InterSystems IRIS for Health Release Notes

Version 2020.3
2021-02-04

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

InterSystems IRIS for Health Release Notes
InterSystems IRIS Data Platform Version 2020.3 2021-02-04

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

About This Book

This book describes the major features that have been added to InterSystems IRIS for Health™ 2020.3, as well as information needed to update custom code from previous versions.

It contains the following sections:

- New and Enhanced Features for InterSystems IRIS for Health 2020.3
- General Upgrade Information
- Compatibility Checklist for InterSystems IRIS for Health 2020.3 — If you are using custom code developed on an earlier version or are accessing durable database or directories that have been used with a previous version, read this section.
- Known Issues and Notes — Describes issues in InterSystems IRIS for Health 2020.3 that you should be aware of and other notes related to the release

General Licensing Notes

InterSystems makes its products and features available under license to customers. While InterSystems may or may not enforce the use of said products or features consistent with the purchased license capabilities, customers are expected to comply with terms of their licenses. Moreover, InterSystems may tighten enforcement at any release without notice.

Developers must be aware that certain license types are required in order to use specific product features such as Multi-Server capability, Mirroring, and Web Services features. The specific requirements are noted in the InterSystems Price List and the Terms and Conditions for licensing. These are available from your local InterSystems representative.

Application Use Of InterSystems Web Server

InterSystems installs an Apache-based web server (often referred to as the "private web server") to assure that the management portals for its products are always available. The private web server is built and configured to meet the management needs of InterSystems administrative servers and is configured to only connect to InterSystems products. The options selected are not, in general, suitable for production use - in particular, security is minimal and the options used are generally unsuitable for a high volume of HTTP requests. Testing, by InterSystems, of the private webserver only covers use of the private web server for managing InterSystems IRIS, HealthShare, and other InterSystems products.

Customers are not required to use this web server to manage our products. You may also use a web server of your choice, located on whatever server you elect to use. The private web server is provided as a convenience to simplify installation and installation dependencies. Many developers also find it useful to use the private web server for unit testing.

UNIX®

The parameters used for the UNIX® build are:

```
--prefix=<installation_location>
--disable-actions
--disable-asis
--disable-auth
--disable-autoindex
--disable-cgi
```

```
--disable-cgid
--disable-charset-lite
--disable-dir
--disable-env
--disable-imap
--disable-include
--disable-negotiation
--disable-setenvif
--disable-shared
--disable-status
--disable-userdir
--enable-access
--enable-alias
--enable-log-config
--enable-mime
--enable-so
--without-berkeley-db
--without-gdbm
--without-ndbm
```

The server produced has defaults using the Apache Group's prefork Multi-Processing Module (MPM). This is the non-threaded server model. The number of requests that can be concurrently served is directly related to the number of Apache worker processes in the pool. The private web server is configured to occupy the smallest possible footprint by allowing a maximum of two worker processes to be created for the pool. The following settings will be found in the Apache configuration (httpd.conf) for the server:

- MinSpareServers: 1
- MaxSpareServers: 2

By contrast, the default Apache configuration for a production grade build is usually the following:

- StartServers: 5
- MinSpareServers: 2
- MaxSpareServers: 20
- ServerLimit: 256
- MaxClients: 256

This configuration will allow Apache to create 5 worker processes at start-up time, increasing to a maximum of 256 as the concurrent load increases. Because of these differences in configuration, the performance of the private web server will be noticeably inferior to that of a production grade Apache build as the concurrent load increases.

Conclusion

If you expect very low volume of HTTP traffic, have limited demands for high availability and secure operation, the private web server may be suitable for your deployment needs. However, if you expect a high amount of HTTP traffic, require high availability in your web server, need to integrate with other sources of web information, or need a high degree of control over your web server, InterSystems recommends installing your own separate copy of Apache, ideally on its own server, and configuring it to use our Web Gateway to communicate with InterSystems products. Review the options above to determine if this is so.

1

New and Enhanced Features for InterSystems IRIS for Health 2020.3

This document describes the new and enhanced features in the 2020.3 release of InterSystems IRIS for Health™, which is a continuous delivery release. The enhancements in this release provide improved FHIR client APIs, improved HL7 migration tools, and features that make it easier to develop and deploy real-time, machine learning-enabled applications that bridge data and application silos.

1.1 Enhancements that Improve Health Interoperability and HL7 Migration Tooling

This release provides the following enhancements to the health-related features:

- New APIs for sending and receiving FHIR request/response messages, allowing your application to perform client-side FHIR operations. For more details, see FHIR Clients.
- eGate Support in the HL7 Migration Tooling. Migrates transformation logic from the eGate interface engine to InterSystems IRIS for Health. For more details, see HL7 Migration Tool.

1.2 Enhancements that Improve Deployment and Operations Experience

This release provides the following enhancements to the deployment and operations experience, both in the cloud and on-premises:

- Configuring a Kubernetes cluster is much easier with the new InterSystems Kubernetes Operator (IKO). See "Using the InterSystems Kubernetes Operator."
- The InterSystems Cloud Manager (ICM) adds support for InterSystems API Manager deployments. See "Deploying InterSystems API Manager."
- Asynchronous mirroring support for sharded clusters.

- You can now manage Work Queues from the System Management Portal.

1.3 Enhancements that Improve Developer Experience

This release provides the following enhancements to the developer experience, including new facilities, higher performance, and compatibility with recent versions of key technology stacks:

- Python Gateway — This release extends the dynamic object gateway to allow you to call Python code from ObjectScript and provides forward and reverse proxy access to Python objects. In previous releases, the dynamic object gateway only supported calls to Java and .NET.
- Support for JDBC and Java Gateway reentrancy.
- .NET Gateway now supports .NET Core 2.1.
- XEP adds support for deferred indexing and indexes can be built as a background process. See “ Controlling Index Updating . ”
- Support for Spark 2.4.4.

1.4 IntegratedML Machine Learning

This is the first InterSystems IRIS for Health release that includes IntegratedML, a new feature that brings “best of breed” machine learning to analysts and developers via simple and intuitive SQL syntax. Developers can now easily train and deploy powerful predictive models from within IRIS, right where their data lives. For details, see Using IntegratedML and Learn IntegratedML in InterSystems IRIS.

For this release:

- • Standard and Community Edition containers are available from the InterSystems Container Registry (ICR), See Using the InterSystems Container Registry for information on the container registry.
- • Community Edition containers are also available from Docker Hub.
- • Kits (and container tarballs) are available from the WRC Software Distribution site.

Note: Full installation kits are provided for a subset of server platforms on the WRC, which will give customers who do not use containers the option to use IntegratedML now, with the option to upgrade to the 2021.1 Extended Maintenance release.

1.5 Other Enhancements and Efficiency Improvements

In each release, InterSystems makes many efficiency improvements and minor enhancements. In this release these improvements include:

- Improved the consistency of the APIs in $system.SQL. See %SYSTEM.SQL.Functions for a list of the new functions and %SYSTEM.SQL for a list of deprecated functions.
- You can use TSQL through JDBC.

- Node.js Native API now includes the List class. See “ Native API Quick Reference for Node.js . ”
- Java Messaging Service (JMS) adapter is able to connect to a broader range of servers.
- InterSystems IRIS on Linux has been enhanced to use Asynchronous I/O for writes to database files, as it always has on all UNIX® and Windows platforms. This is coupled with automatic use of direct I/O instead of buffered I/O. This change optimizes the disk I/O characteristics for database files in the following ways:
  - Improves application responsiveness at higher scaling levels by more fairly sharing I/O bandwidth with database reads and journal writes.
  - Improves integrity check performance by allowing integrity check to read multiple blocks asynchronously.
  - Improves effectiveness of asynchronous reads performed by $prefetchon.

1.6 Continuous Delivery Releases of InterSystems IRIS for Health

InterSystems IRIS for Health 2020.3 is a continuous delivery release. There are now two streams of InterSystems IRIS for Health releases:

- Extended maintenance (EM) releases — These are annual releases and provide maintenance releases. These releases are ideal for large enterprise applications where the ease of getting fixes in maintenance releases is more important than getting early access to new features.
- Continuous delivery (CD) releases — These are quarterly releases provide that access quickly to new features and are ideal for developing and deploying applications in the cloud or in local Docker containers.

The quarterly schedule of continuous delivery releases reduces the time between when a customer requests a feature and we deliver it to them. Having regular schedules for both the continuous delivery releases and the major extended maintenance releases provide customers with a predictable schedule that helps them plan and schedule updates.

Continuous delivery releases are provided in container format and are available on Amazon Web Services (AWS), Google Cloud Platform (GCP), Microsoft Azure, Docker Hub, and the InterSystems WRC download site. You can run a continuous delivery release on any of these cloud platforms or a local system using Docker container. InterSystems does not provide maintenance releases for continuous delivery releases, but instead fixes issues in subsequent continuous delivery releases.

The extended maintenance releases are provided on all Supported Platforms Guide, including UNIX®, Windows, the cloud platforms, and the Docker container.

If your application runs on a non-container platform, you can only use an extended maintenance release for that application but can consider using the continuous delivery releases for:

- Evaluating new features and testing your custom code — this will reduce your upgrade costs when you upgrade to the next extended maintenance major release.
- Using it for new projects that can be deployed in the cloud or in local containers.

In addition to providing fully-suppported releases, InterSystems provides access to prerelease software for developers who want to get an early look at new features.

2

General Upgrade Information

This section provides information on upgrading from earlier versions. The ultimate goal of InterSystems is to have a release which can be installed with no, or little, effect on the applications it supports.

2.1 Important Considerations

2.1.1 Compatibility

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 the release-specific changes, 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.

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

2.1.2 Preview Release

Toward the end of development for each release, InterSystems may make available a preview version of the product to its customers. Notifications of the preview are published on the website and on public blogs. The purpose of this is two-fold:

- It provides an early opportunity for customers to determine how the changes and enhancement in the release affect existing applications, to report issues found, and verify those issues have been resolved.
- It also provides early exposure of new features. Customers have a chance to try out the proposed ideas and give feedback on the usefulness of this feature for their business area.

InterSystems strongly encourages customers to plan on obtaining a preview release and to test their application against it.

Important: InterSystems does not support upgrading from a preview version.

Unresponsive Systems

One of the goals for preview release is to expose the new release to real-world operating challenges to assure its reliability. Therefore, it is possible, although unlikely, that an unanticipated sequence of events may render InterSystems IRIS for Health unresponsive. In this situation, it is extremely important to gather system diagnostic information while in the hung state for InterSystems to analyze. Should an instance of InterSystems IRIS for Health become unresponsive,

- Log in as an administrator
- In a terminal window, run the IRISHung script in the directory, <install-dir>/bin.

  The scripts corresponding to supported systems are:

  – Windows: IRISHung.cmd

  – UNIX®, Linux, AIX, and so on: IRISHung.sh

- Send the resulting output file to the InterSystems Worldwide Response Center. You can email the file to support@inter-systems.com, open a new problem using the WRC Online, or call the Center directly for additional assistance.

2.2 Upgrade Specifics

This section contains specific instructions applicable to this transition.

2.2.1 Upgrading Containers

Because a containerized application is isolated from the host environment, it does not write persistent data; whatever it writes inside the container is lost when the container is removed and replaced by a new container. Therefore, an important aspect of containerized application deployment is arranging for data to be stored outside of the container and made available to other and future containers.

The durable %SYS features enables persistent storage of instance-specific data — such as user definitions, audit records, and the log, journal, and WIJ files — when InterSystems IRIS for Health is run in a container, allowing a single instance to run sequentially in multiple containers over time. For example, if you run an InterSystems IRIS for Health container using durable %SYS, you can upgrade the instance by stopping the original container and running a new one that uses the instance-specific data created by the old one. For information about upgrading, see Upgrading InterSystems IRIS Containers; for detailed information on durable %SYS, see Durable %SYS for Persistent Instance Data.

Important: In this release, the distribution container has a nonroot default user. This improves the security of your container. If you are using a durable %SYS from a 2019.2 or earlier instance with this 2020.3 release, you need to change some file ownerships in the host's durable directory before running InterSystems IRIS for Health 2020.3. Please contact your InterSystems sales engineer or the InterSystems Worldwide Response Center for instructions on changing the file ownerships. If you do not make these changes, the container will encounter an error starting InterSystems IRIS for Health.

2.2.2 Classes

InterSystems recommends that customers recompile all their classes contained in each namespace. This will assure that:

- Subclasses derived from the InterSystems product library will see improved product behavior if a method call results in executing code in its superclass(es).
- All embedded SQL will use the latest versions of the SQL infrastructure.

- All projections involved in language bindings will be updated.
- All generated routines and classes will be updated.

2.2.2.1 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 for Health the class was compiled in

2.2.3 Routines

ObjectScript routines do not need to be recompiled after upgrade with the following exception:

- Routines containing embedded SQL must be recompiled.

2.2.4 Cached Queries

Cached queries are always purged during upgrade. They are recompiled and cached as needed.

2.2.5 Web Services and SOAP

It is not necessary to re-import any Web Service Definition (WSDL) files.

2.2.6 Frozen Plans for SQL Queries

When you upgrade InterSystems IRIS for Health 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. For details, see "Software Version Upgrade Automatically Freezes Plans"

in the *InterSystems SQL Optimization Guide*.

3

Upgrade Compatibility Checklist for InterSystems IRIS for Health 2020.3

The purpose of this chapter is to highlight those features of the 2020.3 release of InterSystems IRIS for Health™ that, because of their difference in this version, affect the administration, operation, or development activities of existing systems.

This document addresses the differences between the 2020.2 and 2020.3 versions of InterSystems IRIS for Health. Customers upgrading their applications from earlier releases are strongly urged to read the upgrade checklist for the intervening versions as well:

3.1 Administrators

The following changes are of special interest to those who are familiar with administering prior versions of InterSystems IRIS and wish to learn what is new or different in this area for version 2020.3. The items listed here are described in the following sections.

- Cloud Containers — Changes for the Arbiter Container with ISCAgent Using a Nondefault Port
- Installation — Change to Default SuperServer Port
- Interoperability Productions — Activity Monitoring Changes
- Interoperability Productions — Message Viewer Changes for XML Objects
- Interoperability Productions — Removal of Deprecated Java Business Hosts, Use PEX Instead
- InterSystems Cloud Manager — JDBCGatewayPort Change
- System — Memory Usage Changes for Global and Routine Buffers

3.2 Developers

The following sections describe compatibility issues that are of interest to developers and administrators. Since developers typically administer development system, developers should read all issues.

3.2.1 Class Compiler — Error Handling Changes

This release improves the class compiler error handling of SQL relationship queries. Some errors that were not reported are now reported; consequently applications may behave differently when compiled or executed. But these new error messages can identify errors in relationship queries that should be fixed to achieve the intended results.

3.2.2 Class Compiler — Locking Changes

In this release, the class compiler always uses a lock when compiling or loading to ensure that the operation is not interrupted by other events; consequently, the 'l' flag and the /lock qualifier are deprecated and will be ignored. This change should not impact any existing user code or procedures other than by avoiding errors in the operation.

3.2.3 Cloud Containers — Changes for the Arbiter Container with ISCAgent Using a Nondefault Port

The arbiter is an independent system hosting ISCAgent for mirror members (see " Arbiter "). By default ISCAgent uses port 2188. In previous releases you could specify a nondefault port by arguments such as `$IRISSYS/startISCAgent.sh 2000` , but in this release you must replace these arguments with `-p 2000` to use port 2000. Using arguments other than `-p` will be ignored and the container will run with port 2188.

3.2.4 Installation — Change to Default SuperServer Port

In this and future releases, the default SuperServer port is 1972. If this port is not available, then the default port will be 51773 or the next free port after this. In previous releases, the default SuperServer port was 51773.

3.2.5 Interoperability Productions — Activity Monitoring Changes

The activity monitoring pages (see " Monitoring Activity Volume ") now distinguish between days based on the combination of UTC and local time. This means when querying the Ens_Activity_Data.Days table for a specific TimeSlotUTC or TimeSlot it is necessary to check if there are 2 rows per unique TimeSlotUTC or TimeSlot : One row when TimeSlot is different from TimeSlotUTC and one row when TimeSlot is the same as TimeSlotUTC. The actual value is the sum of these two rows.

3.2.6 Interoperability Productions — Creating Interoperability Namespace with %Installer Changes

In previous releases if you used the %Installer.Installer class to make an interoperability production enabled namespace, it would set the node to a fixed portal URL. It now sets it to the empty string, which is the typically desired behavior. If you wish to set the node to a fixed portal URL, you can do this with the following:

```
Set ^%SYS("Ensemble","InstalledNamespace",tNSUpper)="<fixed-url>"
```

3.2.7 Interoperability Productions — Custom Business Process Accessing %MessageSent or %MessageReceived Changes

In this release if custom business process code accesses the internal %MessagesSent or %MessagesReceived array then the class parameter SKIPMESSAGEHISTORY needs to be overwritten and set to 0.

3.2.8 Interoperability Productions — Message Viewer Changes for XML Objects

To improve efficiency in displaying XML-enabled objects, large XML objects may not be fully serialized in the message viewer Contents tab. To see the fully serialized value, you must now use the Full Contents tab.

3.2.9 Interoperability Productions — PEX Class Changes

The PEX class hierarchy has changes to reduce errors when users overloaded methods rather than overrode them. In this release, all methods that should be overrode by user code are marked as abstract. If you define a subclass and do not override these methods, you will get a compile error. In previous versions, the compile would succeed but the user callback code would not execute correct because the default code would be called instead. The documentation stated that these methods were abstract and needed to be overridden but it was not enforced by the code.

3.2.10 Interoperability Productions — PEX Library Changes

This release uses a different serialization library; it no longer uses the jackson library. Consequently, your production settings should point to the intersystems-utils-3.1.0.jar provided and not to the jackson library. If your code references the jackson library, you must update it or install the jackson library.

3.2.11 Interoperability Productions — RecordMap Error Handling Changes

The error detection for the RecordMap service is now stricter with the ParseOnly flag selected. In previous releases, errors would be ignored with ParseOnly even if the AlertOnError flag was enabled. In this release, if the AlertOnError flag is enabled and a record fails validation, then an alert is sent even if ParseOnly is selected.

3.2.12 Interoperability Productions — X12 Changes

There are several changes in how X12 documents are handled:

- Change handling poorly formatted documents — Improved handling of certain poorly formatted documents means that the issues with these documents are logged in the Event Log and then handled by sending an appropriate Reply document, rather than these documents resulting in the service failing and documents not being deleted. If you are relying on service failures to identify flaws in the X12 document, you should look at the Reply document instead.
- Validation change — This change now includes code values in validations that were previously ignored. If you are using validation with the flags r, u, or t, then some errors that would have been missed in previous versions will now be caught. This change also adds a new segment, CTX-11, to the validation style schema for HIPAA_5010:999. This does not impact any code for setting the CTX-5() segment, but does mean that if a user has code for getting values from the CTX segments, and the document they are inspecting has a Business Unit Identifier, then that CTX segment will have to be accessed using path 2000().2100().CTX-11 instead of the 2000().2100().CTX-5() which is used for Segment Context. This makes no difference in either setting or getting values using the new style schema.
- Change in allowed order of segments — This change allows segments to be in any order permitted by the standard rather than requiring them to be in the specific order as listed in the standard. This allows valid documents to pass validation where in previous releases, they would have failed. This change should not require any change in your code unless you were using the wrong property path to get the value from some segment to workaround this incorrect behavior, in that case you need to switch to using the correct property paths.

3.2.13 Interoperability Productions — XML VDoc Character Escape Changes

In XML VDoc, when you use setValueAt() to set the value at a node, you can specify a value that consists of mixed content (that is, a value that consists of a mix of element and text nodes). This change ensures that if a < (left-angle) character appears without a closing XML /> or </, then the < is treated as normal text and is XML escaped (replaced by <). In previous releases, this would have been treated as mixed content and would have caused an error. If you have code to handle this expected error, you should examine this case. See “ Using Mixed Content When Setting Paths . ”

3.2.14 Interoperability Productions — Removal of Deprecated Java Business Hosts, Use PEX Instead

Java Business Hosts was deprecated in release 2019.3 and has been removed from this release. PEX replaces the Java Business Hosts feature. For information on migrating existing Java Business Hosts productions to use PEX, see the community article Migrate from Java Business Host to PEX. For a description of PEX, see PEX: Developing Productions with Java and .NET.

3.2.15 InterSystems Cloud Manager — JDBCGatewayPort Change

If you were overriding the JDBCGatewayPort parameter using the JSON field "JDBCGatewayPort", you should delete that parameter and instead do so directly in their custom CPF file (pointed to by "UserCPF"). For example:

```
[Gateways] %JDBC Server=JDBC,127.0.0.1,53773
```

3.2.16 Language Bindings — IRISNative and Gateway Changes

This release contains the following changes to the language bindings IRISNative API and to the gateways:

- Access to the IRISReference class — you now have to use the accessor methods getValue() and setValue() to access the value. Previously, you could access the value directly
- Object returned as IRISList — if a user method returns an object of IRISList, the IRIS side now gets a $list literal. Previously, the IRIS side would get a proxy object of %Net.Remote.Object. If your code is expecting an object of %Net.Remote.Object, you need to change it. There were significant problems including inefficiency with the previous behavior.
- Private classes — in this release you cannot access private Java or .NET classes using public properties or methods. You must change the private classes to be public in Java or .NET if you want to continue using those properties and methods. This change fixes some errors and inconsistencies accessing private classes with public properties or methods.
- %Net.Remote.Object property name change — in this release the %Net.Remote.Object internal Gateway property is renamed to %gateway. This is an internal property and should not be used, but if your code accesses it, you should change it to use the new name.
- Empty strings — in this release, the language bindings map an empty string in the external language to an empty string in IRIS. In previous releases, these were mapped to `$c(0)` .

3.2.17 Language Bindings — Removing Ability to Connect to Caché from .NET Provider or ODBC

In order to enable new features, neither the .NET Provider nor ODBC supports connections to Caché instances in this release.

3.2.18 Language Bindings — New Version of XEP Compatibility Issues

This release has a new version of XEP support, which includes enhanced indexing and a new underlying client/server communication method. Consequently, you cannot use a XEP from a previous version of InterSystems IRIS with this version as the server and you cannot use this version's XEP with a previous version as the server. In addition, there may be some minor code changes that you need to make to any code that called the previous version of XEP support. See " XEP Requirements and Configuration " for details. For example, there is the following change:

XEP updateObject() method — The updateObject() method will now throw an exception if the object to be updated does not exist in the extent of the class. If your application depends on updateObject reverting to insert when the object does not exist then you will need to update the application to catch the exception and properly invoke the store() method to insert a new object.

3.2.19 Language Bindings — 32-bit ODBC Driver Removed on UNIX®

In this release, the 32–bit ODBC drivers libirisodbcu35.so and odbcgatewayu.so have been removed on UNIX® systems. If you have used these drivers, you should use the 64–bit equivalents libirisodbcur6435.so and odbcgatewayur64.so. If you are switching over to the 64-bit unixODBC client driver (libirisodbcu6435.so), you should also make sure you are using the 64-bit unixODBC driver manager and that your applications are built against 64-bit unixODBC. If your app or driver manager is 32-bit and is trying to use the 64-bit driver, this could cause unexpected problems. This change only impacts the 32-bit ODBC drivers built against the unixODBC driver manager. The 32-bit iODBC-based drivers are still included in the installation.

3.2.20 Language Bindings SQL.JDBC — API Cleanup

Typically, developers use the standard JDBC APIs, which are unchanged in this release, but some proprietary public classes and methods have moved or become more restricted in access permissions. These accessible proprietary classes and methods may have been used by customers in their applications, and will need to be updated. For example, if your code is accessing directly the members of the ConnectionParameters class, you must update your code and use the getter and setter methods.

3.2.21 Objects — IDENTIFIEDBY Deprecated Use Parent/Child Relationships

The %Library.Persistent IDENTIFIEDBY parameter is deprecated. If you have used IDENTIFIEDBY to define relationships, you should replace it with a parent/child relationship.

3.2.22 Security — On Red Hat Linux Only Support libcrypto.so.1.1 FIPS Mode in Version 8.2 and Later

On Red Hat Linux, you can only use libcrypto.so.1.1 FIPS mode on Red Hat Linux version 8.2 and later. You cannot use this mode on earlier versions.

3.2.23 SQL — SELECT with INTO Clause Changes

This release contains a change to the behavior of an embedded SELECT statement with an INTO clause. If the execution of the SELECT statement results in SQLCODE=100, or in the case of a FETCH that results in SQLCODE=100, any INTO variables in the statement will be set to null ("") if they are defined. If the INTO variable was undefined prior to the execution of the SELECT, and the SELECT does not set the variable (or any previous FETCH does not set the variable for cursor selects), the variable will remain undefined. If the variable was defined prior to the SELECT, or the SQL SELECT or FETCH sets the variable, the variable will be set to "" when SQLCODE=100 or SQLCODE<0 is returned.

This is a change that might result in some applications requiring updating in order to avoid application errors, or incorrect results returned.

3.2.24 SQL — Remove Unused Environment Variables

Some environment variables that were not used by SQL or the class compiler have been removed. These were not used by the product, but if you have referenced them, you should remove them from your application. The following variables are no longer defined in 2020.3: envRoutineSizeGet, envUseOldJavaGeneratorGet, envUseJavaGenerator1Get, envUseJavaGenerator2Get, envUseJavaGenerator3Get, envUseSortGet, and envTimeChangedGet. The following variables are still defined but return an error if you try to update the setting: envDefaultOHandleClassNameGet, envDefaultSerialStateClassNameGet, and envDefaultStorageClassNameGet.

3.2.25 SQL — Change to TuneTable and TuneSchema Defaults

In this release, the second argument of $SYSTEM.SQL.TuneTable and $SYSTEM.SQL.TuneSchema, update, now has a default=1. Prior to this change, the default was 0 for update. Calling `$SYSTEM.SQL.TuneTable(` *tablename* `)` and `$SYSTEM.SQL.TuneSchema(` *schemaname* `)` without specifying a value for update will result in a change of behavior from previous versions.

3.2.26 System — $ZREFERENCE Variable Changes

In this release, the $ZREFERENCE variable, which contains the last global referenced, can have a different value after running %ETN error trap utility than it had in the same context in previous releases. If your code is parsing the $ZREFERENCE value to determine where the error information was written, you must modify your code to determine this information using the return value from LOG^%ETN or BACK^%ETN, which is a $LIST of the first two subscripts in the ^ERRORS global.

3.2.27 System — GLOSTAT Changes

As part of a project to improve the performance of counters, certain GLOSTAT statistics have been changed:

- %SYS.ProcessQuery property 'DataBlockWrites' and the equivalent $zu(67,38,pid) now counts any database block queued for writing by this process, whereas it previously counted only blocks internally typed as data blocks. This more accurately reflects the write load induced by this process than before (without adding the unnecessary complication of an additional statistic).
- The undocumented $zu(190,6,0) and $zu(190,6,7) functions to zero system statistics are removed and now throw <FUNCTION>.
- The iris stat -p32 columns formerly labelled 'rfcnt' and 'bfhit' are now labelled 'phyrd' and 'wdqbk' and they reflect, for the process in that row, the number of database reads and database blocks queued respectively

3.2.28 System — Memory Usage Changes for Global and Routine Buffers

This release improves performance on newly installed systems where the database cache size has not been configured. Under most circumstances, you should carefully configure cache sizes and Configure Huge and Large Pages for optimal system performance. Configuring cache sizes is especially important for live production systems, systems with heavy loads, and systems with multiple instances.

However, when users initially evaluate InterSystems IRIS for Health, they typically run it without configuring the database cache size. The new behavior in allocating memory on systems where cache size has not been configured is:

1. When database cache (global buffers) is unspecified in the configuration, the instance auto-selects 25% of total physical memory for database cache. Previously, the calculation used 12.5% with an upper limit of 1GB (so effectively a flat 1GB for most systems). There is now no upper limit imposed. Additionally, this changes the calculation on Windows systems to be based on physical memory rather than "available" memory so that the behavior is consistent across restarts (and also consistent with Linux and UNIX® platforms).

   While this may result in a large allocation, If the instance isn't used to load a large amount of data, then most of this memory is not touched and, in most operating system environments, will not occupy physical memory. If the operating system has enough large pages available for this memory allocation, InterSystems IRIS for Health will use them, and thus will occupy physical memory. In this case, however, that large page memory is known to be available. Control over use of large pages is available via the memlock configuration parameter .
2. On startup, if database cache is unspecified, a message "Global buffer setting requires attention. Auto-selected 25% of total memory." will be emitted.
3. Leaving routine cache (routine buffers) unspecified in the configuration no longer means a minimal amount routine cache (36MB) but instead is taken as 10% of the amount of global buffers with a minimum of 80MB and maximum of 1020MB. (More precisely, it's the number of global buffers multiplied by 8KB multiplied by 10%, so the simple 10% applies as long as only 8KB-size global buffers are used). This applies regardless of whether the database cache was configured or auto-selected as described above. In this way, most systems may be able to select only database cache size and leave routine cache size unaltered.
4. The absolute minimum database cache allowed is changed from 1.6MB to 16MB. The previous minimum could not accommodate the longest possible string.

3.2.29 System — Asynchronous I/O for Database Writes on Linux and UNIX® Systems

InterSystems IRIS for Health on Linux has been enhanced to use Asynchronous I/O for writes to database files, as it always has on all UNIX® and Windows platforms. This is coupled with automatic use of direct I/O instead of buffered I/O. This change optimizes the disk I/O characteristics for database files, allowing higher scaling on busy systems without compromising application responsiveness. Of course, synchronization still occurs at key points to guarantee data integrity.

This change may require attention in certain configurations as file system buffering may have allowed InterSystems IRIS for Health to perform acceptably with an under-configured database cache in previous versions. Check that database cache is sized appropriately on your system.

Additionally, InterSystems IRIS on Linux and all UNIX® platforms are enhanced to use asynchronous I/O for writing to the write image journal to provide optimal I/O characteristics. This change, however, requires no special attention.

For details, see “ Buffered I/O vs. Direct I/O . ”

3.2.30 Web Gateway — SSLCC_Protocol Parameter Changes

To allow for TLS 1.3, there are changes to the way you access the SSLCC_Protocol parameter. If you are using an external script or the Web Gateway registry methods (particularly %CSP.Mgr.GatewayMgr.GetServerParams or %CSP.Mgr.GatewayMgr.SetServerParams) to read/write the SSLCC_Protocol parameter in CSP.ini, you must adapt your code to use the SSLCC_Protocol_Min and SSLCC_Protocol_Max parameters instead.

4

Known Issues and Notes

This chapter describes issues that InterSystems is aware of in InterSystems IRIS for Health 2020.3 and notes related to this version of the product. These issues will be addressed in future releases.

4.1 Issue Using MultiValue Feature

InterSystems has identified some issues with InterSystems IRIS MultiValue. We are investigating those and will fix them in a future release. If you are upgrading a MultiValue based application from Caché or Ensemble to InterSystems IRIS, you should thoroughly test your application on InterSystems IRIS before upgrading a production system. If you encounter any issues or want more information about the issues we have encountered, contact your sales engineer or the InterSystems Worldwide Response Center.

4.2 EnsLib.HL7.Segment GetValueAt() 32 KB Limitation

The **GetValueAt()** method of the EnsLib.HL7.Segment class truncates values larger than 32 KB.

As a workaround, use one of the following methods:

- **GetFieldStreamRaw()**
- **GetFieldStreamUnescaped()**
- **GetFieldStreamBase64()**

See the entry for EnsLib.HL7.Segment in the *Class Reference* for details.