



Installation manual

1.	Download XModeler.....	2
2.	Integrate XModeler in Eclipse	4
1.	Adding XModeler to Eclipse	4
2.	Setting the Java Build Path	8
3.	Editing ini.txt.....	10
4.	Editing the Program Arguments	11
5.	Done!	14
3.	XDoc.....	15

1. Download XModeler



Go to the **LE4MM** Homepage <https://www.wi-inf.uni-duisburg-essen.de/LE4MM/>

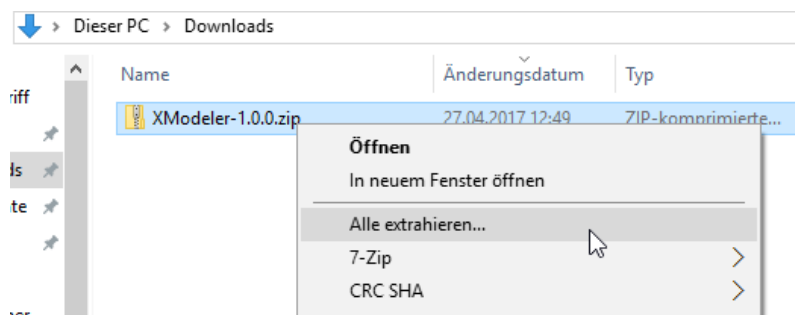
Under **Resources/Downloads** you find a direct download link for the latest XModeler version.

Downloads

Software

- **XModeler-1.0.1.zip** 82mb 
- **Github development web-site** with **Bug-Tracking** system

There is also a link provided which leads to our Github development website including a bug tracking system.

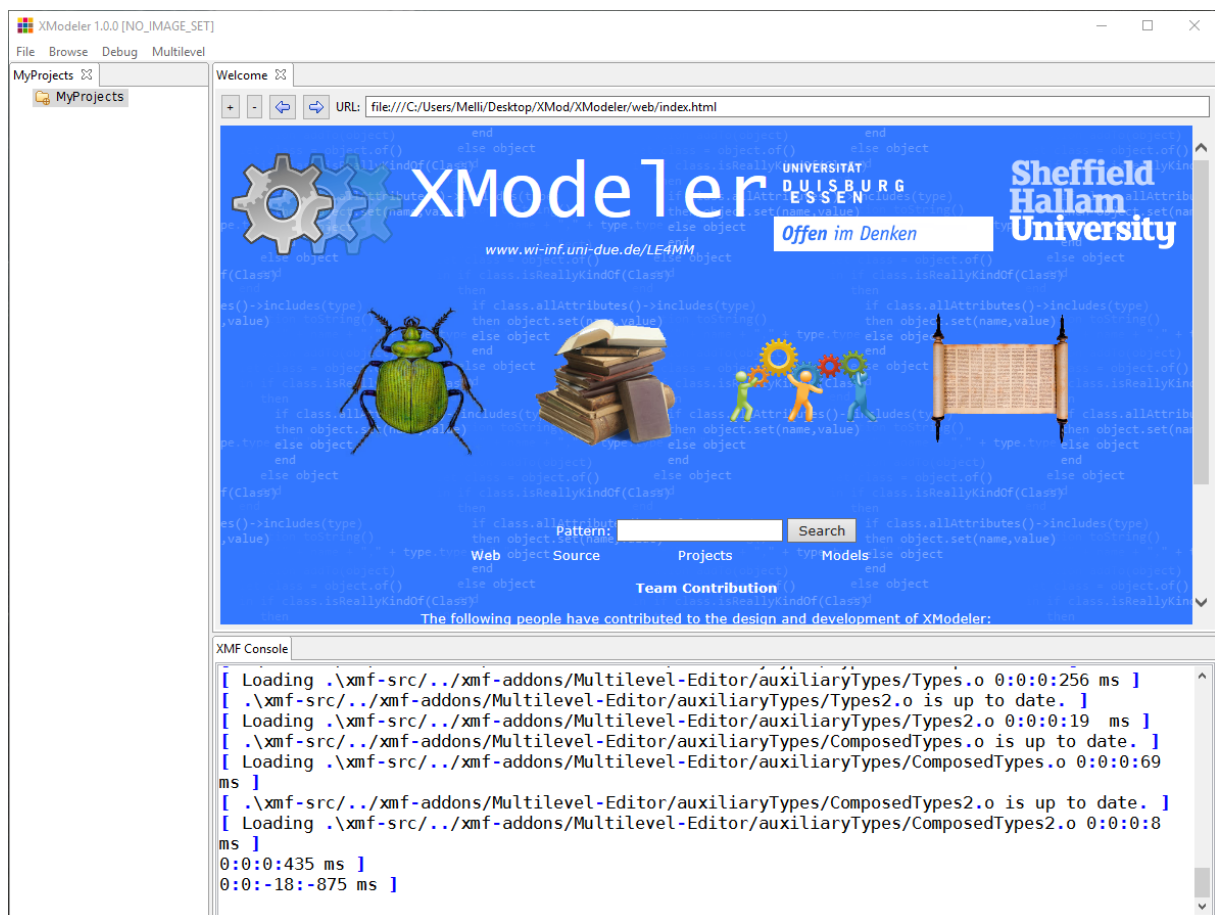


Unwrap the downloaded file.

xmf-src	14.02.2017 15:46	Dateiordner	
changelog.txt	22.05.2017 08:44	Textdokument	1 KB
ini-linux.txt	22.05.2017 08:46	Textdokument	1 KB
ini-win.txt	22.05.2017 08:46	Textdokument	1 KB
xmodeler-linux32.sh	16.05.2017 10:03	Shell Script	2 KB
xmodeler-linux64.sh	15.02.2017 09:39	Shell Script	2 KB
xmodeler-macosx64.sh	15.02.2017 09:39	Shell Script	2 KB
xmodeler-win32.bat	19.05.2017 09:23	Windows-Batchda...	2 KB
xmodeler-win64.bat	15.02.2017 09:38	Windows-Batchda...	2 KB

After unwrapping the downloaded file, double-click the application file suitable for your Windows version (e.g. xmodeler-win64.bat). Your XModeler is now ready to use.

Windows might warn you about opening the file with a security advice, but the XModeler software is safe to use.



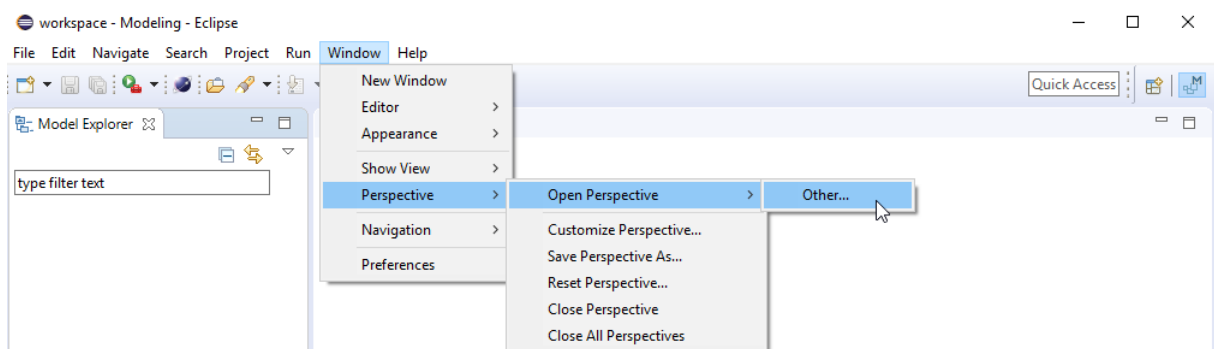
2. Integrate XModeler in Eclipse

1. Adding XModeler to Eclipse

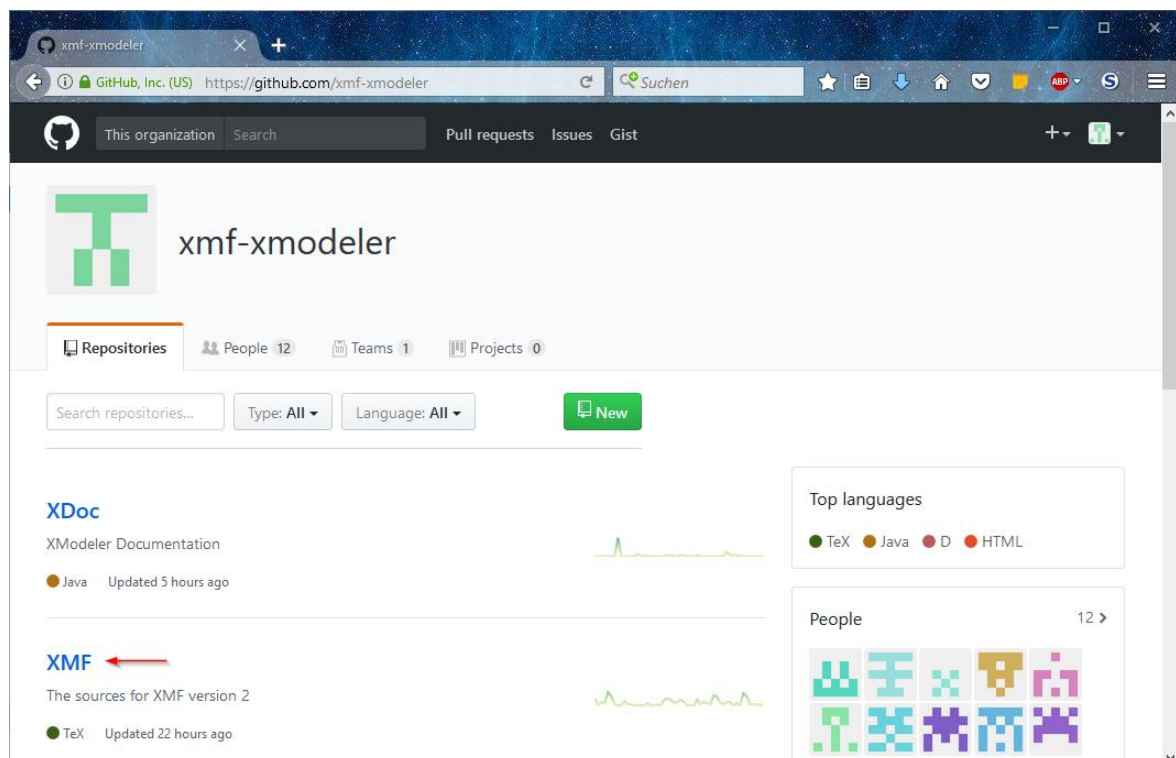
For using the XModeler in an Eclipse development environment you will need the Eclipse IDE for Java Developers (or an equivalent distribution) and the latest Java version installed on your Computer. To download the sources for the XModeler from Github you also need a Github account.

In your Eclipse, switch to the Git View.

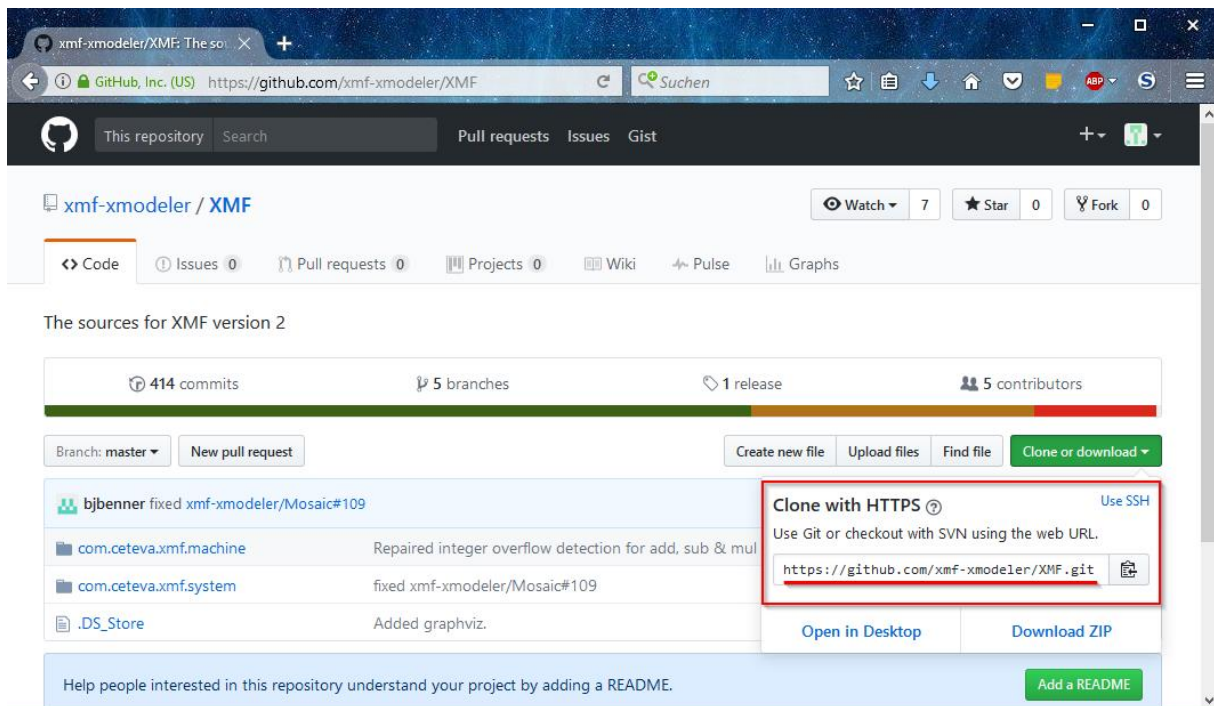
Under **Window** → **Perspective** → **Open Perspective** → **Other...**, then choose Git.



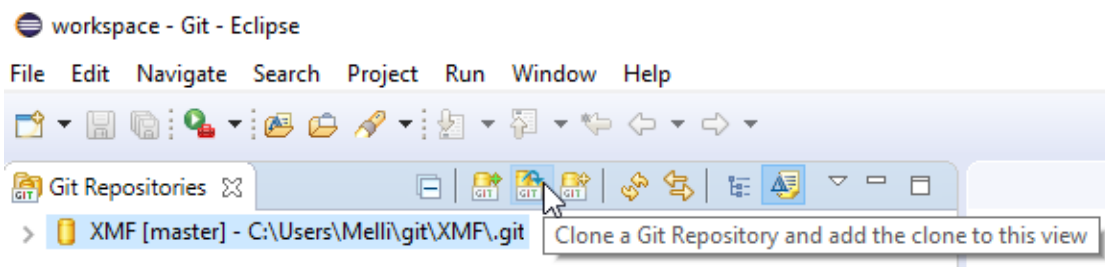
In the Git View, we will clone a Git Repository from our Code on Github.



In your browser, visit the Github development website where you find the data to copy (<https://github.com/xmf-xmodeler>). The first repository to clone is located under **xmf-xmodeler/XMF** where you find the sources for the 2nd XMF version.



Simply copy the URL provided when clicking the “**Clone or download**” button and add it as Source for our Clone Git Repository (<https://github.com/xmf-xmodeler/XMF.git>).



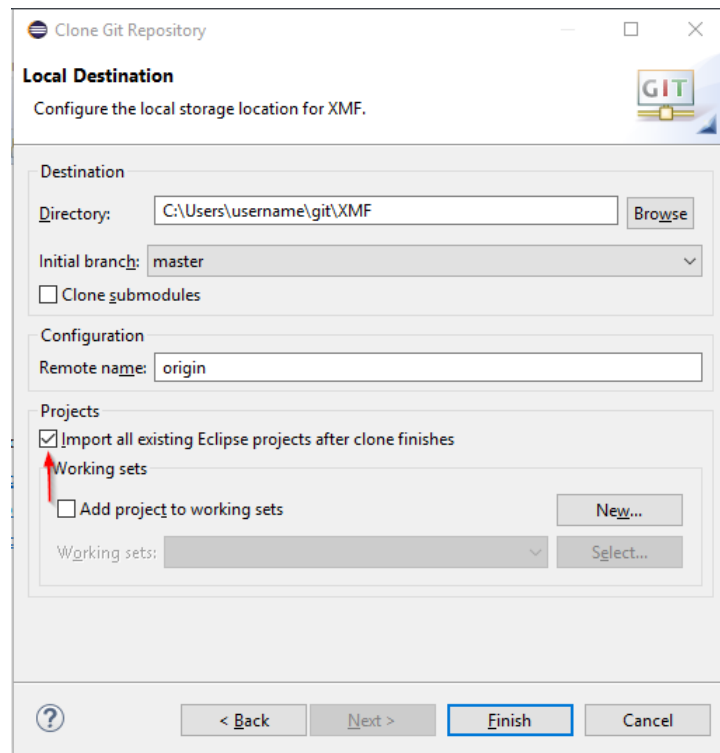
In the Git Perspective we choose **Clone a Git Repository** to clone and add the clone to our project view. In the picture, you can already see the package you are going to add.

The screenshot shows the 'Clone Git Repository' dialog box with the 'Source Git Repository' tab selected. The dialog has a title bar with a Git logo and standard window controls. Below the title bar, the text 'Source Git Repository' is followed by the instruction 'Enter the location of the source repository.' The main area is divided into three sections: 'Location', 'Connection', and 'Authentication'. The 'Location' section contains fields for 'URI:' (filled with 'https://github.com/xmf-xmodeler/XMF.git'), 'Host:' (filled with 'github.com'), and 'Repository path:' (filled with '/xmf-xmodeler/XMF.git'). There is a 'Local File...' button next to the URI field. The 'Connection' section has a 'Protocol:' dropdown set to 'https' and an empty 'Port:' field. The 'Authentication' section has a 'User:' field filled with 'user', a 'Password:' field filled with dots, and a checkbox for 'Store in Secure Store' which is unchecked. At the bottom, there are navigation buttons: a help icon, '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a blue border.

Here you add the **URL**, **Host** and **Repository path** which should fill automatically when choosing the Git Clone option. If you have Github user credentials with writing permission to the repository, enter your username and password in the "**Authentication**" section.

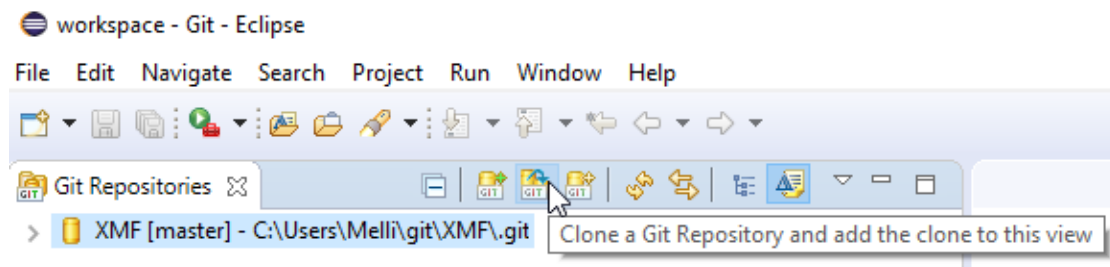
The screenshot shows the 'Clone Git Repository' dialog box with the 'Branch Selection' tab selected. The dialog has a title bar with a Git logo and standard window controls. Below the title bar, the text 'Branch Selection' is followed by the instruction 'Select branches to clone from remote repository. Remote tracking branches will be created to track updates for these branches in the remote repository.' The main area shows 'Branches of https://github.com/xmf-xmodeler/XMF.git:' followed by a list of branches: 'AddonsEssen', 'development', 'intrinsic', 'master', and 'xmf-integration'. Each branch has a checked checkbox and a small Git logo icon. Below the list are 'Select All' and 'Deselect All' buttons. At the bottom, there are navigation buttons: a help icon, '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a blue border.

Leave all branches selected and click on **Next**.



In the last window you check the box **“Import all existing Eclipse projects after clone finishes”** to display the cloned projects in your Eclipse Workspace.

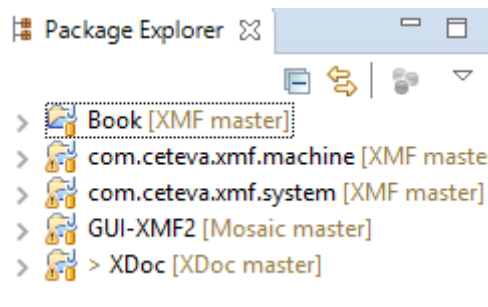
Click on Finish and the objects from Github then will be copied. This will take some time.



Next we need to add one more Git Repository. Select Clone again and follow the steps above.

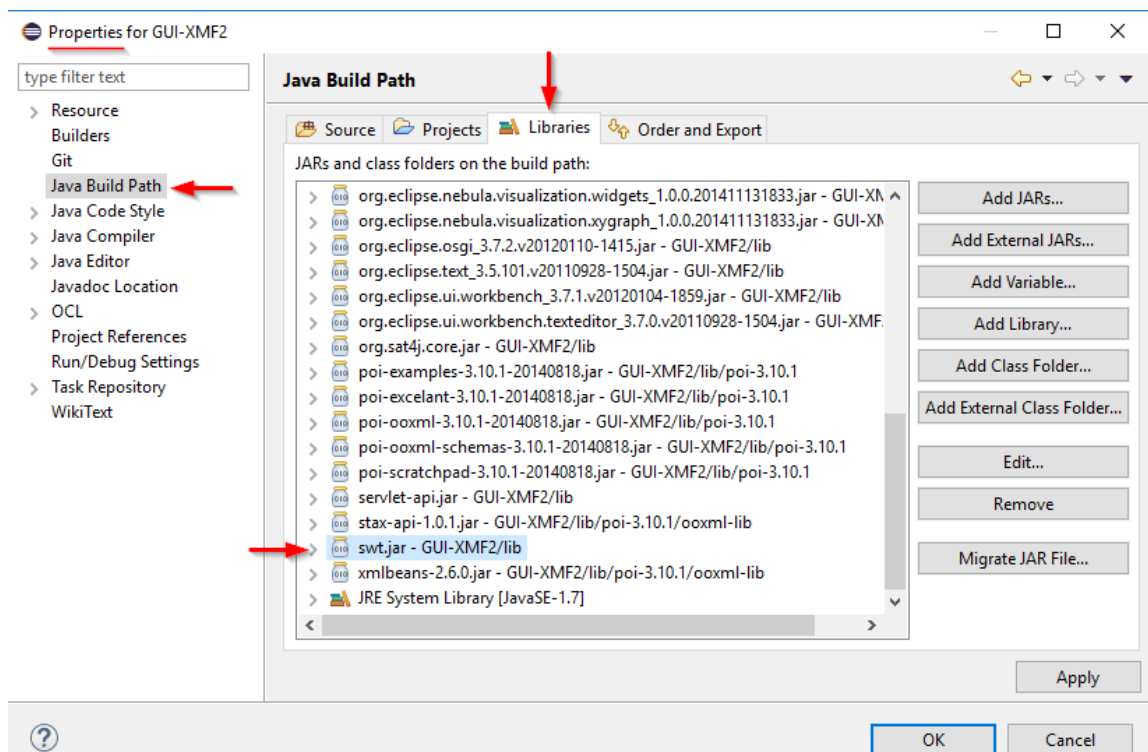
The next Repositories you need to add, is the folder **xmf-xmodeler/Mosaic** where you find the Java client for XMF2 (<https://github.com/xmf-xmodeler/Mosaic.git>).

2. Setting the Java Build Path

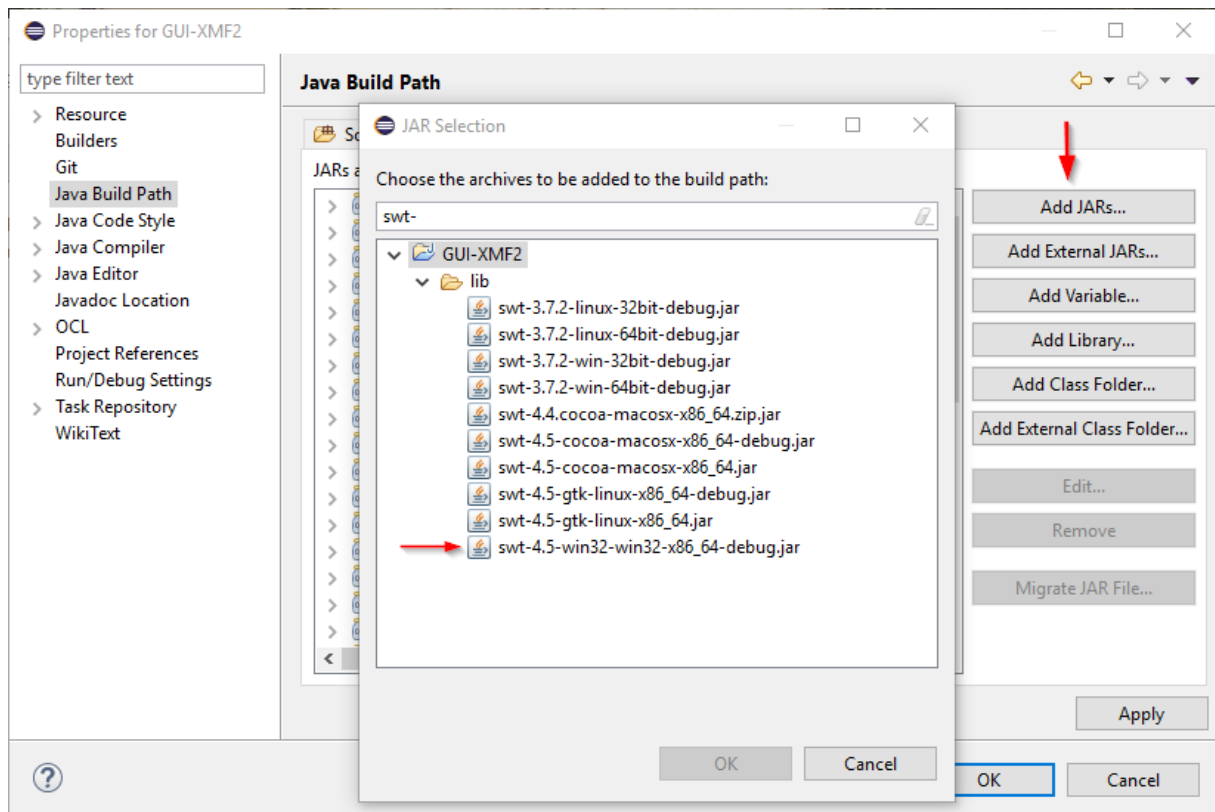


Switch back to the **Java** perspective.

You should then see these projects in your Package Explorer. The documentation project “**Book**” is not required and may be disabled by a right-click → **Close Project**.

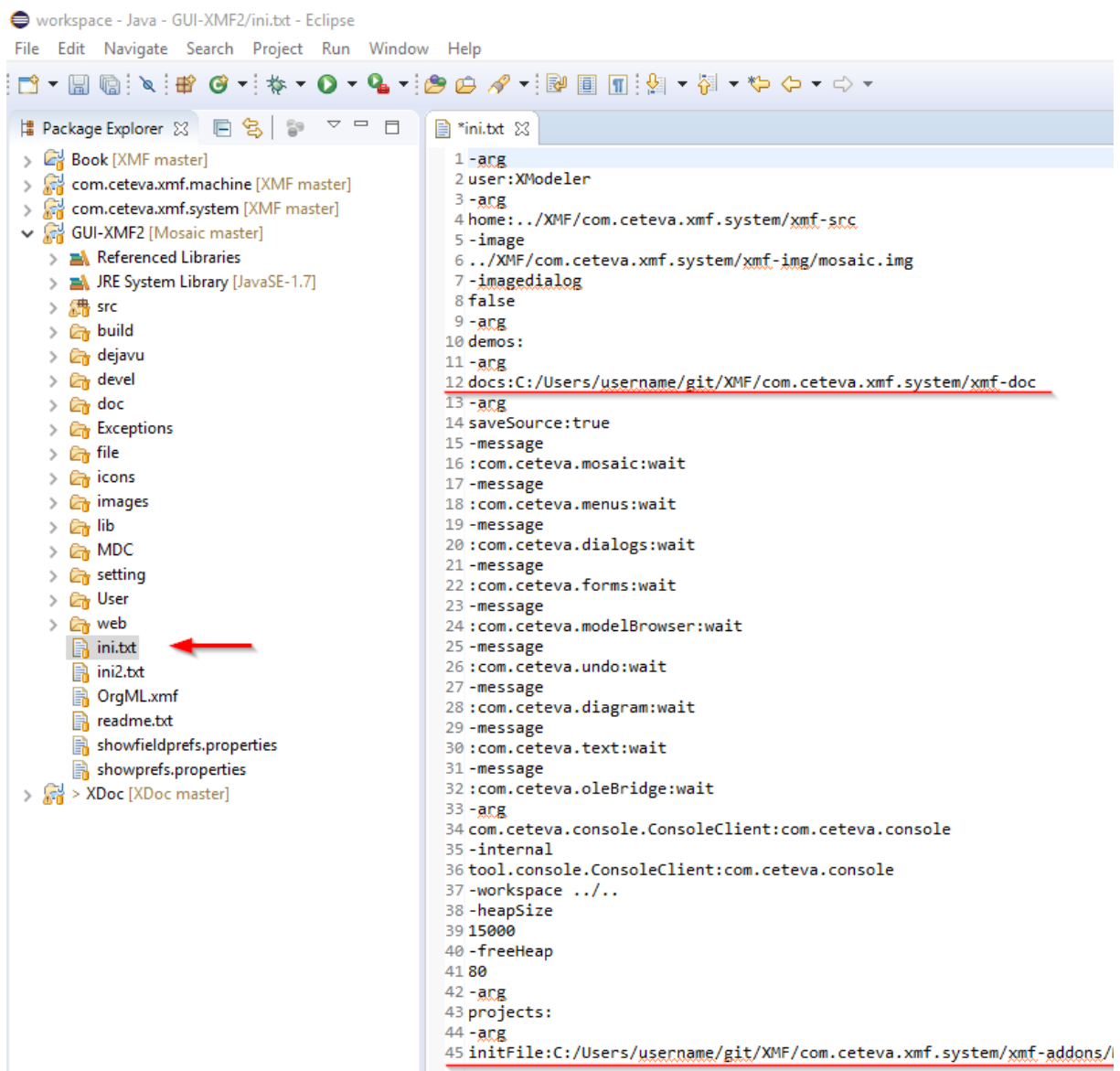


Right-click the **GUI-XMF2** project and choose **Properties**. In the **Java Build Path** section select the **Libraries** tab and **remove** the entry “**swt.jar**”.



Add via "Add JARs..." the file "swt-4.5-XXX-YYbit-debug.jar" suitable for your system (e.g. "swt-4.5-win32-win32-x86_64-debug.jar" for Windows 64bit editions). You find it under **GUI-XMF2/lib** or filter for "swt-". Then "Apply" and "OK".

3. Editing ini.txt

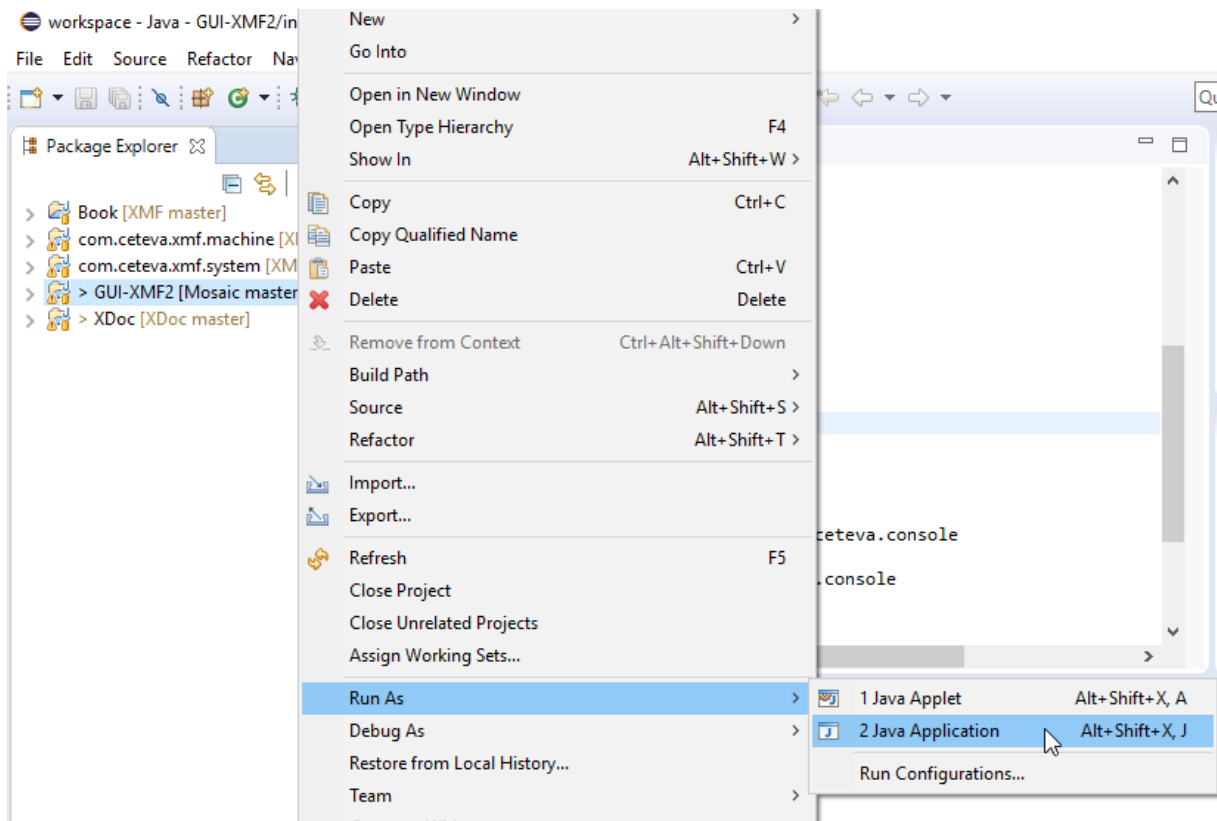


For the program to find the downloaded folders on your computer you need to change the Configuration parameters and therefore the program paths in the document “**ini.txt**”.

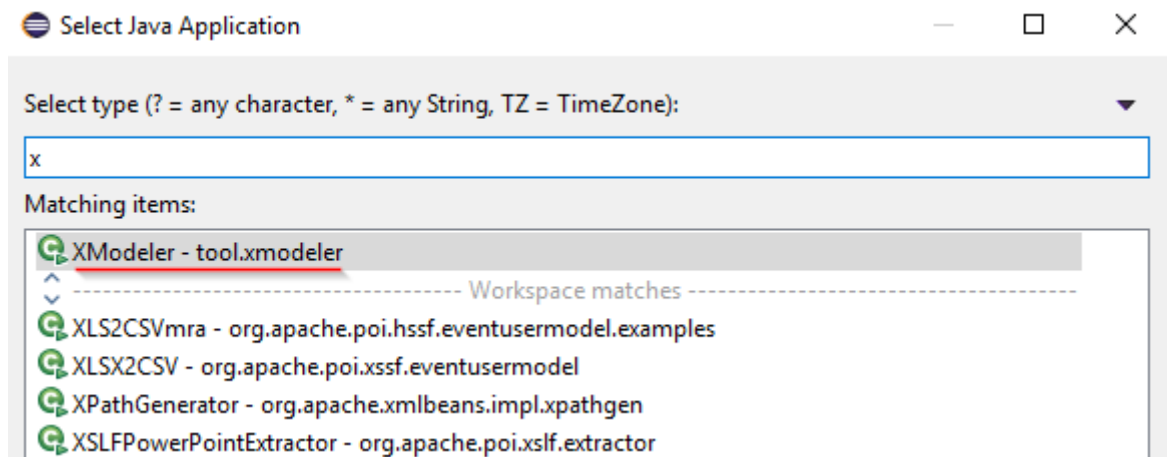
Check on your computer where the cloned XModeler files are located and correct the paths in line **12** and **45**. Don’t forget to change the username.

4. Editing the Program Arguments

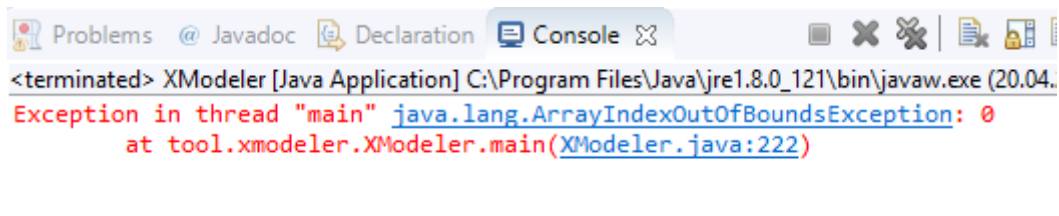
To be able to edit the program arguments we first need to run the **XModeler** once to add it to our **Eclipse Run History** to get access to the configuration menu.



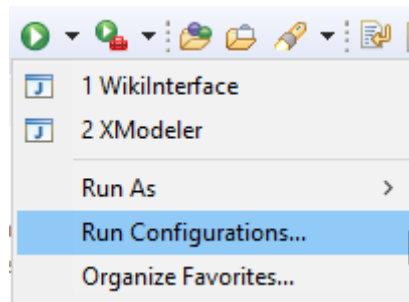
Start XModeler2 for the first time by right-clicking on the **GUI-XMF2** project → **Run As** → **Java Application** and select "**XModeler - tool.xmodeler**" from the list of available main classes.



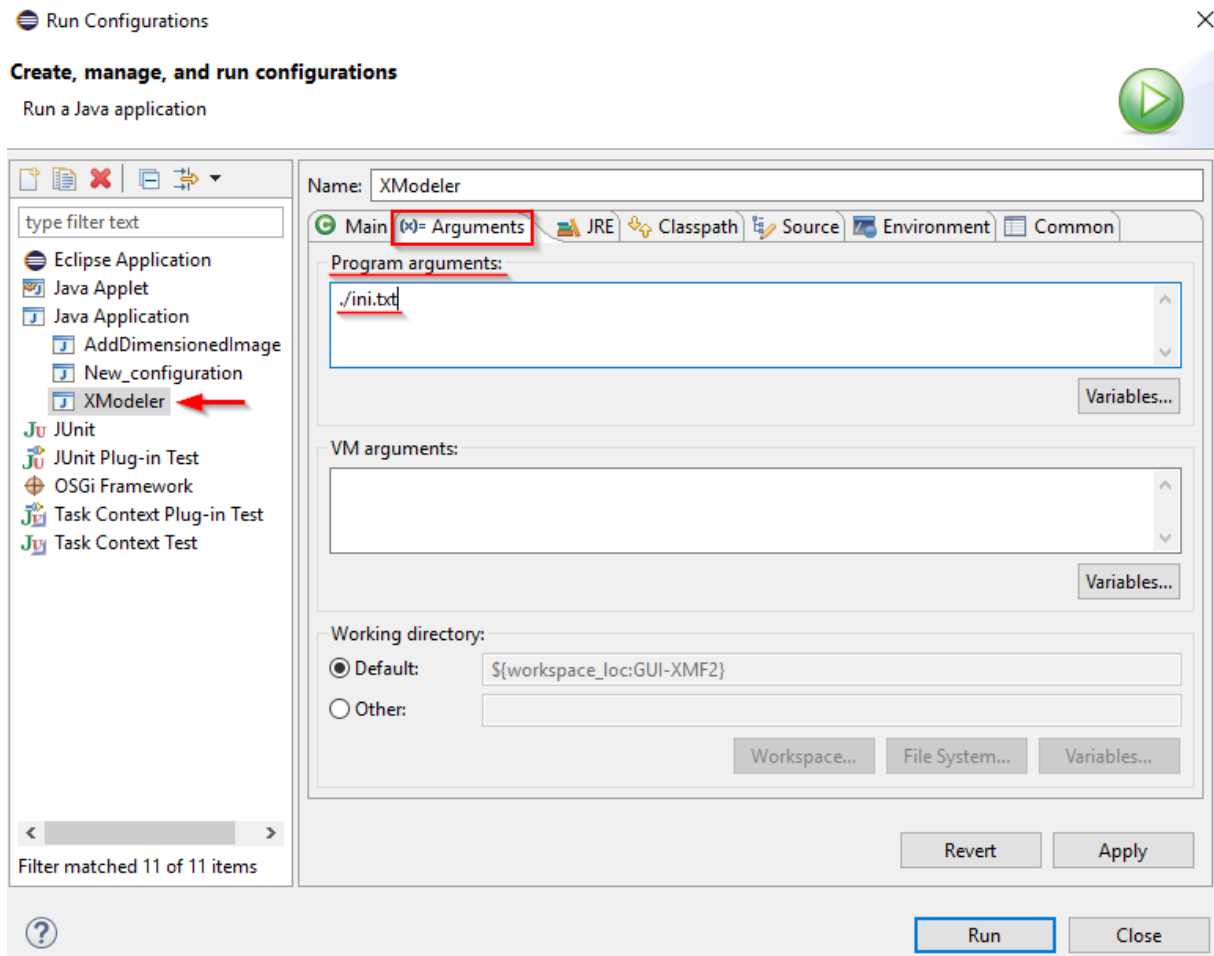
Select "**XModeler - tool.xmodeler**".



The XModeler will terminate with an "**ArrayIndexOutOfBoundsException**". To fix this issue, we need to configure how Eclipse should run the XModeler.



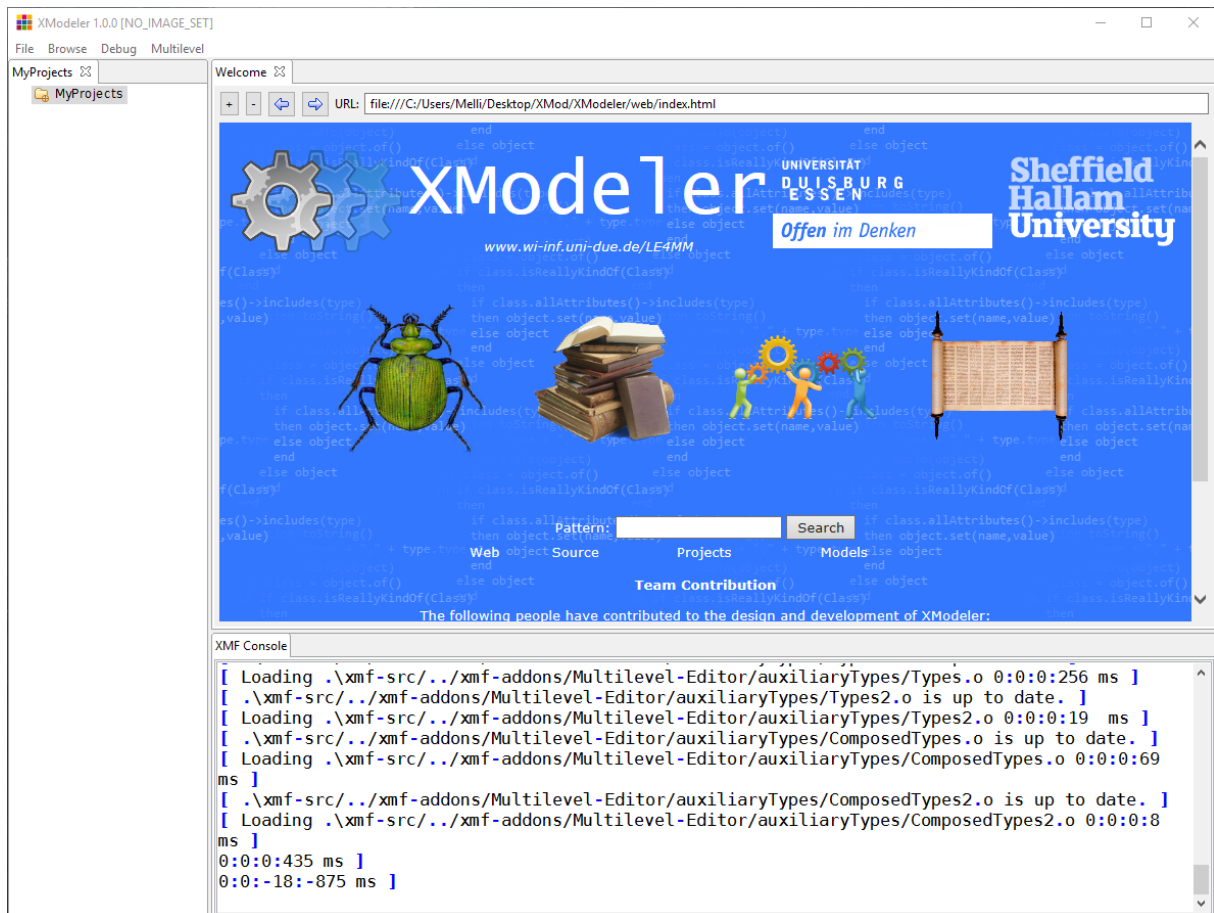
After running **XModeler** the application will be included in **Eclipse's Run History** list. Click on "**Run Configurations...**" in the Eclipse's run history list and select "**XModeler**".



Click on **"Arguments"** and insert **"./ini.txt"** into **"Program arguments"**. Click on **"Run"** in order to execute the XModeler2.

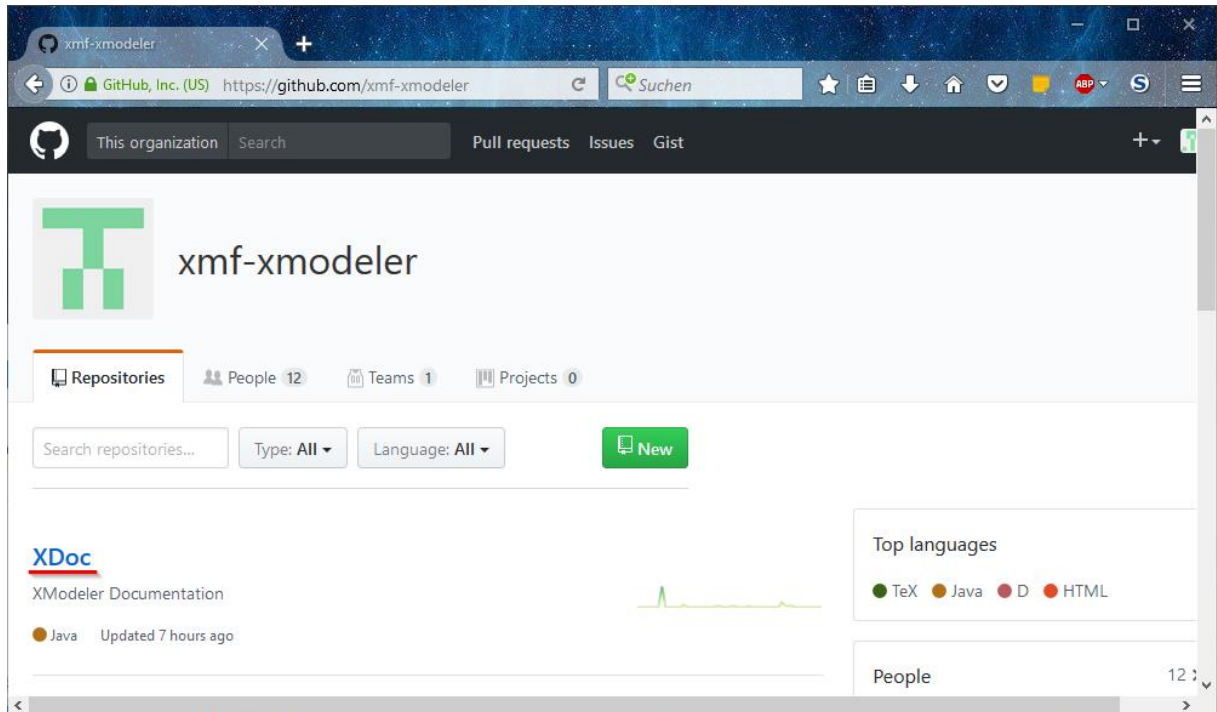
5. Done!

Now the **XModeler** can quickly be started by pressing the round green "Run" button in the Eclipse toolbar.

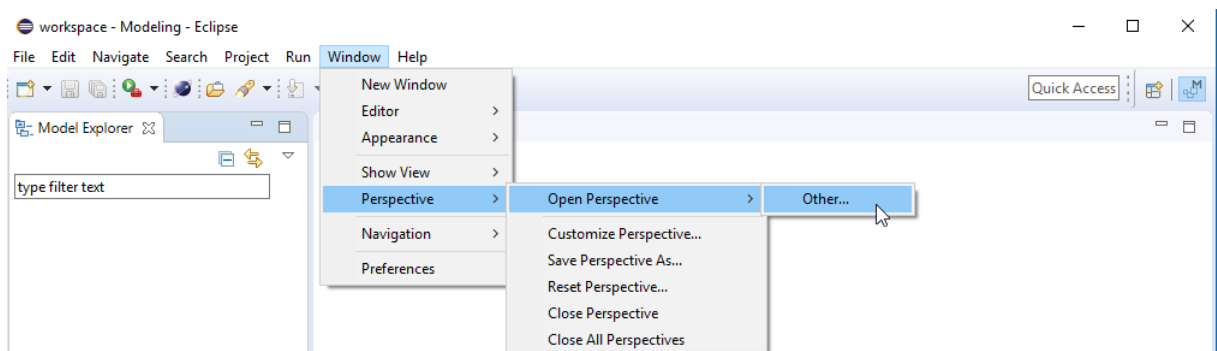


3. XDoc

For Testing the XModeler and using our Test tool in Eclipse, you need to add another Repository to your Package Explorer in Eclipse.

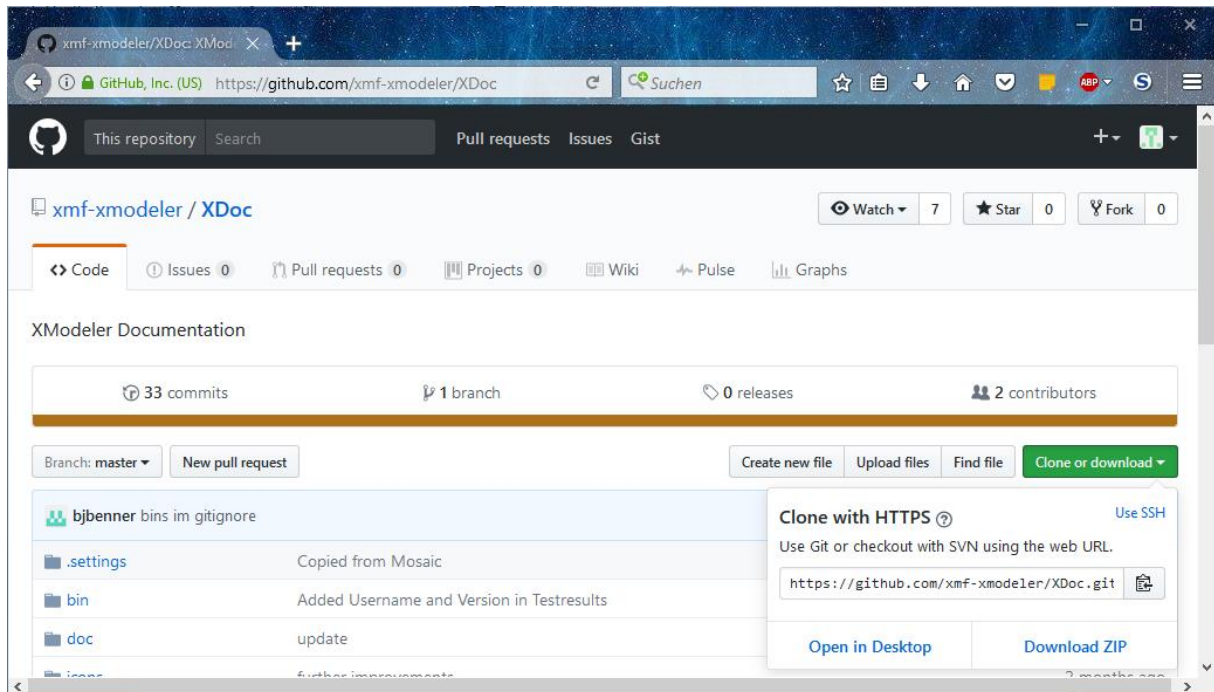


You can find the required files for the XModeler Documentation on our Github development website in the folder **xmf-xmodeler/XDoc** (<https://github.com/xmf-xmodeler/XDoc.git>).



In Eclipse, you need to switch from the Java perspective to the Git perspective by either clicking on the Git button or via **Window → Perspective → Open Perspective → Other...**

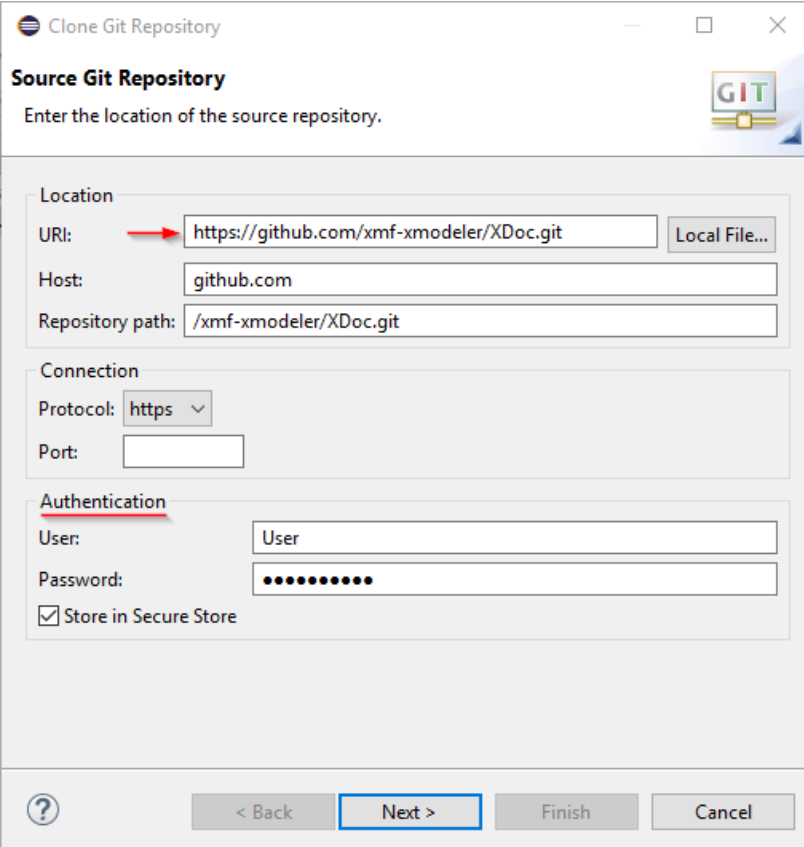
In the Git View, we will clone a Git Repositories from our Code on Github.



Simply copy the URL provided when clicking the “**Clone or download**” button and add it as Source for our Clone Git Repository. (<https://github.com/xmf-xmodeler/XDoc.git>).

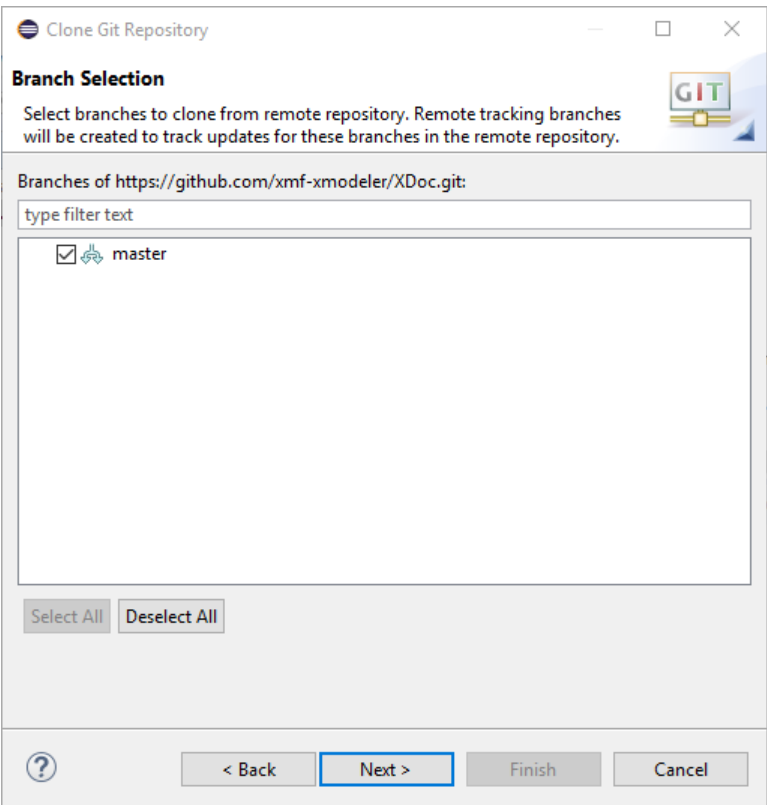


In the Git Perspective we choose Clone a Git Repository to clone and add the files for the documentation to our project view.



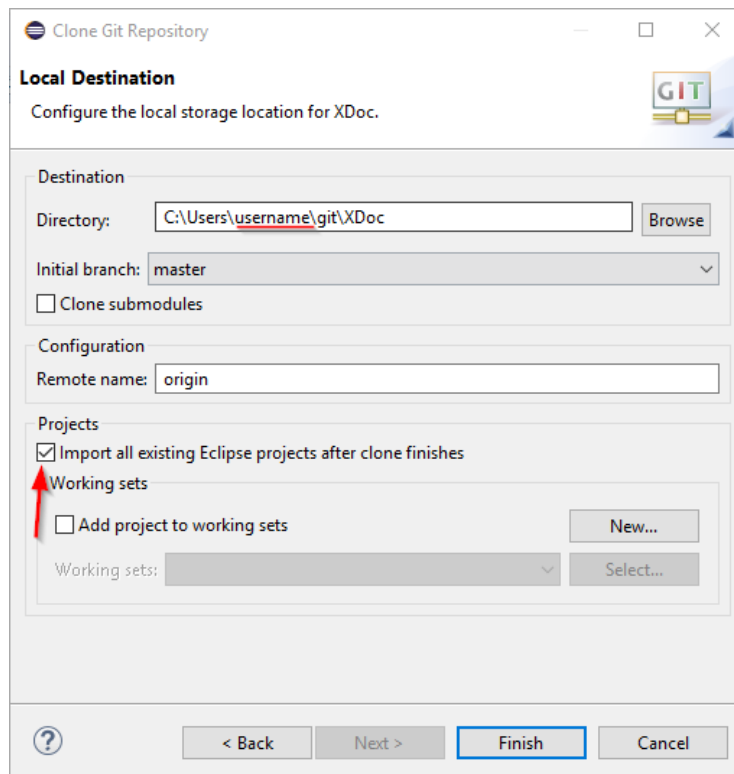
The "Clone Git Repository" dialog box is shown, with the "Source Git Repository" section active. The "Location" group contains fields for "URI:" (with a red arrow pointing to it, containing "https://github.com/xmf-xmodeler/XDoc.git"), "Host:" (containing "github.com"), and "Repository path:" (containing "/xmf-xmodeler/XDoc.git"). A "Local File..." button is to the right of the URI field. The "Connection" group has a "Protocol:" dropdown set to "https" and an empty "Port:" field. The "Authentication" group has a "User:" field (containing "User"), a "Password:" field (filled with dots), and a checked "Store in Secure Store" checkbox. At the bottom are buttons for "?", "< Back", "Next >", "Finish", and "Cancel".

If you have Github user credentials with writing permission to the repository, enter your username and password in the "**Authentication**" section.



The "Clone Git Repository" dialog box is shown, with the "Branch Selection" section active. It displays "Branches of https://github.com/xmf-xmodeler/XDoc.git:" and a list box containing a single entry: "master" with a checked checkbox and a branch icon. Above the list is a search filter labeled "type filter text". Below the list are "Select All" and "Deselect All" buttons. At the bottom are buttons for "?", "< Back", "Next >", "Finish", and "Cancel".

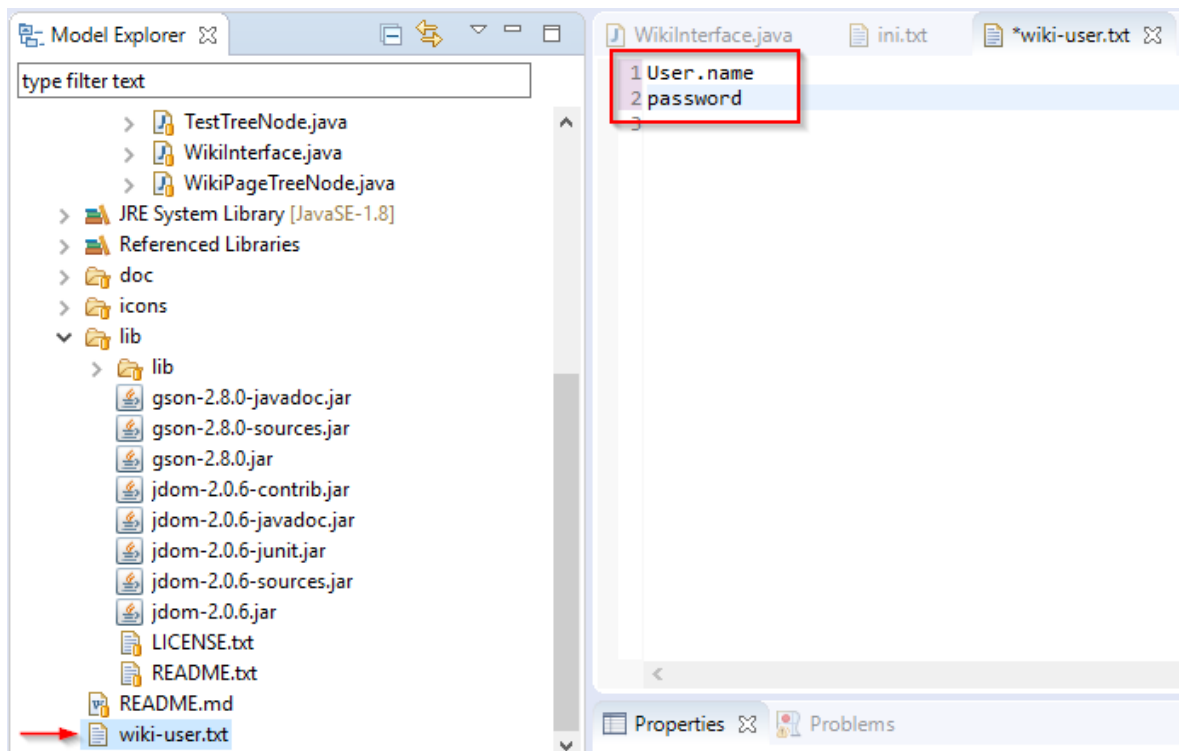
Leave the checkbox selected.



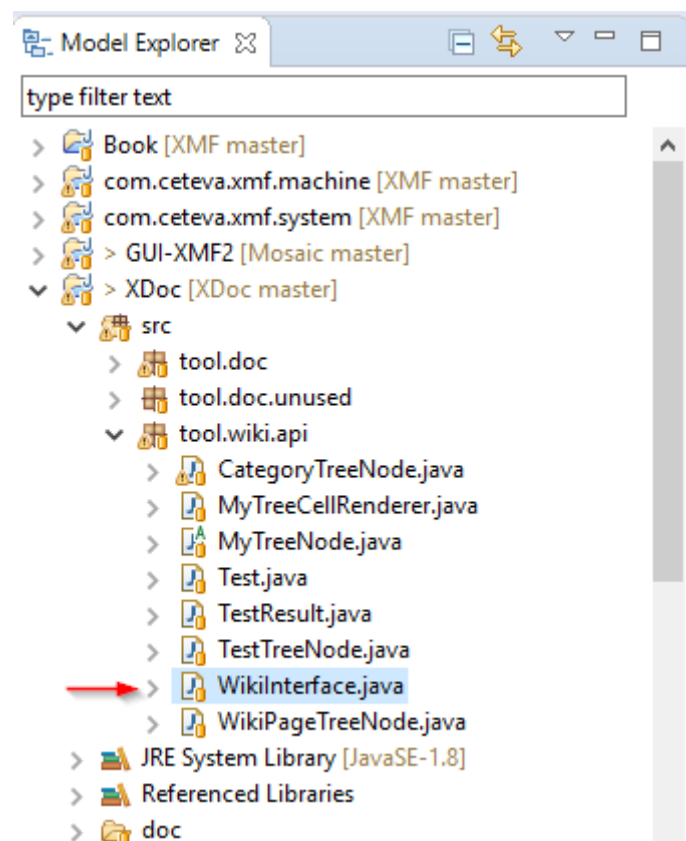
In the last window you can check the box **“Import all existing Eclipse projects after clone finishes”** for importing your projects existing in Eclipse for the clone.

Click on Finish and the objects from Github then will be copied.

To use the Bug-Tracking system you need to log in via the WikiInterface. Therefore you need to enter your user data in the wiki-user.txt file you find under **XDoc**.



Use the username and password you use for the XModeler Wiki website.



You can use the testing tool which you can find under **XDoc** → **src** → **tools.wiki.api** → **Wikiinterface.java**.

Right Click on **WikiInterface.java** and **Run As** → **Java Application** to open the test tool and automatically add it to your Eclipse run history.

The screenshot shows the 'Wiki Interface Test' window. On the left is a tree view of tests under 'XModeler' > 'Tests' > 'DiagramTest' > 'DiagramClassTest'. The 'Create Class' test is selected. The right pane shows the configuration for this test:

Comment	EMPTY
Preconditions	Choose a valid name (like "Dog"). Check there is no class on the diagram with that name
Action	Add a class via the Toolbar with the chosen name (left of diagram)
Postconditions	A class node and a class box appear with the chosen name
Last tested	<input type="text"/>
Result	EMPTY
Priority	<input type="text" value="1.0"/>
<input type="button" value="Report Test Result"/> <input type="button" value="Save (UM-Wiki:DiagramClassTest/5)"/>	

In the Wiki Interface Test Tool, you can navigate to the function type you want to test. For each test, you find a short instruction. We use the Comment field for our results. To make your results visible on our Wiki page you need to click on the Save button.