
# First Look: Data Transformations

Version 2020.3
2021-02-04

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

*First Look: Data Transformations*
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

**First Look: Data Transformations** ..... **1**

1 Data Transformation and Interoperability ..... 1

2 Exploring the DTL Editor ..... 1

2.1 Before You Begin ..... 1

2.2 Creating the Data Transformation ..... 4

2.3 Transforming the Data ..... 4

2.4 Compiling the Data Transformation ..... 8

2.5 Performing Preliminary Testing ..... 8

2.6 Adding Transformation to Production ..... 9

2.7 Testing the Data Transformation ..... 9

3 Other Important Features ..... 10

4 Learn More About Data Transformation ..... 11

# First Look: Data Transformations

This First Look helps you transform data with InterSystems IRIS® data platform integration productions.

To browse all of the First Looks, including those that can be performed on a free evaluation instance of InterSystems IRIS, see InterSystems First Looks.

# 1 Data Transformation and Interoperability

Data transformation is at the heart of the interoperability of InterSystems IRIS. You have the power to change the format and content of incoming data from one system to meet the requirements of a downstream system, allowing the systems to communicate with each other. Simply put, messages sent from one system can be transformed into messages that the other application can understand. With InterSystems IRIS, data transformations are simple to create, test, and maintain.

Suppose you have two retail systems that include the price of products. When System A sends data to other systems, it includes the base price of a product without tax added. However, System B needs the price to include taxes for the region. A data transformation in an InterSystems IRIS interoperability production can convert the price received from System A into the tax-added price before the data is sent to System B. Once defined, the data transformation takes care of the modifications automatically.

Common transformations include:

- Copying values from the source message to the target message.
- Performing calculations based on values of the source message and copying the results to the target message.
- Assigning new literal values to the target message.
- Rearranging the order of data.

# 2 Exploring the DTL Editor

You can write DTL (data transformation language) code to create a data transformation or you can create one using the DTL Editor. The DTL Editor allows non-technical users to create data transformations without having to write code. For example, its graphical user interface allows you to quickly map values from the source message to the target message with a drag-and-drop action. The following tour of the DTL Editor guides you through its major features by creating a data transformation within a production. For an introduction to InterSystems IRIS productions, see First Look: Connecting Systems Using Interoperability Productions.

In this demo, data about a student's final grade in a class must be converted so that a different application can use the data.

Want to try an online video-based demo of InterSystems IRIS interoperability features? Check out the Interoperability QuickStart!

## 2.1 Before You Begin

Before starting this tour of the DTL Editor, you need to do the following:

- Select an InterSystems IRIS instance.

- Download an InterSystems IRIS production and sample file from GitHub.
- Create an interoperability-enabled namespace.
- Import the downloaded production into the namespace.

### 2.1.1 Selecting an InterSystems IRIS Instance

To use the procedure, you will need a running InterSystems IRIS instance. Your choices include several types of licensed and free evaluation instances; the instance need not be hosted by the system you are working on (although they must have network access to each other). For information on how to deploy each type of instance if you do not already have one to work with, see Deploying InterSystems IRIS in *InterSystems IRIS Basics: Connecting an IDE*.

### 2.1.2 Downloading from GitHub

Download the production and data file that are used with this demo from https://github.com/intersystems/FirstLook-DataTransformations. The XML file is a representation of an InterSystems IRIS production that you will deploy on your system.

The FirstLook-DataTransformations sources must be accessible by the instance. The procedure for downloading the files depends on the type of instance you are using, as follows:

- If you are using an ICM-deployed instance:

  1. Use the **icm ssh** command with the **-machine** and **-interactive** options to open your default shell on the node hosting the instance, for example:

     ```
     icm ssh -machine MYIRIS-AM-TEST-0004 -interactive
     ```

  2. On the Linux command line, use one of the following commands to clone the repo to the data storage volume for the instance. For a configuration deployed on Azure, for example, the default mount point for the data volume is /dev/sdd, so you would use commands like the following:

     ```
     $ git clone https://github.com/intersystems/FirstLook-DataTransformations
     /dev/sdd/FirstLook-DataTransformations
     OR
     $ wget -qO- https://github.com/intersystems/FirstLook-DataTransformations/archive/master.tar.gz
      | tar xvz -C /dev/sdd
     ```

     The files are now available to InterSystems IRIS in /irissys/data/FirstLook-DataTransformations on the container's file system.

- If you are using a containerized instance (licensed or Community Edition) that you deployed by other means:

  1. Open a Linux command line on the host. (If you are using Community Edition on a cloud node, connect to the node using SSH, as described in *Deploy and Explore InterSystems IRIS*.)
  2. On the Linux command line, use either the **git clone** or the **wget** command, as described above, to clone the repo to a storage location that is mounted as a volume in the container.
     - For a Community Edition instance, you can clone to the instance's durable %SYS directory (where instance-specific configuration data is stored). On the Linux file system, this directory is /opt/ISC/dur. This makes the files available to InterSystems IRIS in /ISC/dur/FirstLook-DataTransformations on the container's file system.
     - For a licensed containerized instance, choose any storage location that is mounted as a volume in the container (including the durable %SYS directory if you use it). For example, if your **docker run** command included the option **-v /home/user1:/external**, and you clone the repo to /home/user1, the files are available to InterSystems IRIS in /external/FirstLook-DataTransformations on the container's file system.

- If you are using an InterSystems Learning Labs instance:

1. Open the command-line terminal in the integrated IDE.
2. Change directories to /home/project/shared and use the **git clone** command to clone the repo:

```
$ git clone https://github.com/intersystems/FirstLook-DataTransformations
```

The folder is added to the Explorer panel on the left under **Shared**, and the directory is available to InterSystems IRIS in /home/project/shared.

- If you are using an installed instance:
  - If the instance's host is a Windows system with GitHub Desktop and GitHub Large File Storage installed:
    1. Go to https://github.com/intersystems/FirstLook-DataTransformations in a web browser on the host.
    2. Select **Clone or download** and then choose **Open in Desktop**.

    The files are available to InterSystems IRIS in your GitHub directory, for example in C:\Users\User1\Documents\GitHub\FirstLook-DataTransformations.
  - If the host is a Linux system, simply use the **git clone** command or the **wget** command on the Linux command line to clone the repo to the location of your choice.

### 2.1.3 Creating an Interoperability-Enabled Namespace

You must create an interoperability-enabled namespace before you can import the production that you downloaded from GitHub. If you have already created an interoperability-enabled namespace on your instance, you can use that for this production. To create a new interoperability-enabled namespace, use the following procedure. (The namespaces created when you first install InterSystems IRIS are not interoperability-enabled.)

1. Open the Management Portal for your instance in your browser, using the URL described for your instance in *InterSystems IRIS Basics: Connecting an IDE*.
2. Select **System Administration** > **Configuration** > **System Configuration** > **Namespaces** to go to the **Namespaces** page.
3. On the **Namespaces** page, select **Create New Namespace**. This displays the **New Namespace** page; follow the instructions for using this page in Create/Modify a Namespace in the "Configuring InterSystems IRIS" chapter of the *System Administration Guide*, making sure that the **Enable namespace for interoperability productions** check-box is selected.
4. Select **Save** near the top of the page and then select **Close** at the end of the resulting log.

### 2.1.4 Importing the Demo Production

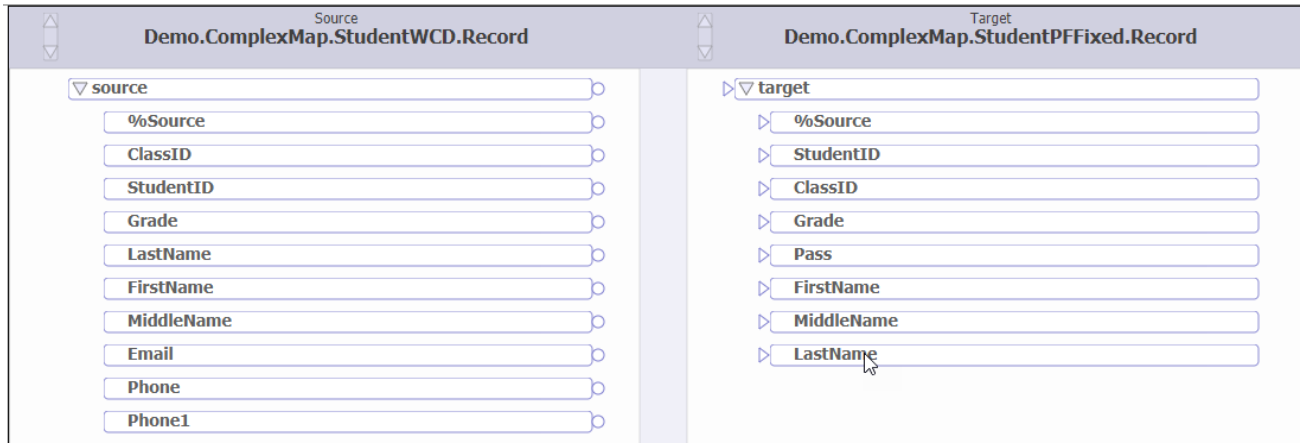
To import the demo production, which will contain your data transformation, use this procedure:

1. In the Management Portal, select **Interoperability > Manage > Deployment Changes > Deploy** to go to the **Deploy Production Changes** page. If prompted, select the namespace that you created before completing this step.
2. Click **Open Deployment**.
3. Go to the directory where you downloaded the files from GitHub, select DTLStudentDemo.xml, and click **OK**.
4. Click **Deploy**.
5. Click **OK**.

## 2.2 Creating the Data Transformation

InterSystems IRIS provides a Data Transformation Wizard to speed up and simplify the process of creating a new data transformation. For this demo, you are creating a data transformation that connects two record maps related to a student's grade in a class. To create the data transformation:

1. From the Management Portal home page, select **Interoperability > List > Data Transformations**.
2. Click **New**. The Data Transformation Wizard opens.
3. Select **Demo** from the **Package** drop-down list.
4. In the **Name** field, enter a unique name such as `DTLDemoTransform`.
5. Click the magnifying glass next to the **Source Class** field.
6. Click **Message Classes > Demo > Complex Map > StudentWCD > Record**.
7. Click the magnifying glass next to the **Target Class** field.
8. Click **Message Classes > Demo > Complex Map > StudentPFFixed > Record**.
9. Click **OK**. The new data transformation opens in the DTL Editor.




## 2.3 Transforming the Data

Now that you have created the data transformation, use the DTL Editor to:

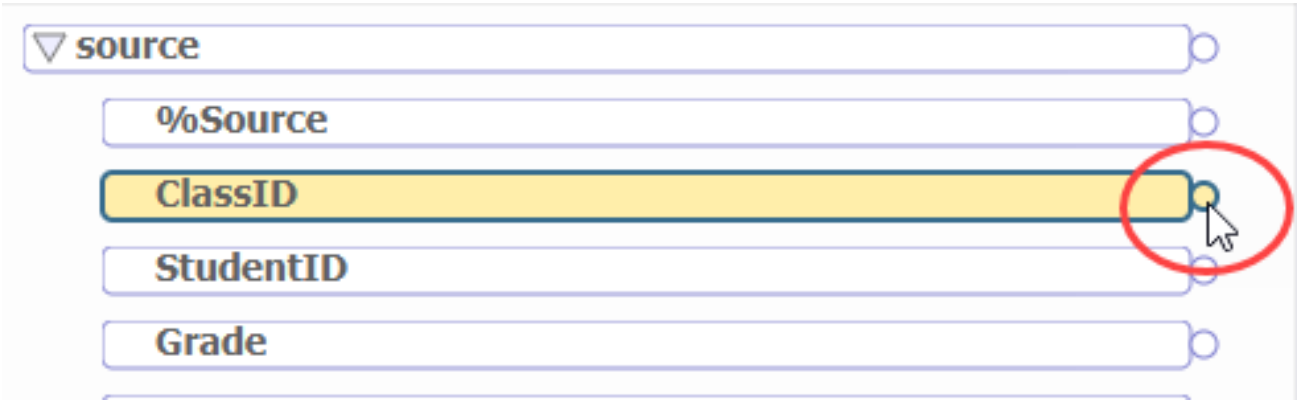
- Copy data from the source to the target, rearranging the order of the data as needed.
- Add a college-specific code to the beginning of the course number.
- Round off the final grade to a whole number.
- Use the final grade to determine whether the student passed or failed, and indicate this result in the target file.
- Ignore data from the source that is not required in the target.

### 2.3.1 Copying Data

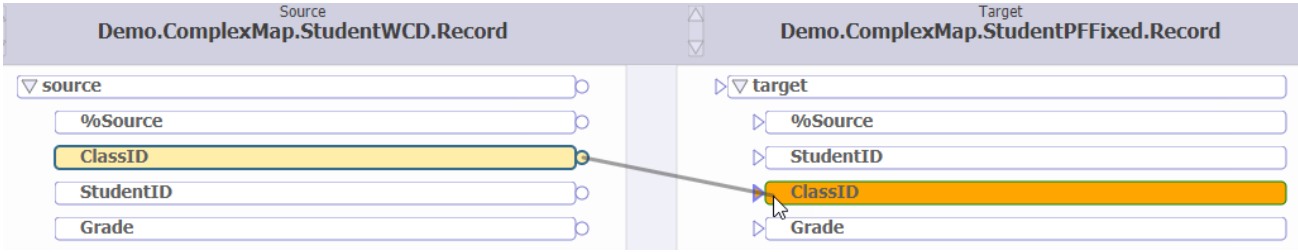
One of the most powerful and time-saving features of the DTL Editor is the ability to drag and drop values from the source to a corresponding property of the target. You simply click and hold the circle on the source's property and drag it to a property of the target message.

To copy values from the source to the target:

1. Click and hold the circle of the ClassID property of the source (on the left-hand side of the Editor).



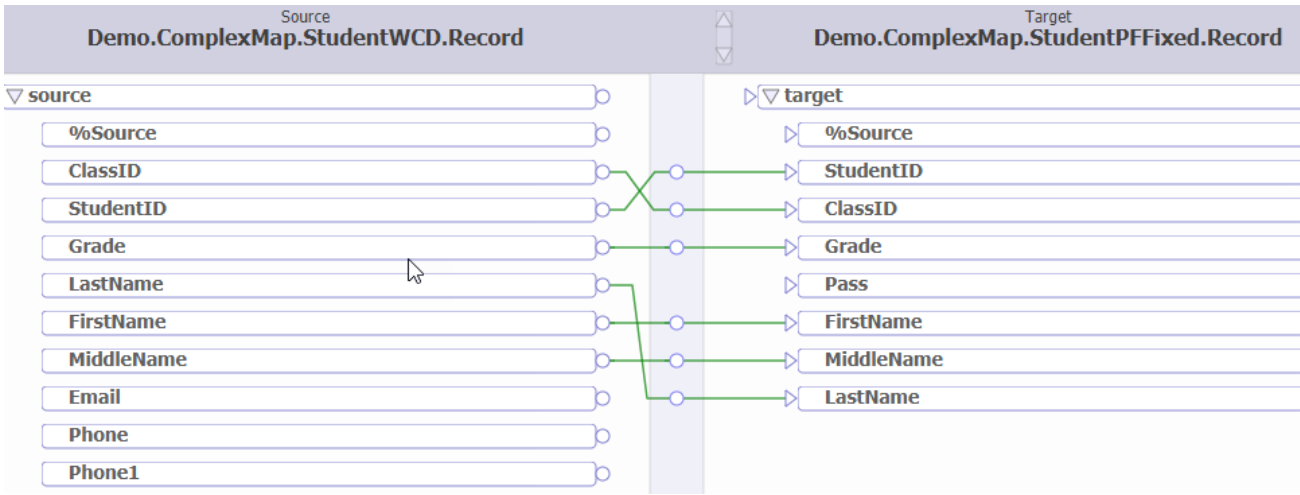

2. Drag your cursor to the ClassID property of the target until it is highlighted, then release the mouse button.




The value of the source's ClassID will be copied to the target's ClassID property.

3. Following this procedure, connect the following source properties to the corresponding target properties:
   - StudentID
   - Grade
   - LastName
   - FirstName
   - MiddleName

The Editor should now look like:

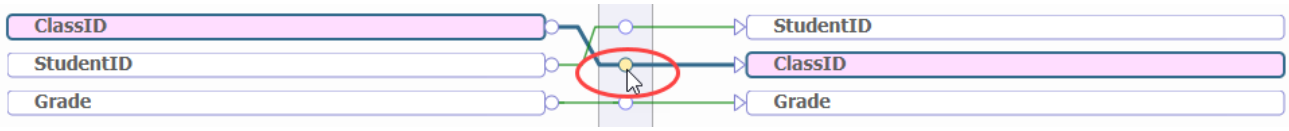


Notice that you have changed the order in which the data will be stored simply by connecting the property pairs.

## 2.3.2 Modifying the Value of a Property

In some cases, you want to change the value of the source's property before copying it to the corresponding target property. In this demo, the target system accepts a ClassID that starts with a two-character identifier that indicates the Source's college. The source's message does not include this identifier, so the data transformation must modify the value of ClassID before copying it to the target message.

1. Click the circle on the line that connects the source's ClassID property and the target's ClassID property.




   Notice that the **Action** tab on the right-hand side of the Editor now shows information about the Assign action that was created when you connected the two ClassID properties. The **Property** field is set to `target.ClassID` and the **Value** field is set to `source.ClassID`.

2. Change the **Value** field of the **Action** tab to: `"UC."_source.ClassID`. `UC` is the college identifier and it is added to the beginning of the ClassID using the underscore, which is the concatenate operator in the InterSystems ObjectScript programming language. It is beyond the scope of this demo, but it gives you an idea of how you can include ObjectScript code in your transformation.

## 2.3.3 Using Functions

You can use built-in functions when setting the value of a property. In this demo, you are rounding the final grade to a whole number using the `Round( )` function.

1. Select the connector between the source's Grade property and the target's Grade property.
2. Click the magnifying glass next to the **Value** field of the **Action** tab.
3. Select **Round ( )** from the **Function** drop-down list.
4. Click **OK**.

The **Value** field of the **Action** tab is now set to: `..Round(source.Grade)`

## 2.3.4 Using the Table of Actions

Notice that as you mapped source properties to target properties, the DTL Editor added **Set** actions to the table below the diagram.

Actions

| # | Action | Condition | Property | Value | Key / Transform |
|---|---|---|---|---|---|
| 1 | set | | target.ClassID | "UC."_source.ClassID | "" |
| 2 | set | | target.StudentID | source.StudentID | "" |
| 3 | set | | target.Grade | ..Round(source.Grade) | "" |
| 4 | set | | target.FirstName | source.FirstName | "" |
| 5 | set | | target.MiddleName | source.MiddleName | "" |
| 6 | set | | target.LastName | source.LastName | "" |

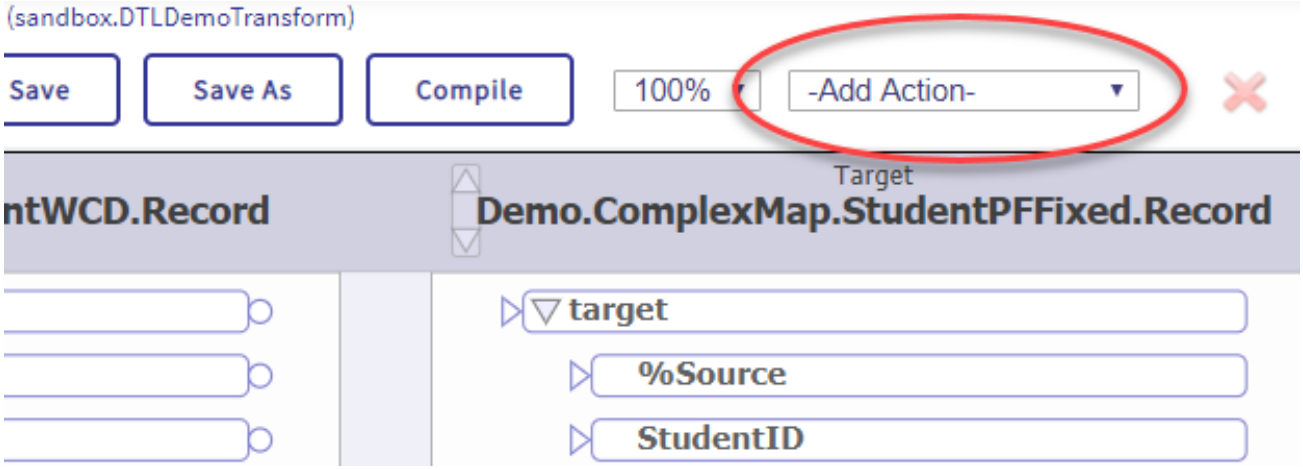
While the graphical portion of the DTL Editor makes it easy to visualize and perform actions quickly, sometimes it is better to use the table below the diagram to further define the transformation. For example, in this demo selecting the connection

between the source property and target property by clicking the graphical connection was the quickest way to select the **Set** action. However, in a more complex transformation, it might be difficult to select the connector. In this case, it is easier to select the **Set** action from the table below the diagram.

### 2.3.5 Defining a Conditional Set of Actions

In transforming data, you can use conditional statements to perform actions only if certain conditions are true. In this demo, you define a transformation to indicate that the student passed the class if the final grade is greater than or equal to 65. To add an **If** statement to accomplish this:

1. Make sure the last item is the table is highlighted. The **If** statement is placed right after the highlighted action.
2. Select **If** from the **Add Action** drop-down list.




   Notice that an **If** block is added to the table below the diagram.
3. With the **If** statement selected in the table, define the **Condition** field on the **Action** tab as: `target.Grade>=65`. The actions that you put inside the **If** block are executed only if the value of the target's Grade property is greater than or equal to 65.
4. Select **Set** from the **Add Action** drop-down list.
5. On the **Action** tab, define the **Property** field as: `target.Pass`.
6. Define the **Value** field as `1`. This is the boolean value indicating that the student passed the class.
7. Select the **else** action in the table.
8. Select **Set** from the **Add Action** drop-down list.
9. On the **Action** tab, define the **Property** field as: `target.Pass`.
10. On the **Action** tab, define the **Value** field as `0`. This is the boolean value indicating that the student did not pass the class.

The **if** block should now look like:

```
7  if      target.Grade>=65
8    set   target.Pass     1
9  else
10   set   target.Pass     0
11 endif
```

### 2.3.6 Ignoring Data from Source Message

In the demo, notice that the Email, Phone, and Phone1 properties were not connected to the target message. No further action is needed to prevent this data in the source message from being copied to the target message.

## 2.4 Compiling the Data Transformation

You have the option to save your work as you go, but you must always remember to compile the data transformation. Your production does not recognize changes until you click **Compile**.

## 2.5 Performing Preliminary Testing

The DTL Editor gives you the ability to quickly test the data transformation without having to run a message through the entire production. You simply input the test message and the DTL Editor displays what the output message will look like. At this point in this demo, you are pasting an XML representation of the record map data into the Test Transform tool. Later, you will test the transformation by running a text file through the production.

Remember to compile your data transformation before testing it.

1. Click the **Tools** tab on the right-hand side of the editor.
2. Click **Test**.
3. Copy and paste the following data into the **Input Message** text box:

```
<test>
<Record>
<ClassID>CS241</ClassID>
<StudentID>930698</StudentID>
<Grade>77.8</Grade>
<LastName>Sutherland</LastName>
<FirstName>David</FirstName>
<MiddleName>Timothy</MiddleName>
<Email>david.sutherland@mail.com</Email>
<Phone>978-343-3940</Phone>
<Phone1>978-343-0951</Phone1>
</Record>
</test>
```

4. Click **Test**.

   The **Output Message** text box should now look like:

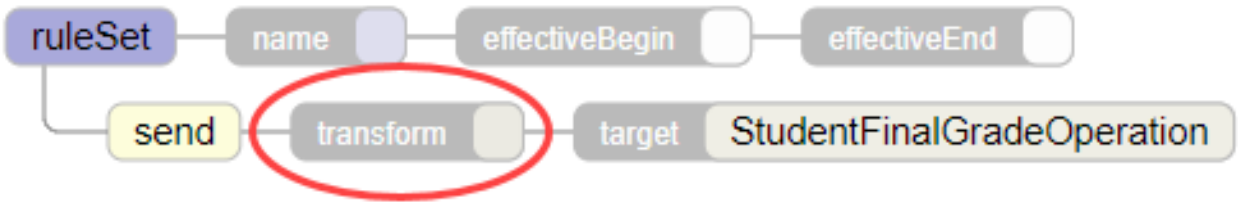
```
<Record>
  <StudentID>930698</StudentID>
  <ClassID>UC.CS241</ClassID>
  <Grade>78</Grade>
  <Pass>true</Pass>
  <FirstName>David</FirstName>
  <MiddleName>Timothy</MiddleName>
  <LastName>Sutherland</LastName>
</Record>
```

**Tip:** When working with HL7 or other EDI messages, you can copy and paste raw text from a sample message to test the transformation. You do not need an XML representation of the message.

## 2.6 Adding Transformation to Production

Now that you have created and compiled a data transformation, you must add it to your production so that source messages flowing through the production get transformed into the message that gets sent to the downstream application. In this demo, you are adding the transformation to a business rule associated with the business process DTLDemoRouter.

1. From the Management Portal home page, go to **Interoperability > List > Productions**.
2. Select `Demo.ComplexMap.SemesterProduction` and click **Open**.
3. Select `DTLDemoRouter` listed under **Processes**.
4. On the **Settings** tab, click the magnifying glass next to the **Business Rule Name** field.
5. Double-click the **transform** shape located in the business rule's **send** action.




6. In the **Data Transform Selector** dialog, select the data transformation that you created for this demo.
7. Click **OK**.
8. In the Business Rule Editor, click **Save**.

For more information about business rules, see *Developing Business Rules*.

## 2.7 Testing the Data Transformation

Now that you have added the data transformation to the production, run the production to see your demo in action. In this step:

- Create the directories where the production accepts and sends the sample files.
- Start the production.
- Copy the sample input file into the correct directory.
- Compare the contents of target file with the source file.

### 2.7.1 Creating Required Directories

The demo production is defined to look in c:\practice\in on the instance's host for the file containing the source data, and to write out the target file to c:\practice\out. If you cannot create the directories on the host's c:\ drive or it is not a Windows system, you must identify or create alternative directories and then modify the business service and business operation of the production before continuing. The **File Path** setting of the business service defines where the production looks for the input file, while the **File Path** setting of the business operation defines where the target file is sent.

The way to create the directories this depends on the type of instance you are using, as follows:

- For an instance deployed by ICM, use the **icm exec** command with the **-machine** and **-interactive** options to open a bash shell inside the container in which the instance is running, for example:

  ```
  icm exec -command bash -machine MYIRIS-AM-TEST-0004 -interactive
  ```

  You can then create the directories on the container file system.

- For any containerized instance, whether licensed or Community Edition, use the command **docker exec -it** *container_name* **bash** to open a bash shell in the container (the name of a Community Edition container is **`try-iris`**). Then create the directories on the container file system.
- For InterSystems Learning Labs, use the command-line terminal in the integrated IDE to create new folders in the Shared folder; you can browse to these in the Management Portal under /home/project/shared.
- For an installed instance, create the directories on the local file system.

This rest of this exercise assumes the directory paths c:\practice\in and c:\practice\out; substitute the correct paths, if different.

### 2.7.2 Starting the Production

To start the production containing the demo data transformation:

1. From the Management Portal home page, go to **Interoperability > List > Productions**.
2. Select `Demo.ComplexMap.SemesterProduction` and click **Open**.
3. Click **Start**.

### 2.7.3 Copying the .CSV File to Directory

The production is defined to accept a .CSV file containing the student information from c:\practice\in. Copy the input.csv file from the directory containing the GitHub files to the input directory, either c:\practice\in or another directory you created and specified.

As long as the production is running, this file should disappear from the directory within a few seconds, indicating that the production has started to process the file successfully.

### 2.7.4 Verifying Data in Output

The demo production takes .CSV file you copied into the production's input directory and uses the data transformation to convert the data into a different format with new content. This outgoing file is copied into the production's output directory, either is c:\practice\out or another directory you created and specified, with a timestamp as the filename.

Open input.csv and the output file and compare the two.

# 3 Other Important Features

This demo provides the basics of developing and testing a data transformation. Other important, commonly used features include:

- **for each** statements that iterate through a collection, stepping through the same sequence of actions for every member of the collection. This collection can be an array or, in the case of HL7 or other EDI messages, a repeating segment. For more information, see Adding a For Each Action.
- Subtransformations, which allow you to modularize your data transformations. A data transformation can use a sub-transform action to run another transformation at any point in the flow of actions. You can create a library of reusable

transformations that are called from other transformations, avoiding duplicate transformation logic. For example, you can create segment transformations that are reused whenever those segments appear in an EDI message.

- Support for X12 and EDIFACT standards. When creating a new transformation using the Data Transformation Wizard, you can specify whether the message type is X12 or EDIFACT. This allows you to select a document type that populates the DTL Editor with the right schema. Visual representations of the segments of these schemas expand and collapse in the DTL Editor.

# 4 Learn More About Data Transformation

For more details about data transformations, see:

- Developing DTL Transformations
- Advanced Data Transformations (online course)
- Business Process and Data Transformation Language Reference