First Look: Connecting Systems in InterSystems IRIS Using Java Business Hosts

Version 2018.1
2018-01-31

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

First Look: Connecting Systems in InterSystems IRIS Using Java Business Hosts
InterSystems Version 2018.1 2018-01-31

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

First Look: Connecting Systems in InterSystems IRIS Using Java Business Hosts 1

1 Solving the Problem of Connecting Systems 1
2 How InterSystems IRIS Productions Connect Systems 1
3 Trying Connecting Systems for Yourself 4
3.1 Getting Your System Ready 4
3.2 Building the JAR Files 4
3.3 Creating a Production-Enabled Namespace 6
3.4 Creating the Credentials 7
3.5 Creating the Production and the Initiator and Generating the Business Hosts 8
3.6 Configuring the Production 8
3.7 Running the Production and Examining the Messages 9
4 For More Information about Java Business Hosts and Productions 10

First Look: Connecting Systems in InterSystems IRIS Using Java Business Hosts

This First Look helps you develop interfaces in Java that connect systems together with an InterSystems IRIS production. An InterSystems IRIS production is an interoperability framework for rapid connectivity and the development of new connectable applications. The production provides built-in connections to a wide variety of message formats and communications protocols. You can easily add other formats and protocols and use a graphic interface to define business logic and message transformations. Productions provide persistent storage of messages, which allow you to audit whether a message is successfully delivered. A production consists of business services, processes, and operations. Business services connect with external systems and receive messages from them. Business processes allow you to define business logic including routing and message transformation. Business operations connect with external systems and send the messages to them.

1 Solving the Problem of Connecting Systems

When connecting systems together, it can be challenging to get them to understand the other system's messages and documents. For example, consider the following problem:

- You have two separate systems: one is collecting data from multiple networked devices and the other is a work order system that tracks broken devices and the repair process.
- The current process depends on human intervention to monitor the devices and initiate the repair process. This has caused delays and is unreliable.
- You have been given the task to connect the two systems together: to monitor the data being collected and to automate initiating the repair process. You know how to detect faulty devices in the data collection system and know how to initiate a repair, but the two systems store data in incompatible formats even when the data represents the same item.
- You also need to record the actions when a repair is initiated from the data collection system.

You can solve this problem using an InterSystems IRIS production. It provides the framework for defining an interface that accepts messages from the data collection system, transforming the message into one that can be understood by the repair system, and then sending it to the repair system. It also stores a record of the message path.

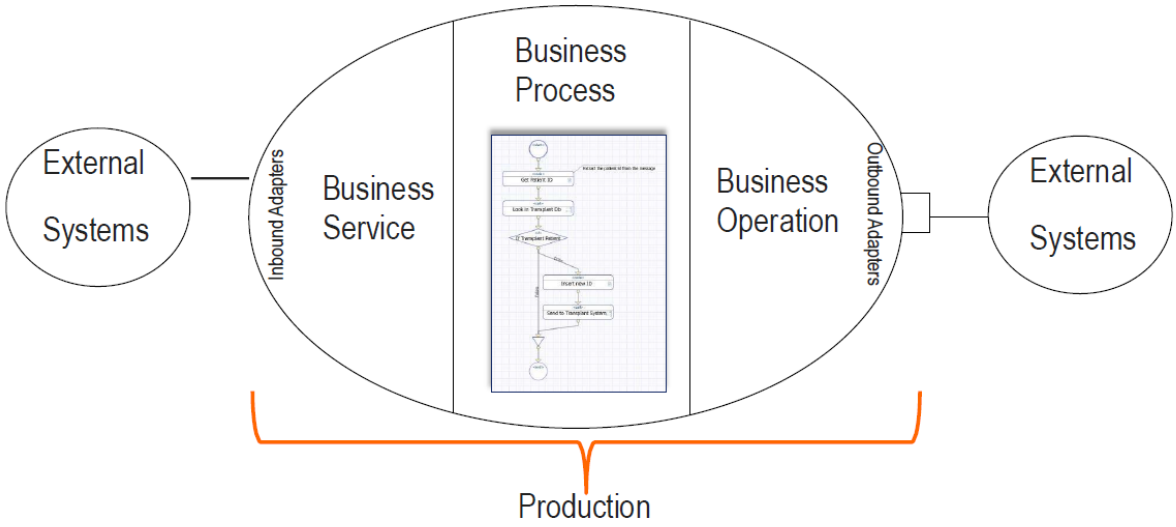
In this *First Look*, you will learn how to connect two Java programs with a simple production. For demonstration purposes, this document uses very simple Java code. A Java program for the data collection system or the work order system would be more complex and require a DTD schema, but you would use the same procedure to connect them with InterSystems IRIS.

2 How InterSystems IRIS Productions Connect Systems

In its simplest form, a production consists of:

- A business service that provides the interface for a message coming from an external system.
- A business process that provides any needed business logic and message transformation.
- A business operation that provides the interface for a message going to an external system.

The following illustrates a simple production:

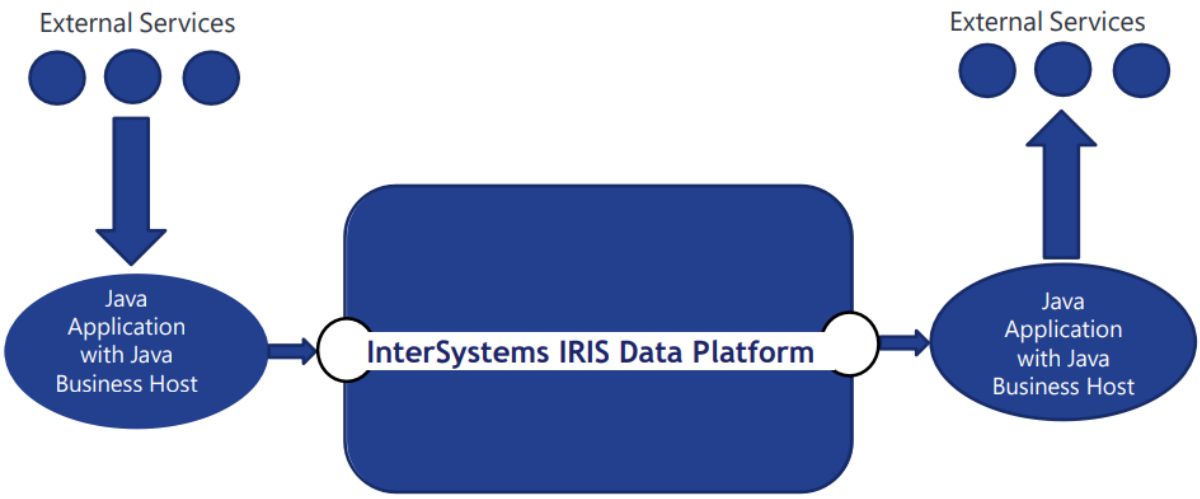


There are some business services and operations provided with InterSystems IRIS. If it has one that supports the message format that a system uses, you can avoid custom coding. But in many cases you will have to develop a custom business service and operation. You can develop these using the InterSystems IRIS ObjectScript or using Java.

Typically, the reason you choose to develop in Java is one of the following:

- There is an available Java library that parses the message format used by the system, and it is quicker to use the library rather than custom coding a parser for the message format.
- You prefer to develop custom code in Java rather than in InterSystems IRIS ObjectScript.

If you are developing a business service or operation in Java, you can use the Java Business Hosts feature to connect your Java code with the production. This allows you to do all of your business service and business operation coding in Java. The following illustration shows how the Java code connects to the InterSystems IRIS production:



You can use Java Business Hosts with the following kinds of messages:

- Plain text
- XML
- X12
- EDIFACT

To connect your Java code to the production, you have to implement the following classes and methods.

- For receiving messages from an external service, you implement a Java application that listens to messages and includes the Java class:

  `com.intersys.gateway.bh.BusinessService`

  with the following methods:

  – OnInit: this method is called when the production starts or the business service is enabled. It typically starts a listener that will receive messages. The listener receives the messages from the external service and then sends them to the business service in the production by calling the method Production.SendRequest(). The production is passed in as an argument to OnInit. Your code should save it so that it can call SendRequest in the listener.

  – OnTearDown: this method is called when the production is stopped or the business service is disabled. It typically stops the listener.

- For sending messages from the production to an external service, you implement a Java application, which includes the Java class:

  `com.intersys.gateway.bh.BusinessOperation`

  with the following methods:

  – OnInit: this method is called when the business operation starts. It typically initializes any structures needed by the OnMessage method. The OnInit method has an argument that specifies settings and values. These settings make it possible for your code to access the values set by the administrator in the Production Configuration page.

  – OnMessage: this method is called when the business operation receives a message. It is responsible for sending the message to the external service.

- OnTearDown: this method is called when the business operation ends. It typically releases any structures created by the OnInit method.

3 Trying Connecting Systems for Yourself

In this section, you will connect two Java hosts in a production. For demonstration purposes, these are very simple Java programs. Rather than getting messages from an external service, the business service just generates a random message. And the business operation writes the message to a log. Connecting to an external server requires more complex Java code, but you would follow the same process to connect the Java code to the production.

3.1 Getting Your System Ready

Before creating this example, you should do the following:

- Install a running, licensed instance of InterSystems IRIS.
- Install the Java Development Kit 8 (JDK) if you do not have it.
- Optionally, install the Eclipse Oxygen and the Eclipse IDE for Java Developers. This example uses this development environment, but you can use the JDK by itself or any other Java development environment to build the JAR files.
- Define the environment variables and paths required to build JAR files. On Windows systems, these are the JAVA_HOME and CLASSPATH environment variables. Typical values are:
  - JAVA_HOME: C:\Program Files\Java\jdk1.8.0_152
  - CLASSPATH: .;..;%JAVA_HOME\lib;C:\InterSystems\IRIS\dev\java\lib\JDK18\isc-gateway-3.0.0.jar
- Clone or download the FirstLook-JavaHosts sample code from github: https://github.com/intersystems/FirstLook-JavaHosts. If you're not familiar with github:
  1. Go to https://github.com/intersystems/FirstLook-JavaHosts in a web browser.
  2. Select **Clone or download**.
  3. Select Download FirstLook-JavaHosts-master.zip.
  4. Extract the files from the zip archive.
  5. In the FirstLook-JavaHosts-master\src\javahosts directory, open javahostsService.java and javahostsOperation.java in a text editor.
  6. In the next section, you will copy the contents of these files into stubs that you create with your Java IDE.

3.2 Building the JAR Files

This section covers using Eclipse to create the JAR files. You will use Eclipse to create a project, add the InterSystems IRIS JAR file that defines the interfaces, and add unimplemented methods. At this point, you would add your custom code. You can copy the custom code from the afl-javahosts example.

1. Open Eclipse and create a new Java project, give the project a name, such as javahosts and select **Next**.
2. Select the **Libraries** tab and select **Add External JARs ...** and add the isc-gateway-3.0.0.jar library to the project. This jar file is in the directory:

   install-dir\dev\java\lib\JDK18\

3. Then select **Finish**.
4. Add new Java classes named javahostsService and javahostsOperation.
5. After the business service class name, type:

```
implements BusinessService
```

6. Hover over “BusinessService” and add the import statement by selecting **import BusinessService**. Then hover over javahostsService and select **add unimplemented methods**. At this point, your business service class should look like:

```
package javahosts;

import com.intersys.gateway.bh.BusinessService;
import com.intersys.gateway.bh.Production;

public class javahostsService implements BusinessService {

 @Override
 public boolean OnInit(Production arg0) throws Exception {
  // TODO Auto-generated method stub
  return false;
 }

 @Override
 public boolean OnTearDown() throws Exception {
  // TODO Auto-generated method stub
  return false;
 }

}
```

7. After the business operation name, type:

```
implements BusinessOperation
```

   and then add the import statement for BusinessOperation and add the unimplemented methods. At this point, your business operation class should look like:

```
package javahosts;

import com.intersys.gateway.bh.BusinessOperation;

public class javahostsOperation  implements BusinessOperation {

 @Override
 public boolean OnInit(String[] arg0) throws Exception {
  // TODO Auto-generated method stub
  return false;
 }

 @Override
 public boolean OnMessage(String arg0) throws Exception {
  // TODO Auto-generated method stub
  return false;
 }

 @Override
 public boolean OnTearDown() throws Exception {
  // TODO Auto-generated method stub
  return false;
 }

}
```

8. In the next steps, you will replace the unimplemented methods with some simple sample code.
9. In the business service class, replace the entire javahostsService.java file with the code from the javahostsService.java in the FirstLook-JavaHosts sample.
10. In the business operation class, replace the entire javahostsOperation.java file with the code from the javahostsOperation.java in the FirstLook-JavaHosts sample.

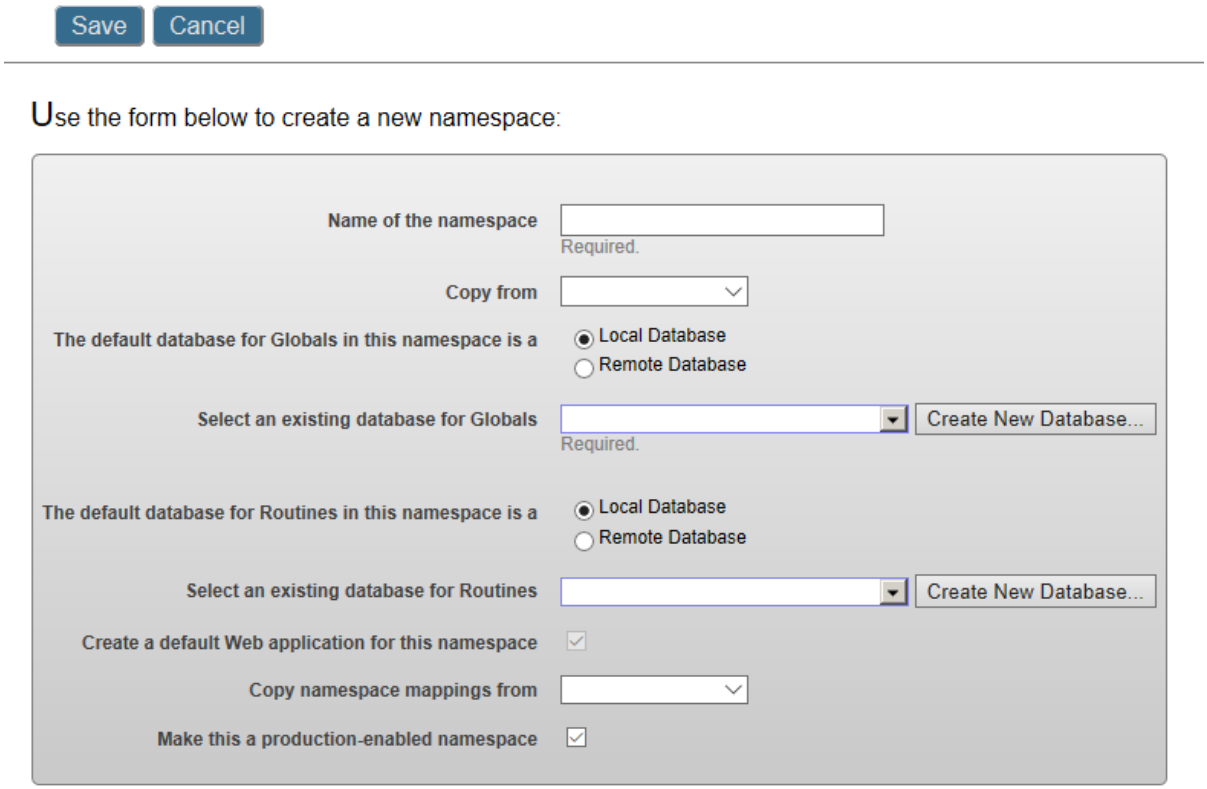
11. Generate the jar files. For the business service, export it as javahostsService.jar and for the business operation, export it as javahostsOperation.jar.

You've finished creating the JAR files, next you create the production-enabled namespace.

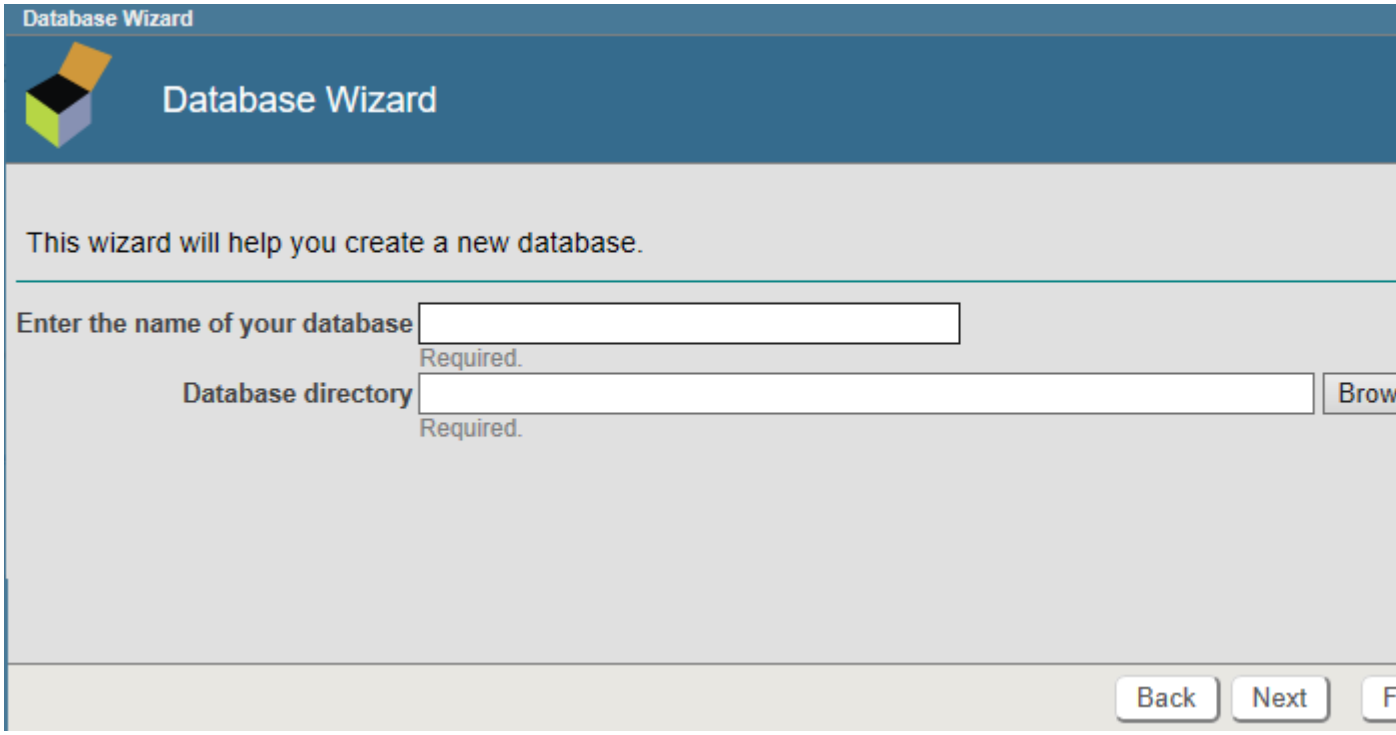
3.3 Creating a Production-Enabled Namespace

In order to create a production, you must have a production-enabled namespaces. The namespaces created when you first install InterSystems IRIS are not production-enabled. You will create a production-enabled namespace in this section. If you already have a production-enabled namespace, you can skip this step. In the Management Portal:

1. Select **System Administration** > **Configuration** > **System Configuration** > **Namespaces** to go to the **Namespaces** page.
2. On the **Namespaces** page, select **Create New Namespace**. This displays the **New Namespace** page:



3. On the **New Namespace** page, enter the name for the new namespace, such as `javahostsns`.
4. Next to the **Select an existing database for Globals** drop-down menu, select **Create New Database**. This displays the **Database Wizard**:



5. On the first page of the **Database Wizard**, in the **Enter the name of your database** field, enter the name of the database you are creating, such as `javahostdb`. Enter a directory for the database, such as `C:\InterSystems\IRIS\mgr\javahostsdb`. On that page, select **Next**.
6. On the next page, select **Finish**.
7. Back on the **New Namespace** page, in the **Select an existing database for Routines** drop-down menu, select the database you just created.
8. Ensure that the `Make this a production-enabled namespace` check box is selected.
9. Select **Save** near the top of the page and then select **Close** at the end of the resulting log.

You have now created an production-enabled namespace.

For more details about creating a namespace and its associated database, see "Create/Modify a Namespace" in the "Configuring InterSystems IRIS" chapter of the *InterSystems IRIS System Administration Guide*. For background information, see "Namespaces and Databases" in the *Orientation Guide for Server-Side Programming*.

3.4 Creating the Credentials

The Java code needs credentials to have access to the production. For this example, you can use the same InterSystems IRIS account that you use to develop a production. For a live system, you will create an account that has the privileges needed to run the production, but not any extra privileges.

To create the credentials, in the Management Portal:

1. Select a namespace that is enabled for productions.
2. Select **Interoperability** > **Configure** > **Credentials**.
3. Specify an ID, such as javahostsCredentials, and a user name and password for an account on the InterSystems IRIS system. Then select **Save**.

3.5 Creating the Production and the Initiator and Generating the Business Hosts

In this step, you will create a new production, include the Java Business Host initiator, and generate the business hosts. In the Management Portal:

1. Select **Interoperability** > **Build** > **Java Business Hosts**.
2. Select **Start New Production**, give the production a name, such as javahostsProd, leave the other fields with the default values, and select **OK** twice. The form should have a message indicating that the production is running and contains a Java Gateway Service. If you don't get this message, you probably have a problem with the environment variables or Java JDK installation.
3. Generate the business service host by:
    a. Select **Browse** and select the jar file generated for the business service.
    b. Select the name of the Java class, such as javahosts.javahostsService, from the dropdown menu.
    c. Accept the default ObjectScript class name, such as JBH.Javahosts.JavahostsService.
    d. For this sample, accept the default Format of Incoming Data, Plain Text. When you are developing your own code, select the appropriate format.
    e. Select the credentials that you created in the previous step from the dropdown menu.
    f. Select **Generate**.
4. Then generate the business operation host by:
    a. Select **Browse** and select the jar file generated for the business operation.
    b. Select the name of the Java class, such as javahosts.javahostsOperation, from the dropdown menu.
    c. Accept the default ObjectScript class name, such as JBH.Javahosts.JavahostsOperation.
    d. Select **Generate**.

You have completed creating the production and generating the business hosts. Next you add the business hosts to the production and configure them.

3.6 Configuring the Production

In this step, you will add the business operation and business service to the production and configure them. In the Management Portal:

1. Select **Interoperability** > **Configure** > **Production**.
2. Select the **Operations** plus sign to display the Business Operation Wizard.
    a. In the **Operation Class** drop-down menu, select the business operation, JBH.Javahosts.JavahostsOperation, that you generated in the previous step.
    b. Leave the **Enabled** check box clear.
    c. Select **OK**.
3. Select the JBH.Javahosts.JavahostsOperation in the production diagram and then in the **Settings** tab:
    a. Expand **Additional Settings** and in the **LogFile** field, enter a file path for the log file, such as c:\practice\javhost-slog.txt.

b. Select the **Enabled** check box.

c. Select **Apply**.

4. Select the **Services** plus sign to display the Business Service Wizard.

   a. In the **Service Class** drop-down menu, select the business service, JBH.Javahosts.JavahostsService, that you generated in the previous step.

   b. Leave the **Enabled** check box clear. You will enable the service in the next step.

   c. Select **OK**.

5. Select the JBH.Javahosts.JavahostsService service in the production diagram and then in the **Settings** tab:

   a. In the **Target Config Names** drop-down menu, select the JBH.Javahosts.JavahostsOperation operation.

   b. Select the **Enabled** check box.

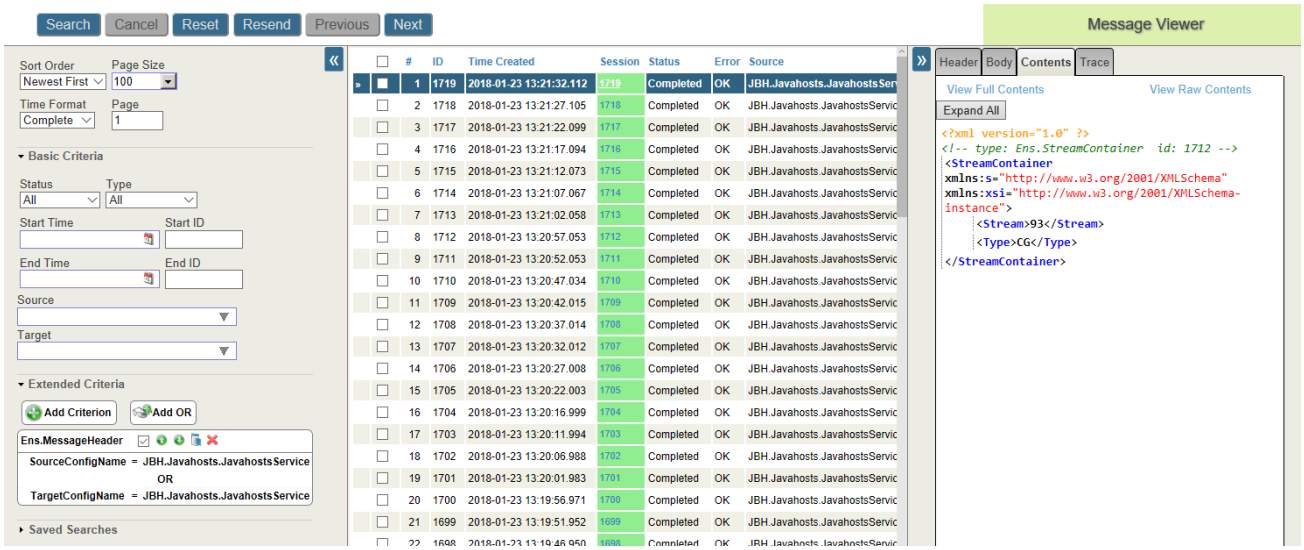
   c. Select **Apply**.

You've finished configuring the business hosts and the production. All the business hosts in the production diagram should be green and the production should be running. In the next section you will examine the messages.

3.7 Running the Production and Examining the Messages

Once you enabled the business service, the production started sending messages. To see the messages, select the **Messages** tab on the Production Configuration page. The messages are displayed as shown by:

To see the contents of a message, select **Go To Message Viewer**. Select **Search** in the message viewer, select a message, and select the **Contents** tab. The Message Viewer shows you the following:



The production continues to send messages. To stop the production:

- Select **Stop** on the Production Configuration page to stop the production.
- You can restart the production by selecting **Start**.

4 For More Information about Java Business Hosts and Productions

For more information about Java Business Hosts, see Java Business Hosts Presentation. For more information about productions, see:

- Introducing InterSystems IRIS Interoperability
- Developing Productions
- Configuring Productions