Upgrading and Modifying InterSystems IRIS

Version 2024.2
2024-09-05

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

Upgrading and Modifying InterSystems IRIS
PDF generated on 2024-09-05
InterSystems IRIS® Version 2024.2

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

1

Upgrade Overview

This topic is intended for customers who are upgrading from an earlier version of InterSystems IRIS®, InterSystems IRIS for Health™, or HealthShare® HealthConnect. Customers looking to migrate from InterSystems Caché® should see Why Migrate to InterSystems IRIS?

Important: InterSystems recommends that each application be thoroughly tested in the upgraded environment **before** it is deployed to customers and begins processing live data.

1.1 Compatibility Goals

The goal of each release is a smooth upgrade to new and improved functionality with no changes required to existing applications. However, because of error corrections, significant enhancements, or changes to applicable standards, this goal cannot always be met. In this case, InterSystems discloses all identified instances where changes to applications are necessary so that customers can gauge effort required to move to the new release.

You may, after reading about the release-specific changes (linked below), conclude that none of them affect your application. Even in this case, regardless of how robustly designed and how well implemented your application is, there is no substitute for quality assurance test results that confirm your judgement and demonstrate the application remains unaffected by the upgrade.

1.2 Before Upgrading

The following upgrade tasks are necessary on all platforms. Perform these tasks before you run the upgrade procedures:

Note: If you are upgrading to a maintenance release, the tasks marked with an asterisk (*) are optional. However, performing all tasks will ensure optimum performance.

1. *Review the* InterSystems Upgrade Impact Checklist (or the Incompatibility History) and make sure to account for incompatibilities as needed.
2. *Read the upgrade chapter in the Release Notes for your product* and make sure to account for incompatibilities as needed.
   - Special Considerations When Upgrading in the *InterSystems IRIS Release Notes*
   - Special Considerations When Upgrading in the *InterSystems IRIS for Health Release Notes*

- Special Considerations When Upgrading in the *HealthShare Health Connect Release Notes*

Follow any specific recommendations that should be performed before installing.

3. *Read the InterSystems Supported Platforms for the version of InterSystems IRIS to which you are updating* — Check that your technologies are supported in the new version of InterSystems IRIS.
4. *Ensure that all source code for the application is available** — After a major upgrade, you must recompile all class code under the new version of InterSystems IRIS or else provide class code that has already been compiled under the new version. Otherwise, you will not be able to run your application after the upgrade. It is recommended that you recompile any routine code.

   Check that you have the code for any classes running in deployed mode (which you can place under deployed mode again when the upgrade is complete), as well as the code for custom routines not generated from classes (*.mac or *.int).
5. *Save custom classes, routines and globals** — On an upgrade, the IRISSYS database is modified. To prevent your own classes, routines and globals in the %SYS namespace from being affected by the upgrade installation, ensure that they have names that begin with "Z", "z", "%Z", or "%z". All .int and .obj routines (except for Z*, z*, %Z*, and %z*) are deleted from the %SYS namespace when upgrading.

   Similarly, on an upgrade, the IRISLIB, IRISTEMP, and ENSLIB databases are completely replaced.
6. *Save user files** — To ensure that your user files are not deleted or replaced during the upgrade, save them in the *install-dir*\mgr\User directory, which is the only directory that is not subject to modification during an upgrade.
7. *Examine huge/large page allocations* — On Linux and AIX® systems, the huge/large page allocation must be greater than the shared memory allocation; otherwise, your system will not use huge/large pages which can cause performance issues.

   If your current huge/large pages allocation is exactly equal to your system's shared memory allocation, you should reallocate huge/large pages before upgrading, adding several additional MB, to account for any upward change in shared memory allocation between versions. For details, see Configuring Huge and Large Pages.
8. *Decide whether to create an installation manifest** — A manifest is useful for silent installs, as you can invoke the post-upgrade tasks to run as soon as the upgrade is complete, eliminating the need for interactive steps. See Creating and Using an Installation Manifest for more information.
9. *Precompile any user code that runs on startup** — If your application calls any user code at instance startup (using **%ZSTART**, **ZAUTHENTICATE**, or other means), you must precompile it on an instance of the new version of InterSystems IRIS and use an installation manifest to install it as part of the upgrade process. If you do not, the instance will fail to start up.
10. *Obtain an updated license key** — See the "Managing InterSystems IRIS Licensing" chapter of the *System Administration Guide* for more information.
11. *Check system integrity* — Run a system integrity check on existing directories to ensure there is no corruption in any of the databases. For more information, see the "Introduction to Data Integrity" chapter of *Data Integrity Guide*.
12. *Back up the system* — Before upgrading, InterSystems recommends that you run a complete backup of your system using your customary full operating system backup procedures. If the upgrade is not successful, you may need to restore from this backup. For information about backups, see the "Backup and Restore" chapter of *Data Integrity Guide*.
13. *Stop all productions and disable auto-start of productions* — If your system is running any productions, follow the instructions in the Stopping a Production section of the "Starting and Stopping Productions" chapter in *Managing Productions*.

    It is important to disable auto-start so that you can recompile code before starting productions up again.

Important: If you have removed all external language gateway configurations, you may encounter validation errors during the upgrade process. These validation errors occur when upgrading *from* an affected version. They will occur regardless of the version *to* which you are upgrading. The affected versions include: 2021.1.0, 2021.1.1, 2021.1.2, 2022.1.0, and 2022.1.1.

To prevent these errors, add a single gateway configuration of type **Remote** that points to the local gateway with an arbitrary port number. For example, you can set the **Server Name / IP Address** to `127.0.0.1` and set the **Port** to `1`, naming it `ForUpgrade`. This can be done at any point prior to upgrading and will not impact normal system operation. This configuration can be deleted after the upgrade is completed.

Note: For users of HealthShare Health Connect or InterSystems IRIS for Health, if your instance was previously migrated from Health Connect/HSAP based on the Caché/Ensemble platform, the components.ini file in your installation directory may have a reference to the database MPRLLIB, which is no longer used by the product. Comment out this reference prior to the upgrade by inserting a semicolon at the start of each line. This will prevent a misleading error message in messages.log saying that this database does not exist.

Example:

```
;[MPRLLIB]
;Version=15.032.9686
[HSLIB]
Version=2018.1.0
Compatibility_HSAALIB=15.0
Compatibility_HSPILIB=14.0
Compatibility_VIEWERLIB=17.0
```

2

Upgrading or Modifying Windows

This section describes the procedure to upgrade a single instance of InterSystems IRIS, InterSystems IRIS for Health, or HealthShare HealthConnect. Before proceeding, be sure to perform the tasks in the Before Upgrading section of this chapter. If you are upgrading a mirror or ECP configuration, also review the information in the Upgrading a Mirror and Upgrading ECP Configurations sections of this chapter.

If you are using a manifest as part of your upgrade process, see the instructions in Creating and Using an Installation Manifest before performing the standard installation steps.

CAUTION: Prior to beginning an upgrade of an instance, it is essential that the instance be shut down cleanly. To verify that the shutdown was clean, examine the messages.log file after the shutdown finishes. If the log contains entries similar to the following, then the shutdown was clean:

```
...
05/03/19-14:24:13:234 (5204) 0 Journal restore not required at next startup
05/03/19-14:24:13:234 (5204) 0 Transaction rollback not required at next startup
...
```

If these entries are not present, the instance did not shut down cleanly. Please contact the InterSystems Worldwide Response Center before proceeding with the upgrade.

2.1 Windows Upgrade Procedure

The installer provides all the functionality required to upgrade to a new version of InterSystems IRIS. To instead perform a silent upgrade, which is necessary when using a manifest, review the information in Unattended Upgrade or Reinstall.

To perform the upgrade:

1. Double-click the installer file, for example, IRIS-win_x64.exe.
2. The installer displays a list of all existing InterSystems IRIS instances on the host. Select the name of the instance you want to upgrade and click **OK**.
3. When prompted for an InterSystems IRIS license key, click **License** and browse for the new iris.key file. If the installer detects a new key file in the *install-dir*/mgr directory, it proceeds without prompting you for the license key.
4. If a local Microsoft Internet Information Services (IIS) web server is detected, the **Local Web Server** dialog box lets you select whether or not the installer should automatically configure the web server for this instance. If you choose to not have the installer configure your web server automatically, you will have to configure it manually after the installation finishes. If a local web server is not detected, the dialog box instead asks if you want to **Abort the installation**

or **Continue the installation without Web Gateway**. If you choose to continue anyway, you will have to configure the web server manually after the installation finishes.

Important: InterSystems recommends using the Microsoft IIS web server because it can be automatically configured during the upgrade process. Make sure it is installed and running before beginning the upgrade process. In most cases, it is not necessary to manually configure the IIS web server.

5. After the installer validates the license key, click **Upgrade**.

   Important: Do not interrupt the installation while it is in progress. If the upgrade fails with any error messages, correct the issues and restart the upgrade installation.

6. After the upgrade is completed, click **Finish**.
7. Examine messages.log, iboot.log, and ensinstall.log in the *install-dir*\mgr directory for any errors. If any fatal error is found, correct the error, and then run the installer again.

Important: If your operating system is configured to use large memory pages, check the startup messages to make sure shared memory is being allocated in accordance with these settings. If you see a message similar to the following, reboot your server to avoid an out-of-memory situation.

```
Failed to allocate 592MB shared memory using large pages.  Switching to small pages.
```

2.2 Reinstall or Uninstall

By running setup and selecting an InterSystems IRIS instance of the same version as the installer, or by selecting **Programs and Features** from Windows Control Panel and selecting an InterSystems IRIS instance, you can make changes to or uninstall the instance.

When you run setup as described in Windows Attended Installation and select an InterSystems IRIS instance of the same version as the installer in the **Select Instance** box, or select an instance in **Programs and Features** and use the **Change** or **Repair** buttons, the **Updating Instance** *instancename* dialog box displays.

Note: When you select the **Uninstall** button in **Programs and Features**, the uninstall operation begins immediately.

Click **Next** to display the **Modify, repair or remove the program** dialog box, then select the appropriate option on this dialog to change, repair, or uninstall the instance.

- Select **Modify** to display the **Custom Setup** dialog box described in Windows Custom Installation. Using this dialog box, you can select the component groups or components you want to add or remove. Components are described in the Components Installed by Setup Type table.
- Select **Repair** to repair problems with the instance such as missing or corrupt files or registry entries.
- Select **Remove** to uninstall the instance.

Important: Use only the InterSystems IRIS installer or Windows Control Panel **Programs and Features** to uninstall InterSystems IRIS. Other uninstall programs are not supported and using them may cause unexpected results.

2.3 Unattended Upgrade or Reinstall

In addition to installing a new instance, the InterSystems IRIS installer can be called on an existing installed instance. To do so, you must use the **/instance** flag to specify the name of the target existing instance. The action the installer takes depends on the version of the installation file compared to the version of the instance, as follows:

- If the installation file is the same version as the target installed instance, the installer reinstalls (repairs) the instance.
- If the installation file is a later version than the target installed instance, the installer upgrades the instance to the new version.

For example, to run an unattended upgrade of an installed instance IRISB that is an earlier version than the installation file, use the following:

```
C:\downloads\IRIS-2019.1.0.516.0-win_x64.exe /instance IRISB /qn
```

You can reinstall one or more specific features, as listed in the Custom-Installable Features table, by specifying the target instance and using the **REINSTALL** property (see the Command-Line Properties table). For example, to reinstall the PDF documentation for the installed instance IRISB, you can use the following command (assuming the installation file and IRISB are the same version):

```
C:\downloads\IRIS-2018.1.0.508.0-win_x64.exe /instance IRISB /qn REINSTALL=documentation,documentation_pdf
```

2.4 Unattended Removal

To launch an unattended removal, specify the instance to uninstall and the **REMOVE=ALL** property, as follows:

```
<path>\<installer>.exe /instance <instancename> /q[b|n] REMOVE=ALL
```

You can also use the **REMOVE** property to remove specific features, as described in the Custom-Installable Features table. For example, to remove the Apache 2.0 Web Gateway from instance IrisC, use the command:

```
C:\downloads\IRIS-2018.1.0.508.0-win_x64.exe /instance IrisC /qn REMOVE=cspgateway,cspapache20
```

2.4.1 Special Consideration

If you do not have access to the original installation package, you can run uninstall in unattended mode by using the Windows® Installer command-line application (**msiexec**) and information in the Registry, as follows:

```
msiexec /x {<product_guid>} /qn /l <logfile>
```

where <*product_guid*> is the **ProductCode** property value of the version you installed.

You can obtain the **ProductCode** property value from the following Registry location:

Processor Type	Registry Location
32–bit	HKEY_LOCAL_MACHINE\SOFTWARE\Intersystems\IRIS\Configurations\<*instance*>
64–bit	HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\Intersystems\IRIS\Configurations\<*instance*>

where <*instance*> is the name of the InterSystems IRIS instance you want to uninstall in unattended mode. The **ProductCode** property value is displayed in a row similar to:

```
ProductCode     REG_SZ   {80E3F658-2D74-4A81-92AD-FD16CD226154}
```

You can also use any of the properties in the Command-Line Properties table with **msiexec**. For information about **msiexec**, see the Microsoft msiexec (command-line options) TechNet article (https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc759262(v=ws.10)).

3

Upgrading or Modifying UNIX®, Linux, and macOS

This section describes the procedure to upgrade a single instance of InterSystems IRIS, InterSystems IRIS for Health, or HealthShare HealthConnect. Before proceeding, be sure to perform the tasks in the Before Upgrading section of this chapter. If you are upgrading a mirror or ECP configuration, also review the information in the Upgrading a Mirror and Upgrading ECP Configurations sections of this chapter.

If you are using a manifest as part of your upgrade process, see the instructions in Creating and Using an Installation Manifest before performing the standard installation steps.

CAUTION: Prior to beginning an upgrade of an instance, it is essential that the instance be shut down cleanly. To verify that the shutdown was clean, examine the messages.log file after the shutdown finishes. If the log contains entries similar to the following, then the shutdown was clean:

```
...
05/03/19-14:24:13:234 (5204) 0 Journal restore not required at next startup
05/03/19-14:24:13:234 (5204) 0 Transaction rollback not required at next startup
...
```

If these entries are not present, the instance did not shut down cleanly. Please contact the InterSystems Worldwide Response Center before proceeding with the upgrade.

3.1 UNIX®, Linux, and macOS Upgrade Procedure

The installation script, **irisinstall**, provides all the functionality required to upgrade to a new version of InterSystems IRIS. To instead perform a silent upgrade, review the information in the UNIX®, Linux, and macOS Unattended Installation section of the "Installing on UNIX®, Linux, and macOS" chapter of this guide.

Note: If the profile of the user executing **irisinstall** has a value set for the *CDPATH* variable, the upgrade fails.

To perform the upgrade:

1. If your installation kit is in the form of a .tar file, you must first uncompress it (see Uncompress the Installation Kit).
2. As a user with root privileges, start the installation procedure by running the irisinstall script, located at the top level of the installation files:

   ```
   # sudo sh /<pathname>/irisinstall
   ```

where <*pathname*> is the location of the installation kit, typically the temporary directory to which you have extracted the kit.

3. The script displays a list of all existing InterSystems IRIS instances on the host.
4. At the **Enter instance name:** prompt, type the name of the InterSystems IRIS instance you want to upgrade, and press **Enter**.
5. When prompted for a license key file, type *keypath*\iris.key, where *keypath* is the location of the new iris.key file. If the installation script detects the new license key in the *install-dir*/mgr directory, the script proceeds without prompting you for the license key file.
6. If a local web server is detected, you will be asked if you would like to use the web server to connect to your installation. If you enter y, the web server will be connected automatically. If you enter n, the web server will not be connected automatically and you will have to configure it manually after the installation finishes. If a web server is not detected, you will be asked if you would like to abort. If you choose to continue the installation, you will have to configure your web server manually after the installation finishes.

   Important: InterSystems recommends using the Apache httpd web server because it can be automatically configured during the upgrade process. Make sure it is installed and running before beginning the upgrade process. In most cases, it is not necessary to manually configure the Apache web server.

7. The installation script summarizes the upgrade options and again asks you to confirm the upgrade. Press **Enter** to continue with the upgrade.

   Important: Do not interrupt the installation while it is in progress. If the upgrade fails with any error messages, correct the issues and restart the upgrade installation.

8. Examine messages.log, iboot.log, and ensinstall.log in the *install-dir*/mgr directory for any errors. If any fatal error is found, correct the error, and then run the installation script again.

After the upgrade finishes, the instance restarts and any installation manifest is run.

Important: If your operating system is configured to use huge memory pages, check the startup messages to make sure shared memory is being allocated in accordance with these settings. If you see a message similar to the following, reallocate huge pages to be greater than the shared memory allocation, then reboot your server to avoid an out-of-memory situation.

```
Failed to allocate 1468MB shared memory using Huge Pages. Startup will retry with standard
 pages.
```

3.2 Uninstalling on UNIX®, Linux, and macOS

To safely uninstall InterSystems IRIS, follow this procedure:

1. Find the name of the InterSystems IRIS instance you wish to delete using the **iris list** command to list all the instances on your machine:

   ```
   iris list
   ```

2. Verify the instance is stopped. If it is not, stop it with the **iris stop** command:

   ```
   iris stop <instname>
   ```

Where *instname* is the instance name that you chose during the installation. If it hangs, force it down using **iris force**:

```
iris force <instname>
```

3. Remove the instance using the **iris delete** command:

   ```
   iris delete <instname>
   ```

4. Remove the installation directory using the following operating system command:

   ```
   rm -r <directory>
   ```

   Important: Be aware that this removes files you may wish to keep. For example: the license key (iris.key), the configuration file (iris.cpf), and the user database file (iris.dat).

The uninstall procedure removes all files installed and created during normal InterSystems IRIS processing, including journal and temporary database files.

Important: The SUSE Linux Enterprise Server 9 platform uses asynchronous scriptlets, so the uninstall process cannot guarantee that InterSystems IRIS stops before it removes files.

4

Upgrading a Mirror

This section provides instructions for upgrading InterSystems IRIS instances, an application, or both on the members of an InterSystems IRIS mirror.

As noted in the "Mirroring" chapter of the *InterSystems IRIS High Availability Guide*, all failover and DR async members of a mirror must be of the same InterSystems IRIS version, and can differ only for the duration of an upgrade. Once an upgraded member becomes primary, you cannot make use of the other failover member and any DR async members (and in particular cannot allow them to become the primary) until the upgrade is completed.

Mirroring does not require reporting async members to be of the same InterSystems IRIS version as the failover members, although application functionality may require it; for more information, see InterSystems IRIS Instance Compatibility in the "Mirroring" chapter of *High Availability Guide*.

There are four mirror upgrade paths to choose from. To determine which procedure you should follow, review the factors in the Choosing Mirror Upgrade Procedure section below. The four procedures are located in the Mirror Upgrade Procedures section below.

Also review the following two sections about upgrading a mirror:

- Mirror Upgrade Terms, which defines some of the terms used in the mirror upgrade procedures.
- Adding Mirror Members During an Upgrade, which discusses when you should add members to a mirror.

Important: On Linux systems supporting **systemd**, although it is possible to use either **systemctl** commands or the /etc/init.d/ISCAgent script to manage the ISCAgent, you must choose one method and use it exclusively; the ISCAgent should always be stopped using the method with which it was started. For more information, see Starting the ISCAgent on Linux Systems in the "Mirroring" chapter of the *InterSystems High Availability Guide*.

When you upgrade InterSystems IRIS on such a Linux system, a running ISCAgent is automatically restarted using **systemd**. If you are using the /etc/init.d/ISCAgent script to manage the ISCAgent, stop the agent before performing the upgrade so that it is not automatically restarted, then restart it using the script after the upgrade.

4.1 Choosing Mirror Upgrade Procedure

There are two primary factors to consider in choosing the procedure appropriate for your mirror upgrade:

- Are you upgrading to a maintenance release of the installed version of InterSystems IRIS, or to a new major version of InterSystems IRIS?

- Does the upgrade involve changes to mirrored databases?

Whenever you upgrade to a maintenance release and are not making any application upgrades, you do not have to consider the second question; you can always use the simple Maintenance Release Upgrade procedure, which renders the mirror unavailable only for the time it takes to execute a planned failover.

However, when you upgrade from one major InterSystems IRIS version to another, perform application upgrades, or both, there is the possibility that your upgrade involves changes to mirrored databases. Such changes *must occur on the functioning primary failover member* as part of the upgrade procedure, while application access is disabled. As a result, the upgrade requires more downtime than if you are not making any mirrored database changes. (The changes are then replicated by the mirror on the backup and any async members.)

Important: Following a major upgrade, InterSystems recommends that you recompile all classes in all application namespaces on the instance and some routines must also be recompiled. Any application upgrade you perform may require changes to application data. In either of these situations, your upgrade may result in changes to mirrored databases.

If you are upgrading to a major release, are performing application upgrades, or both, you must determine whether any of the following conditions apply:

- Classes and routines are stored in mirrored databases that also contain application data. Classes must be recompiled on the primary (even if the application has not been upgraded). Recompiling routines on the primary is recommended.
- Data in mirrored databases must be upgraded due to an application upgrade; these changes must take place on the primary.

Note: Since all Interoperability productions have mirrored application code, these conditions will always apply for mirrored environments with Interoperability productions.

If your major upgrade and/or application upgrade involves any of these conditions, use the Major Version Upgrade (Mirrored Database Changes) procedure, which renders the mirror unavailable only for the time it takes to execute a planned failover and make the required mirrored database changes.

If your upgrade does *not* involve any of the listed mirrored database changes, consider the Major Version Upgrade (No Mirrored Database Changes) procedure, which renders the mirror unavailable only for the time it takes to execute a planned failover.

If you have a significant planned maintenance window and minimizing mirror downtime is not a major concern, you may choose to use the simpler Major Version Upgrade with Planned Downtime procedure instead.

In the case of major upgrades, the general sequence of each procedure applies to upgrades of the mirror's InterSystems IRIS instances, application code, or both; adapt individual steps depending on what you are upgrading.

Note: Your circumstances may call for additional steps and details in a given procedure. If you are still unsure which procedure is appropriate for you, or whether a particular procedure is appropriate as stated, please contact the InterSystems Worldwide Response Center (WRC).

4.2 Adding Mirror Members During an Upgrade

If you plan to add members to a mirror, you may want to defer this until you plan to perform one of the upgrades described in this section and add members of the new version during the upgrade, so that you do not have to upgrade them later. You can always add members of a newer version to a mirror during an upgrade, provided you immediately continue with and complete the upgrade as described, with one restriction: once a member of the newer version becomes primary, it must remain primary until all other members have been upgraded.

Adding new members during an upgrade can be very helpful when migrating a mirror to new hardware. Rather than upgrading one of the failover members before failing over to it, you can install a new instance of the new version on the target system, add it to the mirror as a DR async member, promote it to backup and then fail over to it, thus migrating the mirror primary to the new system. By repeating this technique you can then migrate the remaining failover member. The procedures in this section include steps for adding a new backup of the new version as an alternative to upgrading the existing backup, to illustrate this approach.

4.3 Mirror Upgrade Terms

In the procedures in this section, the following terms are used:

Upgrade classes and routines

Recompile all classes in all application namespaces on the instance. This should be done after an InterSystems IRIS major version upgrade (as described in Post-Upgrade Tasks) and/or application upgrade, which may involve one or more of the following operations, depending on your situation:

- As noted in the foregoing, when classes exist in any mirrored database that also contains application data, they must be recompiled locally on the functioning primary failover member (regardless of whether they are modified by an application upgrade). It is recommended that any existing routines are recompiled.
- Classes and routines stored in nonmirrored routines databases that are separate from application data can be recompiled on a mirror member regardless of whether it is the current primary.
- Classes and routines stored in separate nonmirrored routines database can also be "precompiled" by recompiling a copy of the database on a system that has already been upgraded to the target InterSystems IRIS release, then distributed to each mirror member following the upgrade.

System A

The mirror member that is *initially* the primary failover member.

System B

The mirror member that is *initially* the backup failover member.

System C

A newly-added DR async of the new InterSystems IRIS version that has been promoted to failover member.

Set no failover

Use the **^MIRROR** routine to set the no failover state so that failover cannot occur; see Avoiding Unwanted Failover During Maintenance of Failover Members in the "Mirroring" chapter of the *InterSystems IRIS High Availability Guide* for instructions.

Perform a graceful shutdown

Use the **iris stop** command to shut the instance down cleanly (see Controlling InterSystems IRIS Instances in the "Using Multiple Instances of InterSystems IRIS" chapter of the *InterSystems IRIS System Administration Guide*).

Promote to failover member

Follow the procedure for promoting a DR async mirror member to failover member as described in Promoting a DR Async Member to Failover Member in the "Mirroring" chapter of the *InterSystems IRIS High Availability Guide*.

Viewing the Mirror Monitor

Use the Mirror Monitor page in the instance's Management Portal to review the status of the mirror and its members; see Using the Mirror Monitor in the "Mirroring" chapter of the *InterSystems IRIS High Availability Guide* for more information.

4.4 Mirror Upgrade Procedures

There are four mirror upgrade paths to choose from. To determine which procedure you should follow, review the factors in the Choosing Mirror Upgrade Procedure section. The four procedures are:

- Maintenance Release Upgrade
- Major Version Upgrade (Mirrored Database Changes)
- Major Version Upgrade (No Mirrored Database Changes)
- Major Version Upgrade with Planned Downtime

The first three procedures are designed to let you perform the upgrade you need with the shortest possible application downtime; they differ to accommodate different upgrade circumstances. The last, which is a bit simpler, can be used for any type of upgrade when minimizing downtime is not a priority.

4.4.1 Maintenance Release Upgrade

If you are upgrading to an InterSystems IRIS maintenance release rather than a new major InterSystems IRIS version and are not making any application changes, use the following procedure, which renders the mirror unavailable only for the time required to execute a planned failover:

1. To prevent failover from occurring until the backup is fully upgraded, set no failover.
2. Perform one of the following two operations:
   - Upgrade the existing backup:
     a. Perform a graceful shutdown of the backup (B).
     b. Upgrade the backup (B) with the new version of InterSystems IRIS. System B becomes backup.
   - Add a new backup of the new version:
     a. Install the new version of InterSystems IRIS on system C.
     b. Add system C to the mirror as a DR async member.
     c. Promote system C to failover member. System C becomes backup and System B becomes a DR async.
3. Ensure that the backup (B or C) has become active by viewing the Mirror Monitor.
4. Clear no failover.
5. Perform a graceful shutdown of the primary (A). The mirror fails over and the backup (B or C) takes over as primary.

6. Upgrade system A with the new version of InterSystems IRIS. System A becomes backup.
7. If system C became primary and system B was demoted to DR async, upgrade system B.
8. Perform a graceful shutdown of the new primary (B or C). The mirror fails over and the backup (A) becomes the primary. If you are using a third instance as a DR async member, you should failover once more, so that each member has become the primary once. This ensures that all automated upgrade and reactivation steps are performed on all mirror members and that all instances are ready to act as primary.

4.4.2 Major Version Upgrade (Mirrored Database Changes)

If you are upgrading to a new major InterSystems IRIS version and/or performing an application upgrade and have determined that changes to mirrored databases are required (as described in Choosing a Mirror Upgrade Procedure), use the following procedure, which renders the mirror unavailable only for the time it takes to execute a planned failover and make the required mirrored database changes:

1. To prevent failover from occurring until the backup is fully upgraded, set no failover.
2. Perform one of the following two operations:
   - Upgrade the existing backup:
     a. Perform a graceful shutdown of the backup (B).
     b. Upgrade the backup (B) with the new version of InterSystems IRIS. System B becomes backup.
     c. Upgrade nonmirrored classes and routines on system B, if any.
   - Add a new backup of the new version:
     a. Install the new version of InterSystems IRIS on system C.
     b. Add system C to the mirror as a DR async member.
     c. Promote system C to failover member. System C becomes backup and System B becomes a DR async.
3. Ensure that the backup (B or C) has become active by viewing the Mirror Monitor.
4. Disallow new user access to the mirror using established practices at your site. Additionally, you must disable any application jobs that would normally start on system B when it becomes primary.
5. Clear no failover.
6. Perform a graceful shutdown of the primary (A). The mirror fails over and the backup (B or C) takes over as primary
7. Upgrade mirrored classes and routines on the new primary (B or C). If an application upgrade requires further changes to mirrored databases, make these changes.
8. Restore user access to the mirror.
9. To prevent system A from taking over as primary until fully upgraded, set no failover
10. Upgrade system A with the new version of InterSystems IRIS. System A becomes backup.
11. Upgrade nonmirrored classes and routines on system A, if any.
12. Clear no failover.
13. If system C became primary and system B was demoted to DR async, upgrade system B and upgrade nonmirrored classes and routines on system B, if any.
14. Perform a graceful shutdown of the new primary (B or C). The mirror fails over and the backup (A) becomes the primary. If you are using a third instance as a DR async member, you should failover once more, so that each member has

become the primary once. This ensures that all automated upgrade and reactivation steps are performed on all mirror members and that all instances are ready to act as primary.

4.4.3 Major Version Upgrade (No Mirrored Database Changes)

If you are upgrading to a new major InterSystems IRIS version and/or performing an application upgrade and have determined that changes to mirrored databases are *not* required (as described in Choosing a Mirror Upgrade Procedure), you may be able to use the following procedure.

This procedure is simplest for static applications where separate nonmirrored routines databases are always maintained. It can, however, be used even if the separate routines databases are normally mirrored by removing these databases from the mirror for the duration of the upgrade.

Important: If the routines databases are normally mirrored, ensure that they do not contain any application data before removing them from the mirror.

This procedure cannot be used with normally mirrored routines databases if ECP application servers connect to the mirror.

1. Enact a code freeze and application configuration freeze to disallow application changes during the upgrade, using established procedures at your site, to ensure that routines databases are not modified during the upgrade.
2. To prevent failover from occurring until the backup is fully upgraded, set no failover.
3. Perform one of the following two operations:
   - Upgrade the existing backup:
     a. If the routines databases are mirrored, remove them from the mirror on system B.
     b. Ensure that the backup (B) has become active by viewing the Mirror Monitor.
     c. Perform a graceful shutdown of the backup (B).
     d. Upgrade the backup (B) with the new version of InterSystems IRIS. System B becomes backup.
     e. Upgrade classes and routines on system B.
   - Add a new backup of the new version:
     a. Install the new version of InterSystems IRIS on system C.
     b. Add system C to the mirror as a DR async member.
     c. Promote system C to failover member. System C becomes backup and System B becomes a DR async.
     d. If the routines databases are mirrored, remove them from the mirror on systems C and B.
     e. Ensure that the backup (C) has become active by viewing the Mirror Monitor.
4. Clear no failover.
5. Perform a graceful shutdown of the primary (A). The mirror fails over and the backup (B or C) takes over as primary.
6. To prevent system A from taking over as primary until fully upgraded, set no failover.
7. Upgrade system A with the new version of InterSystems IRIS. System A becomes backup.
8. Upgrade any nonmirrored classes and routines on system A.
9. If system C became primary and system B was demoted to DR async, upgrade system B and upgrade nonmirrored classes and routines on system B, if any.

10. For any mirrored routines databases that you removed from the mirror on backup (B or C) before it became the new primary, do the following:

    a. Remove the routines databases from the mirror on system A.

    b. Add the databases to the mirror on the new primary (B or C), then back them up and restore them on the backup (A) and DR async (B, if C is the primary), using the procedures for adding an existing database to a mirror provided in Adding Databases to a Mirror in the "Mirror" chapter of the *InterSystems IRIS High Availability Guide.* Retain these backups, as they represent the first backups of newly-added mirrored databases; older backups made prior to the upgrade cannot be used to restore these databases in the event of disaster.

11. Clear no failover.

12. Perform a graceful shutdown of the new primary (B or C). The mirror fails over and the backup (A) becomes the primary. If you are using a third instance as a DR async member, you should failover once more, so that each member has become the primary once. This ensures that all automated upgrade and reactivation steps are performed on all mirror members and that all instances are ready to act as primary.

4.4.4 Major Version Upgrade with Planned Downtime

If you are upgrading to a new major InterSystems IRIS version and/or performing an application upgrade and have a significant planned maintenance window that obviates the need to minimize mirror downtime, you may prefer to use the following simpler procedure, regardless of whether changes to mirrored databases on the primary are required:

1. Disallow all user access to the mirror using established practices at your site. Additionally, you must disable any application jobs that would normally start on instance startup.
2. Perform a graceful shutdown of the backup (B).
3. Perform a graceful shutdown of the primary (A).
4. Upgrade system A with the new version of InterSystems IRIS. System A becomes primary.
5. Upgrade mirrored classes and routines on system A. If an application upgrade requires further changes to mirrored databases, make these changes.
6. Upgrade nonmirrored classes and routines on system A, if any.
7. Perform one of the following two operations:

   - Upgrade the existing backup:

     a. Upgrade system B with the new version of InterSystems IRIS. System B becomes backup.

     b. Upgrade nonmirrored classes and routines on system B, if any.

   - Add a new backup of the new version:

     a. Install the new version of InterSystems IRIS on system C.

     b. Add system C to the mirror as a DR async member.

     c. Promote system C to failover member. System C becomes backup.

     d. Upgrade system B with the new version of InterSystems IRIS. System B becomes a DR async.

     e. Upgrade nonmirrored classes and routines on system B, if any.

8. If system C became primary and system B was demoted to DR async, upgrade system B and upgrade nonmirrored classes and routines on system B, if any.

9. Perform a graceful shutdown of the new primary (B or C). The mirror fails over and the backup (A) becomes the primary. If you are using a third instance as a DR async member, you should failover once more, so that each member has become the primary once. This ensures that all automated upgrade and reactivation steps are performed on all mirror members and that all instances are ready to act as primary.
10. Restore user access to the mirror.

5

Upgrading ECP Configurations

In general, ECP application servers should be upgraded before the data servers they connect to. Application servers must have either local routines databases to recompile following upgrade or access to “precompiled” routines database (previously recompiled on a separate system running the target InterSystems IRIS version) on the data server.

Downtime can be minimized using rolling upgrades with users connecting to both upgraded and unupgraded application servers as well as the database server if one of the following is true:

- There are no application changes.
- There are application changes, but the new code does not generate any new data structures, meaning that old code can work with data generated by new code.

If the new application code can work against old data, but can generate new data structures not understood by the old code, follow this procedure:

1. Upgrade the application servers on a rolling basis, but do not allow users to connect to an application server once it is upgraded. (Application server capacity is gradually reduced.)
2. When enough application servers have been upgraded, restore user access to the upgraded application servers and end all user connections to the remaining (not upgraded) application servers and the data server (if need be), ensuring that users have no access to these systems until you enable it.
3. Upgrade the remaining application servers, restoring user access to each application server after it is upgraded.
4. Upgrade the data server and restore user access as needed. (Upgrading the data server causes a pause in application activity, the length of which depends on the amount of downtime involved in the upgrade.)

If you have questions or concerns about how to upgrade your ECP configuration, please contact InterSystems Worldwide Customer Support (WRC).

6

Post-Upgrade Tasks

The post-upgrade tasks required depend on whether you have upgraded to a new major version of InterSystems IRIS or to a maintenance release of the installed version. Also note the following points:

- If you changed an instance from 8–bit to Unicode during the upgrade, InterSystems IRIS does not automatically convert your databases. You need to do this conversion yourself; for more information, see the Database Portability section in the "Namespaces and Databases" chapter of *Orientation Guide for Server-Side Programming*.
- It is not necessary to re-import any OpenAPI 2.0 specification files.
- It is not necessary to re-import any Web Service Definition (WSDL) files.
- Cached queries are always purged during upgrade. They are recompiled and cached as needed.
- If you did not configure your web server automatically during the installation process, you will need to connect it manually.
- If your web server is using a port number other than 80, you will need to change the CPF WebServerPort and WebServerURLPrefix parameters to your web server's port number in order to connect with your IDE.
- If you upgraded from an instance with a private web server (2023.1 or older), you should disable and remove the private web server.
- If you are on Linux or AIX® and are using huge or large pages, confirm that they are still being used:

  In the <install-dir>/mgr/messages.log file, confirm there is a message indicating shared memory has been allocated using huge or large pages.

  If huge/large pages are not enabled, you may need to adjust the allocation to account for increases in shared memory with the new version. After reallocating, restart your system to allow the changes to take effect. See Configuring Huge and Large Pages for details on this process.

6.1 Maintenance Release Post-Upgrade Tasks

When the upgrade is complete, if your system runs any productions, follow the instructions in the Starting a Production section of the "Starting and Stopping Productions" chapter in *Managing Productions* to restart the productions.

Typically, maintenance release upgrades do not require you to change external files and clients, or to recompile classes and routines.

If you choose to recompile, review the information in the Major Version Post-Installation Tasks section.

6.2 Major Version Post-Installation Tasks

After upgrading InterSystems IRIS to a new major version, it is important to follow the guidance provided in the General Upgrade Information in the *Release Notes*. You must also perform the following tasks (if you have not already done so as part of one of the previous procedures in this chapter):

- *Recompile classes and routines* — InterSystems recommends that customers recompile all classes and all routines in each namespace. See How to Compile Namespaces for instructions. You may have your own tools and procedures for this purpose, taking into account any dependencies and other needs particular to the instance that was upgraded.
- *Recompile %SYS classes and routines* — The %SYS namespace is skipped by default when compiling all classes and routines. Make sure you recompile this namespace as well. See How to Compile Namespaces for instructions. Only classes and routines beginning with "Z", "z", "%Z", or "%z" will be preserved on upgrade. These classes and routines cannot be precompiled.
- *Regenerate proxy classes* — You must regenerate any proxy classes you had generated in the upgraded instance by following the instructions in the appropriate guides in the *InterSystems IRIS Language Bindings* set.

  This item does not apply to web services and web clients; it is not necessary to re-import any Web Service Definition (WSDL) files.
- *Clear browser cache* — Your browser cache may contain JavaScript files that are no longer compatible with the installed version of InterSystems IRIS and may cause errors; clear your browser cache immediately after completing the upgrade.

The following tasks may be necessary depending on your environment and the components you use:

- *Review query plans.* When you upgrade to a new major version, existing Query Plans are automatically frozen. This ensures that a major software upgrade will never degrade the performance of an existing query. For performance-critical queries, you should test if you can achieve improved performance.
- *Restart productions* — If your system runs any productions, follow the instructions in the Starting a Production section of the "Starting and Stopping Productions" chapter in *Managing Productions*.
- *Upgrade the Web Gateway* — If your Web Gateway is on a separate machine from the InterSystems IRIS server you are upgrading, you must also upgrade the Web Gateway on that separate machine. You can accomplish this by following the Windows Upgrade Procedure in this chapter, or performing a silent installation with the **ADDLOCAL** property as discussed in Windows Unattended Installation.
- *Update web server files* — Following upgrades you must deploy these using established practices at your site.

6.2.1 How to Compile Namespaces

After an upgrade, you must compile the code in each namespace so that it runs under the new version of InterSystems IRIS. If the compiler detects any errors, you may need to recompile one or more times for the compiler to resolve all dependencies.

After your code compiles successfully, you may want to export any updated classes or routines, in case you need to run the upgrade in any additional environments. Importing these classes or routines during future upgrades allows you to update your code quickly, minimizing downtime.

Note: If you are using a manifest as part of your upgrade process, you can compile your code from within the manifest. See the instructions in the Perform Post-Upgrade Tasks section of the Creating and Using an Installation Manifest appendix of this guide.

6.2.1.1 Compiling Classes

To compile the classes in all namespaces from the Terminal:

```
do $system.OBJ.CompileAllNamespaces("u")
```

To compile the classes in a single namespace from the Terminal:

```
set $namespace = "<namespace>"
do $system.OBJ.CompileAll("u")
```

To compile the classes in the %SYS namespace from the Terminal:

```
set $namespace = "%SYS"
do $system.OBJ.CompileList("%Z*,%z*,Z*,z*","up")
```

Note: If your namespaces contain mapped classes, include the `/mapped` qualifier in the call to **CompileAllNamespaces()** or **CompileAll()**:

```
do $system.OBJ.CompileAllNamespaces("u /mapped")
do $system.OBJ.CompileAll("u /mapped")
```

6.2.1.2 Class compiler version utility

To assist customers in determining which class compiler version a class or classes in a namespace have been compiled with, InterSystems provides two assists

- Method – $System.OBJ.CompileInfoClass(<classname>)

  This method returns the version of the class compiler used to compile this <classname> and the datetime the class was compiled

- Query – $System.OBJ.CompileInfo(<sortby>)

  This query generates a report for the current namespace that includes all classes, the version of the compiler used to compile each one, and the datetime each was compiled. The first argument <sortby> may have the following values:

  – 0 – the time the class was compiled

  – 1 – the class name

  – 2 – the version of InterSystems IRIS the class was compiled in

6.2.1.3 Compiling Routines

To compile the routines in all namespaces from the Terminal:

```
do ##Class(%Routine).CompileAllNamespaces()
```

To compile the routines in a single namespace from the Terminal:

```
set $namespace = "<namespace>"
do ##Class(%Routine).CompileAll()
```

To compile the routines in the %SYS namespace from the Terminal:

```
set $namespace = "%SYS"
do ##Class(%Routine).CompileList("%Z*,%z*,Z*,z*","up")
```