Preamble

• The Taverna 2 platform is constantly changing; while the look and feel of the workbench may change, the functionality wont!
Getting started

If you don’t see a ‘Biomoby’ folder, you need to tell Taverna to go and fetch it for you!
Getting started

From the ‘Service Panel’:

Import new services → Biomoby service...

Choose the ‘default’ registry (more on this later!)
Getting started

Now we see that Taverna has fetched Biomoby services for us to use
Installing a Plugin

We will also make sure that the Spreadsheet import plugin is installed!
To install the plugin, from the toolbar, click on **Advanced** and then on **Updates and plugins**.
Installing a Plugin

In the resultant window, click on **Find New Plugins**
Installing a Plugin

Among others, we discovered the **Taverna 2 Spreadsheet** activity. Check the box and then click on **Install**.
Installing a Plugin

The plugin is being installed! Once it has been fully installed, you will have to **restart** the workbench.
Installing a Plugin

Here we see the activity under the **Service templates** node in the service panel.
The *Service panel* lists all of the services available to a workflow designer.
Under the node ‘Biomoby’ Moby services are shown.

Services are sorted by their Biomoby Service type.
If you wish to use registries other than the default one, you can add new *Biomoby Activities* exactly as we did for the default registry.

Taverna even remembers the registry (or registries) that you chose when you restart the workbench!
Creating Workflows
We will start by adding a Biomoby datatype to the workflow.

From the menu bar, click on

Advanced → Biomoby → Datatype Browser → http://...
The Datatype viewer should be visible now.

Context click on the root node of the tree (Object) and

Add Datatype – ‘Object’ to the workflow.
The Datatype viewer should be visible now

- Context click on the root node of the tree (Object) and
  *Add Datatype – 'Object' to the workflow.*
The Datatype viewer should be visible now

Context click on the root node of the tree (Object) and

Add Datatype – ‘Object’ to the workflow.
The Workflow Explorer now shows that we have a Processor called **Object**
- Has 3 input ports: id, namespace and article name
- Has 1 output port: mobyData

The Workflow diagram illustrates our processor
If we click on our Datatype, and then on the Details Tab, we can do interesting things with our *Object*.
The Details tab provides us with some information on our datatype.

Please click on the *Datatype registry query* button to proceed.
To find those services that operate on a specific domain, we can restrict our search to only those that operate on the namespace(s) that we specify.

Click *Yes*, then navigate to the namespace *NCBI_gi* and click *Done.*
The resulting window illustrates what services produce and consume our datatype.

- Navigate to the `bioinfo.icapture.ubc.ca` node and context click on `getGenBankFasta`.
- Add the service to the workflow.
Notice how Taverna automatically made the appropriate connection from our datatype to our Biomoby service.
The **Workflow Explorer** now shows that we have a Processor called **getGenBankFasta**

- Has 1 input port: Object(identifier)
- Has 1 output port: FASTA(fasta)

The Workflow diagram illustrates our processor
To discover more services that we can use, click on the `getGenBankFasta` activity, then click on the `Details` tab and finally click on the `Browse Biomoby service details` button.
The resultant window displays the services’ inputs and outputs.

There are also tool tips that show up when your mouse hovers over any particular input or output that tells you what namespaces the data type is valid in.
FYI

- If we context (right) click on the leaf `Object('identifier')`, we can bring up a menu that will allow us to add that datatype to our workflow.
  - Not only would the datatype be added, but Taverna will attempt to make the appropriate connections for you too!
Context clicking on an output reveals a menu with 3 options.

- A brief search for services that consume our datatype
- A semantic search for services that consume our datatype
- Adding a parser to the workflow that understands our datatype
The result of choosing to add a parser for FASTA to our workflow.

The parser allows us to extract:

- The namespace and id from FASTA
- The namespace and id from the child String
- The textual content from the child String
The result of choosing to conduct a brief search for services that consume FASTA
We will add the service getDragonBlastText to our workflow by choosing ‘Add service - getDragonBlastText to the workflow?’ from the context menu.
The current state of our workflow shown graphically.

- Taverna made a guess to determine the appropriate connections. Sometimes the guess isn’t correct, but usually it is.
A more complex view of our workflow
Finding services that consume `NCBI_BLAST_Text` starts by browsing the details for the Biomoby service `getDragonBlastText`
conduct a brief search
Add the service `parseBlastText` to our workflow
Our current workflow
Workflow inputs are added by clicking on the little red triangle (located on the main menu bar)
The result from adding 2 inputs:

- Id
- namespace
The workflow input id will be connected to Object’s input port ‘id’. You can make the connection by clicking on the workflow input id and dragging it to the Object’s input port ‘id’
Workflow after connecting the workflow input ‘id’
The workflow input namespace will connect to Object’s input port ‘namespace’.

Workflow after connection of the workflow inputs.
Workflow outputs are added by clicking on the green triangle (located on the toolbar).
The result from adding 2 workflow outputs:
- moby_blast_ids
- fasta_out
The output `moby_blast_ids` will be connected to `parseBlastText`'s output port `Object(Collection - 'hit_ids')`. You can make the connection by dragging the `parseBlastText`'s output port to the workflow output port.
The output fasta_out will be connected to Parse Moby Data(FASTA) output port fasta_'content'
To run the workflow, click on **File** from the toolbar
Choose ‘Run workflow’
A prompt to add values to our 2 workflow inputs
To add a value to the input ‘id’ click on the id tab and choose ‘New value’
Enter 656461 as the id
Choose the namespace tab and click on ‘New value’
Enter *NCBI_gi* as the value for namespace
Once you are done, click on ‘Run workflow’
Our workflow in action
Once the workflow is complete, we can examine the results of our workflow.

We may not have results for moby_blast_ids*

* we may need to configure the service to be less stringent. More on this later!
Without the parser, FASTA is represented as a Moby message, fully enclosed in its wrapper.

Non-moby services do not expect this kind of message
  - Example for moby_blast_ids:

```xml
<moby:MObY xmlns:moby="http://www.biomoby.org/moby">
  <moby:mobyContent>
    <moby:mobyData moby:queryID="a0">
      <moby:Collection moby:articleName="hit_ids">
        <moby:Simple>
          <moby:Object moby:id="Z368" moby:namespace="DragonDB_Sequence" />
        </moby:Simple>
        <moby:Simple>
          <moby:Object moby:id="EM:AMA568115" moby:namespace="DragonDB_Sequence" />
        </moby:Simple>
        <moby:Simple>
          <moby:Object moby:id="EM:AJ794110" moby:namespace="DragonDB_Sequence" />
        </moby:Simple>
        <moby:Simple>
          <moby:Object moby:id="EM:AB012941" moby:namespace="DragonDB_Sequence" />
        </moby:Simple>
      </moby:Collection>
    </moby:mobyData>
  </moby:mobyContent>
</moby:MObY>
```
Non-moby services expect the just the sequence and using the Parse Moby Data (FASTA) processor, we can extract just that
Moby services can interact with the other services in Taverna.

Let’s add a Soaplab service.
► We will choose a nucleic_restriction Soaplab service called ‘restrict’
► Drag it into our workflow
We will connect the output port `fasta_'content'` from the service *Parse Moby Data (FASTA)* to the input port `sequence_direct_data` from the service *restrict*.

Context click on the *Parse Moby Data (FASTA)* and choose `Link from output ... → fasta_'content'`
Then click on the Soaplab service
In the resulting list menu, choose `sequence_direct_data`
The result of our actions so far.

We will need to add another workflow output to capture the output of restrict.
Create an output called restrict_out
Connect the output port `outfile` from the service `restrict` to the workflow output `restrict_out`
Once the connections have been made, run the workflow again using the same inputs.
The workflow on top has some extra services added to it.

- **FASTA2HighestGenericSequenceObject** from the authority bioinfo.icapture.ubc.ca, a conversion service
- **runRepeatMasker** from the authority genome.imim.es, an analysis service
- A Moby parser for the output **DNASequence** from runRepeatMasker.
- A workflow output **Masked_Sequence**

Add them to your workflow
The service runRepeatMasker is configurable, i.e. it consumes Secondary parameters.

To edit these parameters, click on the service, and in the Details tab choose ‘Configure’
<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>engine</td>
<td>crossmatch</td>
</tr>
<tr>
<td>species</td>
<td>None</td>
</tr>
</tbody>
</table>

- The name of the parameter is on the left and the value is on the right.
- Clicking on the Value will bring up a drop down menu, an input text field, or any other appropriate field depending on the parameter.
The parameter species contains an enumerated list of possibilities.
Select human.
When you have made your selection, close the window.
• Before we run the workflow again, we will make the workflow input port *id* a port that can take in a list of strings
  – Context click on the id port and choose to edit the port. Then give the port a *List of depth 1* port type
Let’s run the workflow
• We will run our workflow with a list of values
  – Click on the id tab and then click on *New value* twice
• Enter 656461 and 654321 as the ids
• Enter NCBI_gi as the value for namespace
• Our workflow will now run using each id with the single namespace
Imagine now that you want to run the workflow using a FASTA sequence that you input yourself (without the gi identifier).

To do this, context click on getDragonBlastText and choose *Browse Biomoby service details*

- Expand the Inputs node and context click on FASTA('sequence')
- Choose Add Datatype – FASTA('sequence') to the workflow

A FASTA datatype will be added to the workflow and the appropriate links created.
Notice the datatype FASTA on the left of the workflow

- Since the datatype FASTA has a String, a String was also added to our workflow and the appropriate connection was made.

- We will now have to add another workflow input and connect it to the String component of FASTA.
• A workflow input ‘sequence’ was added to the workflow and a connection was made from the workflow input to the input port ‘value’ of String.

• We also removed the link between getGenBankFasta and getDragonBlastText by context clicking on the link on the workflow diagram and choosing to remove the link.

• Now when we choose to run our workflow, we will also have the chance to enter a FASTA sequence.

• In addition, we configured the getDragonBlastText service and made the evalue 10,000.
• Go ahead and enter any FASTA sequence as the input to the workflow input ‘sequence’
• Run the workflow
Any results can be saved by simply choosing to *Save result*
  - You will be prompted to enter a directory to save the results.
Using the Spreadsheet Import Plugin

Let’s remove the **Sequence** workflow input from our workflow, so that our workflow looks similar to the one pictured here.
Using the Spreadsheet Import Plugin

- We would like to import a spreadsheet with our data in it.
- Our data has 2 columns; sequence id and sequences.
- You can obtain the demo spreadsheet from:

  http://dev.biordf.net/~kawas/sequences.xls
Using the Spreadsheet Import Plugin

From the Service panel, we need to navigate to the node `SpreadsheetImport` located directly below the `Service templates` node.
Using the Spreadsheet Import Plugin

When we add the import plugin, we are asked to configure it.
Using the Spreadsheet Import Plugin

- In the columns section, we will only import our sequences (B to B).
- In the rows section, we will import all rows and exclude the header row.
- In the column to port name mapping, we will map column B to `fasta_sequence`. 
Using the Spreadsheet Import Plugin

Once we click on the Ok button, we will see our SpreadsheetImport activity on the canvas.

Notice that the output is fasta_sequence.
Using the Spreadsheet Import Plugin

Since our spreadsheet contains FASTA sequences, we need to connect the output port of the import activity to the value input port of our String activity.
Next we will add a widget that will prompt us for our spreadsheet.

Navigate to the Select File service located under Local services → ui node.

Add it to the workflow!
Using the Spreadsheet Import Plugin

Once we add the Select File widget to our workflow, we need to connect the output port, `selectedFile`, to the input port of SpreadsheetImport (`fileurl`).
Using the Spreadsheet Import Plugin

We need to now give the input ports for the Select File activity constant values!

This can be done by context clicking on each of the input ports and choosing *Set constant value* from the resulting menu.
Using the Spreadsheet Import Plugin

- **fileExtensions** – the extensions of the files we are interested in; set to *xls*
- **title** – the title to give the select file widget; set to *Select spreadsheet*
- **fileExtLabels** – the labels to give the extensions we provided above; again, use *Excel Spreadsheet*

Note: if the input ports for this widget are not filled in, it will fail to run!
Using the Spreadsheet Import Plugin

Run the workflow again using the inputs that we have previously used.

You will be prompted to open your spreadsheet. Do so and see Taverna utilize your data!
Using the Spreadsheet Import Plugin

Some results produced by using the spreadsheet data
DOWNLOAD THE WORKFLOW

Get the workflow for this tutorial from

http://dev.biordf.net/~kawas/t2_bioomby_tutorial.t2flow

http://dev.biordf.net/~kawas/t2_bioomby_tutorial.t2flow