Delete** link in the appropriate row. The Delete Database page displays information about the database you are deleting, and lets you:

- Select the namespaces mapped to this database for deletion. You cannot delete a database if a namespace is mapped to it, so unless you select all of the listed namespaces you cannot delete the database.

  You cannot delete namespaces that are also mapped to other databases. When this is the case, a link is provided to take you to the Namespaces page, where you can modify the database mappings of the namespaces involved. After you delete all mappings to another database, that database will be removed from the list of databases you have to delete.
- Confirm that you want to delete the database after reviewing information about it by clicking **Delete Database Now**.

This action simply removes the database from the local instance’s remote database configuration; the actual database and its local configuration on its host are not affected.

6.3 See Also

- Configuring an Application Server
- Configuring Local Databases

7

Using Resources to Protect Assets

Authorization protects InterSystems IRIS® data platform components from inappropriate use. InterSystems uses the following terminology when discussing these components:

- An *asset* is something that is protected. For instance, an InterSystems IRIS database is an asset, the ability to connect to InterSystems IRIS using SQL is an asset, and the ability to perform a backup is an asset.
- *Resources* protect assets. Sometimes there is a one-to-one correspondence between assets and resources – that is, a single asset (such as a database) is protected by one resource. In other cases, multiple assets are protected by a single resource, in order to simplify security management. For example, a variety of system management functions are protected by a single resource.
- A *privilege* grants permission to do something with one or more assets protected by a resource, such as being able to *read* the *orders database*. A privilege is written as a resource name followed by a permission separated by a colon, such as `%DB_Sales:Read`.

  For more information on privileges, see Privileges and Permissions.

This topic addresses issues related to resources and the assets that they protect. InterSystems IRIS includes a set of resources for assets that it protects — and provides access for users based on the rights that they hold. You can also define your own resources.

7.1 Types of Resources

There are multiple resource types:

- System resources — For controlling the ability to perform various actions for an InterSystems IRIS instance. For more information on these resources, see System Resources.

  These resources are **`%Admin_ExternalLanguageServerEdit`**, **`%Admin_Journal`**, **`%Admin_Manage`**, **`%Admin_Operate`**, **`%Admin_RoleEdit`**, **`%Admin_Secure`**, **`%Admin_Task`**, **`%Admin_UserEdit`**, **`%Development`**, **`%DocDB_Admin`**, **`%IAM`**, **`%System_Callout`**, **`%System_Attach`**, and **`%Secure_Break`**.
- Database resources — For controlling read and write access to InterSystems IRIS databases. For more information on these resources, see Database Resources.

  The database resources for a newly installed InterSystems IRIS instance are **`%DB_IRISLOCALDATA`**, **`%DB_IRISAUDIT`**, **`%DB_IRISLIB`**, **`%DB_IRISLOCALDATA`**, **`%DB_IRISSYS`**, **`%DB_IRISTEMP`**, **`%DB_ENSLIB`**.
- Gateway resources — For controlling access to external language servers. For more information on these resources, see Gateway Resources.

The gateway resources for a newly installed InterSystems IRIS instance are **`%Gateway_Object`**, **`%Gateway_SQL`**, and **`%Gateway_ML`**.

- Service Resources — For controlling the ability to connect to InterSystems IRIS using various InterSystems connection technologies. For more information on these resources and the functionality that they control, see Services.

  Not all services have associated privileges — only those services for which InterSystems IRIS provides user-based access; other services, such as data check, are not user-based and, as a result, do not have associated security resources. For more information on managing services, see Services.

  The service resources are **`%Service_CallIn`**, **`%Service_ComPort`**, **`%Service_Console`**, **`%Service_DocDB`**, **`%Service_Login`**, **`%Service_Native`**, **`%Service_Object`**, **`%Service_SQL`**, **`%Service_Telnet`**, **`%Service_Terminal`**, and **`%Service_WebGateway`**.

- Application resources — Either for controlling the whole of a user-defined application or for perform authorization checks anywhere in user code. For information on these resources generally, see Application Resources. For information on creating these resources, see Create or Edit a Resource.

7.2 System Resources

InterSystems IRIS comes with a set of built-in resources that govern actions in relation to the installed InterSystems IRIS instance. System resources include:

- Administrative Resources
- The %Development Resource
- The %DocDB_Admin Resource
- The %IAM Resource
- The %System_Callout Resource
- The %System_Attach Resource
- The %Secure_Break Resource
- The %Service_Native Resource

System resources also include the resources associated with resource-based services. For more details on services, see Services.

7.2.1 Administrative Resources

The administrative resources are:

- %Admin_ExternalLanguageServerEdit
- %Admin_Journal
- %Admin_Manage
- %Admin_Operate
- %Admin_RoleEdit
- %Admin_Secure
- %Admin_Tasks

- %Admin_UserEdit

Note: Privileges on `%Admin_*` resources allow users to carry out administrative functions without having any database privileges (`%DB_<database-name>:R/W`). For example, a user such as a system operator with the `%Admin_Operate:Use` privilege can perform backups without holding privileges on any of the databases being backed up. This is because there is no reason for the operator to have access to the contents of databases other than through applications such as the InterSystems IRIS database backup system.

7.2.1.1 %Admin_ExternalLanguageServerEdit

This resource controls the ability to create, modify, or delete an external language server (also known as a gateway), including changing the gateway resource associated with it and using the methods in the %SYSTEM.java.SQL and %SYSTEM.python.SQL classes.

This resource takes the Use permission.

By default, the `%Manager` role holds the `%Admin_ExternalLanguageServerEdit:USE` privilege.

7.2.1.2 %Admin_Journal

This resource allows users to set and clear the no-journaling process flag via the **DISABLE^%SYS.NOJRN** and **ENABLE^%SYS.NOJRN** entry points, respectively, in programmer mode in the Terminal. This resource allows you to establish users who can perform this action without having to give them the Use permission on the `%Admin_Manage` resource, which might give them more privileges than necessary or desired.

This resource takes the Use permission (not Read or Write).

7.2.1.3 %Admin_Manage

This resource controls multiple sets of privileges:

- It controls access to various pages in the Management Portal, including the System Administration page.
- It controls the ability to:
  - Create, modify, and delete InterSystems IRIS configurations.
  - Create, modify, and delete backup definitions.
  - Add databases, modify database characteristics, and delete databases.
  - Modify namespace map.
  - Perform database and journal restores.
  - Set and clear the no-journaling process flag via the **ENABLE^%SYS.NOJRN** and **DISABLE^%SYS.NOJRN** entry points, respectively, in programmer mode in the Terminal. Note that if you wish for a user to be able to perform this task without other managerial privileges, use the `%Admin_Journal` resource.

This resource takes the Use permission.

7.2.1.4 %Admin_Operate

This resource controls multiple sets of privileges:

- It controls access to various pages in the Management Portal, including the System Operation page.
- It controls the ability to:
  - Start and stop InterSystems IRIS.

  - Examine and terminate processes.
  - Mount and dismount databases.
  - Perform integrity checks.
  - Start, stop, and switch journals.
  - Perform database backups.
  - Examine and delete locks.
  - Examine logs.
  - Start and stop services.

The **`%Admin_Operate:Use`** privilege is required to mount a database, either explicitly (such as when using an ObjectScript utility) or implicitly (such as when making a global reference to an un-mounted database).

This resource takes the Use permission.

7.2.1.5 %Admin_RoleEdit

This resource controls the following privileges:

- For SQL, it controls the ability to:
  - Create or delete a role.

This resource takes the Use permission.

7.2.1.6 %Admin_Secure

This resource controls multiple sets of privileges:

- It controls access to various pages in the Management Portal.
- When working with the RBAC security model, it controls the ability to:
  - Create, modify, or delete a user.
  - Create, modify, or delete a role.
  - Create, modify, or delete application definitions and application resources.
  - Modify audit settings.
  - Modify services.
- For SQL, it controls the ability to:
  - Create, modify, or delete a user.
  - Create, modify, or delete a role.
  - See the privileges granted to a user.
  - See the privileges granted to a role.
  - Revoke SQL privileges that were granted by another user.

This resource takes the Use permission.

7.2.1.7 %Admin_Tasks

This resource's privileges include the ability to create, modify, or run a task, such as through the Management Portal's Task Manager (**System Operation** > **Task Manager**).

This resource takes the Use permission.

7.2.1.8 %Admin_UserEdit

This resource controls the following privileges:

- For SQL, it controls the ability to:
  - Create, modify, or delete a user.

This resource takes the Use permission.

7.2.2 The %Development Resource

The **`%Development`** resource controls access to InterSystems IRIS development facilities and various pages in the Management Portal. Specifically, it controls the ability to:

- Enter direct mode.
- Use Studio. The **`%Development:Use`** privilege is checked whenever the Studio connects to a server.
- Use the global, routine, class, table, or SQL capabilities of the InterSystems IRIS system manager utility. (This privilege is also required to call any APIs that provide programmatic access to this functionality.)
- Use the debugging facilities of InterSystems IRIS, including the **BREAK** and **ZBREAK** commands and the debug option of the process display in the InterSystems IRIS system manager utility.

The **`%Development:Use`** privilege works in conjunction with database privileges to control developer access to InterSystems IRIS as follows:

- For Studio, the **`%Development:Use`** privilege is checked whenever the Studio connects to a server. In order to connect, the user must have the **`%Development:Use`** privilege for the server and be able to read the default globals database for the namespace (that is, have the **`%DB_<database-name>:R`** privilege for it). In order to open a routine, class or other definition, the user must have the Read privilege for the database in which it is stored (which may or may not be the default routines database). In order to compile or save a definition, the user must have the Write privilege for that database.
- For the global, routine, or class capabilities of the InterSystems IRIS system manager utility, the user must have the Read or Write privileges for the database to access or modify globals.
- For the SQL capabilities of the InterSystems IRIS system manager utility, the user must have the appropriate SQL privileges for the tables, views, stored procedures, or other SQL assets. If you have some form of SQL access to a table in a database, you are also granted Read or Write access to that database.

To debug an InterSystems IRIS application, you need no specific database privileges. If you hold the **`%Development:Use`** privilege for the system, you can set a breakpoint in any routine stored in any database on that system. However, you must have the Read privilege for a database to:

- View routine source via the debugger
- Execute a routine

7.2.3 The %DocDB_Admin Resource

The **`%DocDB_Admin`** resource controls the ability to manage a document database application. For more information on this feature, see the Using Document Database guide.

7.2.4 The %IAM Resource

The *%IAM* resource controls the ability to obtain a license from InterSystems IRIS to run the InterSystems API Manager (IAM).

7.2.5 The %System_Callout Resource

The **`%System_Callout`** resource controls access to various tools that perform actions outside of InterSystems IRIS. These include:

- In ObjectScript, using the $ZF(-100) function, which supports invoking operating system commands from within ObjectScript code. Also see Issuing Operating System Commands, which includes detailed instructions for adding the **`%System_Callout:Use`** privilege.
- At the Terminal, using “`!`” and “`$`” as control characters for operating system access. For details, see the $ZF(-100) documentation.
- In local interprocess communication with ObjectScript, opening an interprocess communications device in Q mode. For details, see the OPEN-only Command Keywords for Interprocess Communications Pipes table.

Note: **`%System_Callout`** also controls interactions with the deprecated $ZF(-1) and $ZF(-2) functions.

7.2.6 The %System_Attach Resource

The %System_Attach resource enables the ability to attach the Studio debugger to a running process.

7.2.7 The %Secure_Break Resource

The **`%Secure_Break`** resource enforces the use of the secure debug shell, which restricts programmer access at a <BREAK> prompt. For more information on the secure debug shell, see Secure Debug Shell.

7.2.8 The %Service_Native Resource

The **`%Service_Native`** resource controls whether the user can issue Native SDK calls via Python, Java, .NET, and Node.js. The user must have the **`Use`** permission. The system-defined roles **`%Developer`** and **`%Manager`** have this privilege by default.

7.3 Database Resources

Database resources control access to the contents of InterSystems IRIS databases. The name of the database resource that governs access to a database is stored in the label block of that database.

All database resource names must start with the string “%DB_” and, for custom resources, the first character after the underscore should not be the percent sign. The default database resource name is **`%DB_<database-name>`**. You can change the resource name assigned to a database by using the Management Portal.

7.3.1 Database Resource Privileges

The available database privileges are:

Table 7–1: Database Privileges

Permission	Enables
Read	Data access and routine execution
Write	Modification and deletion of data (including executable code)

Read and Write permissions provide access to all contents of a database, including source and executable code as well as data. InterSystems security management utilities automatically grant the Read permission for any database resource where there is Write access.

Database privileges do not enable protection of individual items, such as routines or globals, within a database. Rather, the same protection is applied to all items of a resource within a database. You can establish higher granularity of protection by storing globals and routines in separate databases. InterSystems IRIS namespace mapping allows you to do this without any application-level modifications.

Note: SQL security grants table-level access, specifying which particular action can be performed, such as **SELECT** or **UPDATE**. For more information on SQL and security, see SQL Users, Roles, and Privileges.

7.3.2 Shared Database Resources

Often, there is a one-to-one correspondence between databases and the resources used to protect them. For instance, protection for the IRISSYS database is specified using the **`%DB_IRISSYS`** resource. However, this is not a requirement and, when several databases share the same security definitions they can share the same security resource.

Consider a sales application with three databases. Rather than define access for each of these individually, the system manager has the choice option to:

1. Create a new database resource, such as **`%DB_SALES`**.
2. Assign this resource to the three databases.
3. Specify suitable access to **`%DB_SALES`** which then governs access to all three databases.

7.3.3 Default Database Resource

When mounting an existing database that has no database resource name, InterSystems IRIS assigns the default resource, **`%DB_%DEFAULT`**, to the database. By default, **`%DB_%DEFAULT`** has the following permissions:

Table 7–2: %DB_%DEFAULT Privileges

Role	Permissions
`%Developer`	Read, Write
`%Manager`	Read, Write

While you can change the privileges associated with **`%DB_%DEFAULT`** resource, you cannot delete the **`%DB_%DEFAULT`** resource itself, since it must be available if an unnamed database is mounted.

7.3.4 Unknown or Non-Valid Resource Names

With one exception (see below), if you attempt to mount a database that has an unknown or invalid database resource name, the attempt fails. (This might occur if a database were moved from one InterSystems IRIS instance to another.) An automatic mount attempt fails with an error and an explicit mount attempt prompts you with the choice of creating the resource named in the database label or changing the database to use a valid resource.

The one exception to this rule is that a user who is a member of the **`%All`** role can mount a database that does not have a resource (such as if its resource was deleted or the database was previously on a different system).

7.3.5 Namespaces

Users and applications interact with InterSystems IRIS databases through namespaces. While there are no privileges associated with namespaces, access to a namespace is granted or denied based on the privileges associated with the underlying databases. More specifically, to access a namespace, you must hold the Read privilege on the default globals database associated with that namespace. This requirement is checked when:

- A process attempts to change to a different namespace, such as by using the **$NAMESPACE** special variable, the **ZNSPACE** command, or the **%CD** utility
- There is an attempt to connect to InterSystems IRIS using any service that connects to a namespace, such as an SQL connection or an object connection

Note: This requirement is not checked when a global or routine reference is made, implicitly or explicitly, to a namespace.

The fact that namespace privileges depend on privileges for the underlying databases can lead to unexpected behavior. For example, suppose that namespace NSCust refers to data in three databases: DBCust1, DBCust2, and DBCust3. Suppose also that the role **`AverageUser`** has the privileges **`%DB_DBCust1:R`** and **`%DB_DBCust3:R`**. Because the role has no privilege associated with DBCust2, any attempt to access data in that database fails (including if it is through the namespace).

7.3.6 IRISSYS, the Manager's Database

InterSystems IRIS ships with a database that provides a repository for administrative routines and globals. This is the IRISSYS database, and is sometimes known as the *manager's database.*

Within this database, there are groups of globals and routines whose names begin with the percent sign (these are often known as "percent globals" or "percent routines"). These globals and routines have a special role in the management of an InterSystems IRIS site and have special rules that apply to them:

- All users have Read permission for percent routines and percent globals.

  Note that via mappings, it is possible to change where these items are stored, but that has no effect on their visibility. Percent routines and percent globals are always visible in all namespaces.
- All percent routines have Write permission for *all* globals located in the same database (percent as well as non-percent). For instance, percent routines in the IRISSYS database have Write access to globals stored in that database, but not to globals in any other database. Simultaneously, percent routines in any other database have implicit Write access to globals stored in that same database but *not* to percent globals in IRISSYS. This implicit Write permission is only available during normal routine execution. It is disabled if the routine has been modified and it is not available in **XECUTE** commands or through argument indirection.

- You can control Write access to percent globals from non-percent routines with the **Enable writing to percent globals** field on the **System-wide Security Parameters** page (**System Administration** > **Security** > **System Security** > **System-wide Security Parameters**); for a description of this page, see System-wide Security Parameters.

CAUTION: Do not move, replace, or delete the IRISSYS database.

7.3.6.1 Special Capabilities

There are special capabilities available to code located in the IRISSYS database. These capabilities are sometimes called "restricted system capabilities." They are:

- Invoking protected **VIEW** commands and **$VIEW** functions.
- Using protected class methods.
- Modifying the roles of a process with a `SET $ROLES = ...` call.
- Invoking the single-argument form of the **$SYSTEM.Security.Login** function (which is the **Login** method of the %SYSTEM.Security class).
- Invoking the two-argument form of the **$SYSTEM.Security.ChangePassword** function (which is the **ChangePassword** method of the %SYSTEM.Security class). (Note that the new password must conform to the general password constraints described in User Account Properties and the instance-specific password constraints described in the Password Strength and Password Policies.
- Invoking one of the $ZF functions, which allow you to call non-ObjectScript programs or functions from ObjectScript routines.

Note: You need no database privileges to read or write database blocks with the **VIEW** command.

The only code that can perform these actions is:

- Routines stored in the IRISSYS database, but only during "normal" routine execution. They are disabled if a **ZINSERT** into the current routine has modified the routine, and they are also not available in **XECUTE** commands or through argument indirection.
- Processes with the Write permission on the **`%DB_IRISSYS`** resource.

7.4 Gateway Resources

Gateway resources control access to the external language servers (also known as gateways) provided in InterSystems IRIS. The gateway resources and the types of external language server they are associated with by default are listed in the following:

- **`%Gateway_ML`** — IntegratedML
- **`%Gateway_Object`** — .NET, Java, Python, R, and XSLT
- **`%Gateway_SQL`** — JDBC

These resources take the Use permission.

The %Admin_ExternalLanguageServerEdit resource controls the ability to create, delete, and modify external language servers, including changing the gateway resources associated with them. You can replace the default gateway resource associated with a gateway with a user-defined resource, or remove it without replacing it, in which case it the gateway is public and can be used by anyone.

Important: InterSystems strongly recommends protecting all gateways by associating a gateway resource.

7.5 Application Resources

InterSystems IRIS supports several forms of custom authorization, all of which depend on user-defined resources, known as Application resources. These include:

- Supplementary authorization checking for a Portal page — for more information, see Use Custom Resources with the Management Portal.
- Authorization checking at a particular point in an application — for more information, see the next section, Create or Edit a Resource.
- Authorization for the whole of an application

For the whole of an application, InterSystems IRIS allows you to create an application definition associated with a user-defined application (which itself is defined as a named entity composed of executable code). Application resources then allow you to perform authorization checking for the application. There are several types of applications:

- Web application definitions
- Privileged Routine application definitions
- Client application definitions
- Document database definitions

Application resources provide a means of controlling access to applications. To use this feature, create a custom resource (as described in Create or Edit a Resource) and use it in association with the application as described in either Edit a Web Application: The General Tab or Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab.

For example, if a web application has an associated resource, then users can only run the application if they have Use permission on the resource. If an application uses other resource-regulated entities, such as databases, then users must also have appropriate permissions on those resources in order to operate the application effectively. For more information on applications, consult Applications.

7.6 Create or Edit a Resource

To create a new resource, on the **Resources** page (**System Administration** > **Security** > **Resources**), click **Create New Resource**.

To edit an existing resource, on the **Resources** page (**System Administration** > **Security** > **Resources**), click the **Edit** button to the right of the resource you wish to edit.

This displays the **Edit Resource** page. The **Edit Resource** page has fields for the following:

- Resource Name — The string by which the resource is identified. For more information on resource names, see Resource Naming Conventions. When creating a resource, this is an editable field; when editing an existing resource, this is a non-editable, displayed string.
- Description — Optional text related to the resource.
- Public Permission —

  - Read — When selected, specifies that all users can view this resource.
  - Write — When selected, specifies that all users can view or change this resource.
  - Use — When selected, specifies that all users can run or otherwise employ this resource.

Once you have added a resource, it appears in the table of resources and is of type Application. You can then use it as part of application-specific authorization. See Check the Privileges of a Process for more information.

7.6.1 Resource Naming Conventions

The names of InterSystems IRIS resources begin with a percent sign character. The names of application-defined resources should not begin with a percent sign character.

Resource names are case-sensitive, but also prohibit naming as if insensitive. This means that:

- Names that differ only by case are prohibited.
- Names are defined using mixed case and the name is preserved exactly as it is entered.
- When a name is looked up, InterSystems IRIS acknowledges differences in case.

For example, if there is a resource named **`Accounting`**, then you cannot create a resource named **`ACCOUNTING`**, **`accounting`** or any combination of letter casing therein. References to the **`Accounting`** resource using capitalization such as **`accounting`** or **`ACCOUNTING`** do not succeed. If you delete the **`Accounting`** resource and subsequently make an **`accounting`** resource, references to **`Accounting`** fail. You must specify the correct letter casing when referencing a resource.

7.7 Use Custom Resources with the Management Portal

By default, the **`%Admin_Manage`**, **`%Admin_Operate`**, **`%Admin_Secure`**, and **`%Development`** system resources control access to the Management Portal. As a supplement to these that allows for more granular Portal security, you can associate an additional custom resource with each Portal page. If a Portal page has an associated custom resource, then the user must hold both the system and custom resource for the page in order to view that page.

For example, access to the **Lock Table** page requires the **`%Operator`** role. You can also associate a custom resource (for example, called **`MyLockTable`**) with the **Lock Table** page; once you have created this association, a user must both be a member of the **`%Operator`** role and also have the **`MyLockTable:Use`** privilege in order to view the **Lock Table** page. With this arrangement, the **`%Operator`** role grants access to fewer pages than in an instance with default settings, and you can define a new role that can view the **`Lock Table`** page and all the other pages to which **`%Operator`** role grants access. You can also create multiple custom resources, so that various roles would have access to various subsets of what a predefined role would have available by default.

This section describes:

- Define and Apply a Custom Resource to a Page
- Remove a Custom Resource from a Page

Important: Because there can be complex interactions among the various pages, resources, and roles, system administrators should plan carefully before implementing custom resources for the Management Portal.

7.7.1 Define and Apply a Custom Resource to a Page

To define and apply a custom resource, the procedure is:

1. Log in as a user who holds the `%Admin_Secure:Use` privilege or is a member of the `%All` role.
2. Create the custom resource. To do this, on the **Resources** page (**System Administration** > **Security** > **Resources**), click **Create New Resource**. When creating the resource, make sure that you properly set its public permissions according to the instance's needs.
3. Associate the privilege to use the custom resource with a role. For an existing role, on the **Roles** page (**System Administration** > **Security** > **Roles**), simply add the privilege to the role; alternately, (also on the **Roles** page), create a new role and then add the privilege to it immediately after creating it. Either way, the privilege consists of the custom resource and the Use permission.
4. Assign the custom resource to the page. To do this:
    a. Use the finder feature of the Portal to select the page. Note that clicking on the name of the page takes you directly to that page; click inside the box (but not on the name itself) to display the page's action pane.
    b. At the very bottom of the page's action pane, click **Assign**. This displays the **Assign Custom Resource** dialog.
    c. In that dialog, select the desired resource from the **Custom Resource Name** list and click **OK**.

7.7.2 Remove a Custom Resource from a Page

To disassociate a custom resource from a page, the procedure is:

1. Log in as a user who holds the `%Admin_Secure:Use` privilege or is a member of the `%All` role.
2. Use the finder feature of the Portal to select the page. Note that clicking on the name of the page takes you directly to that page; click inside the box (but not on the name itself) to display the page's action pane.
3. At the very bottom of the page's action pane, click **Assign**. This displays the **Assign Custom Resource** dialog.
4. In that dialog, select the empty item from the **Custom Resource Name** list and click **OK**.

8

Privileges and Permissions

Permissions allow users to perform some action, such as reading or writing data, or using a tool. Permissions are associated with resources, forming privileges. A privilege is written as a resource name followed by a permission separated by a colon, such as `%DB_Sales:Read`, which describes an action a user can perform.

A group of privileges, in turn, is called a role. Performing an action requires a user to be a member of a role that holds the appropriate privilege. This model provides precision when specifying what tasks a user (or group of users) can do – to make an adjustment, simply adjust the privileges in that user's role.

8.1 How Privileges Work

A privilege associates a resource with a permission, so that a role that holds the privilege can perform a particular action, such as read or write to a database or use an application. The possible permissions are:

- Read — View (but not change) the contents of a resource, such as in a database
- Write — View or change the contents of a resource, such as in a database
- Use — Run or otherwise employ an executable program or tool, such as an application or an InterSystems service

The meaning of each permission depends on the resource with which it is used. Permission names can appear as the full word or the first letter of the word; their names are not case-sensitive.

8.2 Public Permissions

For each resource, permissions can be designated as Public. Effectively, this is equivalent to all users holding that permission on the resource. For example, if the **`%DB_SALES:Read`** privilege is Public, then any user can read from any database protected by the %DB_SALES resource. This does not, however, enable all users to write those databases because (in this example) the **`%DB_SALES:Write`** privilege is not Public.

The following database privileges are Public by default:

Table 8–1: Default Public Privileges

Resource	Permission
%DB_IRIS	Read
%DB_IRISLIB	Read
%DB_IRISTEMP	Read, Write

8.3 Check the Privileges of a Process

InterSystems IRIS® data platform provides a method, **$SYSTEM.Security.Check**, to check on privileges held by the current process. Its one-argument form lists what privileges the process holds on a particular resource; its two-argument form returns whether or not the process holds privileges for a particular resource. (There are also methods with built-in privilege checks, described in the next section.)

The one-argument form returns a comma-delimited list of the permissions held by the process on a resource. For example:

```
$SYSTEM.Security.Check("%DB_TESTDATABASE")
```

returns `READ,WRITE` if the process holds Read and Write permissions for %DB_TESTDATABASE. The permission names are always returned as full words in uppercase letters. The function returns an empty string if the process holds no permissions on the resource.

The two-argument form returns True or False (1 or 0) to indicate whether the process holds a specific privilege. For example:

```
$SYSTEM.Security.Check("%DB_TESTDATABASE", "WRITE")
```

returns 1 if the process holds the Write permission on the %DB_TESTDATABASE resource.

You can also call the function with a list of permissions, such as:

```
$SYSTEM.Security.Check("%DB_TESTDATABASE", "WRITE,READ")
```

It returns 1 if the process holds all of the requested permissions and 0 otherwise. You can also simply use the first letter of the privileges to be checked:

```
$SYSTEM.Security.Check("%DB_TESTDATABASE", "W,R")
```

The method has the following general behaviors:

- The method always returns 1 for a public resource privilege, whether or not the process explicitly holds that privilege.
- Permission names are not case-sensitive.
- Permission names can be fully spelled out, as in the example above, or abbreviated by their first character. Also, permission names are not case-sensitive. Thus, “WRITE,READ”, “W,R”, and “R,Write” are equivalent.

8.4 Use Methods with Buillt-In Privilege Checks

InterSystems IRIS allows methods to require that the process that calls them has certain specified privileges.

This feature uses the Requires method keyword. The Requires method keyword has a quoted string value that is a comma-delimited list of privileges. Each privilege names a resource and its associated permission (Use, Read, or Write) in standard format.

For example, if the **MyAction** method requires the **`Service_FileSystem:Use`** privilege, its signature might be:

```
ClassMethod MyAction() [ Requires="Service_FileSystem:Use"]
{
 // Method content
}
```

If the Requires keyword has a value, the method may only run if the calling process has the required privilege at the time that it invokes the method. If the process does not have the required privilege, the system generates a <PROTECT> error.

Methods that inherit this keyword may require additional resources by overriding and setting a new value for the keyword. There is no way to remove requirements.

8.5 When Changes in Privileges Take Effect

InterSystems IRIS maintains a persistent database of the security settings. When InterSystems IRIS starts, it extracts this information and places it into a segment of shared memory that allows quick access to the consolidated settings. While a process is executing, it maintains its own per-process cache of the privileges it has been granted. This is updated as new privileges are needed (and authorized).

Editing roles, privileges, and so on makes changes to the persistent copy of the information. This becomes visible to users or applications the next time they are subsequently authenticated.

9

Roles

A role is a named collection of privileges. Roles are useful because multiple users often need the same set of privileges. For example, all users of an application or all developers working on a particular project might need a common set of privileges. By using a role, such sets of privileges can be defined once (which makes future modification much easier) and shared by the relevant users.

Privileges are assigned exclusively to roles; privileges are not assigned directly to users. To assign some privileges to a single user, create a role for that purpose.

Note: For SQL access to data in tables, InterSystems IRIS® data platform supports row-level security. For information on setting this up, see Adding Row-Level Security.

9.1 About Roles

Every role has the following properties:

Table 9–1: Role Properties

Property Name	Property Description
Name	Unique role identifier. See Naming Conventions for more information on valid names.
Description	Any text.
Privileges	Resource-permission pair(s) associated with the role. A role can hold zero or more privileges.
Members	Users or roles that have been assigned to the role (listed on the **Members** tab of the **Edit Role** page).

These are displayed on the **General** tab of the **Edit Role** page, which is accessible by selecting Edit in the row for any role in the table on the Roles page (**System Administration** > **Security** > **Roles**).

Each role also may have members that are assigned to it or other roles to which it is assigned. These relations are described in Roles, Users, Members, and Assignments.

9.1.1 About Role Assignment

InterSystems IRIS also supports various role-assignment mechanisms. A *role-assignment mechanism* allows you to associate particular roles with particular authenticated users. InterSystems IRIS uses these associations to determine the authorized activities for the user. Each role-assignment mechanism is associated with one or more authentication mechanisms; configuring InterSystems IRIS includes specifying the supported combination(s) of authentication and role-assignment mechanisms.

The available role-assignment mechanisms are:

Table 9–2: Authentication Mechanisms and Role-Assignment Mechanisms

Authentication Mechanism	Role-Assignment Mechanisms
Delegated authentication (**ZAUTHENTICATE**)	**ZAUTHENTICATE**
Instance authentication	Native authorization (the primary approach described by this guide)
LDAP	LDAP
Kerberos	Options of: • Delegated authorization (**ZAUTHORIZE**) • Native authorization
Operating System authentication	Options of: • Delegated authorization (**ZAUTHORIZE**) • LDAP • Native authorization

For an instance that supports unauthenticated access, all users hold the privileges associated with the UnknownUser and _PUBLIC accounts; these accounts are described in The UnknownUser Account and The _PUBLIC Account respectively.

Note: Regardless of how role assignment occurs, role *management* — that is, associating particular privileges with particular roles — occurs within InterSystems IRIS.

9.1.2 Maximum Number of Roles

Each instance of InterSystems IRIS can have up to 10,240 roles.

9.2 Roles, Users, Members, and Assignments

A role is a container that holds one or more privileges. If a user is associated with a role, then that user is able to exercise the role’s privileges. The terminology for the association of a user and role is:

- The user *is assigned to* the role.
- The user *is a member of* the role.
- The role *includes* the user.

These phrases are all equivalent in meaning to each other.

Each user can be assigned to multiple roles and each role can have multiple users as its members. Similarly, each role can also be assigned to multiple roles and can also have multiple roles as its members. A role can have both users and roles as its members.

Suppose one role is assigned to another role. In this case, if role A is assigned to role B, then role A is described as a “member” of role B; this is equivalent to saying that role A is assigned to role B or that role B includes role A.

If one role is assigned to another, that first role holds the privileges associated with the second role. This is analogous to the relationship of assigning a user to role, whereby the user then holds the privileges associated with the role. Hence, if a user is a member of one role and that role is a member of another role, then the user holds privileges associated with both the roles.

For example, suppose a university has three roles available for its students: **`UndergraduateStudent`**, **`GraduateStudent`**, and **`GeneralStudent`**. Each student is assigned to either **`UndergraduateStudent`** or **`GraduateStudent`**, and these two roles are both assigned to **`GeneralStudent`**. If Elizabeth is assigned to **`GraduateStudent`**, she holds the privileges associated with both **`GraduateStudent`** and **`GeneralStudent`**; if James is assigned to **`UndergraduateStudent`**, he holds the privileges associated with both **`UndergraduateStudent`** and **`GeneralStudent`**.

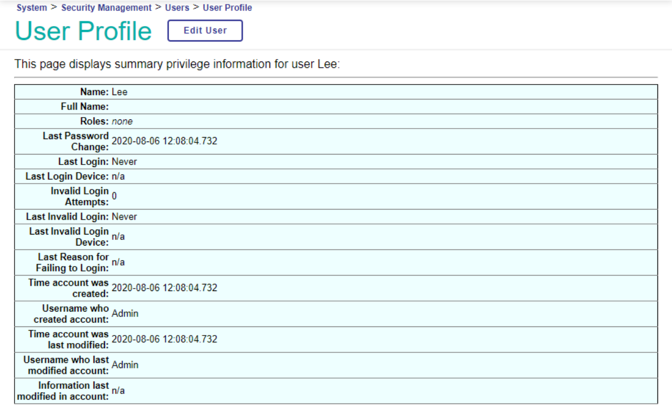
A role’s members are listed on the **Edit Role** page’s **Members** tab; on this tab, you can also assign new members to a role. If a role has been assigned to other roles, these are listed on the **Assigned To** tab of the **Edit Role** page; you can also assign a role to additional roles on that tab.

9.2.1 An Example of Multiple Role Assignment

This section provides an example of how users and roles interact in InterSystems IRIS.

Suppose there is a user named Lee, a role named **`FirstRole`**, and a role named **`SecondRole`**. **`FirstRole`** protects a resource called **`FirstResource`** and **`SecondRole`** protects a resource called **`SecondResource`**.

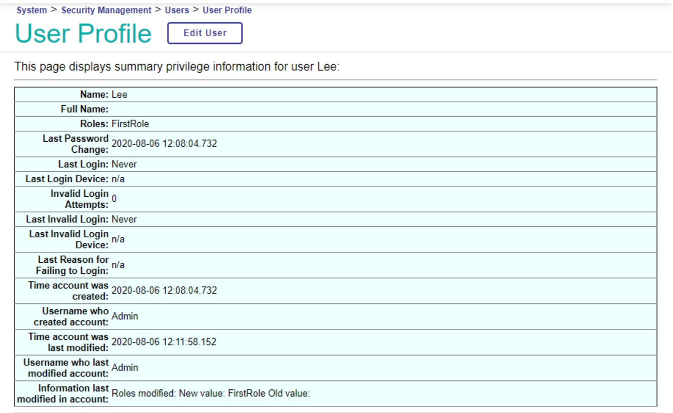
When first created, Lee is not a member of any roles. This is reflected in Lee’s profile:



This page displays summary privilege information for user Lee:

Name:	Lee
Full Name:	
Roles:	*none*
Last Password Change:	2020-08-06 12:08:04.732
Last Login:	Never
Last Login Device:	n/a
Invalid Login Attempts:	0
Last Invalid Login:	Never
Last Invalid Login Device:	n/a
Last Reason for Failing to Login:	n/a
Time account was created:	2020-08-06 12:08:04.732
Username who created account:	Admin
Time account was last modified:	2020-08-06 12:08:04.732
Username who last modified account:	Admin
Information last modified in account:	n/a

When Lee is assigned to the role **`FirstRole`**, this changes Lee’s profile:



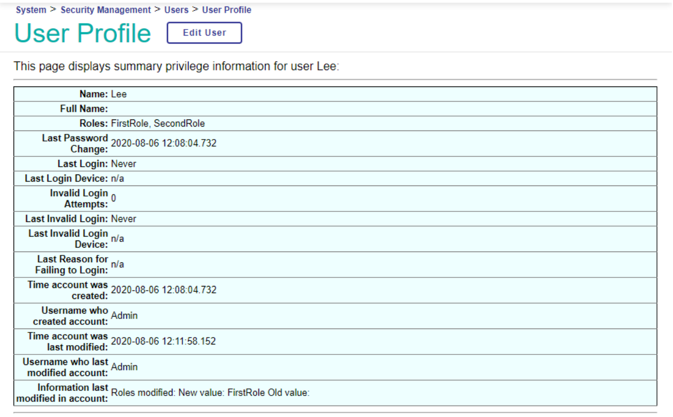
System > Security Management > Users > User Profile

User Profile **Edit User**

This page displays summary privilege information for user Lee:

Name:	Lee
Full Name:	
Roles:	FirstRole
Last Password Change:	2020-08-06 12:08:04.732
Last Login:	Never
Last Login Device:	n/a
Invalid Login Attempts:	0
Last Invalid Login:	Never
Last Invalid Login Device:	n/a
Last Reason for Failing to Login:	n/a
Time account was created:	2020-08-06 12:08:04.732
Username who created account:	Admin
Time account was last modified:	2020-08-06 12:11:58.152
Username who last modified account:	Admin
Information last modified in account:	Roles modified: New value: FirstRole Old value:

When the role FirstRole is assigned to the role **`SecondRole`**, this also changes Lee’s profile:



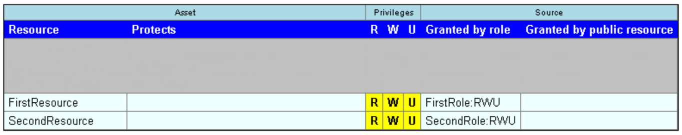
System > Security Management > Users > User Profile

User Profile **Edit User**

This page displays summary privilege information for user Lee:

Name:	Lee
Full Name:	
Roles:	FirstRole, SecondRole
Last Password Change:	2020-08-06 12:08:04.732
Last Login:	Never
Last Login Device:	n/a
Invalid Login Attempts:	0
Last Invalid Login:	Never
Last Invalid Login Device:	n/a
Last Reason for Failing to Login:	n/a
Time account was created:	2020-08-06 12:08:04.732
Username who created account:	Admin
Time account was last modified:	2020-08-06 12:11:58.152
Username who last modified account:	Admin
Information last modified in account:	Roles modified: New value: FirstRole Old value:

The list of Lee’s privileges specifies which privileges originate with which roles:



Asset		Privileges			Source	
Resource	**Protects**	**R**	**W**	**U**	**Granted by role**	**Granted by public resource**
[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
FirstResource		R	W	U	FirstRole:RWU	
SecondResource		R	W	U	SecondRole:RWU	

9.3 Create Roles

You can define roles for use by developers, operators, system managers and other classes of users. Once created, there are various features available to edit a role.

To create a new role:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Create New Role**. This displays the **Edit Role** page.
3. On the **Edit Role** page, the **General** tab is visible. Here, enter values for the following properties:
    - **Name** (required) — Specifies the name of the new role. See the following section, Naming Conventions, for naming rules.
    - **Description** (optional) — Specifies descriptive information about the role.

The role’s resources are listed, but empty, as a role cannot receive resources until it has been created, except under the conditions described in the next step.

4. As a shortcut, if you wish to create multiple roles with similar characteristics, you can use the **Copy from** field on the **Role** page to begin the process of creating a new role. When you select an existing role from this field’s drop-down menu, all its privileges appear in the list of resources; you can then add or delete privileges as desired, and modify its Description property.
5. Click **Save** to create the role.

Once a role exists, you can edit it as described in Manage Roles. For example, the **Resources** table allows you to add new privileges to the role; do this by clicking **Add**.

Note: InterSystems recommends that you do not modify predefined roles.

9.3.1 Naming Conventions

The name of a user-defined role is subject to the following rules in its use of characters:

- It can include all alphanumeric characters.
- It can include symbols, except for the following prohibited characters: “,” (comma), “:” (colon), and “/” (slash).
- It cannot begin with “%” (the percent-sign character), which is reserved for InterSystems IRIS predefined roles.
- It can include Unicode characters.
- It cannot be the same as an existing username.

Also, a role name is not case-sensitive. This means that:

- For names that are defined using mixed case, the name is preserved exactly as it is entered.
- Names that differ only by case are prohibited.
- When a name is looked up, InterSystems IRIS ignores differences in case.

A role name can be up to 64 characters long.

For example, if there is a role named **`BasicUser`**, then you cannot create a role named **`BASICUSER`**. References to the **`BasicUser`** role using capitalization such as **`basicuser`** or **`BASICUSER`** will succeed.

9.4 Manage Roles

Once you have created a role, you modify it in a number of different ways, each of which is described in one of the following sections. The actions fall into several categories:

- General tasks. This includes:
  - View Existing Roles
  - Delete a Role
- Creating, modifying, and removing a role’s privileges. This includes:
  - Give New Privileges to a Role
  - Modify Privileges for a Role

  - Remove Privileges From a Role

- Creating and removing assignments among roles and users. This includes:
  - Assign Users or Roles to the Current Role
  - Remove Users or Roles From the Current Role
  - Assign the Current Role to Another Role
  - Remove the Current Role From Another Role

- Modify a Role’s SQL-Related Options

Note: Changing a user’s roles or changing a role’s privileges does not affect the assigned privileges associated with the user’s existing processes. For new privileges to become active, the user must log out and log in again, restart the process, or perform an equivalent action.

9.4.1 View Existing Roles

To view a list of the currently existing roles, see the **Roles** page in the Portal (**System Administration** > **Security** > **Roles**). This page displays information on the following fields:

- Name — The role’s name (cannot be edited)
- Description — Any text that has been provided to describe the role
- Created By — What user created the role

For each role, you can

- Edit the role’s properties, which includes all actions for privilege management, assignment management, and SQL-related options.
- Delete the role

9.4.2 Delete a Role

To delete a role:

1. On the **Roles** page (**System Administration** > **Security** > **Roles**), for the role you wish to delete, click **Delete** in that role’s row.
2. InterSystems IRIS displays a confirmation dialog. Click **OK** to delete the role and **Cancel** otherwise.

9.4.3 Give New Privileges to a Role

To give a role new privileges:

1. On the **Edit Role** page (**System Administration** > **Security** > **Roles** > **Edit Role**) for an existing role, click **Add** in the **Privileges** table.
2. This displays a page listing all resources. To select a resource to assign to the role, click it. You can select multiple resources simultaneously using the **Ctrl** or **Shift** keys.
3. To add the selected resources to the role, click **Save**. This gives the role all possible permissions on the resource; you can then modify the available permissions for the resource (such as changing permissions on a database from Read-Write to just Read).

9.4.4 Modify Privileges for a Role

To modify the privileges that a role holds:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Edit** for the role you wish to modify. This displays the **Edit Role** page.
3. On the **Edit Role** page, in the **Resources** table, click **Edit** for the resource whose privileges you wish to modify.
4. This displays the page for editing the permissions on the selected resource. Check or clear the boxes for each permission as appropriate.

   Note: This page does not let you clear all permissions for an individual resource. This is because eliminating all a role's permissions for a resource is equivalent to deleting the role's privileges for the resource.

5. Click **Save** to save the privileges in their new state.

These changes should be reflected in the **Resources** table on the **Role** page.

9.4.5 Remove Privileges From a Role

To remove privileges from a role:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Edit** for the role you wish to modify. This displays the **Edit Role** page.
3. On the **Edit Role** page, in the **Resources** table, click **Delete**. This removes the privileges for the resource from the role.
4. Click **Save** to save the privileges in their new state.

9.4.6 Assign Users or Roles to the Current Role

A role can have users or other roles as members that are assigned to it. If a user is assigned to a role, then that user holds the privileges associated with that role. If one role is assigned to another role, then a user assigned to the first role holds the privileges associated with both roles.

The role being edited is known as the "current" role. The users and roles that are assigned to the current role are listed on the **Members** tab of the **Edit Role** page (these users and roles are known as its *members*).

To assign a user or role to the current role, the procedure is:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Edit** for the role you wish to modify. This displays the **Edit Role** page.
3. On the **Edit Role** page, select the **Members** tab.
4. On the **Members** tab, choose either the Users or Roles option to specify whether to assign users or roles to the role. (Users is the default.)
5. The list of users or roles that can be assigned to the current role appears in the **Available** list. You can move them to and from the **Selected** list using the arrow buttons between the lists.
6. When you have the final list of users or roles to add, click **Assign** or **Assign with Grant Option**. Clicking **Assign** simply assigns the new members (users or roles) to the role being edited. Clicking **Assign with Grant Option** also gives the new members the ability to assign other users or roles to the current role when using SQL commands.

9.4.7 Remove Users or Roles From the Current Role

If a user or role has been assigned to the current role, it is known as a member of that role. The procedure to remove a member from a role is:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Edit** for the role you wish to modify. This displays the **Edit Role** page.
3. On the **Edit Role** page, select the **Members** tab.
4. On the **Members** tab, there is a table of users and roles assigned to the current role. For the specified members, click the **Remove** button in the right-most column of the member's row.
5. A prompt appears to confirm the removal. Click **OK**.

The user or role has now been removed from the current role.

9.4.8 Assign the Current Role to Another Role

One role can be assigned to another role. If one role is assigned to another role, then a user assigned to the first role holds the privileges associated with both roles.

The role being edited is known as the "current" role. The roles to which the current is assigned are listed on the **Assigned To** tab of the **Edit Role** page.

To assign the current role to another role, the procedure is:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Edit** for the role you wish to modify. This displays the **Edit Role** page.
3. On the **Edit Role** page, select the **Assigned To** tab.
4. The list of roles that the current role can be assigned to appears in the **Available** list. You can move them to and from the **Selected** list using the arrow buttons between the lists.
5. When you have the final list of roles, click **Assign** or **Assign with Grant Option**. Clicking **Assign** simply assigns the current role to the selected roles. Clicking **Assign with Grant Option** also gives the current role the ability to assign other users or roles to the selected role(s) when using SQL commands.

9.4.9 Remove the Current Role From Another Role

If the current role has been assigned to another role, it is known as a member of that role. The procedure to remove the current role from another role is:

1. From the Management Portal home page, go to the **Roles** page (**System Administration** > **Security** > **Roles**).
2. On the **Roles** page, click **Edit** for the role you wish to modify. This displays the **Edit Role** page.
3. On the **Edit Role** page, select the **Assigned To** tab.
4. On the **Assigned To** tab, there is a table of roles to which the current role is assigned. To remove the current role from one of these roles, select the **Remove** button in the right-most column of that role's row.
5. A prompt appears to confirm the removal. Click **OK**.

The current role has now been removed from that role.

9.4.10 Modify a Role’s SQL-Related Options

For every role, you can grant or remove the following SQL-related characteristics:

- General SQL Privileges
- Privileges for Tables
- Privileges on Views
- Privileges for Stored Procedures

9.4.10.1 General SQL Privileges

On the **SQL Privileges** tab of the **Edit Role** page, you can add or remove SQL privileges for a role:

- To add a privilege to a role, first move the privilege from the **Available** list to the **Selected** list (either double-click it or select it and then click the single right-arrow); click **Assign** to give the privilege to the role. To also add the privilege of being able to grant the added privilege to other roles, select the relevant check box below the **Available** list.
- To add all privileges to a role, click the double-arrow pointing from the **Available** list to the **Selected** list; click **Assign** to give the privileges to the role. To also add the privileges of being able to grant the added privileges to other roles, select the relevant check box below the **Available** list.
- To remove a privilege from a role, click **Remove** to the right of privilege name.
- To remove all privileges from a role, click **Remove All** below the table listing the currently assigned privileges.

The following privileges are available:

- %ALTER_TABLE — For a given namespace, allow the member of the role to run the ALTER TABLE command.
- %ALTER_VIEW — For a given namespace, allow the member of the role to run the ALTER VIEW command.
- %CREATE_FUNCTION — For a given namespace, allow the member of the role to run the CREATE FUNCTION command.
- %CREATE_METHOD — For a given namespace, allow the member of the role to run the CREATE METHOD command.
- %CREATE_PROCEDURE — For a given namespace, allow the member of the role to run the CREATE PROCEDURE command.
- %CREATE_QUERY — For a given namespace, allow the member of the role to run the CREATE QUERY command.
- %CREATE_TABLE — For a given namespace, allow the member of the role to run the CREATE TABLE command.
- %CREATE_TRIGGER — For a given namespace, allow the member of the role to run the CREATE TRIGGER command.
- %CREATE_VIEW — For a given namespace, allow the member of the role to run the CREATE VIEW command.
- %DROP_FUNCTION — For a given namespace, allow the member of the role to run the DROP FUNCTION command.
- %DROP_METHOD — For a given namespace, allow the member of the role to run the DROP METHOD command.
- %DROP_PROCEDURE — For a given namespace, allow the member of the role to run the DROP PROCEDURE command.
- %DROP_QUERY — For a given namespace, allow the member of the role to run the DROP QUERY command.
- %DROP_TABLE — For a given namespace, allow the member of the role to run the DROP TABLE command.
- %DROP_TRIGGER — For a given namespace, allow the member of the role to run the DROP TRIGGER command.

- %DROP_VIEW — For a given namespace, allow the member of the role to run the DROP VIEW command.

9.4.10.2 Privileges for Tables

On the **SQL Tables** tab of the **Edit Role** page, you can add or remove table-related SQL privileges for a role:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace's tables appears.
2. To change privileges for a table, click **Edit** in that table's row. This displays a window for altering privileges.
3. In this window, you can check or clear any of the following items:
    - ALTER
    - DELETE
    - INSERT
    - REFERENCES
    - SELECT
    - UPDATE
    - Give the GRANT option to the role
4. After making your selection(s), click **Apply** to establish the new privileges for the table.

If a role has privileges for a table, it is listed in a table on this page. To revoke the role's privileges for a table, click **Revoke** at the far right of the role's row. Clicking this displays a message containing the namespace and the full name of the table (including the schema), such as the "SAMPLES Sample.Company" namespace and table.

9.4.10.3 Privileges on Views

On the **SQL Views** tab of the **Edit Role** page, you can add or remove view-related SQL privileges for a role.

To add privileges for the view:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace's views will appear.
2. To change privileges for a view, click **Edit** in that view's row. This displays a window for altering privileges.
3. In this window, you can check or clear any of the following items:
    - ALTER
    - DELETE
    - INSERT
    - REFERENCES
    - SELECT
    - UPDATE
    - Give the GRANT option to the role
4. After making your selection(s), click **Apply** to establish the new privileges for the table.

If a role has privileges for a view, it is listed in a table on this page. To revoke the role's privileges for a view, click **Revoke** at the far right of the role's row. Clicking this displays a message containing the namespace and the full name of the view (including the schema).

9.4.10.4 Privileges for Stored Procedures

On the **SQL Procedures** tab of the **Edit Role** page, you can add or remove a role's SQL privileges related to stored procedures.

To add privileges for a stored procedure:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace's stored procedures then appears.
2. Below this window, click **Add**, which displays the **Grant procedure privilege...** dialog.
3. In this dialog, near the top, select the schema from the drop-down that contains the procedure that you wish to add. This displays a list of the schema's procedures in the **Available** window on the left part of the page.
4. Move one or more procedures into the **Selected** window. Make sure the **EXECUTE** box is checked, so that the role has the privilege to execute the stored procedure.
5. Optionally, you can grant the roles the ability to grant this privilege on other roles; to do this, click the **Grant privilege** box near the bottom of the page.
6. Click **Apply** to grant the privilege(s) to the role.

If a role has privileges for a stored procedure, it is listed in a table on this page. To revoke the role's privileges for a stored procedure, click **Revoke** at the far right of the role's row. Clicking this displays a message containing the namespace and the full name of the stored procedure (including the schema).

9.5 Predefined Roles

InterSystems IRIS includes a number of predefined roles. These include:

- **`%All`** — The ability to perform all operations.
- **`%Developer`** — The privileges typically associated with application development. These are roughly the privileges associated with the Portal's System Exploration menu. They include the ability to use the System Explorer, WebStress, and UnitTest pages in the Management Portal, as well as the documentation class reference (sometimes known as Documatic).
- **`%Manager`** — The privileges typically associated with system management. These are roughly the privileges associated with the Portal's System Administration and System Operation menus.
- **`%Operator`** — The privileges typically associated with system operation. These are roughly the privileges associated with the Portal's System Operation menu.
- **`%SQL`** — The privileges typically associated with SQL-related tasks.
- **`%SecureBreak`** — The **`%Secure_Break:Use`** privilege, which enforces use of the secure debug shell. For more information on the secure debug shell, see Secure Debug Shell.

Note: InterSystems recommends that you do not modify predefined roles. Rather, create a new role based on the predefined role and modify the role that you have created.

The following table has a column for each role. Each row of the table lists a system-defined resource and the privilege, if any, that the role holds on it.

Table 9–3: Predefined Roles and Their Privileges

Resource	%Developer	%Manager	%Operator	%SQL	%SecureBreak
`%Admin_Manage`		Use			
`%Admin_Operate`		Use	Use		
`%Admin_Secure`		Use			
`%Admin_Task`		Use			
`%DB_IRISLOCALDATA`	Read	Read	Read		
`%DB_IRISAUDIT`		Read			
`%DB_IRISLIB`	Read	Read, Write			
`%DB_IRISSYS`		Read, Write	Read, Write		
`%DB_IRISTEMP`	Read, Write	Read, Write	Read, Write		
`%DB_%DEFAULT`	Read, Write	Read, Write			
`%Development`	Use	Use			
`%DocDB_Admin`	Use	Use			
`%Secure_Break`					Use
`%Service_Console`					
`%Service_DocDB`	Use	Use	Use		
`%System_Native`	Use	Use			
`%Service_Object`	Use	Use			
`%Service_SQL`	Use	Use		Use	
`%Service_Telnet`	Use	Use			
`%Service_Terminal`	Use	Use			
`%Service_WebGateway`	Use	Use	Use		

The definitions of these predefined roles are set during a new InterSystems IRIS installation and are not modified during an upgrade installation. With the exception of **`%All`**, the use of predefined roles is optional.

The **`%Admin_Secure`** resource is designed to make all the necessary security assets available or restricted as a single unit. This makes it easy to separate these resources for use by the security administrator.

Note: The **`%Operator`** role does not hold the **`%Admin_Task:Use`** privilege by default; if you wish for members of that role to be able to manage tasks, include **`%Admin_Task:Use`** among the role's privileges. Further, any custom roles based on **`%Operator`** must add the **`%DB_IRISSYS:RW`** privilege in order to use the Portal's **Operator** menu. They may also add the **`%Admin_Task:Use`** privilege so that they can manage tasks.

9.5.1 %All

The predefined role, **`%All`**, always holds all privileges for all resources on the system. This is why, for example, a user belonging to the **`%All`** role can still mount a database for which there is no resource available. (One exception is the restrictive **`%Secure_Break:Use`** privilege, which must always be explicitly granted.)

This role cannot be deleted or modified, and there must always be at least one user account holding the **`%All`** role. If there is only one such account, it cannot be deleted or disabled. This is designed to protect a sole InterSystems IRIS system administrator from being inadvertently locked out of the system.

Important: A user who is assigned to the **`%All`** role does not automatically have access to rows in a table that are protected with row-level security. The application must explicitly provide the **`%All`** role with access to such a row. For detailed information about how to do this, see Adding Row-Level Security.

9.5.2 Default Database Resource Roles

When you create a database resource, the system automatically creates a role with the name **`%DB_<database-resource-name>`** that has Read and Write permissions for that resource. The **`%DB_<database-resource-name>`** roles are read-only and therefore cannot be modified; hence, for each of these roles, you cannot add privileges for other resources in addition to the RW access to the database resource for which the role is named.

9.6 Login Roles and Added Roles

Each InterSystems IRIS process has, at any point in time, a set of roles that determine the current privileges for that process. The set of roles includes both *login roles*, which come from the definition of the user (received at login time) and *added roles*, which come from the currently running application (received by application role escalation). From a security standpoint, the origin of a role is immaterial: a process either has a required privilege or it does not.

When an application is started, each role currently held by the process is looked up in a table and any associated application roles are added.

For example, suppose there is an order entry application with two classes of users: normal users, who are assigned the **`OrderEntryUser`** role, and managers, who are assigned the **`OrderEntryManager`** role. Both of these roles allow someone to run the order entry application (that is, both are assigned the **`%Application_OrderEntry:Use`** privilege.) But, when the application does role escalation, different roles are used (**`OrderEntryAppNormal`** versus **`OrderEntryAppSpecial`** and **`OrderEntryAppReporting`**) to enable the application to perform different functions on behalf of these user classes.

Matching Role	Added Roles
`OrderEntryUser`	**`OrderEntryAppNormal`**
`OrderEntryManager`	**`OrderEntryAppSpecial`**, **`OrderEntryAppReporting`**

During the matching sequence, each role held by the process is considered, even if a match has already been found. In other words, multiple roles may match and multiple sets of new roles may be added. However, this process is not recursive: roles added as a result of the matching process are not considered for further matches.

Note: There is no way to restrict a user's roles to fewer than the login roles.

9.6.1 A Note on Added Roles and Access in the Management Portal

When a user goes to a new Portal page, the Portal resets the process to have only the user's login roles. The Portal then checks if the page's application requires a resource; if it does, then the Portal checks if the user has the appropriate permissions on that resource. If the user's privileges do not include the required privileges, the page will not be available.

If the user does have the required privileges, the Portal then adds any application roles and any applicable target roles. The Portal then checks if any links on the page require custom resources; if the user has the appropriate resource(s), the Portal displays those links.

9.7 Programmatically Manage Roles

Certain routines can directly modify the application roles of a running process by setting the *$ROLES* system variable, such as

```
SET $ROLES = "Payroll"
```

$ROLES contains a comma-separated list of the role names assigned to the current process. The union of all the privileges granted to all roles in the list determines the privileges that the process possesses. *$ROLES* initially contains the roles assigned at authentication (that is, login roles).

This command can only be invoked either from a routine that is part of the IRISSYS database or if the current privileges held include Write permission for the IRISSYS database (**`%DB_IRISSYS:W`**).

Note that setting *$ROLES* only alters a process's added roles, not its login roles. Thus if a process currently has the login roles Employee and Manager and the added role **`Payroll`**, after the statement

```
SET $ROLES = "Accounting"
```

$ROLES has the value "Employee,Manager,Payroll,Accounting".

A role can be added to the process's current roles by concatenating it to the current roles, with a call such as:

```
SET $ROLES = $ROLES _ ",Payroll"
```

The statement

```
SET $ROLES = ""
```

removes all added roles.

The **NEW** command can be used with *$ROLES* to stack the current set of roles (Login and Added) and the current value of *$USERNAME*. This allows code to modify the list and, whether control leaves the containing block normally or abnormally, the changes are undone upon exit.

With the exception of a null string argument, `SET $ROLES = <role_name>` is a system capability. `NEW $ROLES` and `SET $ROLES = ""` can be executed by any code.

10

User Accounts

User accounts represent the actual users of InterSystems IRIS® data platform. The roles a user account is a member of determine what resources that use can access. You can also give user accounts specific SQL privileges.

10.1 User Account Properties

Each user account in InterSystems IRIS has a number of properties. To view the properties for an account, navigate to the **Users** page of the Management Portal (**System Administration** > **Security** > **Users**) and select the user account you want to view.

The **General** tab for the user account contains the following properties. For information about the other tabs, see Modify a User's Roles and Modifying a User's SQL-Related Options.

Table 10–1: User Account Properties

Property Name	Property Description
Name	Unique user identifier that is up to 128 characters long. Once the user account is created, this cannot be changed. This can include any character except "@" or "*". A name is not case-sensitive. All usernames can include Unicode characters. A username cannot be the same string as an existing role.
Full Name	The user account's displayable name.
Comment	Any text.
Password	New password value. This value is never visible, regardless of the privileges of user viewing this page; a user either with the **`%Admin_Secure:Use`** privilege or assigned to the **`%All`** role can change another user's password, such as if that user's password has been forgotten or lost. A password is case-sensitive, may include Unicode characters, and must conform to the pattern (types of characters and length) specified in the **Password Pattern** field on the **System Security Settings** page (**System Administration** > **Security** > **System Security** > **System-wide Security Parameters**).
Confirm Password	Confirmation of new password value.

Property Name	**Property Description**
Change password on next login	A check box specifying whether or not the user is required to change the password at the next login.
Password never expires	A check box specifying whether or not the system-wide password expiration limit applies to this user. If selected, the user's password does not expire, even if it has been unchanged for longer than the system limit. To set the password expiration limit, see the System-wide Security Parameters page.
User enabled	A check box specifying whether or not the account is currently enabled.
Account Never Expires	A check box specifying whether or not the system-wide account inactivity limit applies to this user. If selected, the user's account does not expire, even if it has been inactive for longer than the system limit. To set the inactivity limit, see the System-wide Security Parameters page.
Account expiration date	The last date on which the account can be used.
Startup Namespace	The namespace in which to begin execution following login from a terminal-type service or the Portal. This property overrides any namespace value provided via the command invoking InterSystems IRIS.
Startup Tag^Routine	The routine to execute automatically following login from a terminal-type service. This property overrides any routine value provided via the command invoking InterSystems IRIS.
Email Address	The email address associated with this account.
Mobile Phone Service Provider	For two-factor authentication, the user's mobile phone service provider. If the user's mobile phone service provide does not appear in the list, you can add a new provide by clicking **Create new provider**; this displays fields for adding a new mobile phone service provider, which will then be visible.
Mobile Phone Number	For two-factor authentication, the mobile phone number at which the user receives a text message containing the second authentication token (factor).
Type (only displayed on the **Users** page)	The kind of user, which is determined by the authentication and role-assignment mechanisms in use. Values can be `Password user`, `Delegated user`, `Kerberos user`, `LDAP user`, or `OS user`. For more information on user types, see About User Types below.

10.1.1 About User Types

The user account *Type* can be one of the following:

- Password user — This type is authenticated through Instance Authentication, Kerberos (without delegated authorization), or the operating system (without delegated authorization). The InterSystems IRIS tools for editing or otherwise altering users are for use with Password users.
- Delegated user — This type is authenticated through a user-defined authentication mechanism. The InterSystems IRIS tools are available only for viewing this type of user's properties; the user's properties must be edited through external means.
- Kerberos user — This type is authenticated using Kerberos when delegated authorization is in use; with delegated authorization, InterSystems IRIS tools are available only for viewing this type of user's properties; the user's properties

must be edited through external means and specified by the **ZAUTHORIZE** routine, as described in Delegated Authorization. If a user is authenticated through Kerberos without using delegated authorization, then the user is of the Password user type.

- LDAP user — This type is authenticated through LDAP. The InterSystems IRIS tools are available only for viewing this type of user's properties; the user's properties must be edited through external means.
- OS user — This type is authenticated through the operating system (OS) when delegated authorization is in use; with delegated authorization, InterSystems IRIS tools are available only for viewing this type of user's properties; the user's properties must be edited through external means and specified by the **ZAUTHORIZE** routine, as described in Delegated Authorization. If a user is authenticated through the operating system without using delegated authorization, then the user is of the Password user type.

Important: A user can only have one type. A user of one type cannot log in using authentication mechanisms associated with another type.

10.2 Manage User Accounts

To view a list of the existing user accounts, see the **Users** page in the Portal (**System Administration** > **Security** > **Users**). This page displays information on the following fields (as described in more detail in the User Account Properties section):

- **User** — A unique identifier for the user account
- **Full Name** — The account's displayable name
- **Enabled** — Whether or not the user account is currently enabled
- **Namespace** (default namespace) — The initial namespace for a terminal-type connection
- **Routine** (default routine) — The initial routine executed for a terminal-type connection
- **Type** — The kind of user account, which is determined by the authentication and role-assignment mechanisms in use

You can perform the following actions from the **Users** page:

- Create a New User Account
- Edit an Existing User Account
- View a User Profile
- Disable/Enable a User Account
- Delete a User Account

10.2.1 Create a New User Account

To create a new user account:

1. From the Management Portal home page, go to the **Users** page (**System Administration** > **Security** > **Users**).
2. On the **Users** page, select **Create New User**. This displays the **General** tab of the **Edit User** page for creating and configuring user accounts.
3. On the **Edit User** page, set values for the user properties described in the User Account Properties section.

> **Note:** As a shortcut, if you wish to create multiple accounts with similar characteristics, you can use the **Copy from** field to begin the process. Select an existing user account from the **Copy from** drop-down menu to fill in the following fields with values from the selected account:
>
> - Full Name
> - Expiration Date
> - Default Namespace
> - Default Tag^Routine

4. Click the **Save** button to create the new user account.

Once you have created a user account, you can then edit its characteristics.

10.2.2 Edit an Existing User Account

Once you have created a user account, you can alter any of its basic properties:

1. From the Management Portal home page, go to the **Users** page (**System Administration** > **Security** > **Users**).
2. On the **Users** page, there is a table of user accounts. To edit an existing account, select the name of the account from the table. This displays the **General** tab of the **Edit User** page for creating and configuring user accounts.
3. On the **Edit User** page, you can alter values for the properties described in the User Account Properties section.
4. Click the **Save** button to save the new values for the user account.

You can also modify other aspects of the user account on this page's other tabs:

- Roles — Lists the roles that the user account currently holds. You can also give a user account new roles (or take them away) on this page.
- SQL Properties — This includes:
  - **SQL Privileges** — Lists all the SQL privileges that a user account currently holds, on a per-namespace basis. You can also assign or revoke SQL privileges on this page.
  - **SQL Tables** — Lists, by namespace, the tables for which a user account has been granted privileges (%ALTER, DELETE, INSERT, REFERENCES, SELECT, and UPDATE). You can also assign or revoke SQL table privileges on this page.
  - **SQL Views** — Lists, by namespace, the views for which a user account has been granted privileges (%ALTER, DELETE, INSERT, REFERENCES, SELECT, and UPDATE). You can also assign or revoke SQL view privileges on this page.
  - **SQL Procedures** — Lists, by namespace, the stored procedures which a user account can run. You can also assign or revoke the right to run procedures on this page.

Note: A change to a user account only takes effect after the user logs out and then logs back in.

10.2.2.1 Modify a User's Roles

On the **Roles** tab of the **Edit User** page, you can assign a user account to a role or remove it from a role:

- To assign a user account to a role, first move the role from the **Available** list to the **Selected** list (either double-click it or select it and then click the single right-arrow); click the **Assign** button to assign the user account to the role.

- To assign a user account to all roles, click the double-arrow pointing from the **Available** list to the **Selected** list; click the **Assign** button to assign the user account to all the roles.

  Note: If you assign a user account to all roles, this includes the predefined %SecureBreak role, which limits (and does not expand) the user account's abilities. If a user account is assigned to the **`%SecureBreak`** role, this enables the InterSystems IRIS secure debug shell, which restricts the commands that the user may issue. This may also have unexpected consequences in other areas.

- To remove a user account from a role, click the **Remove** button to the right of role name.
- To remove a user account from all roles, click **Remove All** below the table listing the currently assigned roles. (This button is only present if a user account is assigned to two or more roles.)

10.2.2.2 Modifying a User's SQL-Related Options

For every user account, you can grant or remove the following SQL-related characteristics:

- General SQL Privileges
- Privileges for Tables
- Privileges on Views
- Privileges for Stored Procedures

General SQL Privileges

On the **SQL Privileges** tab of the **Edit User** page, you can add or remove SQL privilege for a user account:

- To add a privilege to a user account, first move the privilege from the **Available** list to the **Selected** list (either double-click it or select it and then click the single right-arrow); click the **Assign** button to give the privilege to the account. To also add the privilege of being able to grant the added privilege to other user accounts, click the relevant button below the **Available** list.
- To add all privileges to a user account, click the double-arrow pointing from the **Available** list to the **Selected** list; click the **Assign** button to give the privileges to the user account. To also add the privileges of being able to grant the added privileges to other user accounts, click the relevant button below the **Available** list.
- To remove a privilege from a user account, click the **Remove** link to the right of privilege name.
- To remove all privileges from a user account, click the **Remove All** button below the table listing the currently assigned privileges.

The following privileges are available:

- %ALTER _TABLE — For a given namespace, allow the user to run the ALTER TABLE command.
- %ALTER_VIEW — For a given namespace, allow the user to run the ALTER VIEW command.
- %CREATE_FUNCTION — For a given namespace, allow the user to run the CREATE FUNCTION command.
- %CREATE_METHOD — For a given namespace, allow the user to run the CREATE METHOD command.
- %CREATE_PROCEDURE — For a given namespace, allow the user to run the CREATE PROCEDURE command.
- %CREATE_QUERY — For a given namespace, allow the user to run the CREATE QUERY command.
- %CREATE_TABLE — For a given namespace, allow the user to run the CREATE TABLE command.
- %CREATE_TRIGGER — For a given namespace, allow the user to run the CREATE TRIGGER command.
- %CREATE_VIEW — For a given namespace, allow the user to run the CREATE VIEW command.

- %DROP_FUNCTION — For a given namespace, allow the user to run the DROP FUNCTION command.
- %DROP_METHOD — For a given namespace, allow the user to run the DROP METHOD command.
- %DROP_PROCEDURE — For a given namespace, allow the user to run the DROP PROCEDURE command.
- %DROP_QUERY — For a given namespace, allow the user to run the DROP QUERY command.
- %DROP_TABLE — For a given namespace, allow the user to run the DROP TABLE command.
- %DROP_TRIGGER — For a given namespace, allow the user to run the DROP TRIGGER command.
- %DROP_VIEW — For a given namespace, allow the user to run the DROP VIEW command.

Privileges for Tables

On the **SQL Tables** tab of the **Edit User** page, you can add or remove table-related SQL privileges for a user account:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace's tables will appear.
2. To change privileges for a table, select the **Edit** button in that table's row. This displays a window for altering privileges.
3. In this window, you can check or uncheck any of the following items:
   - ALTER
   - SELECT
   - INSERT
   - UPDATE
   - DELETE
   - REFERENCES
4. After making your selection(s), click the **Apply** button to establish the new privileges for the table.

Privileges on Views

On the **SQL Views** tab of the **Edit User** page, you can add or remove view-related SQL privileges for a user account.

To add privileges for the view:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace's views will appear.
2. To change privileges for a view, select the **Edit** button in that view's row. This displays a window for altering privileges.
3. In this window, you can check or uncheck any of the following items:
   - ALTER
   - SELECT
   - INSERT
   - UPDATE
   - DELETE
   - REFERENCES
4. After making your selection(s), click the **Apply** button to establish the new privileges for the table.

Privileges for Stored Procedures

On the **SQL Procedures** tab of the **Edit User** page, you can add or remove a user account’s SQL privileges related to stored procedures.

To add privileges for a stored procedure:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace’s stored procedures will appear.
2. Below this window, click the **Add** button, which displays the **Grant procedure privilege...** dialog.
3. In this dialog, near the top, select the Schema from the drop-down that contains the procedure that you wish to add. This displays a list of the schema’s procedures in the **Available** window on the left part of the page.
4. Move one or more procedures into the **Selected** window. Make sure the **EXECUTE** box is checked, so that the user account has the privilege to execute the stored procedure.
5. Optionally, you can grant the users the ability to grant this privilege on other user accounts; to do this, click the **Grant privilege** box near the bottom of the page.
6. Click the **Apply** button to grant the privilege(s) to the user account.

To remove a user account’s stored procedure privileges:

1. Choose the relevant namespace from the drop-down near the top of the page. A list of the namespace’s stored procedures will appear.
2. To change privileges for a stored procedure, select the **Edit** button in that table’s row. This displays a page for altering privileges.
3. On the page that appears, uncheck the **EXECUTE** check box and the **GRANT privilege** check box as appropriate.
4. Click the **Apply** button to change the privilege(s) for the user account.

10.2.3 View a User Profile

A user profile provides security information about a user account, such as the roles to which the user account is assigned and the time of the user’s last login. To view a user profile, the procedure is:

1. From the Management Portal home page, go to the **Users** page (**System Administration** > **Security** > **Users**).
2. On the **Users** page, in the row for the user, click **Profile**. This displays the user profile.

Alternately, if the **Edit User** page is visible for a user account, click **Profile** in the upper-left corner of the page.

The following properties are listed as part of the user profile.

Table 10–2: User Profile Properties

Property Name	Property Description
Name	Unique user identifier. This can include any characters except the @, which is used to identify a domain. This is editable on the **Edit User** page.
Full Name	The user account’s displayable name. This is editable on the **Edit User** page.
Roles	A comma-separated list of roles assigned to user account. These are editable on the **Roles** tab of the **Edit User** page.
Last Password Change	The date and time of user account’s most recent password change.

Property Name	**Property Description**
Last Login	The date and time of most recent successful login or 0 if there has not yet been a successful login. Read-only.
Last Login Device	The IP address of the host from which the user last logged in.
Invalid Login Attempts	The number of invalid login attempts since the most recent successful login. Read-only.
Last Invalid Login	The date and time of most recent invalid login attempt. Read-only.
Last Invalid Login Device	The IP address of the host from which the user last unsuccessfully attempted to log in.
Last Reason for Failing to Login	The error thrown for the most recent invalid login attempt. Read-only.
Time account was created	The date and time at which the user account was created. Read-only.
Username who created account	The account name associated with the user who created the account. Read-only.
Time account was last modified	The date and time at which the account was last modified. Read-only.
Username who last modified account	The account name associated with the user who last modified the account. Read-only.
Information last modified in account	A list of properties that were last modified for the account. Read-only.

10.2.4 Disable/Enable a User Account

You can disable or enable a user account. For example, you can disable an account to make it temporarily unavailable; when you enable it later, you then do not have to reconstruct its properties, roles, and so on.

To disable or enable a user account:

1. From the Management Portal home page, go to the **Users** page (**System Administration** > **Security** > **Users**).
2. On the **Users** page, for the user account you wish to disable or enable, click the name of the user account. This displays the**General** tab of the **Edit User** page for that user.
3. On the **Edit User** page, clear or select the **User enabled** field.
4. Click the **Save** button to save the user account in its new state.

10.2.5 Delete a User Account

To delete a user account:

1. From the Management Portal home page, go to the **Users** page (**System Administration** > **Security** > **Users**).
2. On the **Users** page, for the user account you wish to delete, select the **Delete** button in that user account’s row.
3. InterSystems IRIS displays a confirmation dialog. Select **OK** to delete the user account and**Cancel** otherwise.

10.3 Predefined User Accounts

Every instance of InterSystems IRIS automatically includes the following accounts:

Table 10–3: Predefined User Accounts

Username	Assigned Roles	Purpose
Admin	%Manager	Default administrator account. This account exists for all instances of all InterSystems products to support instance administration. InterSystems recommends that you change the password for this account from its initial value and disable the account prior to going into production.
CSPSystem	(None)	Default account representing the Web Gateway when it connects to InterSystems IRIS via Instance Authentication for Normal and Locked-down instances. InterSystems recommends that you change the password for this account from its initial value prior to going into production. This user account is used internally by HealthShare and should not be disabled. **Note:** If you change this user's password for the instance, you must also change it for the web gateway.
IAM	IAM_API	Default account required to obtain a license for InterSystems API Manager (IAM) from InterSystems IRIS. To use the IAM user, you must enable it and change its password; setting up the IAM user is part of setting up IAM generally.
SuperUser	%All	Default account with all privileges available. This account exists for all instances of all InterSystems products to provide complete access to all aspects of the product. InterSystems recommends that you change the password for this account from its initial value and disable the account prior to going into production.
UnknownUser	%All (Minimal security) or None (Normal or Locked-Down security)	Default account for a non-logged in user.
_PUBLIC	(None)	Set of privileges given to all users. This is not a login account, so it is always marked disabled, but it still gives assigned privileges to all users.
_SYSTEM	%All	Default SQL account. This account exists for all instances of all InterSystems products to provide SQL access. InterSystems recommends that you disable this account for production systems.
_Ensemble	%All	Interoperability manager (not a login account). Only on InterSystems IRIS instances. This account is used internally by HealthShare and should not be disabled.
HS_Services	%HS_ServiceRole	This account is used by healthcare products for internal purposes. By default, it is disabled for InterSystems IRIS for Health™ and HealthShare® Health Connect, and must be enabled before using FHIR®-based IHE profiles with these products. For other HealthShare solutions, the account is enabled by default and must not be disabled.

There is also an account called a “privileged user account,” which is created during Normal and Locked Down installations and for which you supply a username and password.

It is not possible to delete the following accounts:

- _Ensemble
- _PUBLIC
- _SYSTEM
- UnknownUser

CAUTION: New installations of InterSystems IRIS use the same password for all the predefined accounts, as described in Initial User Account Passwords. The default password is a security vulnerability, particularly in a Minimal Security installation. To address this issue, disable the accounts or change their passwords. InterSystems recommends disabling the accounts after creating a unique account with the **`%All`** role, as InterSystems IRIS requires at least one enabled account with the **`%All`** role.

This is a critical concern with containerized instances in particular; see Authentication and passwords for more information, including ways in which you can address the issue.

Additionally, there is a user account called %System, which is not visible and which exists for InterSystems IRIS to use internally. You cannot log in to, edit, or delete this account. This account runs with the **`%All`** role. Certain routines, such as **%ZSTART** and **%ZSTOP**, are run as this user account; to run such a routine as a different user, call **$SYSTEM.Security.Login()**.

While it is not recommended, you can delete predefined user accounts. However, there must be at least one account with the **`%All`** role in addition to %System.

10.3.1 Notes on Various Accounts

10.3.1.1 The UnknownUser Account

For certain applications, or certain parts of an application, unauthenticated users may have a legitimate reason to use InterSystems IRIS, such as for a retail system to display availability of products, prior to the user initiating a purchase. For this type of situation, InterSystems IRIS supports the UnknownUser account. When an unauthenticated user connects, a special name, UnknownUser, is assigned to $USERNAME and the roles defined for that user are assigned to $ROLES.

Unauthenticated access is *not* used when authentication fails. For example, suppose that a user attempts to connect to InterSystems IRIS via terminal and supplies a username and password that fails authentication. Here, the user is not connected to InterSystems IRIS, even if unauthenticated access is permitted; on the other hand, if unauthenticated access is permitted and that same user connects to InterSystems IRIS without supplying a username (such as by pressing the Enter key at the username prompt), then that user is connected as UnknownUser, the unauthenticated user. Similarly, if an ODBC client attempts to connect with null strings for username and password, that connection will be accepted if unauthenticated access is permitted for the service; if the same ODBC client provides a non-empty username and password values that fail authentication, that client is not connected even if unauthenticated access is permitted.

10.3.1.2 The _PUBLIC Account

The predefined user account, _PUBLIC, is a special account that does not exist for logins. Rather, it holds a set of roles. These roles are specified as the default roles for any user who connects to the system. This ensures a minimum set of roles for any user. For example, if you associate the %Operator role with the _PUBLIC user, then the value of *$Roles* for any user will always include %Operator. These roles are assigned even when _PUBLIC is marked disabled.

10.4 Validate User Accounts

If you need to validate user accounts in application code, you can do this by creating a simple routine that attempts to log the user in with the one-argument form of the **$SYSTEM.Security.Login** method. If the login succeeds, the user account is valid; if the login fails, the user account is not valid. When the routine exits (regardless of the login's success or failure), the current user account will be the one that invoked the routine.

Here is a sample routine to perform this task, called **ValidateUser**:

ObjectScript

```
ValidateUser(TestUser) {
    Write "Validating ",TestUser,"...",!
    New $Roles
    Set sc = $SYSTEM.Security.Login(TestUser)
    If sc = 1 {
        Write $Username," is a valid user.",!
        Write $Username," belongs to the following login roles: ",$Roles,!
    } Else {
        Write TestUser," is not a valid user.",!
    }
    Quit sc
}
```

This routine takes as its single argument, a string that is the name of the user account to be validated. It then performs the following actions:

1. The call of `New $Roles` stacks both the *$Roles* variable and the *$Username* variable. For more information on *$Roles*, see the $Roles reference page.
2. It then invokes the one-argument form of the **$SYSTEM.Security.Login** method, which attempts to log the user in and does not require the user's password. If the login succeeds, the method returns 1; this determines the information that the routine displays and the routine's return value.
3. When the routine exits, this implicitly logs out the user, if there has been a successful login.

Important: This routine uses the one-argument form of the **$SYSTEM.Security.Login** method. To successfully invoke the one-argument form of **$SYSTEM.Security.Login**, a user account must have the **`IRISSYS:Write`** and **`%Service_Login:Use`** privileges. For more information on **$SYSTEM.Security.Login**, see the reference page for the %SYSTEM.Security class.

Here is a sample routine that demonstrates invoking **ValidateUser**, called **VUTest**. It is hard-coded to test two users, one called ValidUser and one called NonexistentUser:

ObjectScript

```
VUTest() {
    Write $Username," is the current user.",!,!

    Set sc = $$^ValidateUser("ValidUser")
    Write !

    Write "Exited validation code. ",$Username," is the current user.",!,!

    Set sc = $$^ValidateUser("NonexistentUser")
    Write !

    Write "Testing complete.",!
    Write $Username," is the current user."
    Quit 1
}
```

Suppose the **VUTest** routine were created in the User namespace of an InterSystems IRIS instance, the PrivilegedUser account were a member of the **`%All`** role, and only the ValidUser were to exist. Here are the results of invoking **VUTest** at a Terminal prompt:

```
Username: PrivilegedUser
Password: ***********
USER>d ^VUTest
PrivilegedUser is the current user.

Validating ValidUser...
ValidUser is a valid user.
ValidUser belongs to the following login roles: %Manager

Exited validation code. PrivilegedUser is the current user.

Validating NonexistentUser...
NonexistentUser is not a valid user.

Testing complete.
PrivilegedUser is the current user.
USER>
```

11

Applications

Applications are the primary way that most users interact with InterSystems IRIS® data platform — and application security provides a key set of tools for regulating user access and user actions. Application security uses InterSystems authorization tools to ensure that only the appropriate users can use an application. An application can also escalate its users' privileges.

This topic covers the following topics:

- Applications, Their Properties, and Their Privileges
- Application Types
- Create and Edit Applications
- Built-In Applications

11.1 Applications, Their Properties, and Their Privileges

From a security standpoint, an application:

- Is an entity that allows users to perform actions
- Is associated with one or more resources
- Has properties that govern its behavior
- Can enhance its users' privileges while they are running it
- Can include programmatic privilege checks

All these characteristics and capabilities are part of the InterSystems authorization tools, which manage how an application and its users can interact with InterSystems products and other security resources. There are several kinds of applications:

- Web Applications
- Privileged Routine Applications
- Client Applications
- Document Database Applications

This section covers:

- Applications and Their Properties
- Associating Applications with Resources

- Applications and Privilege Escalation
- Check for Privileges Programmatically

11.1.1 Applications and Their Properties

Applications allow you to specify a set of permitted actions, such as reading from and writing to databases or using other assets. To do this, InterSystems IRIS supports what is called an *application definition*, which is a set of information that represents the application within InterSystems IRIS. (The relationship of an application definition to an application is analogous to that of the relationship of a resource to an asset.) By establishing an application definition, you can control and manage the application.

Important: Applications and application definitions are frequently referred to interchangeably. The distinction is only important in settings where the executable code or user experience of that code differs from the representation of that code within InterSystems IRIS. The former is the application itself and the latter is the application definition.

Each application, through its application definition, has the following properties:

Name

The name of the application. This must start with a forward slash (“/”) and must be followed by an alphanumeric or any of the following characters:

/ , - , _ , . , or %.

The application definition’s name is independent of the name of any resource.

Description

A description of the application.

Enabled

A switch that specifies if the application is available for use. If the application is not enabled, no one can run the application — not even a member of the **`%All`** role. For more details on how this property governs each kind of application, see the appropriate section: Web Applications, Privileged Routine Applications, or Client Applications.

Resource

A resource that manages application behavior. The resource has different effects for different application types: for web applications and client applications, it controls whether or not users have access to the application; for privileged routine applications, it controls the application’s privilege escalation. If a web or client application has no resource, then any user can run the application; if a privileged routine application has no resource, then the application escalates privileges for any user.

Each application definition can only be associated with a single resource. For more details on how this property affects each kind of application, see the appropriate section: Web Applications, Privileged Routine Applications, or Client Applications. For more information on how applications interact with resources, see Associating Applications with Resources.

Application Roles

One or more roles to which application users are assigned. While running the application, the user is assigned to its application roles by appending these roles to the list of roles in the *$Roles* variable. For more information on using application roles, see Applications and Privilege Escalation.

Matching Roles

> One or more roles that cause the user be assigned to some additional roles (called "target roles") while running the application. If users are assigned to a matching role, then, while using the application, they are also assigned to any target roles. This is done by appending these roles to the list of roles in the *$Roles* variable. For example, if **`%Admin_Manage`** is a matching role, then being a member of that role might cause the application user to also become a member of the target role of **`%Admin_Secure`**. For more information on using matching roles, see Applications and Privilege Escalation.

All applications have these properties. Each of the application types also has its own other, unique characteristics.

11.1.2 Associating Applications with Resources

When an application (and therefore its application definition) is a single, unitary whole, it has a single resource — a one-to-one relationship. You can also specify that multiple applications (and therefore multiple application definitions) are associated with a single resource — a many-to-one relationship. For any number of applications, the situation reduces to some combination of these two conditions.

A more complex case is when an application is composed of separate parts, each of which is known as a *sub-application.* An application made up of a group of sub-applications is designed to behave as a single, unitary whole, while allowing different sub-applications to require different kinds of or levels of security. In this situation, it is useful to give each sub-application its own application definition and to associate it with a separate resource. This way, each sub-application can have its own independent security-related behavior. While, from the application perspective, there are multiple sub-applications that compose the larger application, from the InterSystems security perspective, there are simply multiple, individual applications — each with its own application definition — and users pass among them without knowing it. Again, this reduces to the one-to-one and many-to-one cases, except that the multiple application definitions represent what appears to the end-user as a single application. Because the users have already authenticated and by that process have established their roles, then passing from one sub-application to another requires no authentication.

For example, suppose there is an expense-reporting application where all employees can enter expense reports, but only accounting officers can generate checks. In this case, the application might appear as a single whole and the functionality to generate checks would be grayed out for all employees except the accounting officers. To accomplish this, there would be two separate sub-applications, one for entering reports and another for generating checks. All users would be able to use the former and only accounting officers would be able to use the latter. For them, there would be no visible difference between what might simply appear as two separate screens in a single application.

11.1.3 Applications and Privilege Escalation

Since you can use application resources to escalate a user's roles, they provide a mechanism for meeting authorization needs that shift dynamically. To perform privilege escalation for an application:

1. Given an existing application, create a resource and then associate the application with the resource.
2. Create one or more roles that hold the Use permission for the resource.
3. Determine the list of privileges that the application requires in order to run. If the application has sub-applications, there may be more than one such list.
4. Associate each list of privileges with a particular role. Establish each role as an *application role* for the application or sub-application.
5. Establish any *matching roles* for the application or sub-application. Each matching role has one or more *target roles* associated with it.
6. When a user successfully invokes an application, InterSystems IRIS performs two actions:
    - For the duration of application use, it assigns the user to any application roles. (For privileged routine applications, this depends on successfully invoking the **AddRoles** method, as described in Privileged Routine Applications.)

- If the user is assigned to any matching role, the application assigns the user to any target roles for the duration of application use. (Again, for privileged routine applications, this depends on successfully invoking the **AddRoles** method, as described in Privileged Routine Applications.)

For example, suppose that an application has its own resource, called **`AppRsrc`**. Two roles hold the **`AppRsrc:Use`** privilege; these are **`AppUser`** and **`AppOperator`**. **`AppOperator`** is also a matching role, where the target role is **`%Manager`**. In this scenario, when a user belonging to the **`AppUser`** role invokes the application, the value of *$Roles* does not change; when a user belonging to **`AppOperator`** invokes the application, the value of *$Roles* expands to include **`%Manager`**. If the application has an application role of **`AppExtra`**, then a user belonging to the **`AppUser`** role receives the **`AppExtra`** role when invoking the application. In the first scenario (matching role only), belonging to the **`AppOperator`** role causes privilege escalation; in the second scenario (matching role and application role), belonging to either role results in privilege escalation.

11.1.3.1 User-Based and Application-Based Security

The InterSystems security model allows for flexible privilege assignment that can be user-based, application-based, or both. The use of an application can be limited to specific users or open to any users. For those users authorized to use the application, there can be several behaviors:

- The application can run with the user’s privileges alone.
- The application can escalate privileges for only some users (using matching and target roles).
- The application can escalate privileges for all users (using application roles).
- The application can escalate some privileges for all users and only escalate other privileges for certain users (using a combination of matching/target roles and application roles).

Hence, you have control of whether application use is limited to specific users or open to any users; simultaneously, you also have control of whether an application runs with the user’s privileges or with its own privileges. This enables InterSystems IRIS to provide a very flexible model:

Table 11–1: Protection/Escalation Matrix for Secured Applications

Privilege Level / Protection Level	Public Application	Restricted Application
With User-Dependent Privileges	1. Any user can run the application. Application runs with user privileges.	2. Only specified users can run the application. Application runs with user privileges.
With Privilege Escalation	3. Any user can run the application. Application runs with (expanded) application privileges through application roles and matching roles.	4. Only specified users can run the application. Application runs with (expanded) application privileges through application roles and matching roles.

Each of the scenarios described in the previous table is commonly used for a different authorization model:

1. Public Application with User-Dependent Privileges

This describes an application available to any authenticated user; when run, the application grants no additional privileges. For example, for a company’s contact database, any user belonging to the company-wide role can get the office phone number and email address for any employee; managers hold greater privileges, which entitle them to view employee home phone numbers; HR staff hold even greater privileges, which entitle them to view and update full records. The application is accessible to all employees, and its behavior depends on privileges that each user already has when invoking it — the application itself grants no roles.

2. Restricted Application with User-Dependent Privileges

This describes an application available only to a user who belongs to a specified role; when run, the application grants no additional privileges. For example, a company may have a payroll application for its hourly employees, which displays the number of hours worked, pay rate, and so on. To run the application, a user has to be a member of either the **`HourlyEmployee`** role or the **`HourlyManager`** role. Once running, the application checks which role was present: members of **`HourlyEmployee`** can see and *not* edit their own data, while members of **`HourlyManager`** can see and edit data for their own reports. An employee who is a member of the **`HourlyEmployee`** role can run the application to check the accuracy of personal data; any other employee (such as one on a salary and who is not a member of the required role) cannot even run the application.

3. Public Application with Privilege Escalation

This describes an application available to any authenticated user; when run, the application escalates privileges based on the roles to which the user belongs. (The application can also escalate privileges only for certain roles.) For example, suppose a university has an application where students can review and update their records. Here, any student is an authenticated user and can edit his or her own contact information. To support this functionality, the application includes code for editing an entry; this code checks that the entry being edited matches the authenticated user and, if so, escalates its own privileges to update the record, and then restores the privileges to their previous state. If one student attempts to update another's record, then the check fails, there is no privilege escalation, and the update does not occur. The application might also check if the user is a member of the registrar's office role, in which case it would be possible to update information more widely.

4. Restricted Application with Privilege Escalation

This describes an application available only to a user who belongs to a specified role; when run, the application escalates privileges based on the roles to which the user belongs. (The application can also escalate privileges only for certain roles.) For example, a hospital's emergency room might have an application that grants the attending doctor special, wider privileges for viewing the records of patients currently admitted for emergency care. Because of the potentially critical nature of emergency-room cases, the doctor needs to be able to view more information in this setting than while simply making rounds; hence, the privileges are escalated.

11.1.4 Check for Privileges Programmatically

An application can also include code to check if its users have privileges required to perform a particular action. To do this, use the **$SYSTEM.Security.Check** method. The syntax of this call is:

ObjectScript

```
Set status = $SYSTEM.Security.Check(app_resource, app_permission)
```

where

- *app_resource* is the resource for which the user must hold a permission
- *app_permission* is the permission that must be held.
- *status* is the method's return value of TRUE or FALSE (1 or 0).

For example, if an application requires a user to have Write permission on the **`Application_Order_Customer`** resource, then the **Check** call would be:

ObjectScript

```
Set status = $SYSTEM.Security.Check("Application_Order_Customer", "WRITE")
```

Note: No privilege is required to call **$SYSTEM.Security.Check**.

11.2 Application Types

There are several types of applications:

- Web Applications
- Privileged Routine Applications
- Client Applications
- Document Database Applications

11.2.1 Web Applications

These applications connect to InterSystems IRIS using the **`%Service_WebGateway`** service.

For web applications, security information is maintained as part of the Web session. That is, the values of *$USERNAME* and *$ROLES* are preserved across page requests. (More specifically, when processing begins for a page, *$ROLES* contains the user’s roles as well as roles defined for the application. It does not contain roles that have been dynamically added during processing of a previous page via `SET $ROLES` or **$SYSTEM.Security.AddRoles**. This is true for both stateless and “state-full” sessions.

With Web applications, the client (that is, the browser) typically does not send a username and password to the server when it connects. Instead, the user requests a page and the server responds with a login page that must be completed before the rest of the application can be accessed. If two-factor authentication is enabled, then, once the user has provided a username and password, the server displays a page for entering the security code; if authentication succeeds, the user has access to the application.

Note: With two-factor authentication, the server always displays the page for entering the one-time security token — even if the username-password pair is not valid. After the user enters the one-time security token, the server displays a message that access is denied, and provides a minimum of information that could be used against the system.

CSP security processing occurs as follows:

1. As each page request is received, its application is determined from the URL. If the application is not enabled, there is no connection.
2. If the application is the same as the application for the last page processed for the web session, then there is already a connection, so no further security checking is required.
3. If the Use permission for **`%Service_WebGateway`** is not public and the user does not hold this permission, there is no connection.
4. If the application or **`%Service_WebGateway`** requires authentication and the user has not already been authenticated, then InterSystems IRIS checks if the request includes *IRISUsername* and *IRISPassword* parameters:
    a. If *IRISUsername* and *IRISPassword* are present, InterSystems IRIS attempts to log in; if the login succeeds, it checks if the user has the Use permission for the application resource. If either of these fail, there is no connection.
    b. If *IRISUsername* and *IRISPassword* are not present, InterSystems IRIS displays an application-specific login page, if one is defined in the web application configuration. (This is the only page in a secure application that can be used prior to login.) If there is no application-specific login page, the username and password fail authentication, or the user does not have the Use permission on the application resource, there is no connection.

For information about editing a web application, see:

- Create an Application

- Edit a Web Application: The General Tab
- Edit an Application: The Application Roles Tab
- Edit an Application: The Matching Roles Tab

Also see Secure Custom Web Application Logins.

11.2.1.1 An Example of Programmatically Escalating Roles in a Web Application

The following example demonstrates how to escalate application roles for a web application. It performs the following actions:

1. It changes control to the %SYS namespace. This is required to invoke the required calls, because the code for role management and escalation is only available there.
2. It adds the application's target roles.
   a. All users receive the **`MYAPP`** role. This is because there is no matching role listed before the colon in the line that initially sets the value of *matchroles*. The syntax is **`matchrole:targetrole`**, where an empty value for **`matchrole`** means that all users receive **`targetrole`**.
   b. It adds the **`MYAPP2`** roles for users who already have the **`MYAPPSPECIAL`** role. The syntax here is **`matchrole1:targetrole1,matchrole2:targetrole2`**. Hence, if a user has the **`MYAPPSPECIAL`** role, then the applications adds the **`MYAPP2`** role. (The leading comma follows the syntax for second and subsequent match and target roles; see the **Security.Applications.Create** method for more details.)
3. It uses the local variable that holds role escalation information to update the application's MatchRoles property. The **Security.Applications.Modify** method only updates the properties for which it has values; it leaves the others unchanged.
4. Finally, if role escalation succeeds, announce this.

The code is:

Class Member

```
Method UpdateRoles() As %Status {
 // ********************* modify application roles *********************
 write !, "All users receive the added MYAPP role."
 write !, "Users who have MYAPP2 receive MYAPPSPECIAL also."

 // Change to the %SYS namespace.
 new $NAMESPACE
 set $NAMESPACE="%SYS"

 // Add roles for the application.
 //
 // Add the MYAPP role for all users.
 set matchroles=":MYAPP"
 // Also add MYAPP2 for users who already have the MYAPPSPECIAL role.
 set matchroles=matchroles_",MYAPPSPECIAL:MYAPP2"

 // Use the matchroles variable to
 // set the applications's MatchRoles property.
 set MyAppProps("MatchRoles")=matchroles
 set status=##class(Security.Applications).Modify("/csp/MyApp",.MyAppProps)

  // Announce success.
  if $$$ISOK(status) {
    write !, "Roles were successfully modified."
 }
}
```

11.2.2 Privileged Routine Applications

A *privileged routine application* grants the privilege to escalate roles to one or more classes or routines for the users of those classes or routines. The classes or routines in a privileged routine application are written in ObjectScript. To use a privileged routine application:

1. Create an application definition in the Management Portal, as described in Create an Application.
2. Add classes or routines to it, as described in Edit an Application: The Routines/Classes Tab.
3. Edit the application definition's classes or routines in your development environment to escalate roles, as described in Escalate Roles in a Privileged Routine Application: The AddRoles Method.

The Portal provides the following pages to edit a privileged routine application (which includes the first two mentioned above):

- Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab
- Edit an Application: The Application Roles Tab
- Edit an Application: The Matching Roles Tab
- Edit an Application: The Routines/Classes Tab

11.2.2.1 Escalate Roles in a Privileged Routine Application: The AddRoles Method

To escalate roles in a privileged routine application, invoke the **AddRoles** method of the %SYSTEM.Security class. To call **AddRoles**, the syntax is:

ObjectScript

```
Set sc = $SYSTEM.Security.AddRoles("AppDefName")
```

where *AppDefName* is the name of the application definition and *sc* is a status code. If a class or routine is part of an application definition and the user is appropriately privileged, then calling **AddRoles** from that class or routine escalates privileges to include any application roles (as described in "Edit an Application: The Application Roles Tab") and any relevant matching roles (as described in "Edit an Application: The Matching Roles Tab").

Important: If a routine does not use curly braces to delimit code in its entry points, then control can pass from one entry point to another, possibly resulting in overprivileged users and unintended levels of access. For more information on structuring routines, see User-Defined Code.

Processing of the call to **AddRoles** occurs as follows:

1. If the call is not from a privileged class or routine, then the call fails.
2. If the required resource specified in the application definition is not public and the user invoking the method or routine does not have Use permission on this resource, then the call fails.
3. Otherwise, the call succeeds.

Tip: To cause the user to give up any application roles and to revert to login roles when control passes out of scope for the routine that escalates privileges, include the following command *prior* to the call to **AddRoles**:

ObjectScript

```
New $Roles
```

For more information on these topics, see Programmatically Manage Roles.

11.2.2.2 An Example of Using a Privileged Routine Application

Suppose there is an application that uses a database called DB1. This application's users hold the **`%DB_DB1`** role only, so they all have privileges for DB1. Some of the application's users also require temporary access to another database, DB2. Those users get access to DB2 through the **PRAEscalate** method ("PRA" for "Privileged Routine Application") of the PRATestClass class, which escalates their privileges; specifically, **PRAEscalate** adds the **`%DB_DB2`** role, which provides access to DB2.

To enable the **PRAEscalate** method to add the **`%DB_DB2`** role for the appropriate users, the following security items must exist:

- A resource called **`PRATestResource`**, which is not public.
- A role called **`PRA_DB2`**, which has only one privilege: **`PRATestResource:Use`**.
- The **`%DB_DB2`** role, which was created when the DB2 database was created.
- A privileged routine application called PRATestApp. Related to PRATestApp:
  - Users must have the **`PRATestResource:Use`** privilege to run the PRATestApp application, Therefore, users who require access to the DB2 database must have the **`PRA_DB2`** role (which grants the **`PRATestResource:Use`** privilege).
  - The PRATestClass class is part of the PRATestApp application. (To include the class in the application, do so on the **Routines/Classes** tab of the **Edit** page for PRATestApp.)
  - The **`%DB_DB2`** role is an application role for PRATestApp. (To specify an application role, do so on the **Application Roles** tab of the **Edit** page for PRATestApp.)

Given this setup and two users, PRATestBasicUser and PRATestDB2User:

- PRATestBasicUser is a member of **`%DB_DB1`** only. Therefore, the PRATestApp application does not escalate PRATestBasicUser's roles, and the user *cannot* use the part of the application that requires access to DB2.
- PRATestDB2User is a member of the **`%DB_DB1`** and **`PRA_DB2`** roles. Therefore, the PRATestApp application does escalate PRATestBasicUser's roles, and the user *can* use the part of the application that requires access to DB2.

Here is the code of **PRAEscalate**:

Class Member

```
Method PRAEscalate()
 {
   Write "This method is a part of the privileged routine application ",!
   Write "called PRATestApp.",!
   Write "The user invoking this routine is ",$Username,!
   Write "The current value of $Roles is ",$Roles,!,!
   Write "Calling the AddRoles method...",!,!
   New $Roles
   Set sc = $SYSTEM.Security.AddRoles("PRATestApp")
   If sc = 1
   {
      Write "Application roles have been added.",!
      Write "$Roles now is ",$Roles,!,!
   } Else {
      Write "The call to AddRoles has failed.",!
      Do $system.Status.DecomposeStatus(sc,.Err)
      Write Err(Err),!
   }
 }
```

Here is the terminal session where PRATestDB2User runs this routine:

```
Username: PRATestDB2User
Password: ********
USER>set x = ##class(PRATestClass).PRATest()
This method is a part of the privileged routine application
called PRATestApp.
The user invoking this routine is PRATestDB2User
The current value of $Roles is %DB_DB1, PRA_DB2

Calling the AddRoles method...

Application roles have been added.
The current value of $Roles is %DB_DB1, %DB_DB2, PRA_DB2
Removing %DB_DB2 from $Roles...
$Roles now is %DB_DB1, PRA_DB2

USER>
```

Here is the terminal session where PRATestBasicUser runs this routine:

```
Username: PRATestBasicUser
Password: ********
USER>set x = ##class(PRATestClass).PRATestMethod()
This method is a part of the privileged routine application
called PRATestApp.
The user invoking this routine is PRATestUser
The current value of $Roles is %DB_DB1

Calling the AddRoles method...

The call to AddRoles has failed.
ERROR #862: User is restricted from running privileged application PRATestApp
-- cannot execute.

USER>
```

11.2.3 Client Applications

These are applications that use the client application type to connect to InterSystems IRIS.

Important: Regarding client applications:

- They are only supported on Windows. Therefore, options in the Management Portal for these applications are only available on Windows.
- For setting up their authentication, use the tools described in Authentication.

To edit a client application, the Portal provides the following pages:

- Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab
- Edit an Application: The Application Roles Tab
- Edit an Application: The Matching Roles Tab

11.2.4 Document Database Applications

These are applications that connect to InterSystems IRIS using the document database.

Important: For applications, use the authentication tools described in Authentication.

To edit a document database application, the Portal provides the following pages:

- Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab
- Edit an Application: The Application Roles Tab

- Edit an Application: The Matching Roles Tab

11.3 Create and Edit Applications

This section describes several topics:

- Create an Application
- Edit a Web Application: The General Tab
- Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab
- Edit an Application: The Application Roles Tab
- Edit an Application: The Matching Roles Tab
- Edit an Application: The Routines/Classes Tab
- Set the Web Application for an Interoperability-Enabled Namespace

11.3.1 Create an Application

To create an application, the procedure is:

1. In the Management Portal menu, select **System Administration** > **Security** > **Applications**, which displays the different application types.
2. Choose **Web Applications**, **Privileged Routine Applications**, **Client Applications**, or **Doc DB Applications**. This displays the page for the selected application type.
3. In the upper-left corner of the applications page, click the button to create a new application. This displays the application editing page for the selected application type. You can then edit the application as if it already existed using the information in either:
    - Edit a Web Application: The General Tab
    - Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab

11.3.2 Edit a Web Application: The General Tab

To edit a web application:

1. In the Management Portal menu, select **System Administration** > **Security** > **Applications** > **Web Applications**.

   This lists configured web applications. The **Type** column identifies an application as a user application (`CSP`) or a system application (`CSP,System`).
2. Select an application, click **Edit**, and enter or change the information.
3. When finished with edits, restart InterSystems IRIS for the new settings to take effect.

11.3.2.1 General Settings

The initial section of the **General** tab displays various options.

Note: The fields described here are only those that are relevant for InterSystems IRIS REST-based web applications. The other type of web application, called a CSP/ZEN application, is a legacy InterSystems application type. Because new applications based on InterSystems IRIS should be REST applications, this section does not describe fields exclusively related to CSP/ZEN applications. (If you have migrated a CSP/ZEN application to InterSystems IRIS from another InterSystems product, documentation is available for the relevant fields.)

Name

An identifier for the application. The name must include a leading slash (`/`), such as in the /myorg/myapp application.

Note that the name `/csp/docbook` is reserved.

Description

A text description of the application.

Namespace

The namespace where this application runs. When you select a different namespace, the dialog immediately displays the default application for that namespace to the right of this drop-down menu.

Namespace Default Application

Whether or not the application is the default application for this namespace. The %System.CSP.GetDefaultApp method returns the default application for the namespace. InterSystems IRIS import functions, such as $system.OBJ.Load or $system.OBJ.ImportDir, use the default application when importing a page without an associated application.

Enable Application

Whether or not the application is available for use. When enabled, an application is available, subject to user authentication and authorization; when disabled, it is not available.

Enable REST or CSP/ZEN

Whether this is a REST application or a CSP/ZEN application, which is a type of InterSystems legacy application. For new applications, InterSystems recommends the use of **REST**, which supports modern third-party front-end technologies.

If you have an existing application that uses CSP, InterSystems supports ongoing development using that technology and provides documentation for the CSP settings in Editing a CSP Application: The General Tab.

Dispatch Class

The corresponding custom subclass of %CSP.REST for implementing a REST service. For more information, see Creating a REST Service Manually.

Redirect Empty Path

Whether the application directs empty paths to `/`. For example, for application `/csp/appname`, a request for `/csp/appname` will be redirected to `/csp/appname/`.

Use JWT Authentication

Whether the application supports JSON web token (JWT) authentication.

JWT Access Token Timeout

The number of seconds until the JWT expires.

JWT Refresh Token Timeout

The number of seconds until the refresh token for the JWT expires.

11.3.2.2 Security Settings

The security settings are:

Resource Required

A resource for which users must have the Use permission so they can run the application. For information on resources and permissions, see About Resources.

Group by ID

Do not use. This field is for migrated legacy applications only and documentation is available for it.

Allowed Authentication Methods

The application's supported authentication mechanisms. The options available here depend on what is selected on the **Authentication Options** page (**System Administration** > **Security** > **System Security** > **Authentication/Web Session Options**). If an application supports multiple authentication mechanisms, authentication occurs as follows:

- If more than one option is enabled *including* **Unauthenticated**, then the user can log in without providing a username and password.
- If multiple options are enabled *and* the user enters a username and password, then InterSystems IRIS attempts cascading authentication.
- If the selected options are Kerberos authentication and instance authentication (password), but **Unauthenticated** is *not* selected, then the user must provide a username and password. InterSystems IRIS attempts to perform authentication first using Kerberos and then instance authentication. If either succeeds, the user is authenticated; if both fail, the user is denied access.

For more information, see Authentication.

Permitted Classes

Do not use. This field is for migrated legacy applications only and documentation is available for it.

11.3.2.3 Session Settings

This settings in this section allow you to manage the session properties for a web application.

Important: To use these settings, you must first set the UseSession parameter of the application's dispatch class to a non-zero value; otherwise, changes in the value of any of these settings have no effect. For more information, see Creating a REST Service Manually.

The session settings are:

Session Timeout

The default session timeout in seconds. You can override this value using the AppTimeout property of the %CSP.Session object.

Note that if a session changes web applications during its life span, the new application does not uses its default timeout to update the session's timeout value. For example, if a session starts out in Web Application A, with a default timeout of 900 seconds, and then moves into Web Application B, which has a default timeout of 1800 seconds, the session still times out after 900 seconds.

To cause an application change to update the session timeout value, then, in a session event class, override the **OnApplicationChange** callback method to add code to update the AppTimeout property of the *%session* object.

If you disable automatic logouts for **Interoperability** pages in the InterSystems Management Portal, the session timeout does not apply to those pages. That is, the pages will not time out. Disabling automatic logouts is not recommended. For more information, see Automatic Logout Behavior in the Management Portal.

Event Class

The default name of the class (a subclass of %CSP.SessionEvents) whose methods are invoked for web application events, such as a timeout or session termination. To override this value, specify the value of the EventClass property of the %CSP.Session object, using a class name without an extension (such as .cls) as the value.

Use Cookie for Session

Whether or not the application tracks the browser session by using cookies or a URL-rewriting technique that places a value in each URL. Choices are:

- **Always** — *Default*. Always use cookies.
- **Never** — Never use cookies.
- **Autodetect** — Use cookies unless the client browser has disabled them. If the user has disabled cookies, the application uses URL rewriting.

Note that this option does not set *whether* an application uses cookies; rather, it controls *how the application manages sessions*, subject to the user's preferences. Further, even with values of **Always** or **Autodetect**, an application only uses cookies if its code is written to do so.

Session Cookie Path

The portion of the URL that the browser uses to send the session cookie back to InterSystems IRIS for this application. If you do not specify a value for this field, the application uses the value of the **Name** field with leading and following slashes as its default scope. Hence, for an application named `myapp`, specifying no value here means that /myapp/ is the scope.

The application only sends the cookie for pages within the specified scope. If you restrict the scope to pages required by a single web application, this prevents other web applications on this machine from using this session cookie; it also prevents any other web application on this web server from seeing the cookie.

Note that a primary application and its subapplications can have different security settings while simultaneously sharing a session cookie (if they all use the primary application's path).

Session Cookie Scope

Controls the default value of the *SameSite* attribute for session cookies associated with the web application. For more details, see "About the SameSite Attribute," below.

User Cookie Scope

Controls the default value of the *SameSite* attribute for user-defined cookies created with %CSP.Response.SetCookie. For more details, see "About the SameSite Attribute," below.

About the SameSite Attribute

The SameSite attribute determines how an application handles cookies in relation to third-party applications (aka *cross-site requests*).

The **Session Cookie Scope** and **User Cookie Scope** fields allow you to set SameSite for an application's cookies. **Session Cookie Scope** sets the value of the attribute for session, login, CSRF, and Group ID cookies; **User Cookie Scope** sets it for user-defined cookies.

SameSite can have a value of:

- **None** — The application sends cookies with cross-site requests. If *SameSite* has a value of **None**, browsers may require applications to use HTTPS connections.
- **Lax** — The application sends cookies with safe, top-level cross-site navigation.
- **Strict** — The application does not send cookies with cross-site requests. (The default for system web applications and new or upgraded user applications.)

To exercise more granular control over an application's cookies, use the %CSP.Response.SetCookie method, which overrides the default *SameSite* value for a particular cookie. If you use %CSP.Response.SetCookie to specify that a cookie has a SameSite value of **None**, then you must use an HTTPS connection.

The SameSite attribute is part of an initiative from the IETF and is addressed in several of their documents.

11.3.3 Edit a Privileged Routine Application, a Client Application, or Document Database Application: The General Tab

The procedure is:

1. In the Management Portal menu, select **System Administration** > **Security** > **Applications**, which displays the different application types.
2. Choose **Web Applications**, **Privileged Routine Applications**, **Client Applications**, or **Doc DB Applications**. This displays the page for the selected application type.
3. On the applications page, select the application to edit by clicking on its name. This displays the **Edit** page for the application.
4. By default, the **General** tab appears. For privileged routine applications and client applications, the page's fields are:

The name field (varies by application type)

An identifier for the application.

Description

A text description of the application.

Enabled

Whether or not the application is available. When enabled, an application is available, subject to user authentication and authorization; when disabled, it is not available.

Resource required to run the application

A resource for which users must have the Use permission (enabled as part of a privilege in a role) in order to perform certain actions. For web and client applications, this resource is required in order to simply operate the application; for privileged routine applications, this resource is required to invoke the **AddRoles** method, which gives the application its ability to escalate roles.

11.3.4 Edit an Application: The Application Roles Tab

For web applications, privileged routine applications, or client applications, you can configure an application so all its users receive certain roles, which are known as *application roles*.

To specify application roles for an application, the procedure is:

1. In the Management Portal menu, select **System Administration** > **Security** > **Applications**, which displays the different application types.
2. Choose **Web Applications**, **Privileged Routine Applications**, or **Client Applications**. This displays the page for the selected application type.
3. On the applications page, select the application to edit by clicking on its name. This displays the **Edit** page for the application.
4. On the **Edit** page, go to the **Application Roles** tab.
5. To specify one or more application roles, click on the roles listed in the **Available** list. Move them into the **Selected** list with the arrows.
6. Click **Assign** to establish the application roles.

11.3.5 Edit an Application: The Matching Roles Tab

For web applications, privileged routine applications, or client applications, you can configure an application to support what are called *matching roles* and *target roles*. If a user is assigned to a matching role, then running the application causes InterSystems IRIS to assign the user to any associated target roles. An application can have multiple matching roles; for each matching role, it can have multiple target roles; and multiple matching roles can have the same target role.

To establish a matching role and its target roles for an application, the procedure is:

1. In the Management Portal menu, select **System Administration** > **Security** > **Applications**, which displays the different application types.
2. Choose **Web Applications**, **Privileged Routine Applications**, or **Client Applications**. This displays the page for the selected application type.
3. On the applications page, select the application to edit by clicking on its name. This displays the **Edit** page for the application.
4. On the **Edit** page, go to the **Matching Roles** tab.
5. On the **Matching Roles** tab, choose the role to be a matching role from the **Select a matching role** drop-down.
6. To select the accompanying target role(s), click on the roles listed in the **Available** list. Move them into the **Selected** list with the arrows.
7. Click **Assign** to establish the matching role and its target role(s).

11.3.6 Edit an Application: The Routines/Classes Tab

This tab is for privileged routine applications only. On this tab, you can specify the classes or routines that are part of a privileged routine application.

To add a class or routine to privileged routine application, the procedure is:

1. In the Management Portal menu, go to the **Privileged Routine Applications** page (**System Administration** > **Security** > **Applications** > **Privileged Routine Applications**).

2. On the **Privileged Routine Applications** page, there is a list of applications that can be edited. Click the **Name** of the relevant application. This displays the **Edit Privileged Routine Application** page for the application.
3. On the **Edit Privileged Routine Application** page, go to the **Routines/Classes** tab.
4. In the **Routine/Class name** field, enter the name of the routine or class to be added to the application.
5. Specify whether you are adding a **Routine** or a **Class** by selecting the corresponding check box.
6. Click **Assign** to add the routine or class to the application.

11.3.7 Set the Web Application for an Interoperability–Enabled Namespace

InterSystems IRIS enables you to use a different web application for each interoperability-enabled namespace in a given instance. Consequently, you can enable different sets of users to access different interoperability-enabled namespaces within the same InterSystems IRIS instance.

To set the web application for an existing interoperability-enabled namespace, the procedure is:

1. Create a web application that is a copy of the initial web application for the namespace.

   When you create a namespace, the system creates an initial web application named `/csp/`*`namespace`*, where *namespace* is the name of the namespace.

   For instructions, see Create an Application. You can use the **Copy from** field to specify the application to copy.

2. Set the `^%SYS("Ensemble","InstalledNamespace","`*`namespace`*`")` global node to the name of the web application that you created. The *namespace* value is the name of the namespace that will use the new web application.

   For example, if you created a web application named `/csp/ensdemocopy` and you want to use the web application for the `ENSDEMO` namespace, then execute the following command in the Terminal:

   ```
   set ^%SYS("Ensemble","InstalledNamespace","ENSDEMO")="/csp/ensdemocopy"
   ```

   When a user navigates to the interoperability pages for the namespace, the new web application appears.

11.4 Built-In Applications

Each InterSystems IRIS instance comes with a number of built-in applications. This includes a group of *system applications*; there is always access to system applications, even if the **`%Service_WebGateway`** service is disabled.

Table 11–2: InterSystems IRIS Built-In Web Applications

Name	Purpose or Managed Interactions	Associated Resource	System Application
/api/atelier	REST API used by the InterSystems VS Code extensions (%Api.Atelier dispatch class).	**`%Development`**	No
/api/deepsee	REST API used by InterSystems IRIS Business Intelligence (%Api.DeepSee dispatch class).		No
/api/docdb	DocDB REST API (%Api.DocDB dispatch class).		No

Name	Purpose or Managed Interactions	Associated Resource	System Application
/api/iam	InterSystems API Manager (IAM) REST API (%Api.IAM dispatch class); used for obtaining an IAM license from InterSystems IRIS.	**`%IAM`**	No
/api/iknow	iKnow REST API (%Api.iKnow dispatch class).		No
/api/interop-editors	Rule Editor REST API (%Api.InteropEditors dispatch class).		No
/api/mgmnt	API Management REST API (%Api.Mgmnt dispatch class).		No
/api/monitor	Monitoring REST API (%Api.Monitor dispatch class)		No
/api/uima	UIMA REST API (%Api.UIMA dispatch class).		No
/csp/broker	Common static file store. For InterSystems internal use only.		Yes
/csp/documatic	InterSystems class reference documentation.	**`%Development`**	Yes
/csp/sys	General Portal access.		Yes
/csp/sys/exp	Data management options in the Portal.	**`%Development`**	Yes
/csp/sys/mgr	Configuration and licensing options in the Portal.	**`%Admin_Manage`**	Yes
/csp/sys/op	Operations options in the Portal.	**`%Admin_Operate`**	Yes
/csp/sys/sec	Security management and encryption options in the Portal.	**`%Admin_Secure`**	Yes
/csp/user	Default application for the USER namespace.		No
/isc/pki	InterSystems public key infrastructure (PKI).		Yes
/isc/studio/rules	Mapping to the CSP rules files.		Yes
/isc/studio/templates	Mapping to system-defined Studio template files.	**`%Development`**	Yes
/isc/studio/usertemplates	Mapping to user-defined Studio template files.		No
/oauth2	When InterSystems IRIS is configured as an OAuth 2.0 authorization server, used by that server.	**`%Admin_Secure`**	No
/ui/interop/rule-editor	User interface for the Rule Editor.		No

12

Configuring TLS

InterSystems IRIS® data platform can support multiple *configurations*, each of which specifies a named set of TLS-related values. All its existing configurations are activated at startup. When you create a new configuration from the Management Portal, InterSystems IRIS activates it when you save it. The page for managing TLS configurations is the **SSL/TLS Configurations** page (**System Administration** > **Security** > **SSL/TLS Configurations**).

12.1 Create or Edit a TLS Configuration

The page for creating or editing a TLS configuration is the **SSL/TLS Configurations** page (**System Administration** > **Security** > **SSL/TLS Configurations**). To create a new configuration, click **Create New Configuration**, which displays the **New SSL/TLS Configuration** page; to edit an existing configuration, click **Edit** to the right of the name of the configuration. (You can also create a new set of configurations for a mirror member by clicking **Create Configurations for Mirror**; for more information on mirroring and TLS, see "Configuring InterSystems IRIS to Use TLS with Mirroring.")

When creating or editing a TLS configuration, the following fields are available:

- **Configuration Name** — The string by which the configuration is identified. Configuration names can contain any alphanumeric character and any punctuation except the "`|`" character. If you are creating a configuration for the InterSystems IRIS superserver, the configuration name must be `%SuperServer`; for more information about this topic, see "Configuring the InterSystems IRIS Superserver to use TLS."
- **Description** — Any text.
- **Enabled** — Whether or not the configuration is available for activation.
- **Type** — Intended purpose for this configuration, where the choice is **Client** or **Server**; the default is **Client**. Clients initiate the use of the protocol and servers respond to the initial request. (The InterSystems IRIS superserver uses a server configuration; TLS clients use a client configuration.) The value chosen for this field determines:
  - Whether the next field is the **Server certificate verification** or **Client certificate verification** field. If the configuration is for a client, the next field is the **Server certificate verification**, which specifies the verification that may be required for the certificate of any server to which the client is connecting; if the configuration is for a server, the next field is the **Client certificate verification**, which specifies the verification that may be required for the certificate of any client that attempts to connect to the server.
  - The behavior of the **File containing trusted Certificate Authority certificate(s)** field.
- **Server certificate verification** or **Client certificate verification** — Specifies whether or not the configuration requires the verification of the certificate of the peer to which it is connecting.

  A configuration for a client must have a specified **Server certificate verification** and supports possible values of:

  - **None** — Continues under all circumstances.
  - **Require** — Continues only if certificate verification succeeds.

  A configuration for a server must have a specified **Client certificate verification** and supports possible values of:

  - **None** — Specifies that the server neither requests nor requires a client certificate.
  - **Request** — Allows the client to provide or not provide a certificate. If the client provides no certificate, then authentication proceeds; if the client provides a certificate and verification fails, then authentication fails.
  - **Require** — Specifies that the client must provide a certificate; authentication depends on verification of the certificate.

- **File containing trusted Certificate Authority certificate(s)** — The path and name of a file that contains the X.509 certificate(s) in PEM format of the Certificate Authority (CA) or Certificate Authorities that this configuration trusts. The configuration uses the certificates of the trusted CA(s) to verify peer certificates. Typically, a production system uses certificates from commercial CAs with publicly available certificates.

  Regarding this field, note the following:

  - You can specify the path of the file as either an absolute path or as a path relative to the *<install-dir>*`/mgr/` directory.
  - On Windows and macOS, you can specify that the configuration uses the list of trusted CA certificates that the local operating system provides. To do so, specify the string `%OSCertificateStore` as the value of this field.

    On Windows, InterSystems IRIS is compatible with the Microsoft Root Certificate Program that uses Windows Update to fetch additional certificates on demand. For more information about how to configure certificate updating, see Configure Trusted Roots and Disallowed Certificates on the Microsoft website.
  - For a server configuration with a **Client certificate verification** value of **None**, this field is not available, since there is no peer verification.
  - Certificates from the Windows Certificate Export Wizard must be in PEM-encoded X.509 format, not the default of DER-encoded binary X.509.
  - With mirroring, the configuration must also have enough information to verify its own certificate.

  For information on how these certificates are used, see Establishing the Required Certificate Chain. For information on file names for these certificates and how to verify a certificate chain, see the OpenSSL documentation on the **verify** command.

- **This client's credentials** or **This server's credentials** — The files (if needed) containing the X.509 certificate and private key for the local configuration:

  - **File containing this client's certificate** or **File containing this server's certificate** — The full location of the configuration's own X.509 certificate(s). This must be PEM encoded and can be specified as either an absolute or a relative path. This can include a certificate chain. For information on how this is used for authentication, see Establishing the Required Certificate Chain. (Note that certificates from the Windows Certificate Export Wizard must be in PEM-encoded X.509 format, not the default of DER encoded binary X.509.)
  - **File containing associated private key** — The full location of the configuration's private key file, specified as either an absolute or relative path.
  - **Private key type** — The algorithm used to generate the private key, where valid options are **DSA** (Digital Signature Algorithm) and **RSA** (Rivest, Shamir, and Adleman, for the algorithm's inventors).
  - **Private key password** — An optional password for encrypting and decrypting the configuration's private key.

> **Note:** If the private key is password-protected and you do not enter a value here, InterSystems IRIS cannot confirm that the private key and the certificate's public key match each other; this can result in mismatched keys being saved as a key pair.

  - **Private key password (confirm)** — A retyping of the optional password to ensure that it is the intended string.

- **Cryptographic settings**:
  - **Minimum Protocol Version** — The earliest version of the TLS protocol that this configuration supports. This is a drop-down menu that lists all the versions that the instance can support and is TLS v1.2 by default. See note below.
  - **Maximum Protocol Version** — The most recent version of the TLS protocol that this configuration supports. This is a drop-down menu that lists all the versions that the instance can support and is TLS v1.3 by default. See note below.
  - **Enabled cipherlist (TLSv1.2 and below)** — The set of ciphers used to protect communications between the client and the server, if you are using TLS v1.2 or an earlier version. See Supported Ciphers Syntax for more information on this topic.
  - **Enabled ciphersuites (TLSv1.3)** — The set of ciphers used to protect communications between the client and the server, if you are using TLS v1.3. See Supported Ciphers Syntax for more information on this topic.
  - **Diffie Hellman Bits** — (Servers only) The size in bits of the key that the Diffie Hellman ciphers use. The minimum key size varies by operating system. The **Auto** option specifies a key size that is at least the minimum required for the local operating system. Note that the Open Web Application Security Project (OWASP) recommends a minimum key size of 2048 bits.

> **Note:** As described in Which TLS Versions Does My Instance of InterSystems IRIS Support?, the versions of TLS that may be available depend on the version of the underlying OpenSSL libraries that are in use. InterSystems IRIS checks what OpenSSL libraries are in use and attempts to present only the relevant available versions of TLS. A system may, however, be configured in ways that InterSystems IRIS cannot account for. For example, Ubuntu 20.04 ships with a configuration of OpenSSL that has been modified from the default to disallow TLS 1.0 and TLS 1.1. In cases like this, InterSystems IRIS error message and logging are designed to help you determine if there is a conflict between the OpenSSL configuration and the InterSystems IRIS configuration; contact the InterSystems Worldwide Response Center (WRC) or your operating system vendor if you encounter unexpected behavior.

- **OCSP Settings**:
  - **OCSP Stapling** — Whether or not the configuration supports OCSP stapling.

    If you enable OCSP stapling for a client, then the client requests it; if the server does not provide a stapled OCSP response or the response fails validation, then the handshake fails. If OCSP stapling is enabled for a server, then the server provides a stapled OCSP response when it receives a request from a client.

    InterSystems IRIS creates the OCSP response file if it doesn't exist when you save the TLS configuration. An expired OCSP response automatically updates when you save the TLS configuration or when the server receives a request. Additionally, the CertCheck System Sensor (%SYS.Monitor.SystemSensors::**CertCheck()**) updates the response periodically. IRIS background jobs and any jobs that initiate a TLS server connection must have read/write permission on the OCSP response file. You can grant read/write access to the InterSystems IRIS effective group to satisfy these requirements.

    Regardless of whether you enable OCSP stapling, any user who can create a server-side TLS socket can do so with any enabled TLS configuration.

Note: The required fields vary, depending on whether the configuration is to be a client or server and on the desired features. Not all fields are required for all TLS configurations.

To complete the process of creating or editing a configuration, use the following buttons, which appear at the top of this page:

- **Save** — Dismisses the dialog, saving and then activating the configuration. This saves changes to an existing configuration or the configuration being created.
- **Cancel** — Dismisses the dialog without saving changes to an existing configuration or without saving a configuration being created.
- **Test** — Checks for valid configuration information. If the configuration's role is as a client, selecting this button also prompts for a server (its host name, not its URL) and a port number; InterSystems IRIS then tries to establish a test connection to that server. (This button is not available when creating a server configuration.)

  Note: The **Test** button may not be able to successfully connect with all TLS servers, even if the configuration has no errors. This is because the connection test performs a TLS handshake followed by an HTTP request. If the server expects a StartTLS message before the handshake (such as for use with LDAP, SMTP, FTPS, or another protocol), then the test fails, even though the actual TLS connection to the server succeeds.

12.1.1 Required Information for Certificates

When a client authenticates a server, the client needs to have the full certificate chain from the server's own certificate to the server's trusted CA certificate — including all intermediaries between the two.

There is an issue when setting up a server TLS configuration and the server's trusted CA certificate is not a root certificate. In order for authentication to work properly, the client needs to have access to all the certificates that constitute the certificate chain from the server's personal certificate to a self-signed trusted CA certificate. This chain can be obtained from the combination of the server's certificate file (sent during the handshake) and the client's trusted CA certificate file. The self-signed trusted root CA certificate must be in the client's CA certificate file, and the server's personal certificate must be the first entry in the server's certificate file. Other certificates may be divided between the two locations. The same constraints apply in reverse when a client authenticates to a server.

Regarding certificate formats, note that certificates from the Windows Certificate Export Wizard must be in PEM-encoded X.509 format, not the default of DER encoded binary X.509. All certificates must be PEM encoded regardless of file extension.

12.1.2 Enabled Cipher Suites Syntax

A configuration only allows connections that use its enabled cipher suites. To specify enabled cipher suites, you can either:

- Provide a list of individual cipher suites, using each one's name
- Use OpenSSL syntax to specify which cipher suites to enable and disable

Both the list of cipher suite names and the syntax for specifying enabled cipher suites is described on the ciphers(1) man page at openssl.org. This syntax allows you to specify guidelines for requiring or proscribing the use of various features and algorithms for a configuration.

The default set of cipher suites for an InterSystems IRIS configuration is `ALL:!aNULL:!eNULL:!EXP:!SSLv2` which breaks down into the following group of colon-separated statements:

- `ALL` — Includes all cipher suites except the eNULL ciphers
- `!aNULL` — Excludes ciphers that do not offer authentication

- `!eNULL` — Excludes ciphers that do not offer encryption
- `!EXP` — Excludes export-approved algorithms (both 40- and 56-bit)
- `!SSLv2` — Excludes SSL v2.0 cipher suites

For more information, see the OpenSSL documentation on the ciphers command.

12.1.3 A Note on InterSystems IRIS Client Applications Using TLS

For certain activities, you can use InterSystems IRIS instances to support client applications that interact with the InterSystems IRIS superserver.

When using client applications that interact with the InterSystems IRIS superserver using TLS, the following aspects of the configuration require particular attention:

- **Configuration Name** — While there are no constraints on the name of clients, this information is required to configure the connection.
- **Type** — Because the instance is serving with a TLS client, the type must be specified to be of type **Client**.
- **Ciphersuites** — The specified cipher suites need to match those required or specified by the server.

It is also necessary to ensure that the client and the server are configured so that each may verify the other’s certificate chain, as described in Establishing the Required Certificate Chain.

12.2 Delete a Configuration

The page for deleting a TLS configuration is the **SSL/TLS Configurations** page (**System Administration** > **Security** > **SSL/TLS Configurations**). To delete a configuration, click **Delete** to the right of the name of the configuration. The Portal prompts you to confirm the action.

12.3 Update Certificates for Existing Configurations

InterSystems IRIS provides a way for updating the associated certificates for a TLS configuration. After obtaining the new certificates, you can update them in the TLS configuration. The changes take effect with a new network connection. Many connections are dropped and reestablished quickly so the change can be immediate. However, certain connections can remain open for longer, for example, mirroring and certain interoperability interface connections. To ensure that the updated certificates take effect, InterSystems recommends you drop and reestablish these connections.

Mirroring environments require an additional step after reestablishing connections. Please see Authorizing X.509 DN Updates for more details.

12.4 Reserved and Required Configuration Names

InterSystems IRIS reserves several TLS configuration names for use with particular features. When using such a feature, you must use the reserved configuration name(s). The reserved configuration names are:

- `%MirrorClient` — For a mirror member when acting as a TLS client. For more information on mirroring and TLS, see “Configuring InterSystems IRIS to Use TLS with Mirroring.”
- `%MirrorServer` — For a mirror member when acting as a TLS server. For more information on mirroring and TLS, see “Configuring InterSystems IRIS to Use TLS with Mirroring.”
- `%SuperServer` — For the InterSystems IRIS superserver when accepting connections from other InterSystems IRIS components. For more information about configuring the superserver to use TLS, see Configuring the InterSystems IRIS Superserver to Use TLS.
- `%TELNET/SSL` — For the Windows Telnet server when accepting connections protected by TLS. For more information on mirroring and Telnet, see “Configuring the InterSystems IRIS Telnet Server for TLS.”

Important: For TLS to function properly, you must use the exact case for each configuration name as it appears here.

12.5 Creating, Editing, and Deleting TLS Configurations Programmatically

To manage TLS configurations programmatically, use:

- The Security.SSLConfigs class
- The applicable configuration merge actions:
  - CreateSSLConfigs
  - ModifySSLConfigs
  - DeleteSSLConfigs

13

Support for OAuth 2.0 and OpenID Connect

This page introduces InterSystems IRIS® support for OAuth 2.0 and OpenID Connect.

13.1 Supported Scenarios

With InterSystems IRIS support for OAuth 2.0 and OpenID connect, you can do any or all of the following:

- Use an InterSystems IRIS web application as a client
- Use an InterSystems IRIS web application as a resource server
- Use an InterSystems IRIS instance as an authorization server

For example, you can use an InterSystems IRIS web application as a client of an authorization server that uses third-party technology. Or you can use third-party clients with an authorization server that is built on InterSystems IRIS. The resource server or resource servers could be implemented in InterSystems IRIS or in a different technology.

In all cases, the authorization server is the most complex element and is generally created first. You create clients later. When you create a client, it is generally necessary to understand the capabilities and requirements of the authorization server, such as the scopes it supports.

13.2 InterSystems IRIS Support for OAuth 2.0 and OpenID Connect

The InterSystems IRIS support for OAuth 2.0 and OpenID Connect consists of the following elements:

- Configuration pages in the Management Portal.

  If you configure a client (or a resource server), use the options at **System Administration** > **Security** > **OAuth 2.0** > **Client Configuration**.

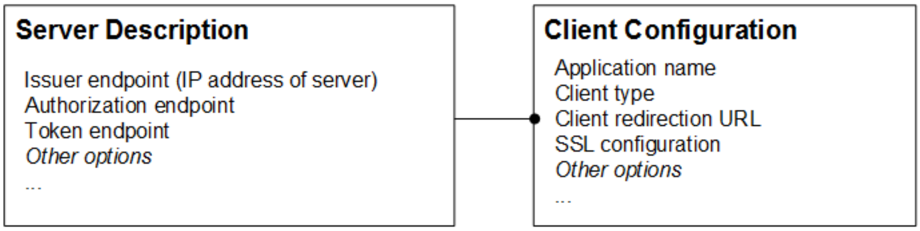
  If you configure an authorization server, use the options at **System Administration** > **Security** > **OAuth 2.0** > **Server Configuration**.

- Classes in the %SYS.OAuth2 package. These classes are the client API. If you define an InterSystems IRIS web application as an OAuth 2.0 client, your client uses methods in these classes.
- Classes in the %OAuth2 package. If you use an InterSystems IRIS instance as an OAuth 2.0 authorization server, you customize the server by subclassing one or more of the classes in the package %OAuth2.Server. Other classes in %OAuth2 provide utility methods for your code to call.
- Classes in the OAuth2 package (in the IRISSYS database). These include persistent classes for internal use by InterSystems IRIS, and you can ignore most of them. However, if you want to create configuration items programmatically, you would use a subset of the classes in this package.

The following subsections provide an overview of the configuration items.

13.2.1 Configuration Items on a Client

Within an InterSystems IRIS instance that is acting as an OAuth 2.0 client, it is necessary to define two connected configuration items for a given client application: a *server description* (which describes the authorization server) and a *client configuration* (which configures the client). A given instance can have any number of server descriptions. Each server description has multiple client configurations, as shown in the following figure, which also indicates some of the information stored in these configuration items:

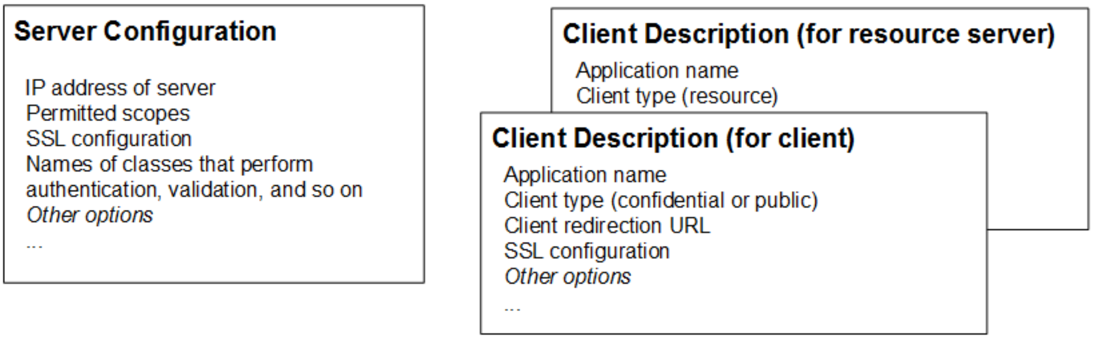


This architecture is intended to simplify configuration, because it enables you to define multiple client configurations that use the same authorization server without needing to repeat the details of the authorization server.

You can create these items via the Management Portal, as described in Using an InterSystems IRIS Web Application as an OAuth 2.0 Client. Or you can create them programmatically, as described in Creating Configuration Items Programmatically.

13.2.2 Configuration Items on the Server

Within an InterSystems IRIS instance that is acting as an OAuth 2.0 authorization server, it is necessary to define a *server configuration* (which configures the authorization server) and a number of *client descriptions*. The following figure indicates some of the information stored in these configuration items.



A given InterSystems IRIS instance can have at most one server configuration and can have many client descriptions. One client description is necessary for each client application. A client description is also necessary for each resource server that uses any endpoints of the authorization server. If a resource server does not use any endpoints of the authorization server, there is no need to create a client description for it.

You can create these items via the Management Portal, as described in Using InterSystems IRIS as an OAuth 2.0 Authorization Server. Or you can create them programmatically, as described in Creating Configuration Items Programmatically.

13.3 Standards Supported in InterSystems IRIS

This section lists the standards that InterSystems IRIS supports for OAuth 2.0 and Open ID Connect:

- The OAuth 2.0 Authorization Framework (RFC 6749) — See https://datatracker.ietf.org/doc/rfc6749
- The OAuth 2.0 Authorization Framework: Bearer Token Usage (RFC 6750) — See https://datatracker.ietf.org/doc/rfc6750
- OAuth 2.0 Token Revocation (RFC 7009) — See https://datatracker.ietf.org/doc/rfc7009
- JSON Web Token (JWT) (RFC 7519) — See https://datatracker.ietf.org/doc/rfc7519
- OAuth 2.0 Token Introspection (RFC 7662) — See https://datatracker.ietf.org/doc/rfc7662
- OpenID Connect Core 1.0 — See http://openid.net/specs/openid-connect-core-1_0.html
- OAuth 2.0 Form Post Response Mode — See http://openid.net/specs/oauth-v2-form-post-response-mode-1_0.html
- JSON Web Key (JWK) (RFC 7517) — See https://datatracker.ietf.org/doc/rfc7517
- OpenID Connect Discovery 1.0 — See https://openid.net/specs/openid-connect-discovery-1_0.html
- OpenID Connect Dynamic Client Registration — See http://openid.net/specs/openid-connect-registration-1_0-19.html
- JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants (RFC 7523) — See https://tools.ietf.org/html/rfc7523
- Proof Key for Code Exchange (RFC 7636) — See https://tools.ietf.org/html/rfc7636

14

Configuring Task Manager Email Settings

You can set the Task Manager to send email notification when a task completes, as described in Using the Task Manager. On the **Task Manager Email Settings** page (**System Administration** > **Configuration** > **Additional Settings** > **Task Manager Email**), you can configure the notification settings described on this page.

14.1 Email Configuration Settings

SMTP Server and Port

Address and port of your outgoing SMTP (Simple Mail Transfer Protocol) mail server

SSL Config

The SSL configuration to use if you want to encrypt the email using SSL/TLS. If there are no SSL configurations on the instance, or you want to create a new one, see Create or Edit a TLS Configuration. If you do not select an SSL configuration, SSL/TLS will not be used.

SMTP Auth User and Password

Required only for SMTP authentication to the SMTP server. See RFC 2554 for details. If you do not provide entries, SMTP username and password are set to NULL.

Sender

Required only for SMTP authentication to the SMTP server. See RFC 2554 for details.

Reply To

Email address to which the recipient should reply

Success Subject

The formatted subject line of a successful task message. See the section “Parameters for Subjects and Messages” below.

Success Message

The formatted message sent after a successful task runs

Failure Subject

The formatted subject line of a failed task message

Failure Message

The formatted message sent after a task fails

Note: You can also configure email settings programmatically through the `%SYS.Task.Config` class.

14.2 Parameters for Subjects and Messages

Format the information in the subject and message text boxes using the task parameters listed at the bottom of the web page and defined in the following table. The web page includes examples.

Task Parameter	Description
ID	Task ID
DESCRIPTION	Task description
NAME	Task name
LASTSTARTED	Last time the task started
LASTFINISHED	Last time the task finished
SCHEDULED	Last scheduled starting time
CURRENTDATE	Date the email sent
CURRENTTIME	Time the email sent
STATUS	Return value of the task: • If the task is successful, this returns `""`. • If the task results in a trapped error, this returns text from a %Status value. • If the task results in an untrapped error, this returns $ZERROR.
TASKCLASS	Task class used for this task; for example, `%SYS.Task.IntegrityCheck` for a database integrity check task,
ERROR	Error code if task failed
SUCCESS	Completed message if task ran successfully

15

Configuring National Language Support (NLS)

A national language locale defines the character set in which all textual data is encoded by InterSystems IRIS® data platform. The character set is 16-bit Unicode UCS-16.

Each locale contains a number of character tables used by InterSystems IRIS when displaying text, collating data (see Collation), converting between uppercase and lowercase letters, matching patterns, and so on. Each locale defines the table to be used for each of these purposes, as well as other details such as date, time, and number formats.

Each InterSystems IRIS instance uses a single current locale; this is determined when the instance is installed, but can be changed at any time. When you change the current locale, some or all of the locale tables used by InterSystems IRIS change.

Installing a new locale does not result in any data conversion, but rather changes how data is represented.

Installing a new locale should not be a frequent operation; it is intended mainly as an upgrade option or the means to correct an installation choice. Always remember that data conversion may be needed and that special attention should be given to global subscripts.

You cannot alter the system locales provided with InterSystems IRIS, which are overwritten when the instance is upgraded.

15.1 Using the NLS Pages of the Management Portal

The **National Language Settings** pages (**System Administration** > **Configuration** > **National Language Settings**) let you browse existing locales and tables as well as create custom locales. You can install a new current locale, load a new table into memory, and more the using the Management Portal. When you select **System Administration** > **Configuration** > **National Language Settings**, the following options are available in the right-hand column:

- Configured Defaults
- Locale Definitions
- Import Locale

15.1.1 Configured Defaults

The **Configured Defaults** page (**System Administration** > **Configuration** > **National Language Settings** > **Configured Defaults**) displays the locale table currently used by default for each purpose within InterSystems IRIS. When writing ObjectScript code or using some utilities, it is possible to specify a particular table for a given purpose; the default table isused when no table is specified.

Each table name is color-coded to show whether the setting was inherited from the current locale at installation or specified using the NLS class packages, as described in Using System Classes for National Language Support.

The configuration defaults are a property of the instance, not of the locale. Therefore, when the instance is upgraded, the default selections are preserved.

15.1.2 Locale Definitions

From the **Locale Definitions** page (**System Administration** > **Configuration** > **National Language Settings** > **Locale Definitions**), you can select a locale at the **Select a locale** drop-down and perform several actions. The drop-down is always set to the current locale when the page first displays.

- Using the **Use locale date/time/number formats for [current locale]** drop-down, indicate whether or not you want to use the date, time, and number formats specified by the current locale. Note that this always applies to the current locale, not a locale you have selected in the **Select a locale** drop-down but not yet installed.
- To view the details of a selected locale, click **Properties**. The next page displays the locale properties grouped into categories. For locales you have added, you can edit the fields and click **Save** to save these changes. You cannot edit the system locales provided with InterSystems IRIS. The properties are as follows:
  - **Basic Properties**
  - **Date, Time, and Number Formats**
  - **Internal Tables** — You have two options when editing the internal tables:
    - **Edit Tables** — You may select or delete a table from the list boxes by double clicking an item, or by selecting an item and then clicking **>** or **<** to move it from the appropriate list.

      Tables that require at least one entry are indicated by an asterisk (`*`); the other tables may be left empty.
    - **Edit Defaults** — You may choose the default from the values you enter in the **Edit Tables** function of the **Internal Tables** category.
  - **Input/Output Tables** — You can edit, add, or remove a table when choosing to edit this category.
    - To edit a table, click the table in the first list. The table name appears in the lower box. You can modify the values and click **Save**.
    - To remove a table, click the table in the first list. The table name appears in the lower box; click **Remove**. A confirmation box displays offering you the option to **Cancel** or **OK** the delete.
    - To add a table, click **Add**. The lower box has the **Table** field enabled and the **Remove** option disabled. You can enter a table name and enter the **Output to** and **Input from** fields.

    Click **Save** when you have made all your updates. If the save is successful, the updated list appears; otherwise, an appropriate error message displays.
  - **Input/Output Defaults**
  - **Strings**
- To take further actions, click the following buttons:
  - **Validate** — Validates the selected locale, displaying an error message if the locale cannot be validated. This is useful when creating custom locales.

- **Copy** — Creates a copy of the selected locale, which you can then customize. The name of the copy must contain four characters beginning with `y` and ending with `8` or `w`. The default description is `Copy of`*%locale*, where *%locale* is the selected locale name. When the copy is created, it is added to the **Select a locale** drop-down.
- **Export** — Exports a locale to an .xml file. For example, you might export a custom locale you created and import it on another instance using the Import Locale page. The default name is loc_%locale.xml, where *%locale* is the selected locale. In addition, you can include the path of the export file; if you do not specify the path, the default location is *install-dir*\mgr.
- **Install** — Installs the selected locale as the current locale for the instance. An initial validation occurs; if it fails, an error message displays, otherwise you can continue with installation.
- **Load Table** — Lets you load a table from the selected locale (the current locale or another) into memory from disk. Select a table type and then a table name from the list populated after you select the type. Click **OK** to load the table or **Cancel** to close the dialog box and return to the **Locale Definitions** page.
- **Delete** — Deletes a locale. You can delete only custom locales; the button is disabled when a system locale is selected. You cannot delete the current locale even if it is a custom locale. You must confirm deletion of the locale before proceeding.

15.1.3 Import Locale

From the **Import Locale** page (**System Administration** > **Configuration** > **National Language Settings** > **Import Locales or Tables**), you can import locales or tables. For example, you can import a custom locale exported (as described in the previous section) from another instance.

1. Select the **Import Type** > **Locale** is the default.
2. Enter a file name and click **Import**. The only valid file extensions are .xml and .goq.
3. A message displays indicating how many locales, tables, and subtables have been imported.

15.2 Using the NLS Class Packages

System Classes for National Language Support contains details on using both the %SYS.NLS and Config.NLS class packages.

The %SYS.NLS Classes section contains details on using the following classes:

- %SYS.NLS.Device — Properties of the current device.
- %SYS.NLS.Format — Date, time, and number formats.
- %SYS.NLS.Locale — Basic properties of current locale (read-only).
- %SYS.NLS.Table — System and process tables (I/O and internal).

The Config.NLS Classes section contains details on using the following classes:

- Config.NLS.Locales
- Config.NLS.SubTables
- Config.NLS.Tables

You can also find details on each of these classes in the *InterSystems Class Reference*.

16

Configuring Work Queue Manager Categories

The Work Queue Manager enables you to distribute work to multiple concurrent processes programmatically in order to improve performance. For more information, see Using the Work Queue Manager.

When your code initializes a set of worker jobs using the Work Queue Manager, that code can specify the category that supplies the worker jobs. A *category* is an independent pool of workers with its own maximum of workers.

The system supplies two categories that you cannot delete: Default and SQL. By default, the maximum number of workers for these categories is **Dynamic**, which is equivalent to twice the number of cores available to your system.

The **Work Queue Manager Categories** page enables you to create, modify, and delete categories as described in the following sections. Any changes that you make are reflected in the configuration parameter file, `iris.cpf`.

16.1 Creating a Work Queue Manager Category

From the **Work Queue Manager Categories** page, you can create new categories.

Category names must be unique and are case sensitive. Additionally, category names can include only letters, numbers, dashes, underscores, and periods, and have a maximum of 64 characters.

To create a new category, do the following:

1. Navigate to **System Administration** > **Configuration** > **System Configuration** > **WQM Categories**.
2. Click **Create Category**.
3. In the **Category name** field, type a name for the category.
4. In the **Max active workers** field, select the maximum number of active worker jobs kept in the pool of jobs that service requests in this category. InterSystems IRIS detects idle jobs and automatically starts new jobs to keep the maximum active job number around this limit. By default, InterSystems IRIS dynamically adjusts this limit based on system resources, up to a maximum of 16 workers.
5. In the **Default workers** field, select the default number of worker jobs assigned to a work group, when a work group in this category is created and no worker job count is specified. By default, InterSystems IRIS dynamically adjusts this value based on system resources, up to a maximum of 8 workers.
6. In the **Max workers** field, select the maximum number of worker jobs kept in the pool of jobs servicing requests in this category. If the number of jobs requested by a work group in this category is greater than this limit, then InterSystems

IRIS provides workers only up to this maximum value. By default, InterSystems IRIS dynamically adjusts this value based on system resources, up to a maximum of 16 workers.

16.2 Editing or Deleting a Work Queue Manager Category

From the **Work Queue Manager Categories** page, you can modify the maximum number of workers for an existing category and delete user-specified categories as follows:

1. Navigate to **System Administration** > **Configuration** > **System Configuration** > **WQM Categories**.
2. Click **Edit** or **Delete** for the relevant category.

17

Maintaining Local Databases

You can review and maintain local databases on the **Databases** page of the Management Portal (**System Operations** > **Databases**). From this page, you can view the following information:

- Databases General Information — Overview information for all databases.
- Databases Free Space Information — Free space information for all databases.
- Database Details Page — Specific information for individual databases. You can also perform maintenance operations to increase free space from this page.

You also have the option to perform an integrity check using the buttons at the top of the page. For more information, see Verifying Structural Integrity.

Most of the information and operations described in this section can also be found and performed using the ^DATABASE command line utility.

17.1 Databases General Information

The **Databases** page (**System Operation** > **Databases**) contains a list of all local databases. For each local database, you see the following information:

Table 17–1: Local Databases Information

Column Heading	Definition
Name	The database name; click this name to display more details. See Database Details Page.
Directory	The system directory in which the database resides.
Max Size (GB)	The maximum size allocated to which the database can grow, in gigabytes.
Size (MB)	The current allocated size of the database, in megabytes. **Note:** This field measures the logical size of the database. As a result, the size reported here may be lower than the physical size of the database, in particular for the IRISTEMP database.

Column Heading	Definition
Status	The status of the database: mounted (including which permissions it has), unmounted, or dismounted. • A mounted database is one for which **Mount Required at Startup** is selected, as described in Edit Database Properties, and which therefore must be mounted for InterSystems IRIS® data platform to start or become primary in a mirror; in this case, it is always mounted and accessible when InterSystems IRIS starts. Alternatively, it is a previously unmounted database that has been mounted dynamically when you accessed it or explicitly mounted it; in this case, it remains mounted until you explicitly dismount it or restart/stop InterSystems IRIS. • An unmounted database is one for which **Mount Required at Startup** is not selected and which therefore does not need to be mounted for InterSystems IRIS to start or become mirror primary, and has been neither accessed nor explicitly mounted; it is mounted dynamically when you access it or explicitly mount it, and remains mounted until you explicitly dismount it or restart/stop InterSystems IRIS. • A dismounted database is one that has been explicitly dismounted; it is inaccessible until you explicitly mount it or restart/stop InterSystems IRIS (that is, a dismounted database is not mounted dynamically if you try to access it). To permanently dismount a database you must remove it from the configuration, as described in Local Databases.
Encrypted	Indicates whether or not the database is encrypted.
Journal	Indicates whether globals in the database are journaled with a `Y` or an `N`.

In addition, the page contains a filter bar that you can use to control the number of databases displayed. For example, to list only the system databases, you might enter `IRIS*` in the **Filter:** text box; and/or to list only five databases per page, enter `5` in the **Page size:** text box; and/or to limit the number of rows displayed to three, enter `3` in the **Max rows:** text box (a **+** sign displayed with the number in the **Results** field indicates there are additional databases that meet the specified criteria, but they are not displayed).

17.2 Database Free Space Information

Managing the free space (empty blocks) in a database is an important aspect of database maintenance. To see free space information, you can display the Free space view of the Management Portal or use the **^%FREECNT** utility. Remember that the size and free space attributes of a database in normal operation change continuously, and that numbers reported by the Portal or the utility at a given point in time are approximations only.

If you determine that a database has more free space than is necessary, you may compact and truncate the database as described in Database Details Page.

17.2.1 Display Free Space Information Using the Management Portal

To display free space information, which shows information about the amount of free space on each local database, navigate to the **Databases** page (**System Operation** > **Databases**) and click the **Free space view** radio button. The following table describes the information displayed:

Table 17–2: Local Databases Free space Information

Column Heading	Definition
Name	The database name; click this name to display more details. See Database Details Page.
Directory	The system directory in which the primary volume of the database resides.
Max Size	The maximum allocated size to which the database can grow, in gigabytes. The default is unlimited when you create a database.
Size	The current allocated size of the database, in megabytes. **Note:** This field measures the logical size of the database. As a result, the size reported here may be lower than the physical size of the database, in particular for the IRISTEMP database.
Expansion Size	Size (in MB) by which to expand the database. The default and recommended setting is zero (0) when you create a database, which indicates the use of system defaults (12% of the current size or 10 MB, whichever is larger). Under this setting, the expansion size will not be greater than 1GB.
Available	The amount of free space (in MB) available in the database.
% Free	The percentage of free space available in the database.
Disk Free Space	The amount of space free on the volume.
Status	The status of the directory, which indicates if the database is mounted and with what permissions.

For information about performing free space management operations from the Management Portal, see Database Details Page.

17.2.2 Display Free Space Information Using ^%FREECNT

InterSystems IRIS also provides the **^%FREECNT** utility, which you run by entering **do ^%FREECNT** in the Terminal, to display the free space available in a database.

When using **^%FREECNT** in the %SYS namespace, you can choose to display the free space of all databases by entering an asterisk (*) at the prompt, or enter one database directory name. For example:

```
%SYS>do ^%FREECNT

Database directory to show free space for (*=All)? *

Databases Selected
------------------
c:\MyIris\mgr\
c:\MyIris\mgr\irisaudit\
c:\MyIris\mgr\irislib\
c:\MyIris\mgr\irislocaldata\
c:\MyIris\mgr\iristemp\
c:\MyIris\mgr\user\
Device:
Right margin: 80 =>


                            Database Free Space
                              Feb 15 2012  7:25 PM
Database                           Max Size  Size     Available %Free    Disk Free
c:\MyIris\mgr\                    Unlimited 191MB    19MB      9.94     60.79GB
c:\MyIris\mgr\irisaudit\          Unlimited 1MB      0.43MB    43       60.79GB
c:\MyIris\mgr\irislib\            Unlimited 319MB    27MB      8.46     60.79GB
```

```
c:\MyIris\mgr\irislocaldata\      Unlimited 1MB     0.55MB    55      60.79GB
c:\MyIris\mgr\iristemp\           Unlimited 4MB     1.5MB     37.5    60.79GB
c:\MyIris\mgr\user\               Unlimited 1MB     0.43MB    43      60.79GB
```

In a namespace other than %SYS, the utility shows the free space of the databases in that namespace. For example:

```
USER>Do ^%FREECNT


Databases Selected
------------------
c:\MyIris\mgr\user\
Device:
Right margin: 80 =>


                            Database Free Space
                              Feb 15 2012  7:28 PM
Database                           Max Size  Size    Available %Free   Disk Free
c:\MyIris\mgr\user\                 Unlimited 1MB     0.52MB    52      42.72GB
```

Note: A `<-` flag (in the %Free column) indicates that the percentage of free space in the specified database has dropped below 5%. Ensure that there is enough space on the file system to handle database expansion.

The **^DATABASE** utility is another way to display free space information, as well as perform other database tasks. For example, you can compact globals in a database, or you can recreate a database, which lets you clear the data in an existing database without changing the database's name or size. See ^DATABASE for information about the utility.

Note: The data structures used by InterSystems IRIS are self-balancing and suffer no performance degradation over time. It is never necessary to take a database down to rebuild it nor to compress data or indexes to regain performance.

17.3 Database Details Page

The **Database Details** page displays detailed information about any database. To access this information from the Management Portal:

1. Display the **Databases** page (**System Operation** > **Databases**).
2. Click the name of the database for which you would like to see more details.

Along the top of the **Database Details** page is a row of buttons, which allow you to perform various database operations:

- **Mount** / **Dismount** — Mount or dismount a database. The new status remains in effect until you explicitly change it or restart/stop InterSystems IRIS; to permanently dismount a database, you must remove it from the configuration.
- Compact — Move free space distributed throughout a database to its end.
- Truncate — Return free space at the end of a database to the underlying file system.
- Defragment — Relocate global blocks so that the blocks representing a given global are in continuous sequence.

The page also includes information specific to the selected database, which is organized into a General Information table and a Database Size table:

Table 17–3: General Information

Field	Definition
Directory	Name of directory where this database resides.

Field	Definition
Resource Name	Resource name assigned to this database.
Mounted	Indicates whether this database is mounted.
Read Only	Indicates whether this database is mounted as read-only.
Read Only Reason	The reason the database is mounted as read-only.
Encrypted	Indicates whether this database is encrypted.
Encryption Key ID	Indicates encryption key ID.

Table 17–4: Database Size

Field	Definition
Block Size	Block size (bytes) for this database.
Blocks	Current number of blocks within this database.
Max Size	Maximum size (MB) allowed for this database.
Size	Current size (MB) of this database.
Expansion Size:	Amount (MB) this database will expand by.
Available Space	Available space within this database.
% Free Space	% Free space within this database.
Disk Free Space	Free space on disk containing this database.
Last Expansion Time	Last time this database expanded.
Full	Indicates that this database is full.

17.4 Compacting a Database

Compacting a database moves free space distributed throughout the database to its end by relocating global blocks. You can then return the free space to the underlying file system by truncating the database. (You can also compact globals; see Compact Globals in a Database.)

When you *compact* a database, you specify the amount of the available free space to be positioned at its end, and the operation rearranges global blocks to ensure that at least that amount of free space is located at the end. (The operation cannot create more free space, so it can never place more at the end than the total available amount.)

For example, suppose the size of a database is 50 MB, with 15 MB of that being free space, and 5 MB of that free space already positioned at the end of the database. If you compact the database and specify more than 5 MB but less than 15MB, global blocks are moved from the end of the database to the beginning until the free space at the end equals the amount you specified; if you specify 15 MB, all possible global blocks are moved to the beginning.

To compact a database:

1. Navigate to the **Databases** page (**System Operations** > **Databases**).
2. Click the name of the database you want to compact. This takes you to its Database Details page.

3. Click **Compact** on the ribbon at the top of the page. This displays the **Compact Database** dialog box, which shows the name and location of the database, its current size, the total available free space, and the amount of free space currently at the end of the file.
4. Specify the amount of free space you want at the end of the file in the **Target free space (in MB) at end of file** text box. Your entry must be within the stated range. Once you have entered an amount, click **OK**. If all of the free space is already at the end, or there is no free space, the prompt does not appear and the **OK** button is disabled.

   Note: For a number of reasons, the operation may move more free space than the amount you specify. Conversely, because the numbers reported are approximations, it is possible that not all of the free space displayed can actually be moved.

5. When the task is complete, a **Background Task Info** dialog box appears. Optionally, you can also view the task on the **Background Tasks** page (**System Operation** > **Background Tasks**).
6. Click **Close** to redisplay the **Database Details** page. The page should automatically refresh to show the new database information.

Note: The compact database operation is designed to run concurrently with normal database activity. The operation does consume some system resources, however, and may not complete if the system is under extremely high load. For these reasons, InterSystems recommends running this and other database reorganization operations (including compacting and defragmenting globals) during off-peak hours, and running only one such operation on a system at a time.

17.5 Truncating a Database

Truncating a database returns free space from the end of the database to the underlying file system. A database is often truncated after being compacted, which moves free space to the end of the database.

When you *truncate* a database, you specify a target size for the database. If there is sufficient free space at the end of database, the operation removes enough to reduce the database to the target size; if not, it removes all that can be removed. (To find out how much of a database's free space is positioned at the end, compact the database; you do not need to complete the operation to display current total available free space and the amount at the end.)

Note: This feature is not applicable to databases with raw volumes.

To truncate a database:

1. Navigate to the **Databases** page (**System Operations** > **Databases**).
2. Click the name of the database you want to truncate. This takes you to its Database Details page.
3. Click **Truncate** on the ribbon at the top of the page. This displays the **Truncate Database** dialog box, which shows the name, location, and current size in megabytes (MB) of the selected database.
4. Enter the **Target File Size (MB)**, which must be less than the current size, and click **OK**. Enter `0` to remove all possible free space from the end of the file.

   Note: Free space for truncation purposes is different than free space for storing data. Certain control structures can be deleted during truncation if they are not being used for data storage. This can lead to a difference in reported free space.

17.6 Defragmenting a Database

Defragmenting a database rearranges global blocks within the database so that all of the blocks containing data for a given global are in consecutive sequence. The operation does not place big string blocks or pointer blocks from a global in sequence, but it does locate them in a contiguous area. As part of the process, the **Defragment a database** option compacts all globals in the same manner as the **Compact globals in a database** option, but with a target density of 70%. (If this is lower than the current global block density of the database, the size of the database does not increase.)

Note: The IRISTEMP database cannot be defragmented.

In general, it is not necessary to run defragmentation on any regular basis. Some workloads, however, particularly those that read large portions of a database sequentially, can benefit from having global blocks organized sequentially.

The defragmentation process requires a certain amount of free space at the end of the database. For this reason, the following possibilities exist:

- If there is enough free space at the end to perform the operation, it completes without any changes to the database beyond global defragmentation.
- If there is not enough free space in the database, the database is expanded as necessary. When defragmentation is complete, you can truncate the database to remove the added free space.
- If there is not enough free space in the database but there is significant free space that could be moved to the end, you are informed of this. In this case, compacting the database before you choose the **Defragment a database** option reduces the amount of expansion required to complete the defragmentation operation.

To defragment the globals in a database, use the following procedure:

1. Navigate to the **Databases** page (**System Operations** > **Databases**).
2. Click the name of the database you want to defragment. This takes you to its Database Details page.
3. Click **Defragment** on the ribbon at the top of the page. This displays the **Defragment Database** dialog box, which shows the name, location, and current size in megabytes (MB) of the selected database, as well as a message describing the free space needed to defragment.
4. Click **OK**.
5. When the task is complete, a **Background Task Info** dialog box appears. Optionally, you can also view the task on the **Background Tasks** page (**System Operation** > **Background Tasks**).
6. Click **Close** to redisplay the **Database Details** page. The page should automatically refresh to show the new database information.

Important: The defragment operation temporarily relocates all of the data in the database, regardless of the degree of global fragmentation in the database prior to running the operation. Subsequent runs of the operation consume similar amounts of resources but do not provide any additional benefit.

Note: The defragment operation is designed to run concurrently with normal database activity. The operation does consume some system resources, however, and may not complete if the system is under extremely high load. For these reasons, InterSystems recommends running this and other database reorganization operations (including compacting a database and compacting globals) during off-peak hours, and running only one such operation on a system at a time.

Global defragmentation can involve a temporary increase in the size of the database being compacted. If this causes the database to reach its configured maximum size (see Local Databases), or if expansion is not possible because there is insufficient space available on the storage volume, the operation is canceled.

17.7 Compacting Globals in a Database

Another way to manage database space is to compact the globals in a database using the **^DATABASE** routine. Compacting globals consolidates global data into fewer blocks, increasing the amount of free space in a database.

When globals are created and updated, InterSystems IRIS typically allocates data in a manner that fills global blocks to about 70% of capacity. (Globals that have grown entirely in collation order may be allocated at closer to 90%.) In general, allowing InterSystems IRIS to manage global block density automatically is sufficient. However, some nonsequential patterns of data deletion may reduce average global block density considerably.

Note: To see the current density of the global blocks in a database on a global by global basis, you can run an integrity check (as described in Verifying Structural Integrity) and examine the `Data Level` output for each global.

When you compact globals, you specify a desired global block density (90% by default) and the operation attempts to come as close to this as possible by consolidating data—for example, rearranging global data that is spread across three blocks into two. Typically (but not always), compacting globals yields a meaningful increase in available free space within a database. (If you specify a target density that is lower than the current global block density of the database, the size of the database does *not* increase.)

To compact the globals in a database, use the following procedure:

1. Open the Terminal and change to the **`%SYS`** namespace.
2. Enter `do ^DATABASE`, and select `7) Compact globals in a database` from the menu.
3. Specify the directory of the database on which you want to run the operation. You can specify multiple databases by entering `?` at the `Database directories to compact?` prompt and then entering a list of numbers.
4. Indicate that you want to compact all globals, or instead enter a list of individual globals to be compacted.
5. Specify the target average global block density, respond to the rest of the prompts, and confirm.

Note: The compact globals operation is designed to run concurrently with normal database activity. The operation does consume some system resources, however, and may not complete if the system is under extremely high load. For these reasons, InterSystems recommends running this and other database reorganization operations (including compacting a database and defragmenting globals) during off-peak hours, and running only one such operation on a system at a time.

Global compaction can involve a temporary increase in the size of the database being compacted. If this causes the database to reach its configured maximum size (see Local Databases), or if expansion is not possible because there is insufficient space available on the storage volume, the operation is canceled.

18

Controlling InterSystems IRIS Processes

An InterSystems IRIS system runs a number of processes. Application code as well as InterSystems IRIS system code executes within these processes. There are three categories of InterSystems IRIS processes:

- User processes, created when a user connects to InterSystems IRIS.
- Background processes, created when a user issues an ObjectScript **Job** command, or by the Management Portal or a utility (see Using the Background Tasks Page).
- InterSystems IRIS system processes.

In this page, the word *process* by itself refers to both user and background processes.

18.1 Available Options

You can manage and control processes using the Management Portal as follows:

Table 18–1: Process Management Functions

Function	How to Access Function From The Portal
Display process information	Display the **Processes** page (**System Operation** > **Processes**).
Display process details	Display the **Processes** page, then click **Details** in the right hand column of the selected process to display the **Process Details** page.
Suspend/resume a process	Display the **Processes** page, then click **Details** in the right hand column of the selected process to display **Process Details** page. Then click **Suspend** or **Resume** on the operations bar, as desired.
Terminate a process	Display the **Processes** page, then click **Details** in the right hand column of the selected process to display the **Process Details** page. Then click **Terminate** or **Terminate with <RESJOB> Error** on the operations bar, as desired.
Display process variables	Display the **Processes** page, then click **Details** in the right hand column of the selected process to display the **Process Details** page. Then click the **Variables** tab to display the process variables.
Broadcast messages to terminals	Display the **Processes** page and click the **Broadcast** button to open the **Broadcast** dialog.

18.2 Display Process Information

To display all the active processes on the system and basic information about each, navigate to the **Processes** page (**System Operation** > **Processes**), which displays a table of the processes with statistics about each in columns.

The following table describes the process information available for display:

Table 18–2: Process Column Information

Column Heading	Definition
Job #	Index of the Processes table.
Process ID	Operating system process identification number (PID).*
Total CPU Time in ms	Total amount of system and user CPU time, in milliseconds, that the process took to execute.
User	Name of the user who owns the process.
Device	Current device the process is using. This can be: • `|TCP|`*IP_address*`:`*Port_number* — For an outbound connection from the instance. • `|TCP|`*Port_number* — For the superserver. • `//./nul` — The null device. This discards any output; if you attempt read from it, there will be no data.
Namespace	Namespace in which the process is running.

Column Heading	Definition
Routine	Name of the routine that the process is currently executing.
Commands	Number of commands executed.
Globals	Number of global references, including updates, executed (database reads and writes) since the process entered InterSystems IRIS.
State	Process state. See the *State* property of the %SYS.ProcessQuery class for an explanation of each state.
Client Name	Name of the client system that connected to, or initiated the connection to, the process.
Client EXE	Name of the executable that called the process.
Client IP	IP Address of the system that initiated the process.
O/S Username	Username assigned to the process by the operating system.
Details	Button appears if you have authority to maintain this process. See Display Process Details.

* An asterisk (*) appears next to the process id if the user entered InterSystems IRIS in programmer mode. A plus or minus sign appears next to Callin processes:

- + Process is in InterSystems IRIS
- – Process is not in InterSystems IRIS

The Callin API is an InterSystems IRIS facility that lets you execute and evaluate ObjectScript commands and expressions from within C programs.

18.3 Display Process Details

The **Process Details** page displays detailed information about any process. To access this information from the Management Portal:

1. Display the **Processes** page (**System Operation** > **Processes**).
2. Click **Details** in the row of the appropriate process. (This option exists only on processes that you have authority to maintain.)
3. Optionally select the **SQL table & statement info** check box. This option adds SQL-related information to the display.

The page also includes information specific to the selected process, which is organized into a General Information table, a Client Application Details table, and a Execution Details table:

Table 18–3: General Information

Field	Definition
Process ID	Process ID (PID) number of this process.
User Name	Name of the user currently logged in for this process.
Login Roles	Login roles for the process.

Field	Definition
Escalated Roles	Additional roles granted to the process. These roles plus the Login Roles is the total set of roles granted to the process.
OS User Name	Username assigned to the process by the operating system.
NameSpace	Namespace in which the process is executing.
Process Priority	Priority level of this process.
Global References	Number of global references made by this process.
Private Global References	Number of private global references made by this process.
Commands Executed	Number of commands executed by this process.
Memory Limit	Amount of memory (KB) allocated for use by this process.
Memory Peak	Peak amount of memory (KB) used by this process.
Memory Used	Amount of memory (KB) currently in use by this process.
Total CPU Time	Total amount of system and user CPU time, in milliseconds, that the process took to execute.
Private Global Blocks	Number of private global data blocks used by this process.
Current Device	Name of the I/O device currently in use by this process.
Open Devices	List of devices currently opened by this process.
Lock	Lock information for this process. Click the link at top of the detail box for additional details (mode, counts, and full reference).

Table 18–4: Client Application Details

Field	Definition
Client Name	Node name of the client that is connected, or initiated the connection, to this process (if any).
EXE Name	Name of the executable client application client connected to this process (if any).
Client IP Address	IP address of the executable client application client connected to this process (if any).
Info	User-defined information (if any).

Table 18–5: Execution Details

Field	Definition
Process State	Current execution state of this process.
In Transaction	Indicates whether or not this process is currently within a transaction.
Last Global Reference	Last global referenced by this process.
Last SQL Table Reference	Last SQL table referenced by this process, if any. (Select the **SQL table & statement info** check box in order to see this data.)
Routine	Name of the routine this process is currently executing.
Source Location	Last reported source location (routine name plus offset) of this process.

Field	Definition
Source Line	Last reported line of source code executed by this process, if available.

Stopping a Process

From this page you can also stop or resume a process. You can stop a process in one of the following ways:

- Suspend a process with the intention of resuming the process later.
- Terminate a process, which entirely cancels the process.

18.3.1 Suspend or Resume a Process

You may want to suspend a process if you are not sure what it is doing and want to investigate, or if a more important process is trying to run and needs the CPU cycles. To access this option from the Management Portal:

1. Display the **Processes** page (**System Operation** > **Processes**).
2. Click **Details** in the row of the appropriate process. This option only exists on processes that you have authority to maintain.
3. Click **Suspend** on the options bar.

You may resume a suspended process at any time by clicking **Resume** from the same page.

18.3.2 Terminate a Process

You may want to terminate a process if it becomes unresponsive or is affecting other processes or users. To access this option from the Management Portal:

1. Display the **Processes** page (**System Operation** > **Processes**).
2. Click **Details** in the row of the appropriate process. (This option exists only on processes that you have authority to maintain. The portal displays the **Process Details** page for the process you selected.
3. Click **Terminate** on the options bar.

   Optionally, to log the status of the process when it terminates, select the **Terminate with RESJOB Error** check box.

   Note: This option is enabled by default.

4. Click **Yes** to confirm that you want to terminate the process. There is no way to resume a terminated process.

18.3.3 Display Process Variables

The **Process Variables** page displays all the variables used in the selected process giving the global name and the value of the global. To access this information from the Management Portal:

1. Display the **Processes** page (**System Operation** > **Processes**) page.
2. Click **Details** in the row of the appropriate process. (This option exists only on processes that you have authority to maintain.) The portal displays the **Process Details** page for the process you selected.
3. Click **Variables** on the options bar.

If you have selected the **SQL table & statement info** check box, this table includes a column via which you can directly view the cached squery, if any.

18.4 Broadcast Messages to Terminals

You can broadcast messages to the terminals associated with a selected process or all processes; this utility is useful, for example, to ask people to sign off the system. However, you must use it carefully or you may cause messages to appear in the middle of reports that may be printing at the time.

The utility temporarily takes control of each terminal as it sends the message. Once the terminal receives the message, the previous process continues. The message appears on the terminal screen; it may disrupt the screen display, but it does not affect user input. The message does not appear in windows running ObjectScript utilities.

To broadcast a message to the terminals associated with a selected process, do the following in the Management Portal:

1. Display the **Processes** page (**System Operation** > **Processes**).
2. Click **Broadcast** (on the options bar) to open the **Broadcast** window.
3. Enter the message to broadcast in the text box.

   (The dialog box notifies you if there are no active processes that can accept a message; you do not see a message text box or list of processes. Click **Close**.)
4. Select the appropriate check boxes for the appropriate processes (PIDs) to receive the broadcast message. Use the **Select All** and **Clear All** buttons accordingly to help with the selection.
5. Click **Broadcast**.
6. After the completed message displays, click **Close**.

19

Services

There are various pathways for connecting to an InterSystems IRIS® instance for users, applications, and even other InterSystems IRIS instances. These pathways are managed by InterSystems *services*, which serve as gatekeepers for connecting to InterSystems IRIS. Because InterSystems services are the primary means by which entities such as users and computers connect to InterSystems IRIS, their management is an essential part of security administration.

19.1 Available Services

The Services page (**System Administration** > **Security** > **Services**) provides a list of services that InterSystems IRIS provides.

There are two groups of services:

- Resource-based Services — These are services that provide user access to InterSystems IRIS. This kind of service needs the authentication and authorization infrastructure of InterSystems security, so it has an associated *resource* and uses the various available authentication mechanisms.
- Basic Services — These are services that provide connections between an InterSystems IRIS server and an InterSystems IRIS application. These do not have associated resources, so they provide little more than the *basic* security functionality of being turned on or off. Enabling or disabling them controls all forms of access.

The following lists the available services, what each controls, and what kind of service it is:

- **`%Service_Bindings`** — SQL or objects; use of Studio [resource-based]
- **`%Service_CacheDirect`** — A proprietary mechanism for connecting to other InterSystems products [resource-based]
- **`%Service_CallIn`** — The CallIn interface [resource-based]
- **`%Service_ComPort`** — COMM ports attached to a Windows system [resource-based]
- **`%Service_Console`** — The local Terminal from a Windows console (analogous to **`%Service_Terminal`** for macOS, UNIX®, and Linux) [resource-based]
- **`%Service_DataCheck`** — The DataCheck utility [basic]
- **`%Service_DocDB`** — Document database applications [resource-based]
- **`%Service_ECP`** — Enterprise Cache Protocol (ECP) [basic]
- **`%Service_Login`** — Use of **$SYSTEM.Security.Login** [resource-based]
- **`%Service_Mirror`** — InterSystems IRIS database mirroring [basic]

- **`%Service_Monitor`** — SNMP and remote monitor commands [basic]
- **`%Service_Shadow`** — Access to this instance from shadow destinations (for use only with existing configurations) [basic]
- **`%Service_Sharding`** — Access to this instance as a shard server [basic]
- **`%Service_Telnet`** — Telnet sessions on a Windows server and remote Windows Terminal sessions [resource-based]
- **`%Service_Terminal`** — The Terminal from a macOS, UNIX®, and Linux console (analogous to **`%Service_Console`** for Windows) [resource-based]
- **`%Service_WebGateway`** — Web application pages [resource-based]
- **`%Service_Weblink`** — WebLink, which is available as a legacy service [basic]

The table of services includes a column for each service property.

19.1.1 Notes on Individual Services

19.1.1.1 %Service_Bindings

For the **`%Service_Bindings`** service, there are three resources that manage access: the **`%Service_Native`** resource, the **`%Service_Object`** resource and the **`%Service_SQL`** resource. Once a user has authenticated, these resources control whether data is accessible to the user through the Native SDKs, as objects, or through SQL respectively. (If a user has table-level SQL privileges on data, then InterSystems IRIS automatically grants an authenticated user the **`%Service_SQL:Use`** privilege for the duration of the connection.)

This service also controls access to Studio. For more information about Studio and security, see Integration with InterSystems Security.

19.1.1.2 %Service_Console and %Service_Terminal

These two services both provide console or terminal-style access to InterSystems IRIS. This functionality is analogous for both Windows and non-Windows systems; **`%Service_Console`** provides this functionality for Windows and **`%Service_Terminal`** provides this functionality for UNIX®, Linux, and Mac.

CAUTION: Terminal or console access is one of the most sensitive aspects of InterSystems security. If an attacker gains access to InterSystems IRIS in one of these ways, it can be possible to read or destroy sensitive data.

19.1.1.3 %Service_DataCheck

This service regulates the use of the DataCheck utility, which provides a mechanism to compare the state of data on two systems. For more details, see Data Consistency on Multiple Systems, and, for security issues, particularly Enabling the DataCheck Service.

19.1.1.4 %Service_ECP

A resource does not govern the use of ECP. Rather, you either enable or disable the service (this makes ECP what is called a “basic service”). This means that all the instances in an ECP configuration, such as a distributed cache cluster, need to be within the secured InterSystems IRIS perimeter.

For details on how privileges work within an ECP-based configuration, see Distributed Cache Cluster Security.

19.1.1.5 %Service_Login

This service controls the ability to explicitly invoke the **Login** method of the %SYSTEM.Security class. Calls to this method are of the form:

ObjectScript

```
 Set Success = $SYSTEM.Security.Login(username, password)
```

where *username* is the user being logged in and *password* is that user’s password.

19.1.1.6 %Service_Mirror

This service regulates the use of InterSystems IRIS database mirroring. For more details about mirroring generally, see Mirroring; for more details about security for mirroring (though the use of TLS), see Configuring InterSystems IRIS to Use TLS with Mirroring.

19.1.1.7 %Service_Sharding

This service regulates the use of an InterSystems IRIS instance as a shard data server. For more details, see Horizontally Scaling for Data Volume with Sharding.

19.1.1.8 %Service_WebGateway

This service manages connections that serve up web pages. Specifically, it manages connections between the web gateway processes running on the web server and the InterSystems IRIS server. You will not interact with this service directly within a web application; instead, the authentication mechanisms are configured within the relevant web application definition.

Under the following circumstances, there is no access to the server via the web gateway:

1. There are authentication mechanisms enabled for the service
2. The web gateway has no valid authentication information for any of the enabled authentication mechanisms
3. Unauthenticated access is disabled for the service

Hence, if you disable unauthenticated access through this service (that is, the Unauthenticated authentication mechanism is disabled), you must ensure that the web gateway has the information it needs to authenticate to the InterSystems IRIS server. For example, for Instance Authentication (password) access, this is a valid username-password pair; for Kerberos access, this is a valid service principal name and key table location. To specify authentication information for the web gateway, use its management interface; for a standard installation, the URL for this is http://localhost:52773/csp/bin/systems/module.cxw, where localhost represents 127.0.0.1 for IPv4 and ::1 for IPv6.

`%Service_WebGateway` controls only the background authentication between the web gateway processes running on the web server and the InterSystems IRIS instance. As a result, authentication mechanisms for web applications are configured and managed within the relevant web application definition, not by editing the allowed authentication methods of %Service_WebGateway itself.

Because **`%Service_WebGateway`** regulates the use of the Portal and its subapplications, disabling **`%Service_WebGateway`** does not disable any system applications, so that there can always be access to the Portal. For more information on system applications, see Built-In Applications.

Important: If you inadvertently lock yourself out of the Portal, you can use emergency access emergency access mode to reach the Portal and correct the problem; this is described in Emergency Access.

19.2 Service Properties

Each service has a set of properties that control its behavior. These can include:

- **Service Name** — Specifies the identifier for the service.
- **Description** — Provides an optional description of the service.
- **Service Enabled** — Controls whether a service is on or off. When enabled, a service allows connections to InterSystems IRIS, subject to user authentication and authorization; when disabled, a service does not permit any connections to InterSystems IRIS.

  At system start up, each service has the same state (enabled or disabled) that it had when InterSystems IRIS was shut down. Note that enabling or disabling a service is not simply a security setting. It determines whether or not a certain capability is provided by InterSystems IRIS and may, for instance, determine whether certain daemon processes are started or memory structures are allocated.

- **Allowed Authentication Methods** — Specifies the available authentication mechanisms for connecting to the service, including either of the two-factor authentication mechanisms; if multiple mechanisms are selected, the user or client can attempt to connect using any of these. The mechanisms available depend on what is selected on the **Authentication/Web Session Options** page (**System Administration** > **Security** > **System Security** > **Authentication/Web Session Options**). If a service supports multiple authentication mechanisms, these are used according to the InterSystems IRIS rules of cascading authentication.

  If either two-factor authentication mechanism is enabled, it has a check box. If visible, these are:

  - **Two-factor Time-based One-time Password** — An InterSystems IRIS user's mobile phone or an authentication device serves as a second authentication "factor"; InterSystems IRIS and the phone or device share a secret key. This key is used to generate a time-based one-time password (TOTP), which the user must enter at a prompt as part of the authentication process.
  - **Two-factor SMS** — An InterSystems IRIS user's mobile phone serves as a second authentication "factor"; InterSystems IRIS sends a eight-digit security token to the phone, which the user must enter at a prompt as part of the authentication process.

  For more details, see Two-Factor Authentication.

  Note: If two-factor authentication is enabled for an instance, this check box appears on the **Edit Service** page for all its services. However, two-factor authentication is only available for **`%Service_Bindings`**, **`%Service_Console`**, and **`%Service_Terminal`** (and only when it is enabled for the instance).

- **Allowed Incoming Connections** — Specifies a list of IP addresses or machine names from which the service accepts connections; if a service has no associated addresses or machine names, then it accepts connections from any machine. This capability can be very useful with multitier configurations; for example, with the web gateway service, it can be used to limit the set of web servers that can connect to InterSystems IRIS. The Allowed Incoming Connections facility for distributed cache cluster data servers has additional features, as described in Distributed Cache Cluster Security.

For a resource-based service, the service can be specified as public. Public services are available to all authenticated users, while non-public services are available only to those users with Use permission on the service's resource. This value is displayed on the main Services page (**System Administration** > **Security** > **Services**) and is set on the **Edit Resource** page for the service's resource. Possible values are:

- N/A — The service has no associated resource; this means that service can simply be turned on or off.
- NO — Access is available to any user holding a role that has the Use permission on the service's resource. This is checked after authentication.

- YES — Access is available to any user.

Note: A change to a service only takes effect after the service is restarted.

19.3 Services and Authentication

Basic services do not support authentication for InterSystems security. They are simply turned on and off. For those services, enabling the service ensures that it accepts all connections. For these services, the assumption is made that all instances or machines using the service are within a secure perimeter and can only be accessed by valid users. This includes **`%Service_ECP`**, **`%Service_Monitor`**, **`%Service_Shadow`**, and **`%Service_Weblink`**.

To enable an authentication mechanism for a resource-based service, you must first enable it for the InterSystems IRIS instance on the **Authentication/Web Session Options** page (**System Administration** > **Security** > **System Security** > **Authentication/Web Session Options**). Resource-based services support authentication mechanisms as listed in the table below. If a service has more than one authentication mechanism enabled, InterSystems IRIS supports cascading authentication.

Table 19–1: Services with Authentication Mechanisms

Service Name	KRB Cache	KRB Login	Del	LDAP	OS	IA	Un
`%Service_Bindings`	N	Y	Y	Y	N	Y	Y
`%Service_CallIn`	N	N	Y	Y	Y	N	Y
`%Service_ComPort`	N	N	Y	Y	N	Y	Y
`%Service_Console`	Y	Y	Y	Y	Y	Y	Y
`%Service_Login`	N	N	Y	Y	Y	Y	Y
`%Service_Telnet`	N	Y	Y	Y	N	Y	Y
`%Service_Terminal`	Y	Y	Y	Y	Y	Y	Y
`%Service_WebGateway`	N	Y	Y	Y	N	Y	Y

Key:

- KRB Cache — Kerberos Cache
- KRB Login — Kerberos Login
- Del — Delegated authentication
- LDAP — LDAP authentication
- OS — Operating system–based authentication
- IA — Instance Authentication
- Un — Unauthenticated access

For each resource-based service, if there are multiple enabled authentication mechanisms, then InterSystems IRIS attempts to authenticate users going from the strongest enabled form of authentication to allowing unauthenticated access (if that is enabled). This is process is described in Cascading Authentication.

19.4 Services and Their Resources

For resource-based services, the properties of the service itself govern access to InterSystems IRIS; at the same time, the properties of the service's resource govern access to and behavior of the service. For all resource-based services except **`%Service_Bindings`**, the service's associated resource has the same name as the service itself; hence the **`%Service_WebGateway`** resource manages access for the **`%Service_WebGateway`** service. (The **`%Service_SQL`** and **`%Service_Object`** resources manage access for **`%Service_Bindings`**.)

A resource itself has only two related properties: whether or not it is public and, if it is public, what its public permissions are; for a service resource, the only relevant permission is Use. If it is public, then all users have Use permission on the service. For more information on resources, see Resources.

Independent of privileges for other resources, service privileges provide little to the user.

20

Using the Task Manager

To access the Task Manager, navigate to **System Operation** > **Task Manager**.

You can also interact with the Task Manager using the Terminal, as described in Using ^TASKMGR. See Tasks for other methods of managing tasks.

Note: The Task Manager polls every 60 seconds to see if there are any Tasks to be run. When you click **Perform Action Now** to schedule a Task, there may be a delay of up to 60 seconds before the newly scheduled Task actually runs.

20.1 New Task

The **New Task** option starts the **Task Scheduler Wizard**. This tool allows you to schedule a new task to run.

Note: You can also use the **Task Scheduler Wizard** to edit an existing task by selecting the existing task from the Task Schedule page.

1. On the first page of the **Task Scheduler Wizard**, provide the following details about the new task:
   - **Task name**
   - **Description**
   - **Namespace to run task in** — Choose from the list of defined namespaces in which to run the task
   - **Task type** — Choose from among the listed tasks. For details about the available tasks, see Task Types below.

     Note: Depending on the task type selected, you may be presented with a form in which to specify additional information; for example, if you are scheduling an `IntegrityCheck`, the form prompts you for **Directory**, **Filename**, and **KeepDays** (number of days to keep the file).

2. **Task priority** — Choose from **Priority Normal**, **Priority Low**, or **Priority High**. For more information about priority, see Priority.
3. **Run task as this user** — Choose from the list of defined users. To choose a different user than the one you are logged in as, you must have the **`%Admin_Secure:Use`** privilege.

> **Note:** If the chosen user is disabled, the task is suspended until the user is enabled and the task is resumed manually. This does not apply to built-in system tasks, which run even when the chosen user is disabled.
>
> Each task consumes a license unit, with the license user ID based on the InterSystems IRIS® data platform username you select; see License Login Special Considerations for more information.

4. **Open output file when task is running** and **Output file** — If the task creates output, the log file is written to the directory specified.
5. **Suspend task on error?** — Specify whether the task will be rescheduled and continue to run after an error, or suspended. The default is **No**, to reschedule and run the task after an error.
6. **Reschedule task after system restart?** — Specify whether you want to reschedule the task when the system restarts (that is, if the system is down when the task is scheduled to run):
   - **No** specifies that the task should run when the system restarts.
   - **Yes** specifies that the task should be rescheduled for the next logical time after the system restarts.
7. **Send completion email notification to** — If you have configured email settings for the Task Manager (see Configuring Task Manager Email Settings), enter a comma-separated list of email addresses to which a notification should be sent when the task ends successfully.
8. **Send error email notification to** — If you have configured email settings for the Task Manager, enter a comma-separated list of email addresses to which a notification should be sent when the task ends in error.
9. **How should task run for Mirror** — If this instance is a mirror member, specify the type of member the task can be run on:
   - primary failover member only
   - backup failover member and async members only (all except primary)
   - all mirror members (primary, backup, and asyncs)

   > **Important:** For a task to run on a mirror member, this option must be set for all tasks on the mirror member. Adding an instance to a mirror member does not automatically update this option in user-defined tasks. As such, you must either:
   >
   > - Define this setting when you create a task even if the instance is not a mirror, so it can run if the instance is added to a mirror.
   > - Make sure you review all user-defined tasks when an instance is added to a mirror and set **How should task run for mirror**.

   > **Note:** When the status of a failover member is in transition, for example when the backup is in the process of taking over as primary, the Task Manager does not run any tasks on that member until its status as primary or backup is established.

   After you have filled in all the necessary fields, click **Next**.
10. On the second page of the **Task Scheduler Wizard**, specify when the new task should run. The options are:
    - **Daily** — A daily interval (for example, every other day, or every third day).
    - **Weekly** — A weekly interval on specific days of the week.
    - **Monthly** — A monthly interval on a specific day of the month.

      > **Note:** If the chosen date does not exist for a particular month, the closest existing date in that month is used.

- **Monthly (by day)** — A monthly interval on a specified day of the week (for example, first Monday, or third Wednesday).
- **After another task completes** — After a specified task runs.
- **On demand** — Only when manually executed.

The **Daily**, **Weekly**, **Monthly**, and **Monthly (by day)** fields allow you to specify what time during the day the task should run:

- **Start Date** — The first day the scheduled task should run.
- **End Date** — The last day the scheduled task should run.
- **Task execution details**, which specify what happens when a task executes as follows:
  - **Run once at this time** — At the specified time, the task executes once.
  - **Run every** ____ — During the specified time range, the task executes repeatedly at the specified interval.

Note: By default, all new tasks expire if they miss their scheduled time for any reason. If you wish to change this behavior, see Changing Task Expiration Behavior below.

After you have filled in all the necessary fields, click **Finish** to schedule the task.

20.2 Task Types

This section describes the predefined task types you can choose when creating or editing a task. Some tasks are only available from certain namespaces.

InterSystems IRIS includes a number of tasks that run by default. This table notes these defaults, which you may edit from the Task Schedule page.

Note: You can define your own task types using the %SYS.Task.Definition API; for details, see the class documentation in the *InterSystems Class Reference*.

Task Type	Corresponding Default Task	Description
CheckLogging	**Check Logging activity** — runs nightly; sends an alert after two days.	Checks that logging mechanisms (such as ^ISCSOAP) have not been left running unintentionally. Creates an alert after *DaysForAlert* days, and shuts off logging after *DaysForShutoff* days.
CumuIncrDBList	None	Runs a cumulative backup of databases in the defined list.
DiagnosticReport	**Diagnostic Report** — runs on demand.	Collects and delivers diagnostic reports to the WRC.
FullAllDatabases	None	Runs a full backup of all databases.
FullDBList	None	Runs a full backup of databases in the defined list.

Task Type	Corresponding Default Task	Description
IncrementalDBList	None	Runs an incremental backup of databases in the defined list.
IntegrityCheck	**Integrity Check** — disabled by default.	Runs an integrity check of databases in the namespace. The Integrity Check task only appears when the %SYS namespace is selected.
InventoryScan	**Inventory Scan** — runs on demand.	Compiles an inventory scan of the instance.
PurgeAudit	**Purge Audit Database** — runs after the **Switch Journal** task.	Purges the audit database after a specified time following a journal switch.
PurgeBackupLog	**Purge Backup Log** — runs nightly.	Purges the backup log after a specified time following a database backup.
PurgeErrorsAndLogs	**Purge errors and log files** — runs nightly.	Purges error globals and messages.log based on settings in the iris.cpf file (see ErrorPurge and MaxConsoleLogSize parameters in the [Startup] section of the *Configuration Parameter File Reference*). **Note:** When System Monitor is running, it monitors and limits the size of the messages log.
PurgeJournal	**Purge Journal** — runs nightly.	Purges journal files that meet the purging criteria.
PurgeTaskHistory	**Purge Tasks** — runs nightly.	Purges Task Manager history files.

Task Type	Corresponding Default Task	Description
RunLegacyTask	None	Runs a legacy task (i.e. your own programmatic task). The line you enter in the text box (for example, **do ^MyCode**) must be executable in the Terminal. **Note:** Do not use the local variable *Status* in `RunLegacyTask` code. InterSystems IRIS uses this variable, and if it is modified by `RunLegacyTask` code, the task is likely to end with an error status. A legacy task that fails to complete due to an error is marked **Suspended due to error** and is not scheduled to run until the error is corrected and you resume the task.
SecurityScan	**Security Scan** — runs nightly.	Disables expired user accounts and expires user passwords and web session tokens.
SwitchJournal	**Switch Journal** — runs nightly.	Switches to a new journal file.

20.3 On-demand Task

The **On-demand Task** page (**System Operation** > **Task Manager** > **On-demand Task**) lists the tasks you have scheduled as *on-demand*. The list includes the task name, a description, and an option to **Run** the task from this page. You can sort the information in the table by clicking any column heading. When you click **Run**, the **Run Task Wizard** page displays the task name and ID, and the date and time the task will run; click **Perform Action Now** to confirm the information and schedule the task.

20.4 Upcoming Tasks

The **Upcoming Tasks** page (**System Operation** > **Task Manager** > **Upcoming Tasks**) lists the tasks scheduled to run within a certain interval. To select an interval, click an option in the **Scheduled to run:** search pane to the left of the task list. If you select the **To a date** option, you can either enter a date in `yyyy-mm-dd` format or click the calendar icon to select a date from the calendar.

You can sort the information in the task list by clicking any column heading. You can **Suspend** or **Resume** the scheduling of each task by clicking the appropriate option:

- **Suspend** — Lets you suspend the task; a **Do you want to reschedule task when task is supposed to run?** drop-down list lets you specify:
  - **No.** Upon resuming the task, the Task Manager attempts to run missed instances of the task before returning to the normal schedule.
  - **Yes.** Upon resuming the task, the Task Manager returns to the normal schedule without attempting to run missed instances.
- **Resume** — Lets you resume a suspended task.

20.5 Task Schedule

The **Task Schedule** page (**System Operation** > **Task Manager** > **Task Schedule**) lists all scheduled tasks. You can sort the information in the table by clicking any column heading. You can view **Details** or **History**, as well as **Run**, a scheduled task by clicking the appropriate option:

- **Task Name** — Lets you view details about the task and perform operations on it.
- **History** — Displays the task's history.
- **Run** — Lets you run the task. A **Run Task** wizard displays the task name and ID, and the date and time the task will run; click **Perform Action Now** to confirm the information and schedule the task.

20.5.1 Task Details

To display detailed information about a scheduled task and perform one of several operations on it, click the task's name in the **Task Name** column. The **Task Details** page displays information and execution details about the selected task. You can perform one of the following operations on the task by clicking the appropriate button:

- **Edit** — Change the task definition and schedule using the **Task Scheduler Wizard**.
- **History** — View the task's history.
- **Resume**/**Suspend** — Suspend a task/resume a suspended task, as on the Upcoming Tasks page.
- **Delete** — Delete the task permanently.

  Note: You cannot delete a system task.

- **Export** — Export a task to a file that can later be imported, including by another InterSystems IRIS instance.
- **Run** — Schedule the task. When you click **Run**, the **Run Task Wizard** page displays the task name and ID, and the date and time the task will run; click **Perform Action Now** to confirm the information and schedule the task.

Note: Some of the actions described are unavailable while a task is running.

20.5.2 Scheduled Task History

To display history information about an individual scheduled task, click the **History** link in the row of the item. The **Task History** page displays detailed history for the selected task. The **Result** column indicates the outcome the last time the task was run, showing either **Success** or an error message. You can sort the information in the table by clicking any column heading.

The **Details** link at the top of the page displays the **Task Details** page for the selected task.

20.6 Task History

The **Task History** page (**System Operation** > **Task Manager** > **Task History**) lists the history of all tasks executed by the Task Manager.

You can sort the information in the table by clicking any column heading. To filter out system tasks, select the **Show only user-defined task types** check box at the top of the table.

20.7 Import Tasks

The **Import Tasks** page (**System Operation** > **Task Manager** > **Import Tasks**) lets you import and run a task by browsing to a previously-exported task file, then clicking **Perform Action Now**. For information about exporting tasks to a file, see Task Details in this section.

Note: Tasks can only be imported from, or exported to, instances running the same version of InterSystems IRIS.

20.8 Using ^TASKMGR

The **^TASKMGR** routine allows you to configure the Task Manager using the Terminal. Except when noted otherwise, **^TASKMGR** and the Management Portal contain the same options for configuring tasks; the one you use is a matter of preference.

To use **^TASKMGR**:

1. Open the Terminal.
2. Enter `set $namespace = "%SYS"` to change to the %SYS namespace.
3. Enter `do ^TASKMGR`.

For more details about how to schedule or edit a task programmatically, see the %SYS.Task class documentation in the *InterSystems Class Reference*.

20.8.1 Changing Task Expiration Behavior

By default, all tasks are configured to expire if they miss their scheduled time. This could happen for a number of reasons; for example, when InterSystems IRIS is down during the scheduled time, or if the previous run of the task extends beyond the next scheduled time. When a task expires, it does not run until the next scheduled time.

The **^TASKMGR** routine (but not the Management Portal) contains the option to change this behavior, such that a scheduled run is never skipped. To do so:

1. Open the Terminal.
2. Enter `set $namespace = "%SYS"` to change to the %SYS namespace.
3. Enter `do ^TASKMGR`.

4. Select option `2.`
5. Enter the task number you want to edit.
6. Press Enter to save the defaults of each option until you reach `Task Expires?`
7. Enter `No`.

Alternatively, you can specify a grace period, such that the task does not expire immediately after it is missed. Instead, the Task Manager runs the missed task as soon as it is able. To adjust this grace period of when a specific task expires:

1. Open the Terminal.
2. Enter `set $namespace = "%SYS"` to change to the %SYS namespace.
3. Enter `do ^TASKMGR`.
4. Select option `2.`
5. Enter the task number you want to edit.
6. Press Enter to save the defaults of each option until you reach `Task Expires?`
7. Enter `Yes`.
8. At the `Expires in how many days?` prompt, enter the number of days before the task should expire.
9. At the `Expires in how many hours?` prompt, enter the number of hours before the task should expire.
10. At the `Expires in how many minutes?` prompt, enter the number of minutes before the task should expire.

20.9 Custom Tasks

To define custom tasks, define a subclass of %SYS.Task.Definition. In this subclass, implement the **OnTask()** method which should perform the required work and return a %Status value.

You can require a specific privilege to execute this task. To do so, specify the *RESOURCE* class parameter in this class. Specify the value in the form *resource:permission*. For example:

Class Member

```
Parameter RESOURCE As String = "%Admin_Manage:USE";
```

For additional information, see the class reference.

After you compile this class, it is available to be added to the Task Manager schedule as described on this page. It is listed there as a User task (as opposed to a System task).

21

Using the Background Tasks Page

A background task is an asynchronous job process that runs in the background, independently of the process that created it. A background task is created when a user issues an ObjectScript JOB command, or by the Management Portal or a utility to execute a job without requiring the user to wait for completion. For example, when you use the Portal to truncate a database, as described in Truncating a Database, a background task is started.

The **Background Tasks** page (**System Operation** > **Background Tasks**) lists past and active background tasks. You can purge the log of past background tasks at any time.

When a background task is active, the process can also be seen on the **Processes** page (**System Operation** > **Processes**), as described in Controlling InterSystems IRIS Processes.

22

Managing InterSystems IRIS Licensing

This topic contains an overview of the InterSystems IRIS® data platform license system and describes how to manage and monitor licenses.

Important: *InterSystems Terms and Conditions* govern how you may use the licensed InterSystems IRIS software. Occasionally, the implementation may be more lenient. Verify that any license-related code you write conforms to these terms and conditions.

If you use license servers, there are important network considerations to take into account.

22.1 Introduction to License Management

Each InterSystems IRIS instance maintains an independent local view of its license capacity and current use, and each instance requires access to a license key (except evaluation installations). To manage these licenses, there are several possibilities, depending on the number of instances you have:

- You can install and activate a local license key file on each instance.
- You can configure a *license server* to manage key files stored in a central location, which it can then distribute to other instances. In this case, each instance must be configured with the LicenseID of the key file, so that it can retrieve a copy of the key at startup.

  The license server coordinates the views of license use maintained locally in every instance. The license server is not an InterSystems IRIS process; it is unaffected if an InterSystems IRIS instance shuts down. One license server can handle multiple instances. Therefore, you need at most one per host regardless of how many InterSystems IRIS instances run on a host. However, each InterSystems IRIS instance must have a local copy of the authorizing license key file installed. Also, because a license server is started by a running instance, it should be configured to run on systems where you expect an InterSystems IRIS instance to be running consistently.

  License servers can run on any computer where you run an InterSystems IRIS instance. A monitor process (the *license monitor*) sends update messages to the license server, which coordinates license allocation when more than one instance shares a license.
- You can configure more than one license server (typically on different hosts) to provide redundancy. The license software selects one of the license server to be the active server. The other servers are available to take over should the active server fail. When configuring license server, decide which machines to use to host the license server. You can configure it to run on as many hosts as you want, but more than three is excessive.

Multiserver licenses can be shared among cooperating instances, either on the same machine or on different machines. Sharing is permitted only with multiserver keys. To use your multiserver licenses, you must configure one or more InterSystems IRIS license servers to allocate the InterSystems IRIS license units authorized by the key. All instances sharing a key must be configured to use the same license server or set of license servers.

Multiple instances with different license keys and running on different platforms *can* use the same license server to coordinate licensing *as long as* each instance has its own copy of the proper iris.key file *and* all instances authorized by the same key use the same license servers. However license units are not summed across license keys. InterSystems IRIS instances using different license keys do not share license units, and users logged in to two instances using different license keys consume a separate license unit from each key.

22.2 License Servers and Network Considerations

If you use license servers, bear in mind the following points:

- Each license monitor queries a license server via a TCP port (using %SYSTEM.License). Thus it is necessary to make sure that TCP traffic to the license server is permitted.

  Important: If a remote license server is protected by a firewall, the license server port must be open for UDP traffic.

- As part of the built-in redundancy for the license server, the License Monitor on an IRIS Instance checks periodically to make sure it's connected to the active license server. If it finds an IP address different than the one it's currently connected to, it assumes the license server has changed and enters a restart mode which resends all current license usage, resulting in warning messages. To avoid this problem, make sure to specify the location of the license server as a stable, consistent IP address.

22.3 Configuring a License Server

You can add or delete license servers from the **License Servers** page (**System Administration** > **Licensing** > **License Servers**) in the Management Portal. The page displays a list of license servers configured for this instance. When there are multiple license servers configured for this instance, the row of the active license server is shaded.

Note: You can also determine which license server is active using the **$System.License.ShowServer()** method.

Click on the name of a listed license server to update its information, or click **Delete** to remove it. To add a new license server:

1. Click **Create License Server** to configure a license server. The **Create New License Server** box appears on the right edge of the screen.
2. Enter a name for the license server in the **Name** box. The license server name identifies the license server in the configuration and must be unique to a configuration.
3. Enter the IP address to host the license server in the **Hostname/IP Address** box. You can enter the IP address in dotted decimal format (`192.29.233.19`) or in alphabetic format (*mycomputer.myorg.com*). If IPv6 is enabled, you can enter a colon separated format IPv6 address (`2001:fecd:ba23:cd1f:dcb1:1010:9234:4085`).
4. Enter the port number used by the license server in the **Port** box. The license server port number must be a number between `1024` and `65535`; InterSystems IRIS uses a default port number of `4002`. Redundant license servers running

on different hosts do not need to use unique port numbers, but must use port numbers that are not already in use at that IP address.

5. (Optional) Enter the central directory where license keys are stored in the **KeyDirectory** box. For more information, see Loading Keys to the License Server.
6. Click **Save** to create the license server.

After adding or deleting a license server, you must restart the InterSystems IRIS instance.

Note: If separate instances all configure the same license server address and port, they all use the same license server; when this is the case, you should delete the default **LOCAL** license server (127.0.0.1) on each instance. If the same key is loaded on each instance, they share the key; if different keys are loaded on each instance, the license server serves each set of instances using each key separately.

22.4 Loading Keys to a License Server

In addition to managing the capacity of shared licenses, the license server may also distribute keys stored in a central directory to InterSystems IRIS instances.

A KeyDirectory property is defined as part of Config.LicenseServers. If this is filled in, the instance which starts the license server reads any valid *.key files found there at startup and sends them to the license server. Each key file must have a unique LicenseID property. The instance logs a count of files successfully loaded and any errors. You may manually reload key files from the directory to update any licenses by calling **ReloadKeys^%SYS.LICENSE** from the %SYS namespace.

Note: Loading a new key with the same LicenseID as an existing key in the license server key tables marks the existing key as replaced. Requests from instances for that LicenseID receive the most recently loaded key. You can use the existing **$System.License.DumpKeys()** method to view the current state of keys in the license server.

22.5 Activating a License Key

InterSystems IRIS uses license keys to ensure proper operation of its registered sites, to define capacity available and to control access to InterSystems IRIS features. (License keys are not required for evaluation installations.) A license key is provided in the form of a license key file, typically named iris.key.

After installing InterSystems IRIS, use the following procedure to activate your license key. You can always use the same procedure to activate a new license key (that is, upgrade the key) for any installed instance. You can activate a license key placed in any location accessible to the Management Portal; as part of activation, the license key is copied to the instance's *install-dir*/mgr directory as iris.key, if it is not named that already.

Note: On Windows, you can also select a license key during installation (see Installing InterSystems IRIS on Microsoft Windows). When you do this, the license is automatically activated and the license key is copied to the instance's *install-dir*/mgr directory as iris.key; the activation procedure described here is not required.

This section also discusses license troubleshooting and upgrading a license from the operating system command line when all license units are in use.

To activate a license key, use the following procedure:

1. Navigate to the **License Key** page (**System Administration** > **Licensing** > **License Key**). Information about the current active license key is displayed. If no license has yet been activated, this is indicated, for example by the notation **Customer Name: License missing or unreadable**. This page includes a **Print** button to let you easily print the displayed information.
2. Click **Activate License Key** and browse for the license key file you want to activate. When you select a file, information about it is displayed so you can verify that you have the right license key before activating it; for example, that it provides the desired capacity, and has the right expiration date. If the key is not valid, this is indicated in an error message. If a license is currently active, information about the current and selected licenses is displayed side by side. If a restart of the instance after activation is required for the license key to take effect, this is noted and the reason for it is provided. This dialog includes a **Print** button to let you easily print information about both the current active license and the new license key you have selected.
3. Click **Activate** to activate the new license key; it is copied to the instance's *install-dir*/mgr directory as iris.key, overwriting the previous license key (if any). A confirmation dialog reminds you to restart the instance if required, and warns you if the new license enables fewer features than the current license.

Instances can be configured to request a license key from the license server by using the LicenseID property of Config.Startup. At instance startup, if there is no iris.key file present and the LicenseID has been defined, then the instance requests and activates the license key from the license server.

Note: The same LicenseID must be in the license key file, as well as defined on the instance that needs to download a license

In general there is no need to restart the instance, but there are constraints when upgrading a license key. Automatic activation of the new key does not occur if you change license types from Power Unit to any other type; this should be a rare event.

Another constraint is the amount of memory the license upgrade consumes from the shared memory heap (*gmheap*) space. If *gmheap* space is not available, the number of license table entries cannot be expanded. If insufficient *gmheap* space is available for a license upgrade, a message is written to the messages log. You can increase the size of the gmheap setting from the **Advanced Memory Settings** page (**System Administration** > **Configuration** > **Advanced Memory Settings**).

If the new license key consumes at least 1000 64 KB pages more *gmheap* space than the existing key, the InterSystems IRIS instance must be restarted to fully activate the new license key. This situation is rarely encountered, since each page represents at least 227 licenses.

22.6 Updating a License Key

To update a license key, replace the key file in the KeyDirectory and run **ReloadKeys^%SYS.LICENSE**. The License Monitor on each instance (**^LMFMON**) checks every 30 minutes to see if there is a different key for the configured LicenseID, and if so, tries to perform an upgrade.

Note: While most upgrades succeed on a live instance, some conditions could require an instance be restarted. The License Monitor logs an error in this case, and does not try to upgrade the key again until the next day (to avoid logging repeated errors). An instance restart loads the new key at startup.

22.7 License Troubleshooting

If only one user can log in after entering your license and restarting InterSystems IRIS, use the Management Portal to investigate. The **License Usage** page (**System Operation** > **License Usage**) shows how many processes are running when you select **Usage by Process**. You can also use the Portal to display license information from the **License Key** page (**System**

Administration > **Licensing** > **License Key**), as described in Activating a License Key. If the key is not valid, the **CustomerName** field contains an explanation.

You can also check the license error messages in the messages log and System Monitor log, which can be viewed in the Portal on the **Messages Log** page (**System Operation** > **System Logs** > **Messages Log**) and **System Monitor Log** page (**System Operation** > **System Logs** > **System Monitor Log**), respectively (see Monitoring Log Files). System Monitor writes license expiration warnings and alerts to these logs, while Health Monitor writes license acquisition alerts and warnings. When the license limit is exceeded, alerts are written to the messages log by the licensing module. In Application Monitor, you can configure license metric-based alerts to send email notifications or call notification methods. See Using System Monitor for more information.

When your license is nearing its expiration date, you will get warnings in the messages log. Your license is valid until the end of the day it expires on. For example, if your license has an expiration date of November 30, 2022, it will be valid until the end of November 30, 2022, but it will no longer work on December 1, 2022.

$System.License.Help displays a list of methods you can use to troubleshoot license problems:

ObjectScript

```
 Do $System.License.Help()
```

This document describes many of these methods.

22.7.1 Administrator Terminal Session

Several problem can prevent you from obtaining a Terminal session. This can happen when InterSystems IRIS fails to start properly and enters *single user mode*, or simply when there are no licenses available. In these cases you may need to create an administrator terminal session, which uses a special license to allow you to resolve the problem.

The command to open an administrator session differs for Windows and for UNIX®, Linux, and macOS.

22.7.1.1 Administrator Session on Windows

Using the command prompt, navigate to *install-dir*\bin. Then, execute the following command as an administrator:

```
irisdb -s<install-dir>\mgr -B
```

This runs the InterSystems IRIS executable from the InterSystems IRIS installation bin directory (*install-dir*\bin), indicates the pathname of *install-dir*\mgr (using the `-s` argument), and inhibits all logins except one emergency login (using the `-B` argument).

As an example, with an instance named *MyIRIS* located in the default directory, the command would look like:

```
c:\InterSystems\MyIRIS\bin>irisdb -sc:\InterSystems\MyIRIS\mgr -B
```

22.7.1.2 Administrator Session on UNIX®, Linux, and macOS

Using the command prompt, navigate to the *install-dir*/bin directory. Then, execute the following command:

```
iris terminal <instance-name> -B
```

As an example, with an instance named *MyIRIS* installed in the default directory, the command would look like:

```
User:/InterSystems/MyIRIS/bin$ iris terminal MyIRIS -B
```

22.8 Upgrading a License from the Operating System Command Line

The **%SYSTEM.License.Upgrade()** method activates a new license key that has been copied to the *installdir*\mgr directory. If all license units are consumed by users, preventing you from opening a Terminal window, you can run this method from the command line to activate a new license key with a greater capacity, as follows:

```
iris terminal <instancename> -U %SYS '##Class(%SYSTEM.License).Upgrade()'
```

For more information on the **iris** command, see Connecting to an InterSystems IRIS Instance.

22.9 Determining License Capacity and Usage

How does one know how many licenses have been used, and by whom? The %SYSTEM.License class provides an interface to the InterSystems IRIS license application programming interface (API) and presents a number of methods and related queries that you can use to query license capacity and current use.

You can run a number of licensing queries using the RunQuery method of the %Library.%ResultSet class. For example:

ObjectScript

```
do ##class(%ResultSet).RunQuery("%SYSTEM.License","Summary")
```

You can view the output of these queries from the **License Usage** page of the Management Portal (**System Operation** > **License Usage**), as detailed in the following table:

Link on License Usage Page	License Query
Summary	**Summary()** — returns license usage summary, as displayed by **$System.License.ShowSummary**.
Usage by Process	**ProcessList()** — returns license use by the operating system process identifier (PID), as displayed by **$System.License.DumpLocalPID**.
Usage by User	**UserList()** — returns license use by User ID.
Distributed License Usage	**AllKeyConnectionList()** — returns current distributed license usage sorted by users. (This is disabled when no license server is connected.)

You can also use the following class methods from the %SYSTEM.License class to display information, or dump the license database to a file:

$System.License.CKEY displays the key. This subroutine is called by the **^CKEY** program which is retained for compatibility:

ObjectScript

```
Do $System.License.CKEY()
```

$System.License.ShowCounts summarizes license use tracked in shared memory on the local system:

ObjectScript

```
Do $System.License.ShowCounts()
```

$System.License.ShowServer displays the active license server address and port:

ObjectScript

```
Do $System.License.ShowServer()
```

If you have developed REST based applications, your licenses will be consumed with use. To prevent this from happening, configure the number of Web Gateway connections that can be made. From the Management Portal in the Web Gateway Management section:

1. Navigate to **Server Access**.
2. Select **State-less Parameters**.
3. Set the **Maximum** to a number 2 or 3 less than the license to allow for server-side logins.

Note: Depending on the server side needs of the application you will need to adjust this.

By doing this when all the available connections are busy, new requests will queue up rather than being rejected. You will not see a rejection due to license counts being exceeded. As volume grows, the response time for the client slows down. That would be the indication that you need to buy more licenses.

The next sections describe several other methods that show license information:

- Methods to Show Local License Information
- Methods to Show License Server Information

22.10 Methods to Show Local License Information

The subroutines listed below dump the contents of license tables contained locally in instance shared memory. In general, they identify the client:

$System.License.DumpLocalAll dumps all local license table entries to the all.dmp file in the current directory:

ObjectScript

```
Do $System.License.DumpLocalAll()
```

An example of the contents of the all.dmp file:

```
License Capacity = 5, Current use = 2, Units Remaining = 3

   0)  User ID = 127.0.0.1, Connections = 2, CSP Count = 0, Time active = 90
   1)  User ID = 192.9.202.81, Connections = 1, CSP Count = 0, Time active = 49
   2)  free
   3)  free
   4)  free
```

$System.License.DumpLocalInUse dumps all local license table entries in use to the inuse.dmp file in the current directory:

ObjectScript

```
Do $System.License.DumpLocalInUse()
```

An example of the contents of the inuse.dmp file:

```
License Capacity = 5, Current use = 2, Units Remaining = 3
```

$System.License.DumpLocalPID dumps local license table use by process ID to the piduse.dmp file in the current directory:

ObjectScript

```
 Do $System.License.DumpLocalPID()
```

An example of the contents of the piduse.dmp file:

```
PID     Process LID     Type    Con     MaxCon  CSPCon  LU      Active  Grace

592     System                  0       0       0       0       0       0
2816    System                  0       0       0       0       0       0
688     System                  0       0       0       0       0       0
```

22.11 Methods to Show License Server Information

The following subroutines dump the contents of license tables maintained by the license server. The output files are in the indicated directory on the host where the active license server is running.

$System.License.ShowSummary displays a summary of license information at the license server. The `Distributed license use` section presents a collective view of license use for all InterSystems IRIS instances currently supported by the license server. The `Local license use` section presents a view of license use for the single InterSystems IRIS instance in which the program is run:

ObjectScript

```
 Do $System.License.ShowSummary()
```

$System.License.DumpServer dumps the license server database information relating to the server from which you run this routine to the file dumpserver.txt on the host running the license server:

ObjectScript

```
 Do $System.License.DumpServer()
```

$System.License.DumpServers dumps the license server database information for all known servers to the file dumpservers.txt on the host running the license server:

ObjectScript

```
 Do $System.License.DumpServers()
```

$System.License.DumpKey dumps the key used by this instance and instances that share it to the file dumpkey.txt on the host running the license server:

ObjectScript

```
 Do $System.License.DumpKey()
```

$System.License.DumpKeys dumps all keys, showing the instances and clients using them to the file dumpkeys.txton the host running the license server:

ObjectScript

```
Do $System.License.DumpKeys()
```

Note: Be aware that the information displayed by the local license methods is more up-to-date than the information shown by the license server methods; the license server is only updated periodically, while the local data is real time.

It is possible to exceed the license limit temporarily because login is controlled locally, but the license server enforces the limit. Each instance permits or denies logins based on its local license table which is maintained in instance shared memory. Each instance sends periodic updates to the license server describing changes to the local license tables. If the combined license use of all instances exceeds the limit, the license server sends a negative acknowledgment to update messages from each instance.

This negative acknowledgment causes each instance to refuse new logins because no additional license units are available. A login is considered new when the license user ID of the InterSystems IRIS process attempting to start does not match the license user ID of any current process. This state persists until the combined use by all instances falls below the authorized limit, at which point the license server begins sending positive acknowledgments in response to instance updates. The individual instances then allow new logins.

22.12 Identifying Users

The InterSystems IRIS licensing system attempts to identify distinct users and to allocate one license unit per user. A user is identified by a license user ID, which can be an IP address, a username, a web session ID, or some other identifier depending on how the user connects.

Multiple processes started by or for a single user share a license unit up to the maximum number of processes per user. If the number of processes exceeds this maximum, a transition occurs and InterSystems IRIS begins allocating one license unit per process for that user ID. The system assumes that if the number of processes associated with a user ID exceeds the maximum, multiple users are accessing InterSystems IRIS through an intermediary (for example, a firewall system), so additional license units are required. (Processes started by the **Job** command are counted under the user ID invoking the command.)

Even if the number of processes under the user ID drops back under the maximum, InterSystems IRIS continues to allocate one license unit per process for that user ID. Only when all connections by the user ID are closed and there are no more processes under the user ID does license allocation reset to one unit for that user ID.

Applications that identify their users by name eliminate problems associated with using a default user ID based on client IP address, web session ID, or other connection-derived user ID.

For example, when firewall or terminal server software is used, InterSystems IRIS cannot differentiate among connecting users, so it falls back on the maximum-connection transition rule. Using mixed connections from the same client also makes it impossible to count users appropriately using automatic ID creation.

When the username serves as the license identifier, these problems disappear. The importance of accurate user identification is expected to grow as organizations implement new access and audit requirements. Using the user identity to control license compliance is a natural corollary to this trend.

This section covers the following topics:

- License Logins
- Username Licensing
- License Login Special Cases

22.12.1 License Logins

There are two modes of license login: automatic and explicit. Automatic login is the default. The licensing system attempts to identify the IP address of the client and uses it as the license user ID. This works well when clients connect directly to the server using IP. It does not work well if a firewall intervenes between the client and the server; all clients appear to have the same IP address. When a terminal server is used with the telnet protocol, automatic login cannot differentiate among users because InterSystems IRIS sees a single IP address for all terminal server ports. Since all connections originate from the same address, all connections have the same user ID. If users connect through a firewall or use the telnet transport from terminal servers, use explicit logins.

When IP is not used as the network transport, the IP address is not available for use as a license user ID. In these cases, the licensing system uses a variety of other sources as the license user ID. Batch processes started by the **at** daemon on UNIX®/Linux systems pose another special case. Such processes do not share a license unit because they are not associated with a user. For these processes, the process ID is used as the license identifier.

When you select explicit login, InterSystems IRIS does not attempt automatic user ID detection. The application must explicitly call the **$System.License.Login(UserIdentifier)** method to supply the license user ID and acquire a license.

Enable explicit login by calling the **$System.License.DeferUserIdentification([0 or 1])** function. You can make this call from the `SYSTEM` entry point in the **^%ZSTART** routine at instance startup. If the argument value is `1`, license acquisition is deferred at login, so an explicit login can be performed. If the argument value is `0`, license acquisition is automatic at process startup.

When you defer login you must call the license login method immediately. A process that has not performed a license login pauses after its first 4000 ObjectScript commands, and then every 1000 ObjectScript commands after that.

Use an explicit login for any case that automatic login does not handle. It is important to remember that, even if automatic login is configured, it is always possible to call **$System.License.Login(UserIdentifier)** to use explicit user identification for licensing purposes.

22.12.2 Username Licensing

You can use the value of *$USERNAME* to identify users for licensing. This enables more accurate counting in situations where you cannot use only the IP address to reliably identify distinct users.

You modify how you specify the license user ID using the **$SYSTEM.License.UserNameLicensing()** method of the %SYSTEM.License class. By default, InterSystems IRIS uses the client IP address to identify a user to the license tracking subsystem. If you installed InterSystems IRIS with higher than Minimal initial security settings, each process has a user ID (*$USERNAME*). You can call the **$SYSTEM.License.UserNameLicensing()** system method to make the InterSystems IRIS license subsystem use *$USERNAME* as the license user identifier.

The **$SYSTEM.License.UserNameLicensing()** method modifies the system state. You can also call it from **SYSTEM^%ZSTART** to enable username licensing at instance startup. The method has the following functions:

- **$SYSTEM.License.UserNameLicensing(1)** — enables *$USERNAME* based licensing and returns the previous state.
- **$SYSTEM.License.UserNameLicensing(0)** — disables *$USERNAME* based licensing and returns the previous state.
- **$SYSTEM.License.UserNameLicensing()** — returns the current state. May return an error if called with an argument for license types that use special login rules.

For example, the following displays whether username licensing is currently enabled or disabled:

ObjectScript

```
 Write " Username Licensing",!
 Write " 1-enabled, 0-disabled",!
 Write $SYSTEM.License.UserNameLicensing(),!
```

The following example enables, then disables username licensing:

ObjectScript

```
Set RC=$SYSTEM.License.UserNameLicensing(1)
Write RC,!
Set RC=$SYSTEM.License.UserNameLicensing(0)
Write RC
```

See the $USERNAME special variable entry in the *ObjectScript Reference* for more information.

22.12.3 License Login Special Considerations

Bear in mind the following special considerations concerning license logins:

- CSP connections are a special case for logins. InterSystems strongly recommends that CSP applications use the %CSP.Session equivalent method, **%CSP.Session.Login**, to identify a user for licensing purposes. If they do not, the web session ID is used as the license user ID. Each session consumes a license unit, which in many cases is unsuitable. For example, a user can have several browser windows open concurrently. Alternatively, a user can connect via several pathways. In this case, you can use the %CSP.Session method, **%CSP.Session.Login(username, password)** to perform an explicit license login for the session.

  Note: When a CSP session ends (from a logout or timeout) and the user has visited only one page, CSP does not immediately release the license. Instead, CSP reserves the license for that user for a *grace period* of up to 10 minutes.

- Anonymous SOAP and REST requests (that is, requests that do not require Instance Authentications) consume a license unit for minimum of 10 seconds. However, any SOAP or REST request that identifies the user requires a license because it is considered a “user request.”
- InterSystems IRIS does not distinguish background processes and count them differently. If a user process starts another process, that child process counts as one more against the user’s overall maximum limit of processes.
- Each task created using the **New Task** page (**System Operation** > **Task Manager** > **New Task**) (see Using the Task Manager) consumes a license unit, with the license user ID based on the InterSystems IRIS username specified by the **Run task as this user** selector and the loopback IP address, **127.0.0.1**, which is converted to the host IP address. This ensures that tasks running as a given user on different hosts are counted together against the maximum limit of processes for that user discussed in Identifying Users.
- Processes started by the user startup routines (**^%ZSTART** and **^ZMIRROR**) are another special case. The process running the routine has no parent process. Therefore, a login is performed for the user ID, *User Startup*, before the routine is called. Processes started by the **Job** command from the routine have this user ID. If you prefer, you can call **$System.License.Login(UserId)** from the routine to change the user ID. This procedure means that the routine can start as many as one less than *maxconn* background processes and only consume one license. If, according to the license terms and conditions, these processes should have a separate license (for example if they drive a piece of laboratory equipment that requires a separate license), you are required to call **$System.License.Login(UserId)** to obtain a license for an appropriate user ID.

22.13 Application Licensing

Application licensing enables InterSystems application partners to take advantage of the InterSystems licensing capabilities for their own licensing purposes. InterSystems IRIS manages customer application licenses just as it does its own application licenses, maintaining usage counts and acquiring and returning user licenses as needed. Application licenses consumed by a process or a web session are automatically released along with the InterSystems IRIS license consumed by the process or session when a process exits, halts or is deleted from the process table, or when a web session times out or is deleted.

An application license is simply a file in standard .ini format, or a section of such a file, containing a section header identifying the application and some number of *keyword*=*value* pairs, unique within the license, representing the features licensed. Any correctly formatted application license can be loaded into InterSystems IRIS by an application at run time.

The application licensing API includes methods and queries that enable applications to consume and return licenses on behalf of a user and programs to obtain information about application and feature licensing, including the number of licenses in use and still available.

22.13.1 Loading an Application License

Any application license can be loaded and activated by an InterSystems IRIS instance at application run time using the **$SYSTEM.License.LoadAppLicenseFile** method, which is documented in the %SYSTEM.License class reference (see Application Licensing API). An application license loaded in this manner is not associated with the active InterSystems IRIS license, but is tracked independently by the InterSystems IRIS instance.

Each application license is contained in a section beginning with `[`*AppName*`]`; the application name (*AppName*) *cannot contain a period (.)*. The remainder of the license consists of a sequence of non-repeating *keyword*=*value* pairs representing the features licensed. See the **$SYSTEM.License.LoadAppLicenseFile** method documentation for more information about the required format.

In the following sample application license, the customer uses *keyword*=*value* pairs to limit the number of licensed users for several application features and enable the Extended Lab Reports feature for all users.

```
[BestLabApplication]
Lab Users=50
Lab Administrators=2
Lab Devices=5
Extended Lab Reports=Enabled
```

An application license is not protected from tampering by InterSystems IRIS, but it can be protected by custom application code. For example, a checksum can be embedded in the keyword section and validated by the application prior to activation.

22.13.2 Application Licensing API

The %SYSTEM.License class provides the following methods for managing application licenses:

LoadAppLicenseFile()

Loads an application license from a file.

GetAppLicense()

Returns the contents of a currently loaded application license as a string of *keyword*=*value* pairs.

GetAppLicenseMaxUsed()

Retrieves the maximum number of consumed license units associated with an application license keyword since the license was loaded

GetAppNames()

Returns the names of applications currently licensed by InterSystems, InterSystems application partners, or both.

IsAppLicensed()

Indicates whether an application or feature is licensed.

ReturnApplicationLicense()

Releases a license unit for an application license and feature, including the license resource associated with the license user ID of the current process, or the session license ID in the case of CSP.

SetConnectionLimit()

Sets the maximum number of connections permitted per user on this instance. This enables a system administrator to prevent a user from consuming all available license units

ApplicationServerLogin()

Indicates whether the number of instances using a currently loaded application license exceeds the limit specified in the key.

GetAppLicenseValue()

Retrieves the value associated with an application license feature keyword.

GetAppLicenseMinimum()

Retrieves the minimum number of free license units associated with an application license feature keyword, where the value associated with the keyword is numeric.

GetAppLicenseFree()

Retrieves the number of free license units associated with an application license feature keyword, where the value associated with the keyword is numeric.

GetProcessAppsTaken()

Retrieves a **$List** of application license types taken for the Process PID or "" for none

ApplicationUserList()

Returns license and application license use by user ID.

23

Managing InterSystems IRIS on Windows

Managing an InterSystems IRIS® data platform instance on the Microsoft Windows platform is straightforward. You can perform most tasks using the Management Portal and the InterSystems IRIS launcher (also known as the cube). You can also control an InterSystems IRIS instance from a command prompt.

This topic uses *install-dir* to refer to the InterSystems IRIS installation directory—you can find the default directory in the Default Installation Directory section of the *Installation Guide*.

CAUTION: *Do not* use Windows file compression on InterSystems IRIS IRIS.DAT database files. (Files are compressed by right-clicking a file or folder in Windows Explorer and selecting **Properties**, then **Advanced**, then **Compress contents to save disk space**; once compressed a folder name or filename is rendered in blue in Windows Explorer.) If you compress a IRIS.DAT file, the instance to which it belongs will fail to start, with misleading errors.

23.1 Managing Access to the InterSystems IRIS Instance

This section discusses the following topics related to managing access to InterSystems IRIS:

- The InterSystems Service
- Restricting Access to the Installation Tree
- Changing the InterSystems Service Account

23.1.1 The InterSystems Service

All InterSystems IRIS jobs and process run from the InterSystems service, *InterSystems IRIS Controller for <instance-name>*. The permissions the InterSystems service has are determined by its associated Windows user account. When this is the local SYSTEM account, InterSystems IRIS has broad access to all files and permissions on the Windows system. To maintain a more secure and restrictive environment, you should select a Windows account for the InterSystems service that only has the needed privileges and access. See Windows User Accounts for more information.

In Normal and Locked Down installations, InterSystems IRIS creates two local user groups that grant access to the instance. When you specify a Windows user account for the InterSystems Service other than the default local SYSTEM account, InterSystems IRIS adds that Windows user account to each group. These groups are:

- **IRISServices**, which grants the privileges to start, stop, and control the InterSystems IRIS instance.

- **IRIS_Instance_***instancename*, which grants access to the installation tree—the directory in which InterSystems IRIS is installed and all its subdirectories.

Note: Regarding these groups and their privileges:

- When the **IRISServices** group is created, it is granted the `Replace a process level token` and `Adjust memory quotas for a process` privileges. Do not remove these privileges.
- The **IRISServices** and **IRIS_Instance_***instancename* groups may not grant all the permissions that InterSystems IRIS requires to perform certain actions. To ensure InterSystems IRIS has the needed access to all instance, journal, and log files that are outside the installation tree, grant the **IRIS_Instance_***instancename* group full access to these files and the directories containing them. You may also grant this group additional permissions if necessary.

Typically you select the Windows account for *InterSystems IRIS Controller for <instance-name>* during installation, as described in the Windows User Accounts. To change the service account after installation, see Changing the InterSystems Service Account below.

23.1.2 Restricting Access to the Installation Tree

By default, any authenticated Windows users can access the installation tree, which may be undesirable. To remove the Windows access control entry (ACE) for authenticated users:

1. On some Windows platforms, for example Windows 10, it may be necessary to first remove inherited permissions. To do so, use the following command

   ```
   icacls <install-dir> /inheritance:d
   ```

2. Then remove the ACE for authenticated users as follows:

   ```
   icacls <install-dir> /remove "NT AUTHORITY\Authenticated Users"
   ```

3. Finally, use the following command to confirm there are no references to “Authenticated Users”:

   ```
   icacls <install-dir>
   ```

   This command lists access permissions on the install directory.

Now only users that are administrators or in the **IRIS_Instance_***instancename* group should have access to the installation tree.

Important: If you do not do this, any user who can log in to the host Windows system can easily modify files, change settings, or disable the InterSystems IRIS instance entirely.

In some cases, you may want to give another Windows account access to the installation tree, in addition to the account used by the *InterSystems IRIS Controller for <instance-name>* service. This could, for example, include accounts running automated tasks, or accounts that log in to the Windows server directly to access InterSystems IRIS (through a local Terminal session or by invoking a custom callin executable). You can give any such account the needed access by adding it to the IRIS_Instance_*instancename* group.

23.1.3 Changing the InterSystems Service Account

Enter the following in the command line to change the Windows user account used for *InterSystems IRIS Controller for <instance-name>*, the InterSystems service:

```
<install-dir>\bin\IRISinstall.exe setserviceusername <instance-name> <username> <password>
```

This command changes the Windows user account to the one you specify. It also adds the user to the **IRISServices** and **IRIS_Instance_***instancename* groups, creating these groups if necessary. After running this command and restarting an InterSystems IRIS instance, the instance runs under the newly specified Windows user account.

23.2 InterSystems IRIS launcher

The primary InterSystems IRIS interface on Microsoft Windows platforms is the InterSystems IRIS launcher. From the InterSystems IRIS launcher, you can start all of the InterSystems IRIS configuration and management tools. You can also invoke each Launcher command from a shortcut or command line.

Correspondingly, you can initiate many of the InterSystems IRIS tools from the Windows program menu by pointing to **IRIS** folder and then to the **Start InterSystems IRIS** for the appropriate InterSystems IRIS instance name.

When you start InterSystems IRIS on a Windows-based system, the InterSystems IRIS launcher IR appears in the system tray of the taskbar.

When you click the InterSystems IRIS launcher, a menu appears with commands to use the ObjectScript utilities and programming environments.

The following table describes the commands available from the InterSystems IRIS launcher menu.

InterSystems IRIS launcher Command	Description
Getting Started	Displays links to tutorials, release notes, documentation, and other related information.
Start InterSystems IRIS	Starts the default instance specified in the square brackets after the menu item, for example **[ii2081]**. If the InterSystems IRIS server is already started, this option appears dimmed—it is unavailable. **Note:** For information about how to prevent an instance from starting automatically, see Memory and Startup Settings.
Stop InterSystems	Shuts down or restarts the local InterSystems IRIS instance. If the InterSystems IRIS server is stopped, this option appears dimmed—it is unavailable.
Studio	Creates, edits, deletes, and compiles InterSystems IRIS class definitions and routines. See Using Studio for more information.
Terminal	Invokes the command line interpreter for InterSystems IRIS. See Using the ObjectScript Shell for more information.
Management Portal	Performs common system management tasks. Creates databases and namespaces, and adjusts all InterSystems IRIS configuration settings. Displays classes, globals, and routines, and functions for managing each. Displays tables and views, perform queries and SQL management functions. See Using the Management Portal.
Documentation	Displays InterSystems IRIS online documentation.

InterSystems IRIS launcher Command	Description
Preferred Server [*server name*]	Shows a list of remote servers and maintains server connections by using the **Add/Edit** command on the submenu. The preferred server appears in brackets and has a check mark next to it in the server list. See Define a Remote Server Connection
About	Displays InterSystems IRIS version and build information.
Exit	Removes the **InterSystems IRIS launcher** icon from the system tray; this does not stop InterSystems IRIS. The Launcher reappears when the instance is rebooted.

23.3 Starting InterSystems IRIS

To start InterSystems IRIS, run the startup procedure at the system level. This procedure runs using either the default configuration file or a configuration file you specify.

Note: If you have any trouble starting InterSystems IRIS, view the messages.log file as described in Monitoring Log Files.

To start InterSystems IRIS on the Windows platform, select **Start InterSystems IRIS** from the InterSystems IRIS launcher. This starts the InterSystems IRIS instance using the specified configuration file. When InterSystems IRIS is not running, the **InterSystems IRIS launcher** icon appears dimmed.

If the InterSystems IRIS launcher is not in the system tray, from the Windows program menu select the **IRIS** folder and **Start InterSystems IRIS** for that instance. To return the Launcher to the system tray, go to the *install-dir*/bin directory and double-click the iristray.exe file.

Alternatively, you can enter these commands from the *install-dir*/bin directory in the **Open** box of the **Run** command on the **Start** menu. For example, to start the instance named MyIris from the MyIris\bin directory, enter the following command:

```
c:\MyIris\bin\iris start iris
```

These methods of starting InterSystems IRIS call the **iris start** command. See Controlling InterSystems IRIS from the Command Prompt for more options and information on the **iris** command.

23.4 Stopping InterSystems IRIS

Normally you leave your InterSystems IRIS system running. However, if your operating system requires a restart, stop InterSystems IRIS before you shut down your system. The maintenance tasks, such as backups and database repair utilities, do not require you to stop InterSystems IRIS.

From the InterSystems IRIS launcher menu click **Stop InterSystems** to shut down or restart the local InterSystems IRIS instance. By default, this option shuts down (or restarts) InterSystems IRIS immediately, using the default shutdown routine. However, it also provides options for setting a timer for a delayed shutdown, for running a user-defined shutdown routine, for broadcasting a warning message to users on the server, and for shutting down without failing over. You can run this same process from the Windows program menu. Select the **IRIS** folder and **Stop InterSystems IRIS** for that instance name. You cannot cancel a shutdown once the countdown reaches 0 and the shutdown procedures have started.

Important: InterSystems recommends that you run **Stop InterSystems IRIS** to shut down InterSystems IRIS to ensure that it closes properly.

These methods of stopping InterSystems IRIS call the **iris stop** command. See Controlling InterSystems IRIS Instances for more options and information on the **iris** command.

Remove Stop InterSystems Command

To prevent unintentional execution of the **Stop InterSystems** command, you can remove the command from the InterSystems IRIS launcher by deleting the irisstop.exe file from the *install-dir*/bin directory of the corresponding InterSystems IRIS instance. You can also remove the **Stop InterSystems** shortcut from the appropriate InterSystems IRIS instance from the **Start** menu. Point to **Programs** and the InterSystems IRIS instance name, then right-click **Stop InterSystems** and click **Delete**.

23.5 Help Information from the Command Prompt

You can control an InterSystems IRIS instance from the Windows command prompt by running the iris.exe program in the *install-dir*\bin directory. For information about the **iris** command, see Controlling InterSystems IRIS Instances.

To display the most current help information for this command, invoke **`iris help`** from the *install-dir*\bin directory. For example:

```
C:\MyIris\bin>iris help
```

The **`iris help`** command will display the most current help information in the terminal. You can also save help information to a file in the *intsall-dir*\bin directory by adding `>outputfilename` to the **`iris help`** command. For example:

```
C:\MyIris\bin>iris help>helpinformation
```

23.6 Connecting to InterSystems IRIS on the Command Line

You can log in to an InterSystems IRIS instance on the command line using the **iris terminal** command. See Connecting to an InterSystems IRIS Instance for more information on **iris terminal**.

24

Managing InterSystems IRIS on UNIX®, Linux, and macOS

This page describes specific administrative procedures on UNIX®, Linux, and macOS.

24.1 UNIX® Users, Groups and Permissions

Every InterSystems IRIS® data platform installation on a UNIX® platform has the following users and groups:

- *Root* — InterSystems IRIS must be installed by `root`, and some processing by InterSystems IRIS system daemons runs as `root`.
- *Owner of instance* — This user owns most installation files and has full control of the instance. If you install with Minimal initial security settings, `root` is the default owner; otherwise, you are prompted for the owner during installation.
- *Effective user for InterSystems IRIS superserver and its jobs* — All InterSystems IRIS processes spawned by the superserver to serve incoming requests run as this user; in addition jobs hosted by job server processes, task manager jobs, and user-defined startup routines (for example, ^%ZSTART) also run as this user. By default, this user is `irisusr`, but you can change the user during a custom installation.
- *Effective group for InterSystems IRIS processes* — All InterSystems IRIS processes automatically run as this group, which allows normal users, while inside InterSystems IRIS, to access InterSystems IRIS database and journal files to which they may not otherwise have been granted access; file permissions on these and other InterSystems IRIS files are set to allow this group to have appropriate access. On a secure system, only the *Effective user for InterSystems IRIS superserver and its jobs* should be a member of this group. By default, this group contains `irisusr`, but you can change the group during a custom installation.
- *Group allowed to start and stop instance* — This group, `root`, and the *Owner of instance* can start and stop InterSystems IRIS.

All journals and journal directories must have the group ownership set to the *Effective group for InterSystems IRIS processes* group and grant full permissions to that group (`rw` for journals, `rwx` for journal directories). The user who owns the journal and journal directories may vary depending on how they were created.

Journals and journal directories created within InterSystems IRIS are created with the appropriate permissions. However, if you move, copy or create journal directories or journals externally (via scripts or administrator action), you must ensure that the proper permissions are maintained. Failure to set the permissions properly may lead to unexpected and serious errors.

The following example assumes the *Effective group for InterSystems IRIS processes* is `irisusr` and the *Owner of instance* is `irisowner`, although the files may have different user ownership depending on the context in which they were created. For example:

```
journal directory    irisowner    irisusr    drwxrwxr-x
20170801.001         irisowner    irisusr    -rw-rw----
```

Note: These settings are maintained, in part, as the set of permissions on the executables within the *install-dir*/bin directory of the InterSystems IRIS installation. Relevant properties include: ownership, group, mode, set-uid, and set-gid bits. It is important that you do not modify these permissions when performing administrative tasks at the operating-system level.

You can use the InterSystems IRIS `filecheck` utility to check whether permissions, owners, and groups for UNIX installations conform to required settings.

24.1.1 Database and Database Directory Permissions

All databases and database directories must have the group ownership set to the *Effective group for InterSystems IRIS processes* group and grant full permissions to that group (`rw` for databases, `rwx` for database directories). The user who owns the databases and database directories may vary depending on how they were created.

Databases and database directories created within InterSystems IRIS are created with the appropriate permissions. However, if you move, copy or create database directories or databases externally (via scripts or administrator action), you must ensure that the proper permissions are maintained. Failure to set the permissions properly may lead to unexpected and serious errors.

The following example assumes the *Effective group for InterSystems IRIS processes* is `irisusr` and the *Owner of instance* is `irisowner`, although the files may have different user ownership depending on the context in which they were created:

```
dataset directory    irisowner    irisusr    drwxrwxr-x
IRIS.DAT            irisowner    irisusr    -rw-rw----
```

24.2 Startup on UNIX®

The InterSystems IRIS instance uses the following resources to control starting, stopping, and creating new processes:

- The iris.ids file in the *install-dir*\mgr directory.
- Shared memory.

24.2.1 Daemon Resource Locks

InterSystems IRIS uses advisory file locking to prevent multiple startups of the same instance on different machines. With advisory file locking, a single lock file (in this case, the file ilock in the *install-dir*/mgr directory) may be used to exclusively lock multiple resources. The Control Process, Write daemon, and Journal daemon each lock a separate section of the lock file. If this section of the ilock file is already locked, startup terminates. The locks held by the different daemons are called *Daemon Resource Locks.*

A file lock is held by a process until the process terminates. Thus if any lock is held, it indicates that some daemon process on some node is running. It does not indicate, however, whether or not the instance is healthy and running normally.

24.2.2 iris.ids File

The iris.ids file contains the name of the node where InterSystems IRIS was started. The existence of the iris.ids file acts as a flag to ObjectScript utilities and customer-written scripts, indicating whether or not the instance is up and running. This file is often ignored during startup. However, if an error occurs when iris.ids is being read, it will prevent InterSystems IRIS from starting up. In previous versions of InterSystems IRIS, the shared memory identifiers were also stored in the iris.ids file, but this is no longer the case.

24.2.3 Startup Sequence

To best understand the startup sequence, imagine that the instance can be run from two (2) different nodes (machines), node A and node B. The iris.ids file is visible to both nodes, as are the Daemon Resource Locks (for shared files). The shared memory itself, however, is visible only on the node on which it was created (that is, the node where you started InterSystems IRIS).

24.2.3.1 Step 1. Check the Status of the Instance

The startup routine runs **irisdb –cV** to find out the status of the instance. It first attempts to attach to shared memory for the instance:

- If there is no shared memory for the instance, a test is made for Daemon Resource Locks:
  - If no Daemon Resource Locks are held, the instance is reported down.
  - If Daemon Resource Locks are held, the instance is reported to be running on the node specified in the iris.ids file. If the iris.ids file does not exist, no information is available on where the daemons are running.

    Action: The user must run **iris stop** or **iris force** to halt the running instance on the appropriate node. This stops the daemons and deletes the iris.ids file.
- If the attach succeeded, the system is assumed to be up and running. This status is reported to the user. Startup halts.
- If an error displays indicating that startup cannot be completed because shared memory is still attached, wait a few minutes for the memory to be released. If the error persists, there may be an error related to the iris.shid file, which tracks the shared memory ID of InterSystems IRIS on all UNIX® platforms except macOS. Contact the InterSystems Worldwide Response Center (WRC) for support correcting the file.
- If startup was initiated using the optional `filecheck` parameter, the InterSystems IRIS `filecheck` utility checks to ensure that permissions, owners, and groups associated with InterSystems IRIS system files match the system requirements.

24.2.3.2 Step 2. Start InterSystems IRIS

The InterSystems IRIS startup process is run. Checks are repeated to ensure that another startup is not competing for the startup resources:

- If Daemon Resource Locks are held, indicating one or more daemons are running on some node for this instance, InterSystems IRIS reports this and exits with an error. Startup halts.

  The node on which the daemons are running is unknown if the iris.ids file does not exist.

  Action: The user must assume that another startup has occurred on some node. To determine on which node the instance has started, examine the iris.ids file.

InterSystems IRIS continues startup.

24.3 Managing InterSystems IRIS

From the shell, a user with any user ID in the `sysmgr` group can run **iris** (see Controlling InterSystems IRIS Instances) which invokes InterSystems IRIS executables and scripts in the *install-dir*/bin directory. The following sections describe how to perform these management tasks on an InterSystems IRIS instance:

- Starting InterSystems IRIS
- Running InterSystems IRIS
- Stopping InterSystems IRIS

Important: The owner of the installation has full privileges to start and stop the instance, to perform system administration, and to run diagnostic programs for that instance.

Only the user ID that is the owner of the instance can and should run all diagnostic activities. This ensures that any files or resources created are owned by the owner of the instance and not root (which may make it impossible to access these resources by a nonroot user). For this reason, it is inadvisable for root to in any way administer an instance not owned by root (including starting and stopping the instance). A user running as root should only administer instances owned by root.

24.3.1 Starting InterSystems IRIS

To start InterSystems IRIS, run the startup procedure at the system level. This procedure activates either a default configuration file or a configuration file you specify.

Note: If you have any trouble starting InterSystems IRIS, view the messages.log file as described in Monitoring Log Files.

If you are not on the console machine, run Telnet and connect to the target machine where InterSystems IRIS is installed. Before you can start InterSystems IRIS on UNIX®, one of the following must be true:

- You are the superuser.
- You have signed on as the root user. (It is acceptable to **su** (superuser) to root while logged in from another account.)
- Your UNIX® group ID matches the group named during the InterSystems IRIS installation as having privileges to stop and start the system.

  See Installing InterSystems IRIS on UNIX® and Linux for information on specifying such privileges during installation.

Start InterSystems IRIS using the **iris** command:

```
iris start <instname>
```

where *instname* is the name of the InterSystems IRIS instance you want to start. See Controlling InterSystems IRIS Instances for more options and information.

From the shell, a user with any user ID in the `sysmgr` group can run **iris start**. This command verifies that the instance is not currently running on the current or another node, creates shared memory and basic InterSystems IRIS daemons, including multiple auxiliary write daemons (AWDs), runs the startup (**^STU**) routine, which creates additional daemons (for example, ECP daemon), and then allows user logins.

24.3.2 Running InterSystems IRIS

From the shell, a user with any user ID and any group ID (`anyuser:anygroup` in this example), can run **iris terminal** (see Connecting to an InterSystems IRIS Instance) which executes irisuxsession in the *install-dir*/bin directory.

Running as `anyuser:irisusr`, InterSystems IRIS runs its standard startup logic, including Kerberos negotiation, to identify a **$USERNAME** and a set of login roles. In many cases, this **$USERNAME** value is associated with the actual user who invoked **iris terminal**. Thus, while any user may run InterSystems IRIS, the activities of that user once in InterSystems IRIS are defined and limited by the security roles assigned to that user.

CAUTION: Do *not* enter InterSystems IRIS by invoking its executable directly from the *install-dir*/bin directory.

The InterSystems IRIS executable is not itself a setgid-executable. It is the responsibility of the **iris terminal** wrapper to set the group properly on behalf of the user entering InterSystems IRIS.

In addition to **iris terminal**, you may run InterSystems IRIS using the irisdb executable in the /usr/bin directory. This command calls **iris terminal <default-instance>**, setting permissions and opening the Terminal for the default InterSystems IRIS instance. To create the irisdb executable, designate a default instance by running the **iris default <instname>** function.

Note: On macOS 10.11 and later, **iris** and **iris terminal** are located in /usr/local/bin, with links from /usr/bin.

24.3.3 Stopping InterSystems IRIS

Normally you leave your InterSystems IRIS system running. However, if your operating system requires a restart, you should stop InterSystems IRIS before you shut down your system. The InterSystems IRIS maintenance tasks, such as backups and database repair utilities, do not require you to stop InterSystems IRIS.

To stop InterSystems IRIS on UNIX®, the same requirements exist as for starting InterSystems IRIS. One of the following must be true:

- You are the superuser.
- You have signed on as the root user. (It is acceptable to **su** (superuser) to root while logged in from another account.)
- Your UNIX® group ID matches the group named during the InterSystems IRIS installation as having privileges to stop and start the system.

To stop InterSystems IRIS, from the command line:

1. Use the **iris stop** command:

   ```
   iris stop <instname>
   ```

   where *instname* is the name of the InterSystems IRIS instance you want to stop. (See Controlling InterSystems IRIS Instances for more **iris** options and information.)

   CAUTION: You can stop InterSystems IRIS with the **iris force** command, but you should do so with caution because it may result in a loss of data.

2. This procedure invokes the InterSystems IRIS **SHUTDOWN** utility, which displays a status report. Check for active processes in the report to determine if the next step is necessary.

3. Should it be necessary, broadcast a message to any users on the system:

   ```
   Do you want to broadcast a message to anyone? No=> Yes
   Send a message to other terminals. Message => Please sign off
   Terminal => /dev/tty/06
   Terminal =>
   Message =>
   ```

4. After sending one message you can send others, until you respond to the `Message` prompt by pressing **Enter**.
5. When the system asks if you would like to see another system status, enter **`Yes`** to see one, or press **Enter** if you do not want another report.
6. If you answer **`Yes`**, when the system status displays again, identify any active terminals.
7. Confirm that you want to halt by answering **`Yes`**. If you answer **`No`**, the shutdown procedure quits and InterSystems IRIS continues running.

Note: On UNIX® platforms, when an InterSystems IRIS instance is stopped, restarted, or forced down, the instance will wait for all processes to detach from shared memory for a maximum of 30 seconds. After 30 seconds, the instance will close. If there are still processes attached to the shared memory after the instance has closed, restarting the instance will fail.

25

Connecting to Remote Servers

This page describes how to use the InterSystems IRIS® data platform Launcher to access remote servers, as well as how to use the InterSystems IRIS Server Manager to define server connections.

The Launcher and the Server Manager are provided only on Windows.

25.1 Launcher Submenus

For quick access to remote InterSystems IRIS servers, the InterSystems IRIS launcher provides the following options that do not require having OS-level permissions on the machine hosting the remote instance:

- The **Remote System Access** submenu, which provides access to the Terminal and the Management Portal for any instance defined in the InterSystems IRIS Server Manager.

  This submenu also provides access to Studio, the documentation, and the class reference, not specifically discussed here.

  Remote System Access also provides an option you can use to connect via Telnet.
- The **Preferred Server** submenu, which lists the instances defined in the InterSystems IRIS Server Manager and which provides options to let you manage that list.

To use the utilities on the **Remote System Access** submenu for a remote instance:

1. Define a remote server connection to add the server to the preferred server list.
2. Click the InterSystems IRIS Launcher and select **Remote System Access**.
3. Select a utility (for example, the Management Portal)
4. Select the server name.

When connecting to the Management Portal, the documentation, or the class reference, you can bookmark the URLs for future use, for quicker access.

25.2 Define a Remote Server Connection

To use the InterSystems IRIS Launcher utilities or other InterSystems IRIS applications on a remote server, the server must be on the connection list in the InterSystems IRIS Server Manager. This is a list of remote servers you have previously defined to which you can quickly connect. A remote server is defined with an IP address for a unique server and a TCP port, which is an instance of InterSystems IRIS on that server.

Important: The InterSystems IRIS superserver must be running on the remote machine and its port must be open on your firewall to use the InterSystems IRIS Launcher utilities on that system.

For security reasons, username and password are not stored with the remote connection information.

To define the remote server:

1. From the InterSystems IRIS Launcher menu, point to **Preferred Server**, and click **Add/Edit** to open the InterSystems IRIS Server Manager. (On Windows systems, you must have Administrator privileges to take this step.)
2. Click **Add** to open the **Add Connection** dialog box.
3. Fill in the fields as described in the following table and click **OK**. Each field is required unless otherwise indicated.

- **Server Name** — A descriptive phrase that identifies the server; it is what appears as a selection in the InterSystems IRIS Launcher.
- **IP Address** — The IP address, host name (if you have a DNS server) or the fully qualified domain name (FQDN) of the remote server. InterSystems IRIS accepts any legitimate name reference for the remote server.
- **Port** — The port number of the superserver. The default port number is `1972`.
- **Telnet Port** — The port number of the telnet connection. The default port number is `23`.
- **Web Server IP Address** — *Optional*. The IP address of the web server you wish to use to manage this InterSystems IRIS instance. Defaults to IP Address if not specified.

  If you plan to use a web server that exists on a separate machine from the InterSystems IRIS instance you are managing, enter the IP address of the web server machine in this field. Defaults to **IP Address** if not specified.
- **Web Server Port** — The port number of the web server, if needed.
- **CSP Server Instance** — *Optional*. The InterSystems IRIS instance name to which you want to connect if you are configuring one web server to connect to multiple InterSystems IRIS instances. See Specifying CSP Server Instance for more information.

  Note: The **CSP Server Instance** must be in all lowercase characters for the request to route properly.
- **HTTPS** — Whether or not links from the InterSystems IRIS Launcher use HTTPS.
- **Authentication Method** — Choose **Kerberos** or **Password**.

  See Kerberos Authentication for details on the use of these fields.

  See Instance Authentication for information about password-based authentication in InterSystems IRIS.
- **Connection Security Level** — For *Kerberos* [1] only. Choose either **Kerberos**, **Kerberos with Packet Integrity**, or **Kerberos with Encryption**.
- **Windows InterSystems IRIS Telnet Server** — For *Kerberos* only. Select this check box if you are defining a connection to a Windows server.
- **Service Principal Name** — For *Kerberos* only. This field is pre-filled with the recommended service principal name format “cache/<FQHN>” (*FQHN* is the fully qualified host name) for the IP address you enter.
- **Comment** — *Optional*. A description of the remote server.

25.3 Specifying CSP Server Instance

If you are configuring one web server to connect to multiple InterSystems IRIS instances, enter the InterSystems IRIS instance name to which you want to connect in the **CSP Server Instance** field. Make sure you enter the name in all lowercase characters.

For example, if you have an IIS web server installed on a Windows machine and you also install two instances of InterSystems IRIS, iris1 and iris2, by default you manage each instance with its own private Apache web server that is installed as part of InterSystems IRIS.

However, you can also manage both instances from the public IIS web server by changing the **Web Server Port** (`80` by default for IIS) and specifying iris1 and iris2 in the **CSP Server Instance** field when creating a server definition for each instance. This automatically creates virtual directories /iris1 and /iris2 on the public web server that point to the corresponding InterSystems IRIS instances.

When you enter a value in the **CSP Server Instance** field and select the Management Portal on the Launcher for this server, the URI is formed as follows:

```
http://<web srvr addr>:<web srvr port>/<csp srvr inst>/csp/sys/UtilHome.csp
```

This places the instance name before the `/csp/sys/UtilHome.csp` portion of the URI and generates the following URIs for the two instances in the example.

```
http://localhost:80/iris1/csp/sys/UtilHome.csp
http://localhost:80/iris2/csp/sys/UtilHome.csp
```

For more complex configurations involving remote web servers, see the sections for your platform in Web Gateway Configuration Guide.

25.4 Connecting via Telnet

You can also connect to a remote instance of InterSystems IRIS from a Telnet session:

1. Click the InterSystems IRIS Launcher and point to **Remote System Access**.
2. Click **InterSystems IRIS Telnet**, connect to the remote server, and log on to the InterSystems IRIS system with your username and password.

You can remotely log in to an InterSystems IRIS instance on any supported platform from a terminal running on a PC or from any workstation client capable of running Telnet. This client may have only utilities and not an InterSystems IRIS server instance. The version of InterSystems IRIS on the client machine in most cases, must be the same or a later version of the InterSystems IRIS system it manages.

Note: On the macOS platform you can also use **SSH** from a command prompt on Windows to connect to your macOS machine and then connect with the following command:

```
sudo /sbin/service telnet start
```

26

Manage InterSystems IRIS Instances: The iris Command

You can install and run multiple instances of InterSystems IRIS® data platform on a single host system. Each instance is a unique, independent InterSystems IRIS environment.

26.1 Options for Managing an Instance

There are many ways to connect to and manage an InterSystems IRIS instance, which may be one of several installed on a given system. Two of the most common methods are as follows:

- The Windows launcher

  Each InterSystems IRIS instance installed on a Windows system has its own launcher in the system tray, which among other options lets you:

  - Connect to the instance by opening the Management Portal, the InterSystems Terminal, and the Studio developer client.
  - Start, stop and restart the instance.
  - Open the user and developer documentation.

  From the launcher, you can also manage multiple remote InterSystems IRIS instances, including but not limited to, running remote backups, editing configuration settings, and creating and compiling remote objects and routines. See Connecting to Remote Servers for more detailed information.

- The **iris** command

  Executing the **iris** command on the operating system command line gives you management access to an InterSystems IRIS instance, which among other options lets you:

  - Connect to the instance using the InterSystems Terminal.
  - Start, stop, and restart the instance.
  - Display information about the instance, and about other instances installed on the system.

  The **iris** command is described in detail in Connecting to an InterSystems IRIS Instance and Controlling an InterSystems IRIS Instance.

To use the **iris** command on a remote server, use a Telnet or SSH client; to use it with a containerized instance, use it inside the container, or use the **docker exec** command to run it from outside the container.

26.2 Connecting to an Instance

This section describes how to connect to an instance and have access to it via a shell, specifically the ObjectScript shell. You can use this shell in any namespace of an InterSystems IRIS instance. The ObjectScript shell is often called the Terminal, although the Terminal is actually a Windows application that provides this shell. To open this shell for a running instance, use the command **iris terminal** *instname*, where *instname* is the name you gave the instance at installation. A containerized instance is typically named **`IRIS`**.

Note: See Using The Terminal (video) on the InterSystems online learning website.

Log in using one of the predefined user accounts, with the password you provided during installation, or an account you created. The prompt that displays indicates the login namespace, for example:

```
# iris terminal IRIS

Node: intersystems2588, Instance: IRIS27

Username: admin
Password: ********
USER>
```

To exit the Terminal and close the window, enter the command **halt**.

When using the **docker exec** command to open the Terminal for a containerized instance (as described in Interact Using the InterSystems Terminal), you are automatically logged in as **irisowner** and do not need to authenticate.

On a Windows system, you must execute the command from its location, the *install-dir*\bin directory of an InterSystems IRIS instance, or include the full path in the command, for example **c:\InterSystems\IRIS27\bin\iris terminal IRIS4**. You can execute the binary of a given instance to connect to that instance or another; the instance name is required either way.

26.3 Controlling an Instance

The **iris** command supports a number of functions beyond **terminal**, and is invoked in the format **iris** *function instname arguments*, where *instname* is the instance name that you chose during the installation and *arguments* depends on the function.

Important: The **iris help** command displays all the command functions and arguments; the file IRISHelp.html is in the *install-dir*\Help directory. Some functions of the **iris** command are not listed in this document, but are shown in the **help** display.

The **iris** command behaves differently depending on the platform, and is described in the following tables:

- The iris Command on Unix®, Linux, and macOS
- The iris Command on Windows

26.4 The iris Command on Unix®, Linux, and macOS

Note: The **iris** command often displays error information in a message box. You can suppress this message box by adding **quietly** as the final argument to the **iris** command, which runs the command non-interactively with minimal dialog. This argument is also useful with other commands, such as when you want to shut the instance down without having to confirm the command.

iris all

Lists summary information for all installed instances, one instance per line, as described below.

Note: If you need complete information, such as for parsing or reporting purposes, use **iris list**.

iris allw

Lists the same information for each instance as **iris all**, without wrapping long field values. Lines longer than 80 characters may result.

iris console *instname* [*arguments*]

Opens the ObjectScript shell command window.

Arguments: same arguments as **iris terminal**.

iris force *instname*

Forces down the instance.

iris help [*arguments*]

Displays the most recent information about the **iris** command.

Arguments: **start**, **stop**, **force** — Display function-specific help for the start, stop, and force functions.

iris list [*arguments*]

Displays information about the installed InterSystems IRIS instances, as described below.

Arguments: *instname* — Optionally specify an InterSystems IRIS instance name to display only information about that instance. For example, `iris list MyIRIS` displays only information about the MyIRIS instance.

iris mdx *instname*

Provides direct terminal access to the DeepSee shell by running ##class(%DeepSee.Shell).%Go().

iris merge *instname* [*arguments*]

Applies a configuration merge file to the instance, updating its CPF (see Automating Configuration of InterSystems IRIS with Configuration Merge).

Arguments: [*merge-file*], [*target-CPF*] — You can optionally specify the location of the merge file to apply, the location of the target CPF (the active CPF for the instance), or both. For example, `iris merge MyIRIS /tmp/merge.cpf /net/home/MyIRIS` applies the merge file /tmp/merge.cpf to the instance named **`MyIRIS`**, the active CPF of which is in /net/home/MyIRIS. If the merge file and/or target CPF are not specified, environment variables are used if they exist; for more information, see How do I reconfigure an existing instance using configuration merge?

iris python *instname*

Provides direct terminal access to the Python shell by running ##class(%SYS.Python).Shell().

Note: For this command to work you must install Python as described in Using Embedded Python.

iris qall

Lists the same information for each instance as **iris all**, except that long lines are truncated to 78 characters plus a terminating tilde (~).

iris qlist [*arguments*]

Similar to **iris list**, but with additional information. The output for each instance (described below) is given on a single line, with fields separated by carets (^).

Arguments: *instname* — Optionally specify an InterSystems IRIS instance name to display only information about that instance. For example, `iris qlist MyIRIS` displays only information about the MyIRIS instance.

iris rename *instname newname*

Renames the instance.

iris restart *instname* [*arguments*]

Restarts the instance; equivalent of **iris stop** *instname* **restart**

Arguments: **nofailover** — Specify this optional argument to prevent triggering a mirror failover.

iris sql *instname*

Provides direct terminal access to the SQL shell by running ##class(%SQL.Shell).%Go()

iris start *instname* [*arguments*]

Starts the instance.

Note: You may be prompted to start in Emergency Mode; if so, see Emergency Access for more information.

Arguments:

- *full CPF path* — By default, InterSystems IRIS reads certain settings from the iris.cpf file located in the *<install-dir>*/mgr directory. You may provide the full path to a different .cpf file to use instead.
- **nostu** — Starts the specified instance without running **^STU**.

iris stat *instname*

Retrieves the same system statistics as the **irisstat** utility (see Monitoring InterSystems IRIS Using the irisstat Utility).

iris stop *instname* [*arguments*]

Shuts down the instance.

Arguments:

- **restart** — Starts the instance after shutting it down.
- **nofailover** — Specify this optional argument to prevent triggering a mirror failover.

- **quietly** — As the final argument, shuts down the instance without requiring confirmation. (Can also be used to run other *iris* commands noninteractively.)

iris stopnoshut *instname* [*arguments*]

Shuts down the named instance without running user shutdown routines, by running **INTNOSHUT^SHUTDOWN**.

Note: Only the instance owner and **`irisusr`** can run **INTNOSHUT^SHUTDOWN** without logging in to the Terminal.

Arguments: **nofailover** — Specify this optional argument to prevent triggering a mirror failover.

iris terminal *instname* [*arguments*]

Opens the ObjectScript shell for the instance.

Arguments:

- **-B** — Enables system administrator emergency login (see Administrator Terminal Session).
- **-b** *partition_size* — Specifies the maximum partition size (in KB) for the process.
- **"[***label***[+***offset***]]^***routine***"** — Specifies the name of an ObjectScript program to run in user mode. In addition to the specified formats, you can pass parameter lists consisting of string and/or numeric literals, as well as omitted (void) parameters, as follows:
  - `"routine[([parameter-list])]"`
  - `"[label]^routine[([parameter-list])]"`
  - `"##CLASS(package.class).method[([parameter-list])]"`

  where, for example, `parameter-list` is specified in the form `"string literal",,-+-000123.45600E+07`, and omitted parameters are passed to the target as `$Data(parameter)=0`.

  Note: Whitespace and shell meta characters must be quoted in an operating-system dependent form.

- **-U** *namespace* — Specifies the login namespace.

  Note: The **-U** argument has no effect if you are starting InterSystems IRIS with a user account whose **Startup Namespace** is specified (see User Account Properties).

26.5 The iris Command on Windows

On Windows, you must run the **iris** command from the *install-dir*\bin directory (or include the full path with the command).

iris all

Lists summary information for all installed instances, one instance per line, as described below.

Note: If you need complete information, such as for parsing or reporting purposes, use **iris list**.

iris allw

Lists the same information for each instance as **iris all**, without wrapping long field values. Lines longer than 80 characters may result.

iris console *instname* [*arguments*]

Opens the ObjectScript shell command window.

Arguments: Same arguments as **iris terminal**.

iris force *instname*

Forces down the instance.

iris help

Displays the most recent information about the **iris** command.

iris list [*arguments*]

Displays information about the installed InterSystems IRIS instances, as described below.

Arguments: *instname* — Optionally specify an InterSystems IRIS instance name to display only information about that instance. For example, `iris list MyIRIS` displays only information about the MyIRIS instance.

iris merge *instname* [*arguments*]

Applies a configuration merge file to the instance, updating its CPF (see Automating Configuration of InterSystems IRIS with Configuration Merge).

Arguments: [*merge-file*], [*target-CPF*] — You can optionally specify the location of the merge file to apply, the location of the target CPF (the active CPF for the instance), or both. If the merge file and/or target CPF are not specified, environment variables are used if they exist; for more information, see How do I reconfigure an existing instance using configuration merge?

iris qlist [*arguments*]

Similar to **iris list**, but with additional information. The output for each instance (described below) is given on a single line, with fields separated by carets (`^`).

Arguments: *instname* — Optionally specify an InterSystems IRIS instance name to display only information about that instance. For example, `iris qlist MyIRIS` displays only information about the MyIRIS instance.

iris restart *instname* [*arguments*]

Starts the instance after shutting it down.

Arguments: **/nofailover** — Specify this optional argument to prevent triggering a mirror failover.

iris run *instname* [*arguments*]

Runs InterSystems IRIS in programmer mode with no input/output device for $Principal.

Arguments: Same arguments as **iris terminal**.

iris runw *instname routine* [*arguments*]

Runs the named InterSystems IRIS routine in application mode with no input/output device for $Principal. When run from a batch script, the command waits for the InterSystems IRIS process to terminate before returning the exit code from the process.

Arguments: *namespace* — Runs the routine in the specified namespace.

Note: The *namespace* argument has no effect if you are starting InterSystems IRIS with a user account whose **Startup Namespace** is specified (see User Account Properties).

iris start *instname* [*arguments*]

Starts the instance.

Note: You may be prompted to start in "Emergency Mode;" if so, see Handling Emergency Situations in the *Encryption Guide* for more information.

Arguments: *full CPF path* — By default, InterSystems IRIS reads certain settings from the iris.cpf file located in the <*install-dir*>/mgr directory. You may provide the full path to a different .cpf file to use instead.

iris startnostu *instname*

Starts the specified instance without running **^STU**.

iris stop *instname* [*arguments*]

Shuts down the instance.

Arguments: **/nofailover** — Specify this optional argument to prevent triggering a mirror failover.

iris stopnoshut *instname* [*arguments*]

Shuts down the named instance without running user shutdown routines, by running **INTNOSHUT^SHUTDOWN**.

Note: Only the instance owner and **`irisusr`** can run **INTNOSHUT^SHUTDOWN** without logging in to the Terminal.

Arguments: **/nofailover** — Specify this optional argument to prevent triggering a mirror failover.

iris stopstart *instname* [*arguments*]

Starts the instance after shutting it down.

Arguments: **/nofailover** — Specify this optional argument to prevent triggering a mirror failover.

iris terminal *instname* [*arguments*]

Opens the InterSystems Terminal (formally the ObjectScript shell) for the instance.

Arguments:

- *routine* — Runs the named InterSystems IRIS routine in application mode in the Terminal for $Principal.
- **"[***label***[+***offset***]]^***routine***"** — Specifies the name of an ObjectScript program to run in user mode. In addition to the specified formats, you can pass parameter lists consisting of string and/or numeric literals, as well as omitted (void) parameters, as follows:
  - `"routine[([parameter-list])]"`
  - `"[label]^routine[([parameter-list])]"`
  - `"##CLASS(package.class).method[([parameter-list])]"`

  where, for example, `parameter-list` is specified in the form `"string literal",,-+-000123.45600E+07`, and omitted parameters are passed to the target as `$Data(parameter)=0`.

Note: Whitespace and shell meta characters must be quoted in an operating-system dependent form.

- *namespace* — Used with *routine*, runs the routine in the indicated namespace.

  Note: The *namespace* has no effect if you are starting InterSystems IRIS with a user account whose **Startup Namespace** is specified (see User Account Properties).

26.6 iris list, qlist, and all

This topic describes contains additional details about some of the **iris** functions.

iris all

Lists the following information about one or more InterSystems IRIS instance:

- Instance status, as follows
  - <blank> (status unavailable, logins disabled)
  - dn (down or has crashed)
  - up (running)
  - st (starting or stopping)
- Instance name
- InterSystems IRIS version
- Superserver port number
- Installation directory

iris list

Lists the following information about one or more InterSystems IRIS instance:

- Instance name (and installation type)
- Installation directory
- InterSystems IRIS version
- Pathname of InterSystems IRIS parameter (.cpf) file
- Superserver and webserver port numbers
- Instance status, as follows
  - running
  - down
  - starting or stopping
  - incomplete start or stop, logins disabled
- Instance’s system health state, if running (see System Monitor Health State) (not included on Windows)

- Mirror member type and status (if a mirror member) (see **%SYSTEM.Mirror.GetMemberType()** and **%SYSTEM.Mirror.GetMemberStatus()**)

iris qlist

Outputs the following information on a single line, separated by carets (**^**), for one or more InterSystems IRIS instance:

1. Instance name (and installation type)
2. Installation directory
3. InterSystems IRIS version
4. Instance status
5. Pathname of the current Configuration Parameter File, relative to the installation directory. Windows systems instead show the full path.
6. Superserver port number
7. Webserver port number
8. JDBC Gateway port number
9. Instance's system health state, if running (always blank on Windows)
10. Product name
11. Mirror member type (if a mirror member)
12. Mirror status (if a mirror member)
13. Data directory (if applicable)

26.7 Configuring Multiple Instances

You can install and simultaneously run multiple instances of InterSystems IRIS on a single machine. Install InterSystems IRIS as for a single installation, giving each instance a unique name, a unique installation directory, and a unique port number for the superserver, web server, and Telnet.

The special considerations for multiple instances are:

- Installing multiple instances is limited by components where only one exists on a system. For example, typically there is only one web server on a system; and as such, the InterSystems IRIS installation configures the Web Gateway for the most recent installation. InterSystems IRIS client components stored in the registry encounter the same issue. InterSystems IRIS stores its ODBC driver in the registry using one name for each. Currently, the last installation updates these components to point to the last instance installed.

  InterSystems makes an effort to move common components to a common directory that can be shared across InterSystems IRIS instances.

- Multiple instances can share the same multiserver key, but if they do, they must use the same license server or set of license servers. Each system running an instance of InterSystems IRIS under the auspices of one or more license servers must have a local copy of the authorizing license key file installed in every instance.
- Multiple instances can be networked.
- Protection is included against simultaneous database use (that is, each instance must have its own databases and cannot access or modify another instance's databases).

- Additional consideration should be given for routing requests to specific instances when multiple instances are configured. For details on this, see Target Applications on Multiple InterSystems IRIS Servers.
- Each instance must have unique port numbers. See the next section for information on how to Set Port Numbers.

26.7.1 Set Port Numbers

For a standard, single instance of InterSystems IRIS, the superserver port number is `1972` by default. For multiple instances of InterSystems IRIS on a single machine, each must have a unique superserver port number. During installation, subsequent instances are assigned a port if you choose to set it automatically, or you can manually enter port numbers during the installation.

You most likely do not need to change the superserver port numbers because of the way the InterSystems IRIS installation assigns them. You can change the superserver port value after installation from the **Memory and Startup** page (**System Administration** > **Configuration** > **System Configuration** > **Memory and Startup**) of the Management Portal.

You do need to assign each instance a unique Telnet port number. You can change the Telnet port values after installation from the **Startup Settings** page (**System Administration** > **Configuration** > **Additional Settings** > **Startup**) and the **Telnet Settings** page (**System Administration** > **Configuration** > **Device Settings** > **Telnet Settings**), respectively, of the Management Portal.

A

Configuring Third-Party Software to Work in Conjunction with InterSystems Products

InterSystems products often run in environments alongside other providers, where interactions between our products and such tools can have deleterious effects. InterSystems guidance about optimal, reliable configurations for deployment presume that our products can be deployed without interference from third-party tools. For instance, InterSystems has observed that software for security, system monitoring, or virus scanning may impact the installation, performance, and functionality of our products. This is particularly true for tools, such as virus scanners, that directly interact with files that are part of or are used by InterSystems products.

InterSystems understands that customers face business, compliance, and other requirements that impact decisions about what software runs in a given environment and how such software is configured.

Virus scanner considerations:

1. To deliver virus-checked software, InterSystems products are delivered out of a sanitized environment to our customers, and InterSystems provides a checksum for verification.
2. Scanners may report false positives for InterSystems products during installation or at runtime. InterSystems has no control over the findings of third-party security products like virus scanners. Please consult with the vendor of the product you are using for documentation details and guidance.
3. It is possible that scanners may detect false positives for virus patterns on InterSystems processes, and they may respond with actions such as file isolation or even deletion. These actions can have potentially catastrophic impact, especially if performed while the instance is running. Files that are modified rapidly, such as databases, journals, etc., have a higher probability of presenting such patterns.
4. Near constant or very high frequency scans on large files (such as databases) very likely will have a considerable impact on the overall performance of InterSystems products.
5. In accordance with your own risk assessments, you may wish to adjust scanning for the following files and directories:
   - The WIJ file and the directory containing the WIJ file; this directory varies based on configuration settings. See the “Write Image Journaling and Recovery” page.
   - All database files (.DAT) and directories containing database files; these vary based on configuration settings. See the “Configuring Databases” page.
     - Any directory in which journal files are stored or processed; this directory varies based on configuration settings. See the “Journaling” page.

- Any other file or directory that is actively required for InterSystems IRIS® data platform to function; again, these vary based on configuration settings. Examples include the alternate journal directory (as described on the “Journaling” page), or any directory being used by a business service or production.

Long-term or permanent exclusions of files from virus scanners might increase the risk of getting “infected.”

CAUTION: EXCLUDING ITEMS FROM MALWARE SCANS MAY INTRODUCE VULNERABILITIES INTO PROTECTED DEVICES AND APPLICATIONS. THE CLIENT ASSUMES ALL RESPONSIBILITY FOR CONFIGURING MALWARE PROTECTION.

B

Feature Tracker Collects Usage Statistics

As part of ongoing efforts by InterSystems to improve its products in line with customer needs, InterSystems includes a software utility in InterSystems IRIS® data platform called Feature Tracker, which gathers statistics on software module usage. This topic describes Feature Tracker.

B.1 Why InterSystems Gathers Statistics

The statistics gathered by Feature Tracker record whether or not software modules are present and used in a given InterSystems IRIS instance. Feature Tracker sends this information to InterSystems weekly via https. These statistics help InterSystems plan development and support.

The information gathered does not include any application data.

B.2 Where Logs Are

If InterSystems IRIS is successful in sending data, it updates the file FeatureTracker.log in the <*installdir*>\mgr directory. This file contains a JSON-format copy of the data that was sent. Each entry has the following form:

```
"FT.<feature>":<value>,
```

If InterSystems IRIS is unable to send the data, it tries several more times. If it is still unsuccessful, it stops trying and tries again at the next regularly schedule time.

InterSystems IRIS updates the messages log file (messages.log) with both successful and unsuccessful attempts. An entry showing a successful send has the following form:

```
mm/dd/yy-hh:mm:ss 0 %SYS.Task.FeatureTracker transferred data to ats.intersystems.com
```

An entry showing an unsuccessful attempt to send has the following form:

```
mm/dd/yy-hh:mm:ss 1 %SYS.Task.FeatureTracker failed to transfer data
```

Transmitted data includes an encoded license key and host name, as well as the instance GUID.

B.3 What Statistics are Gathered

Feature Tracker logs whether the following software features are enabled:

- BitTrakCare
- BI User (Runtime) and BI Development
- C-Type License
- ExtremeNoUserLimit
- Healthshare, Healthshare Foundation
- IPNeutral
- Web License

Feature Tracker also logs the following information:

- Database (DB) Encryption: Whether it is enabled, and number of mounted encrypted databases available at snapshot time
- Mirroring: Whether this instance is connected to a mirror, number of failover members, number and type of async members, and number of mirror sets each async member is a member of
- System: Authorization, InterSystems IRIS version, operating system, host name, instance name, instance ID, and order ID

B.4 How to Deactivate Feature Tracker

Feature Tracker is enabled by default. You can deactivate it to prevent it from sending data to InterSystems (though it will still collect the data). To deactivate Feature Tracker, use the Task Manager as follows:

1. In the Management Portal, select **System Operation** > **Task Manager** > **Task Schedule**.
2. Locate the Feature Tracker line and select **Feature Tracker**.
3. On the displayed task detail page, select **Suspend**.
4. Answer the question `Do you want to reschedule task when task is supposed to run?` as appropriate.
5. Select **Perform this action now** to suspend the task.

If you upgrade InterSystems IRIS, the upgrade preserves the state of Feature Tracker. If the task was scheduled before the upgrade, it remains scheduled, and if the task was suspended, it remains suspended.