Transforms By Example - Guided Tour

This note describes how to install the Transforms By Example toolset (TBE), and run through a guided tour of its main facilities.

You should have available to you:

|  |  |
| --- | --- |
| **Description** | **File name** |
| This document | Transforms By Example Guided Tour.docx |
| Eclipse plugins for the TBE tools | mapper\_230818.zip |
| The guided tour mapper project | TBE\_Guided\_Tour.zip |
| Source code (Eclipse plugin project) for the FHIRRunner project | FHIRRunner.zip |

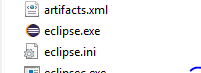
# Installing Eclipse and the TBE Tools

TBE consists of a runtime transform engine (currently Java, FHIR Mapping Language or XSLT) and a set of transform generation tools, which incorporate the runtime engine. The transform generation tools are Eclipse plugins, so first you need to install Eclipse (Modelling Edition), and then install the plugins on Eclipse.

Download Eclipse Modelling Edition (Neon edition or later) from

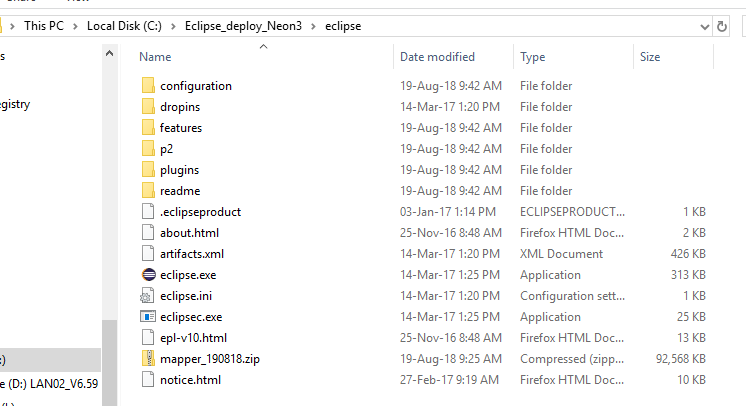
http://www.eclipse.org/downloads/packages/release/Neon/2

This is a zip file of approximately 400 MB. To install Eclipse, put the zip file in a folder and unzip it there. This will create a sub-folder 'Eclipse' including an Icon 'Eclipse.exe' used to run Eclipse:

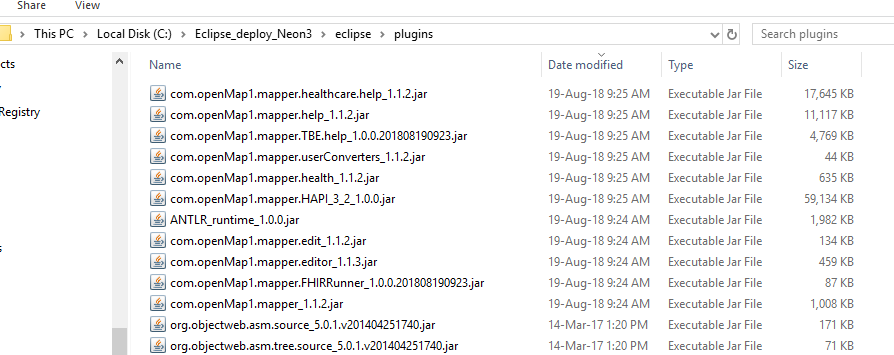


When you first run Eclipse, it will ask you to choose a folder for a workspace - where it will hold your Eclipse projects. Note where this is. Eclipse requires Java 1.8, so you may need to install that.

To install the TBE plugins, copy the zip file containing the plugins (with name mapper\_<date>.zip) into the Eclipse folder:



Most of the size of the zip file (92 MB) is taken up with help files with screenshots, and with the HAPI Java libraries. Unzip this file with a command 'Extract here'. This should leave 11 TBE plugin files in the 'plugins' sub-folder of the Eclipse folder:



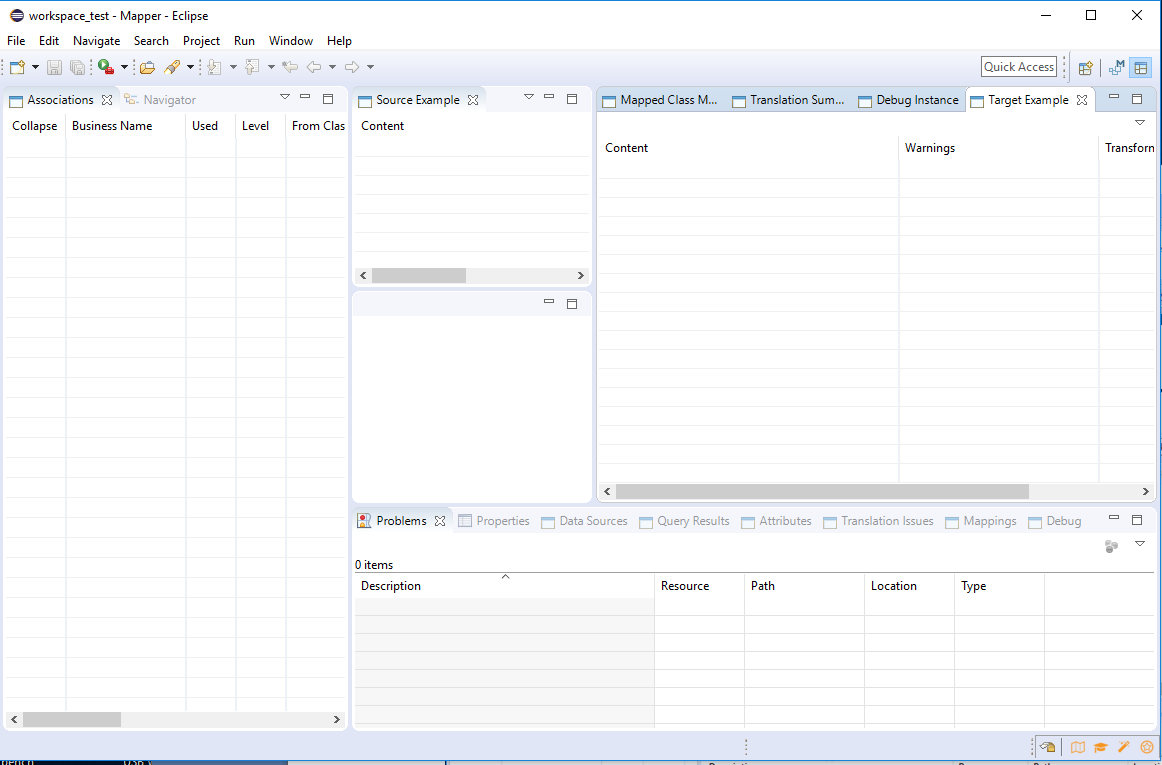
Start Eclipse. Go to the workbench by following



Follow the sequence of menu selections:

Window => Perspective => Open Perspective => Other => Mapper

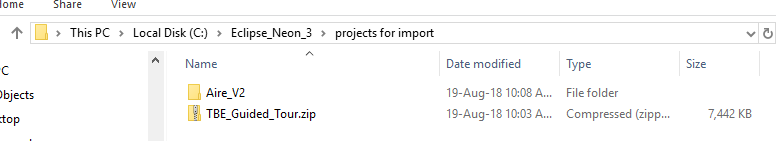
Eclipse should then show the 'Mapper Perspective' (set of windows) which is active when using the TBE tools:



This shows you have successfully installed the TBE tools.

# Importing the Guided Tour Project into the Workspace

Next you need to import the TBE Guided tour project into the workspace. Unzip the file 'TBE\_Guided\_tour.zip' in any empty folder:

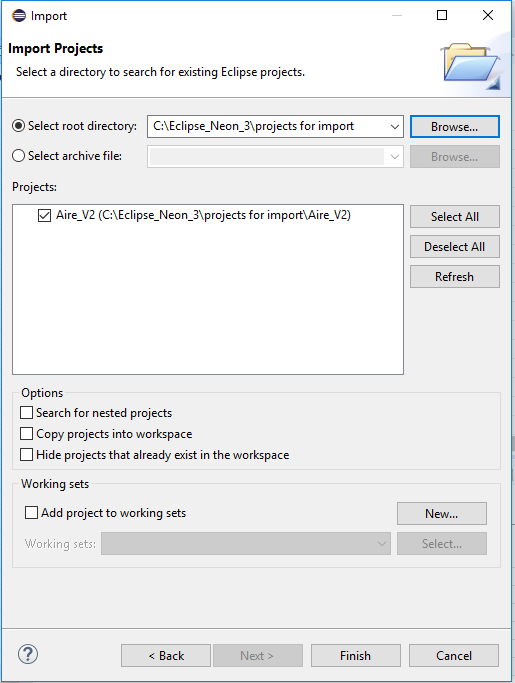


This gives a set of sub-folders for mapper projects. In this case, there is only one project, called 'Aire\_V2'

In Eclipse, follow the menu path:

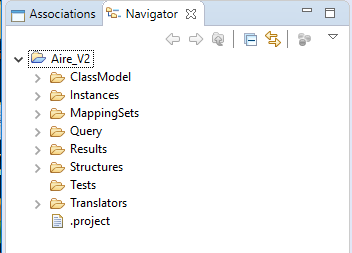
File => Import => General => Existing Projects into Workspace

Then select the folder you unzipped the projects into:



Check the one project 'Aire\_V2', and check 'Copy projects into workspace'. Then Finish.

Now when you select the left-hand 'Navigator' view, you should see a folder for the project 'Aire\_V2', with the sub-folder structure:

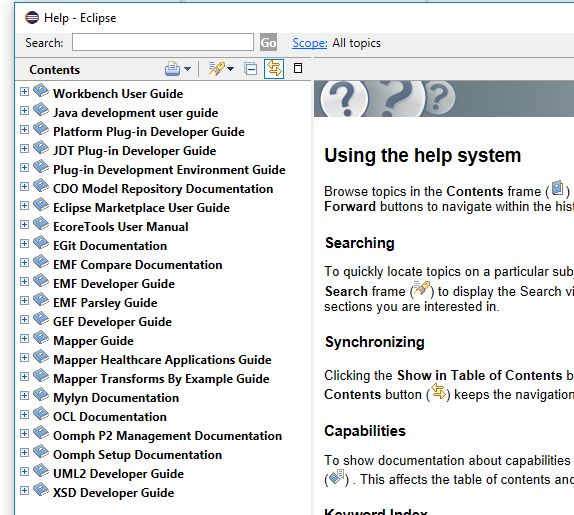


You have now installed the guided tour project, and are ready to tour it.

# Using the TBE Help Files

During the guided tour and afterward, you may want to read some of the TBE Help files. Here is how to do that:

Use the menu option Help > Help Contents to show:

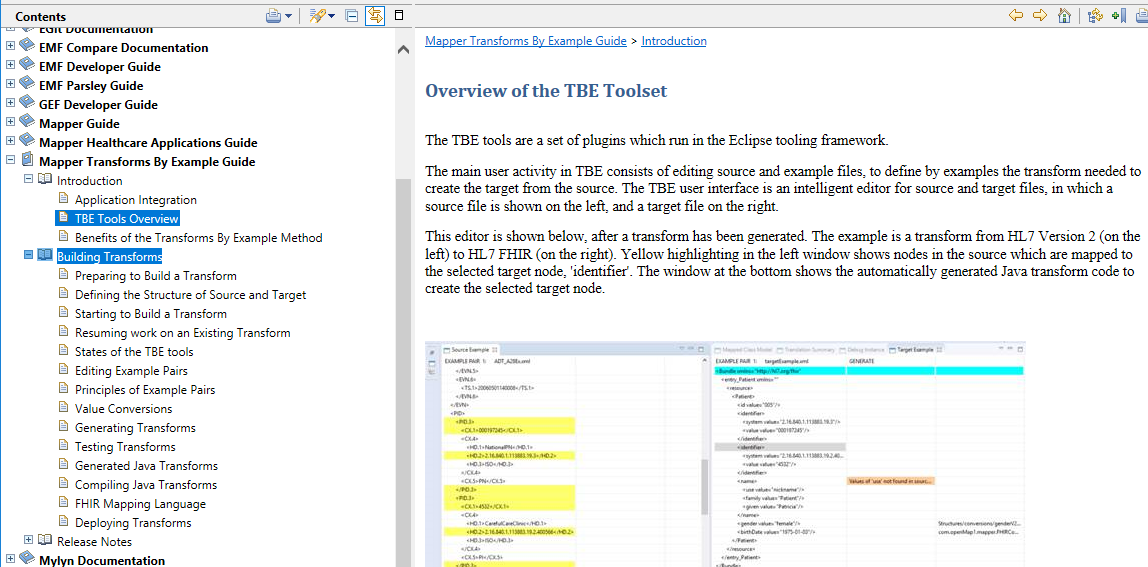


The three sections of most interest to you are:

1. Mapper Guide
2. Mapper Healthcare Applications Guide
3. Mapper Transforms By Example Guide

The first two of these sections provide background information, about the mapping and transform tools which the TBE tools are built on.

You will first want to look at the Mapper Transforms By Example Guide. When expanded, this looks like:



For instance, you may want to look at the 'TBE tools Overview' at this stage.

The section 'Building Transforms' contains detailed guidance for the tasks involved in creating a transform by TBE.

# Fundamental Concepts of TBE

To develop any data transform, you usually need to either write transform code, or to define mappings from the source to the target. You also need to create test cases - examples of the source, and the resulting examples of the target - to test that your transform does what you want it to do.

To create a transform by TBE, your main activity is to create and refine the test cases. Once you have done this, the TBE tools use the test cases as examples, automatically generate transform software from the examples, and test it. Thus the TBE tools can short-cut a large amount of manual work - in writing transform code, or creating mappings - but still give you transform code in Java, which you can modify as required (e.g. to address data quality problems in the source). If you are looking to create transform software with the least effort and with reliable, tested results, TBE is the way to do it. It is, we believe, the most reliable and economical way to create transform software.

To generate working mappings and transforms from some XML (the source) to a target such as a FHIR Bundle, using TBE, you follow these steps:

1. Capture definitions of the allowed structure of the source (e.g. from an XML schema) and the target (from FHIR profiles)
2. For one or more examples of the source structure, create the example of the FHIR target which should result from the transform. This makes a **source-target example pair**.
3. Find or make the needed property value conversions, defining what property values in the source XML give what values in the FHIR target (where source and target use different code systems, e.g. for patient gender, or different data formats, e.g. for dates and times)
4. In two automatic steps:
   1. Generate the transform from the example pairs.
   2. Test the generated transform, to verify that it gives the required FHIR XML from the source, for each example pair or test case
   3. Make the other outputs such as Java and FHIR Mapping Language
5. Iteratively extend and refine the transform, by refining the example pairs to illustrate the transform you require.
6. Inspect and deploy the outputs of the transform generation, such as FHIR Mappings, or generated Java code.

These steps are all described in the TBE Eclipse Help files. For the purposes of the guided tour, steps (1) - (3) have been done already.

# The Example Project Used in the Guided Tour

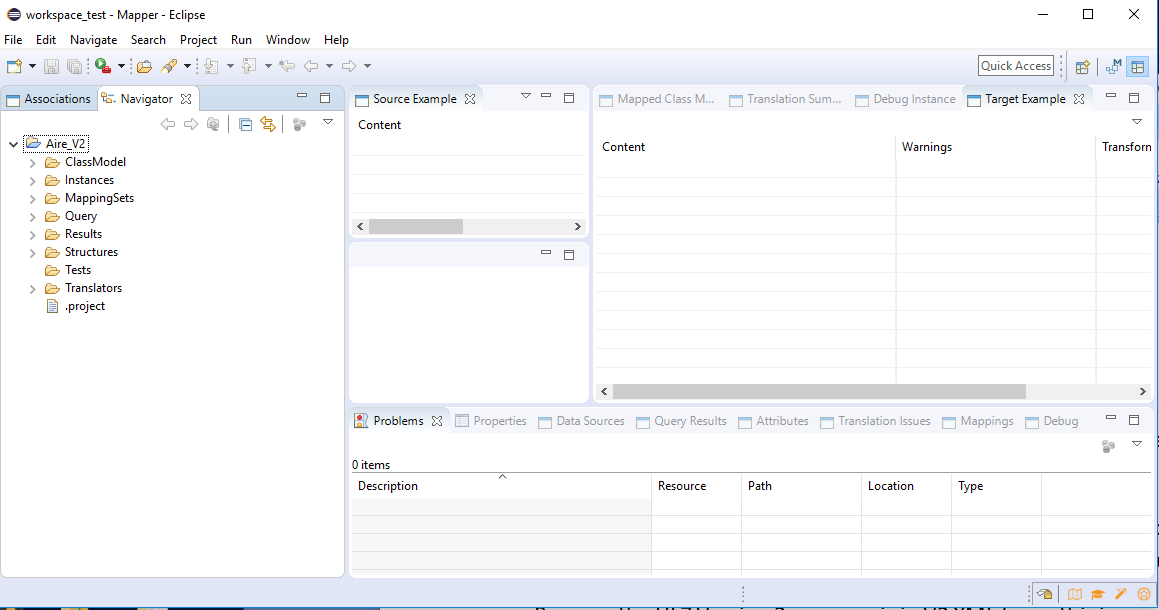
The example project used in the guided tour, called 'Aire\_V2' , is a transform between an HL7 Version 2 ADT message, describing patient demographics, and a FHIR Patient resource.

The V2 ADT message is commonly used in hospitals across the world, and the FHIR patient resource is fundamental to many FHIR applications. Its definition can be found at [https://www.hl7.org/fhir/patient.html](https://www.hl7.org/fhir/patient.html%20) .

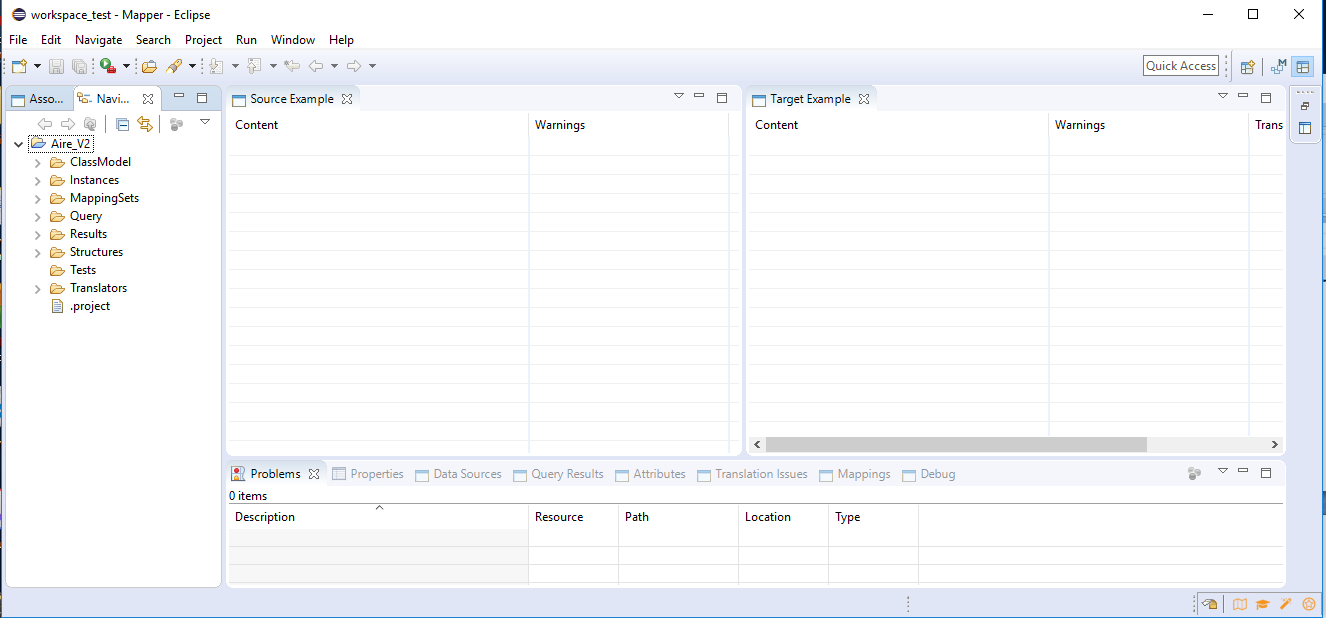
Because the HL7 Version 2 message is in V2 XML form, this is an example of conversion of XML to FHIR. The FHIR output can be in either XML or JSON form. During the tour, we shall view it mainly in XML form.

# Following the Tour

At the start of the tour, when you have set the tools to the mapper perspective, and imported the tour project, Eclipse will look something like this:



Hide the central pane (called a view), and arrange the 'Source Example' view and 'Target view so they are prominent; those are the two main views you will use to edit the example pairs and examine test results:

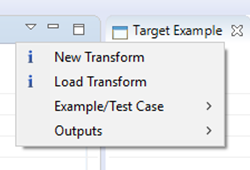


At the top right-hand corner of each of these views, you will see small inverted triangles:

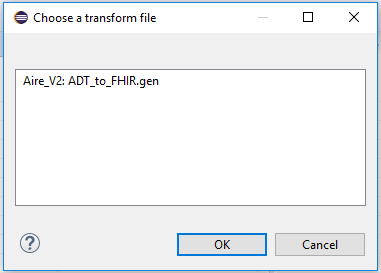


These are both drop-down menus, which you will use for many important operations of TBE.

To open a transform ,which is defined by a set of example pairs, you use the drop-down menu option 'Load Transform':

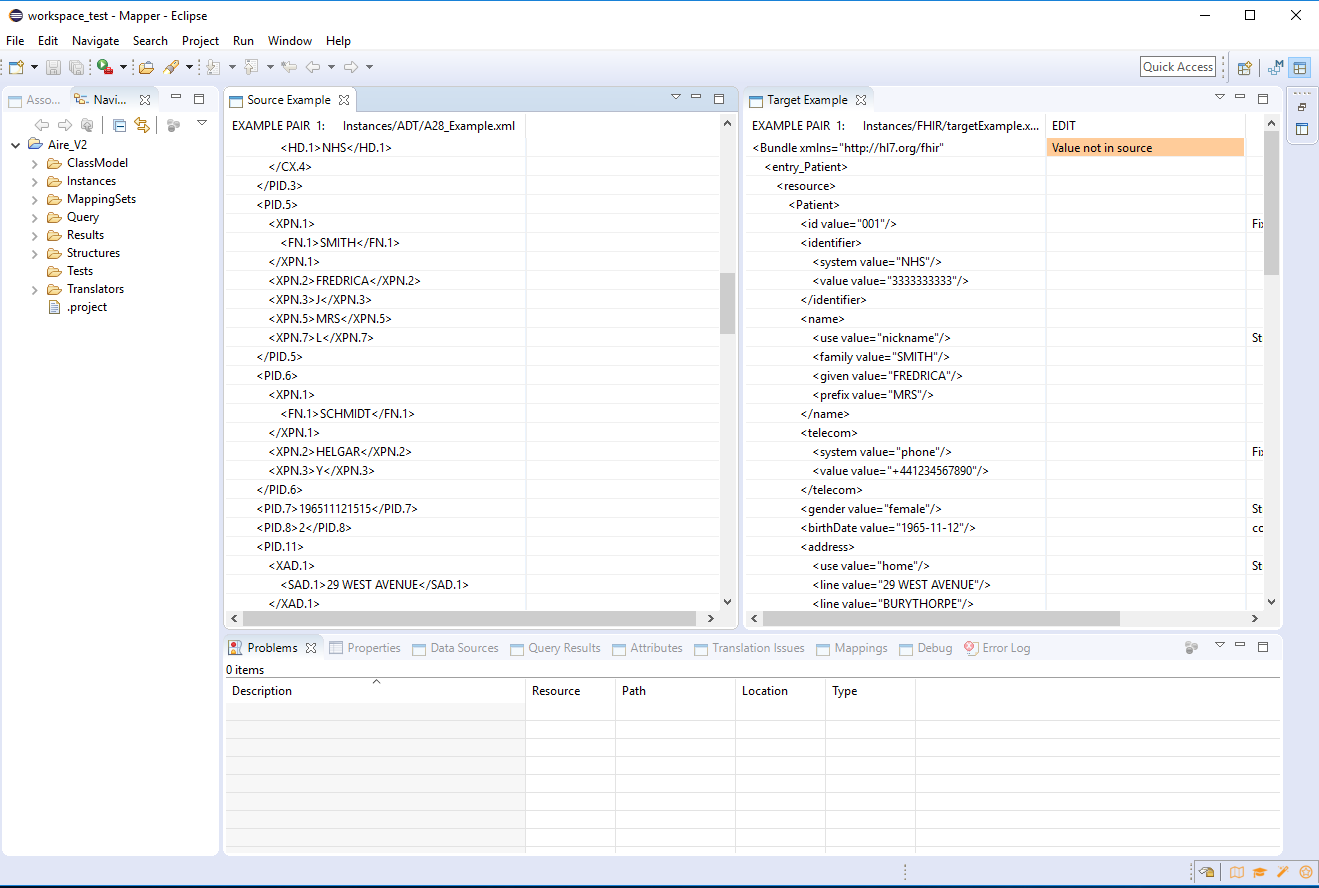


This shows you all the transforms available to load in all projects in the workspace. In this case there is only one, in the project 'Aire\_V2':



Information about each transform is defined in a small text file with extension '.gen' which you can inspect if you need to, but need not.

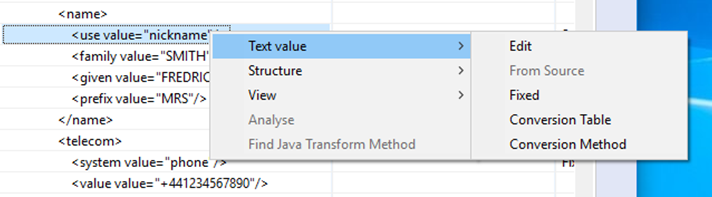
Picking the one .gen file available, and selecting 'OK' gives:



This is the main view that you will see most of the time when you are using the TBE tools, in an intelligent editor that you can use to edit source and target examples side by side, and then generate and test the resulting transform.

The source view (of the V2 ADT message) has been scrolled down, to show fields in the patient's name (MRS FREDRICA SMITH), which you can also see in the right-hand target view, of the FHIR Patient resource. This is the essential property of source-target example pairs, which the TBE tools use to discover and generate the correct transform. Unless there are matching data values in the source and target examples, the TBE tools cannot discover a transform. (sometimes a data conversion is required to convert from source to target values; you will need to provide these).

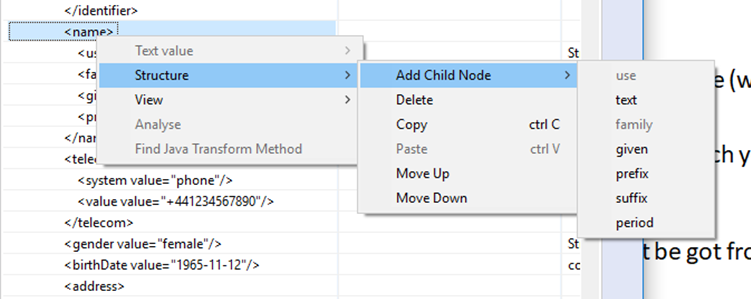
You can see some of the intelligent editor facilities if you right-click any node in the target, to see what you can do to the data value on that node:



* You can edit it with a text editor
* You can give it a value from a node you have selected in the source
* You can say it is a fixed value for that node (which need not be found in the source)
* You can say it is got from some value in the source by applying a conversion table (which you will provide)
* You can say it is got from a value in the source by some conversion method (which you sill supply)

The editor will warn you whenever any value on a target node cannot be got from the source in any way. Those are the values that no transform can produce from the source example.

Similarly, you can edit the target structure - by adding and removing nodes - but only in ways that are allowed by its structure definition:



Here, the user is trying to add a child node to the 'name' in the FHIR Patient resource. The TBE tools know that there is already a 'use' child and a 'family' child node, and that they cannot be repeated; so these cannot be added. You are prevented from creating a structurally invalid target or source.

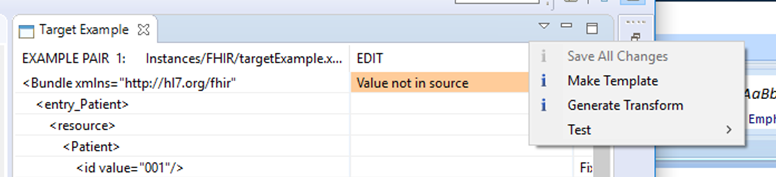
If there is more than one source-target example pair (which there is not in this case) the editor allows you to move between the different pairs.

The TBE view of the target and source examples has three different modes:

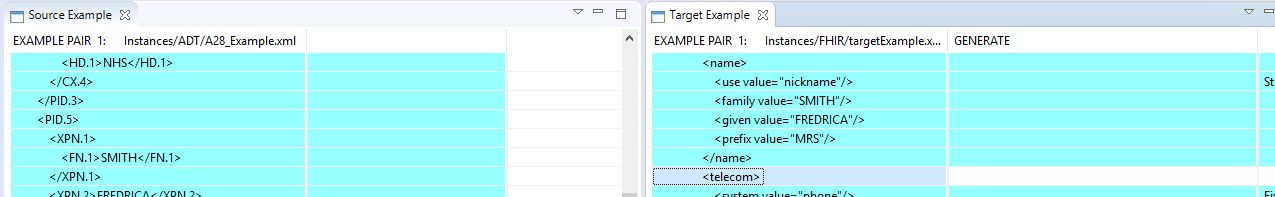
1. The **editing** mode, when you are making changes like those above
2. The **generate** mode, when TBE has just generated a transform, and you can find out about the transform or any problems in the generation process
3. The **test** mode, when you have just run the generated transform against all example pairs or other tests, and you can view problems that came up.

These three modes are easily distinguished by the background colour of the source and target views.

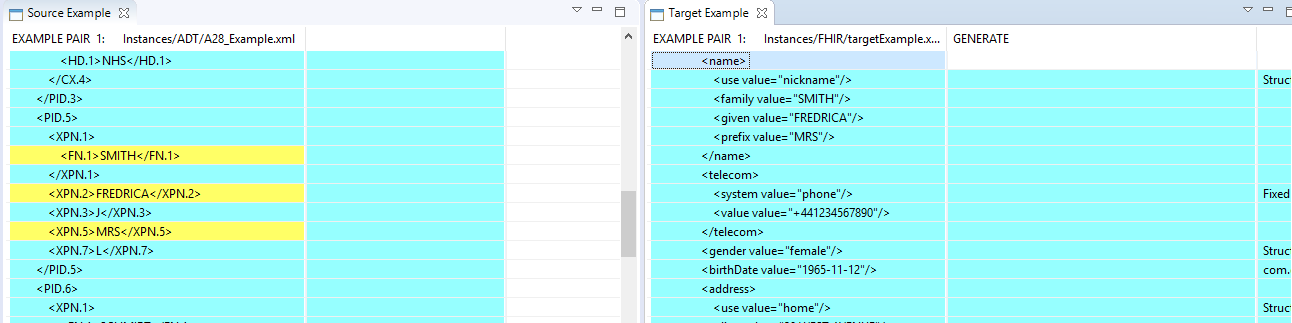
To move between these modes, use the dropdown menu in the target view:



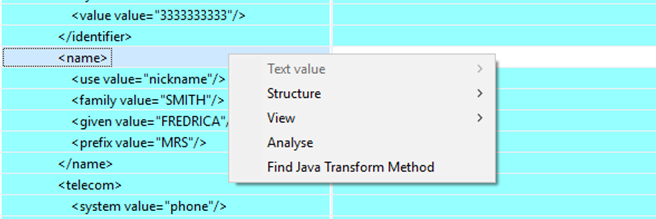
When you select 'Generate Transform' the tools will generate a new transform from the current state of the example pairs, and make new information available. First the background colour changes to the 'Generate' colour - light blue:



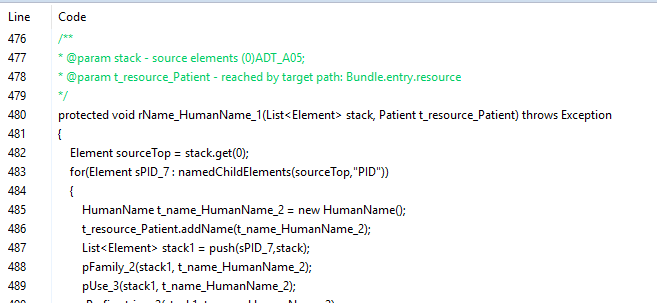
Next, if you select any node in the target, the tools will highlight in yellow the nodes in the source which the selected target subtree is derived from:



You can also right-click on any node in the target to find out more:

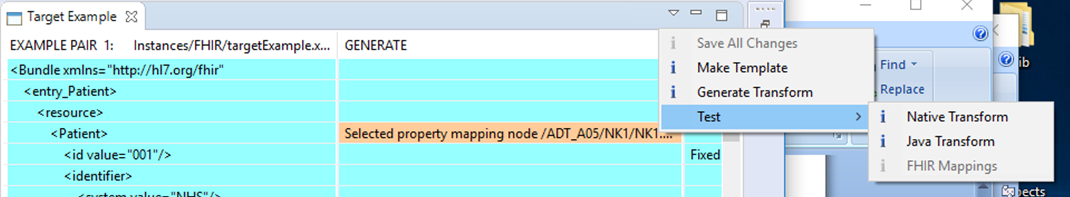


When you select 'Find Java Transform Method', the lower 'Java Code' view shows the method in the generated Java transform which creates the target selected target node:



The generated Java code follows a simple and consistent design pattern, which is easy to understand and modify if necessary (preferably by subclassing, to allow you to re-generate later) . The generated code is more concise than most hand written transform code.

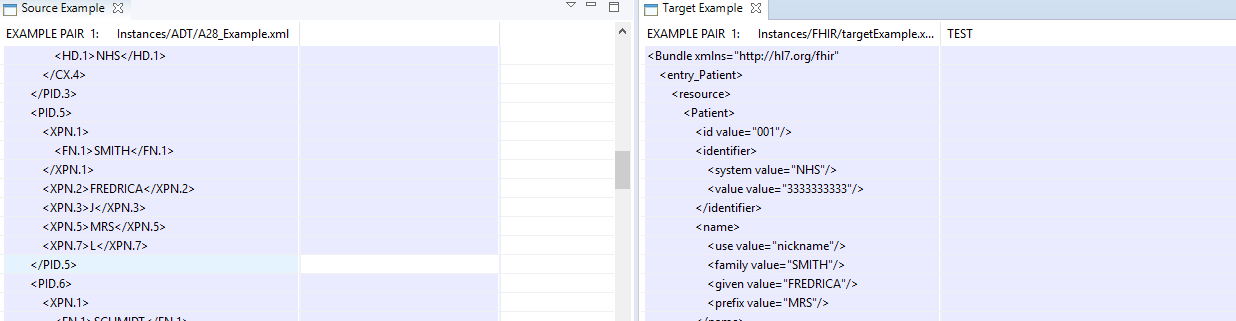
Finally you will want to test the generated transform. There are three ways to do this:

The option 'Native Transform' is always available, and uses a transform engine which is a part of the tools.

The option 'Java Transform' is only available if the generated Java code has been compiled (a separate step) which in this case it has.

The option 'FHIR Mapping Language' is only available if FHIR mappings have been generated, which in this case they have not (they could easily be).

If you choose the 'native transform ' option, the background colour changes to Lilac, for the test mode:



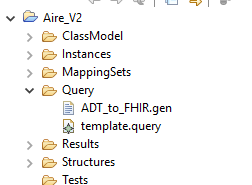
If most of the two views are lilac, your transform is in good shape - because any transform problems, discovered automatically by comparing the test outputs with the target example - are highlighted in a different colour. In this case, there are very few problems.

So the steps in going from Edit mode to Generate Mode to Test mode are very quick. You can start with a very basic transform - which only transforms a few values from source to target - and incrementally refine it, in a rapid edit/generate/ test cycle, until it does all that you require.

# Files Used in the Guided Tour

Looking in the folders of the mapper project 'Aire\_V2', you can find all the files used to generate the transform, and the generated transform code.

You could start by looking at the '.gen' file, which defines where all the other files are found:



The .gen file is a small comma-separated-value text file (created and maintained by the tools) which defines:

* where to find the definitions of source and target structure
* where to find the source- target example pairs
* where the resulting transform will be stored
* where test result files will be stored
* other information required to generate a transform , such as which target nodes have fixed values, and which target nodes need data conversions.

All these files are stored in folders in the project, usually using the following conventions:

* definitions of source and target structures - such as FHIR profiles - are stored in the 'Structures' folder.
* Source-target example pairs are stored in the 'Instances' folder
* Test results and generated transform code are stored in the 'results' folder

You can explore these folders to see what is there. The TBE help files give more information.