MAVERYX®
Test it simple!

Quick Start Guide
for Windows

Examples with Eclipse & JUnit

Some Useful Definitions

- **AUT**: Application Under Test
- **AUT Launch File**: the execution's configuration file that the Maveryx Test Automation Frameworks uses to launch an AUT
- **Keyword-driven testing**: a codeless approach to write test cases for non programmers
- **Data-driven testing**: a methodological approach to separate test cases from test data

Requirements

To work with Maveryx, your system shall meet the following minimum requirements:

- Windows 7 or later
- Java Runtime Environment ver. 1.8.0_161 or later (https://www.java.com/en/download/)
- .NET Runtime Environment ver. 4.6 or later (https://www.microsoft.com/en-us/download/details.aspx?id=48130)

All the examples in this Quick Start Guide use Eclipse IDE for Java EE Developers, Mars version (4,5) or later

(http://www.eclipse.org/downloads/packages/)

Summary

- Install and configure Maveryx
- Get license key
- Run the Demo project
- Create and run your first test

Test it simple!

Cap I

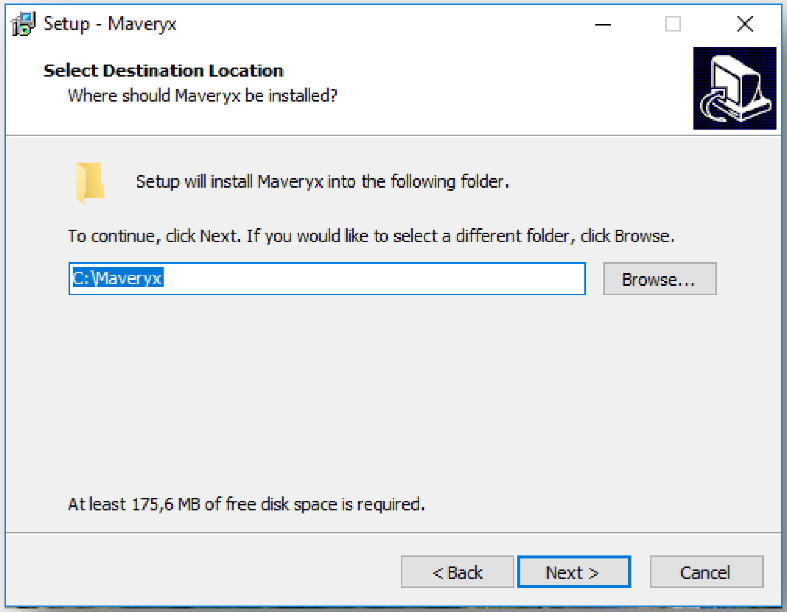
- **Install and configure Maveryx**
- Get license key
- Run the Demo project
- Create and run your first test

Test it simple!

Installation (1)

To install the Maveryx Test Automation Framework and its Eclipse Plugin on your system, run ***Maveryx_Win_2.X.y_Professional.exe*** and follow the steps of the setup wizard.

Choose the directory into which you want to install the Maveryx software. You must have write permissions to this directory.


Setup - Maveryx

Select Destination Location
Where should Maveryx be installed?

Setup will install Maveryx into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

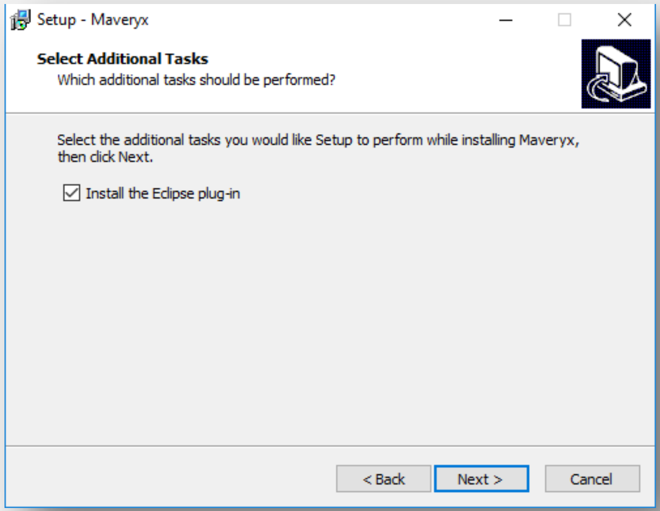
C:\Maveryx

At least 175,6 MB of free disk space is required.

Test it simple!

Installation (2)

You may click on **Install the Eclipse plug-in** to install the Maveryx Eclipse Plugin, then select the Eclipse installation directory.

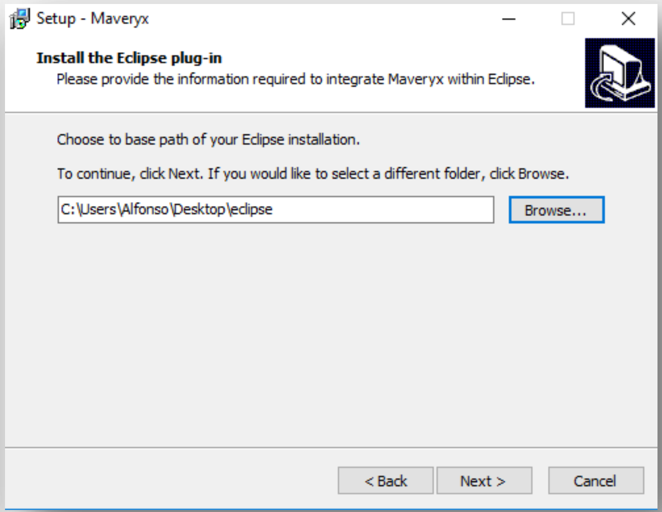


Setup - Maveryx

Select Additional Tasks
Which additional tasks should be performed?

Select the additional tasks you would like Setup to perform while installing Maveryx, then click Next.

☑ Install the Eclipse plug-in



Setup - Maveryx

Install the Eclipse plug-in
Please provide the information required to integrate Maveryx within Eclipse.

Choose to base path of your Eclipse installation.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\Users\Alfonso\Desktop\eclipse

If you want to install the Maveryx Eclipse Plugin later, copy the files in ***MAVERYX_HOME\tools\EclipsePlugin*** folder into the **/dropins** directory of your Eclipse installation.

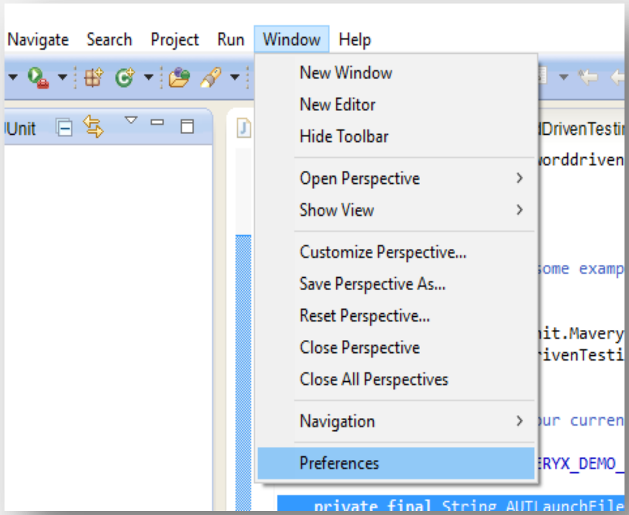
MAVERYX®
Test it simple!

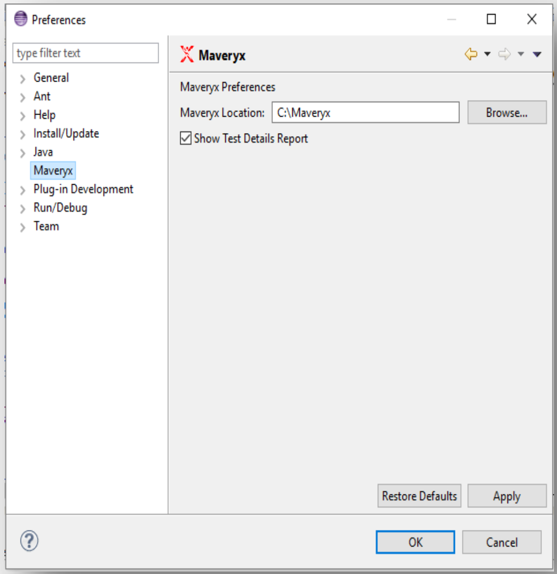
Setting Up Eclipse

- Make sure that the Maveryx Eclipse Plugin files
  - com.maveryx.ide_2.0.1.202004010017.jar
  - com.maveryx.report.chart.lib_2.0.1.202004010017.jar
  - com.maveryx.report.lib_2.0.1.202004010017.jar

  are in the /***dropins*** folder of your Eclipse installation directory

- Run Eclipse and
  - Click on "**Window > Preferences**" menu on the menu bar to open the Preferences dialog
  - Select the item "**Maveryx**" to open the Maveryx's preferences page
  - Click "**Browse…**" to select the Maveryx installation directory
  - Click "**OK**"





Maveryx Preferences
Maveryx Location: C:\Maveryx
☑ Show Test Details Report

Cap II

- Install and configure Maveryx
- **Get license key**
- Run the Demo project
- Create and run your first test

Test it simple!

Maveryx License

- Maveryx uses a **Node-Locked** license type. A node-locked license for Maveryx lets you run the application on a specific machine or workstation. This license type is considered a single-user license, although it's bound to the machine, not the user.

- Trial versions of Maveryx (which is always licensed as Node-Locked) have a time-limited license. After it expires, you can no longer use the product.

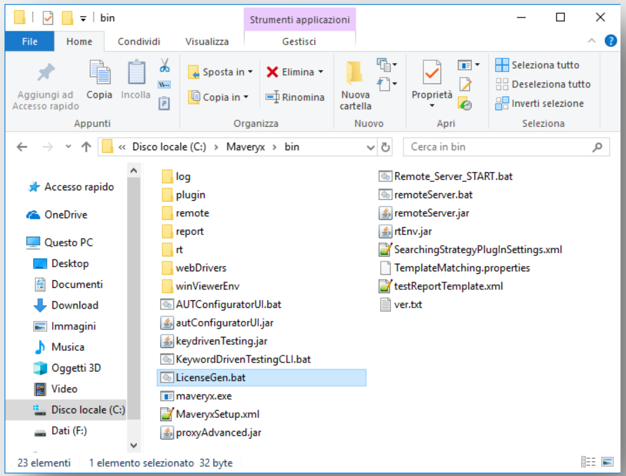
- After a commercial license for Maveryx expires, you can continue using the product. However, you will not be able to get updates for the product and technical assistance from the Maveryx Support team.

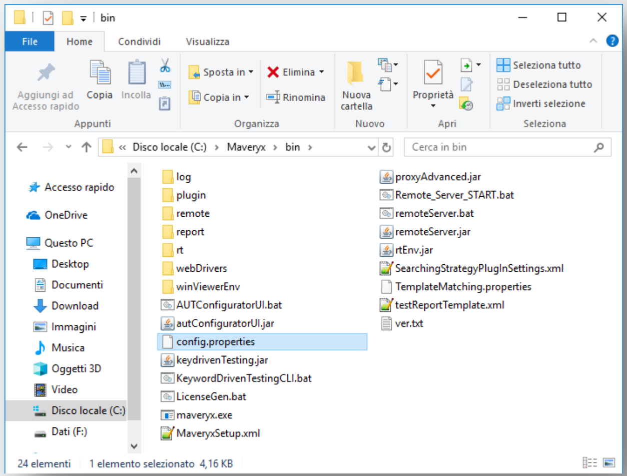
Test it simple!

Collecting Node Data

To generate a license open **MAVERYX_HOME/bin/** folder and run the **LicenseGen.bat** file.

This utility will automatically collect all hardware and software information needed to generate a valid Maveryx license by saving them into the **config.properties** file in **MAVERYX_HOME/bin/** folder.





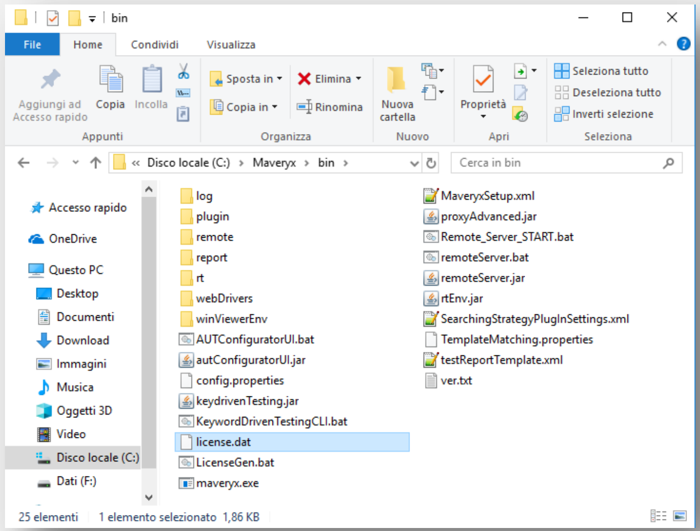
MAVERYX®
Test it simple!

License Key

When you will have the **config.properties** file, send it by email to license.manager@maveryx.com with the subject "**License Key Request**".

In reply to your email, you will receive your license key file (**license.dat**) as attachment.

Save the **license.dat** file into the **MAVERYX_HOME/bin/** folder.



« Disco locale (C:) › Maveryx › bin ›

log
plugin
remote
report
rt
webDrivers
winViewerEnv
AUTConfiguratorUI.bat
autConfiguratorUI.jar
config.properties
keydrivenTesting.jar
KeywordDrivenTestingCLI.bat
license.dat
LicenseGen.bat
maveryx.exe
MaveryxSetup.xml
proxyAdvanced.jar
Remote_Server_START.bat
remoteServer.bat
remoteServer.jar
rtEnv.jar
SearchingStrategyPlugInSettings.xml
TemplateMatching.properties
testReportTemplate.xml
ver.txt

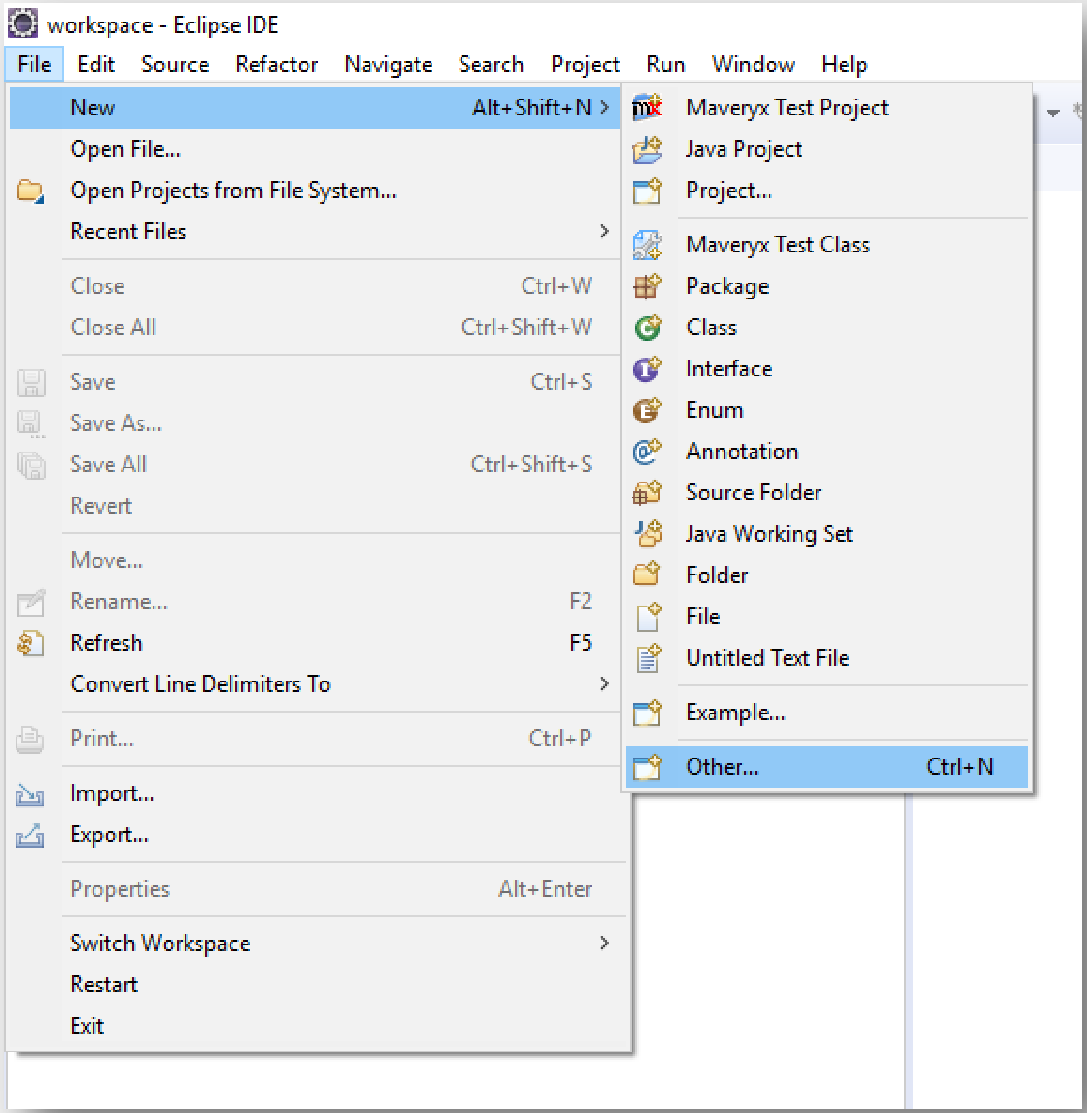
25 elementi 1 elemento selezionato 1,86 KB

Cap III

- Install and configure Maveryx
- Get license key
- **Run the Demo project**
- Create and run your first test

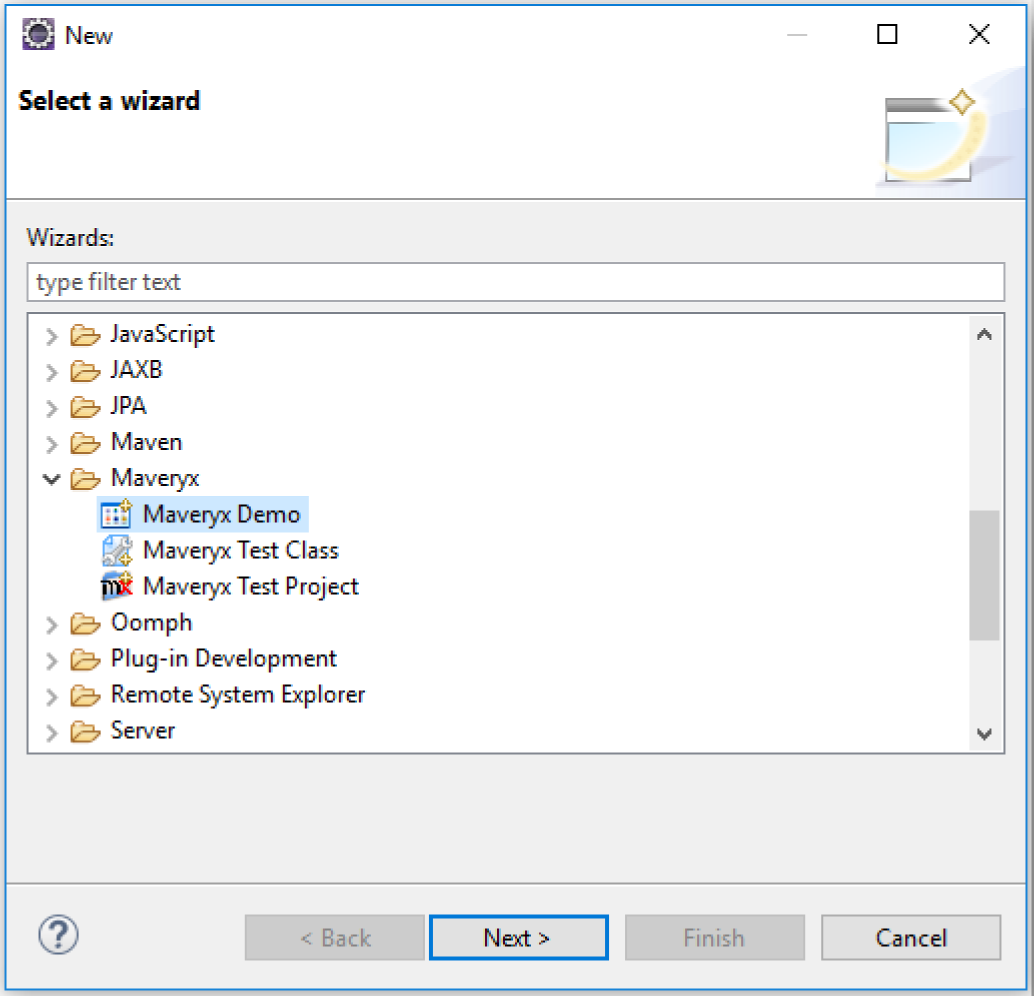
Test it simple!

Creating a Demo Project step 1



Select **File** → **New** → **Other...**

Creating a Demo Project step 2

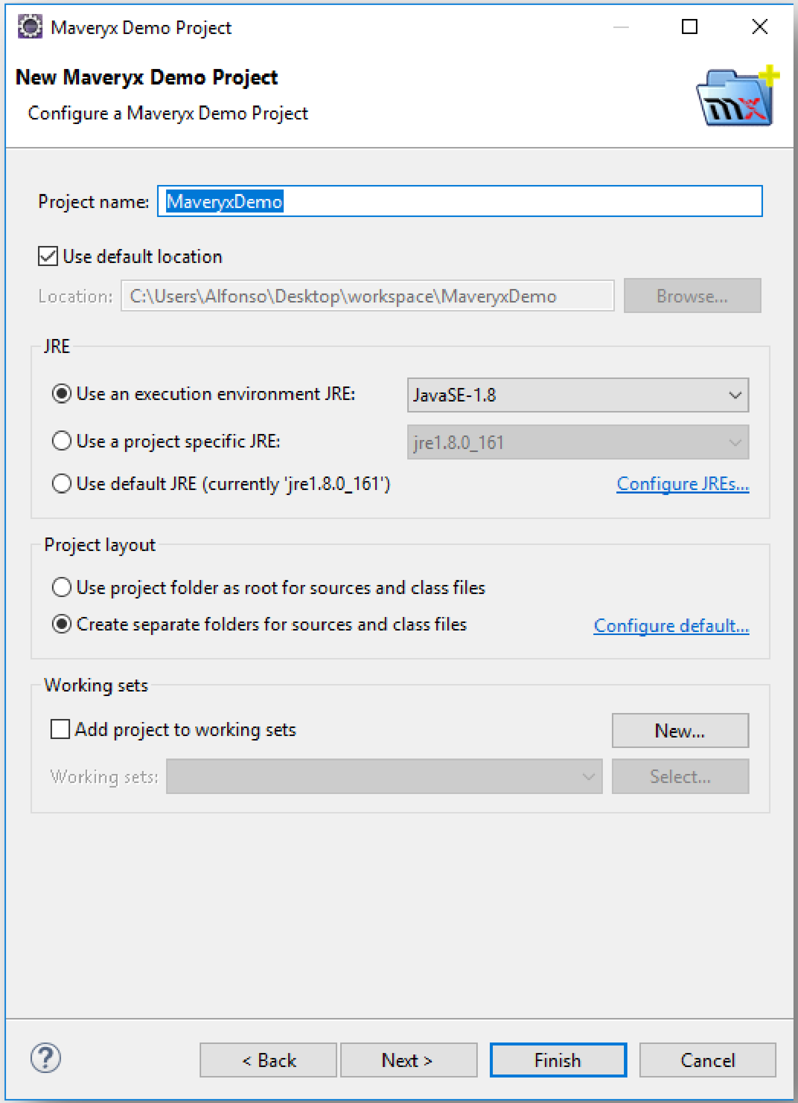


Select **Maveryx** → **Maveryx Demo**

Click **Next >**

MAVERYX® Test it simple!

Creating a Demo Project step 3


Maveryx Demo Project

New Maveryx Demo Project

Configure a Maveryx Demo Project

Project name: MaveryxDemo

[✓] Use default location

Location: C:\Users\Alfonso\Desktop\workspace\MaveryxDemo

JRE

(●) Use an execution environment JRE: JavaSE-1.8

( ) Use a project specific JRE: jre1.8.0_161

( ) Use default JRE (currently 'jre1.8.0_161')

Project layout

( ) Use project folder as root for sources and class files

(●) Create separate folders for sources and class files

Working sets

[ ] Add project to working sets

Working sets:

In the **Maveryx Demo Project** window

1. enter the Project name (default "MaveryxDemo")
2. in the **JRE** section make sure that Java/JRE 8 or higher is selected

Click **Finish**

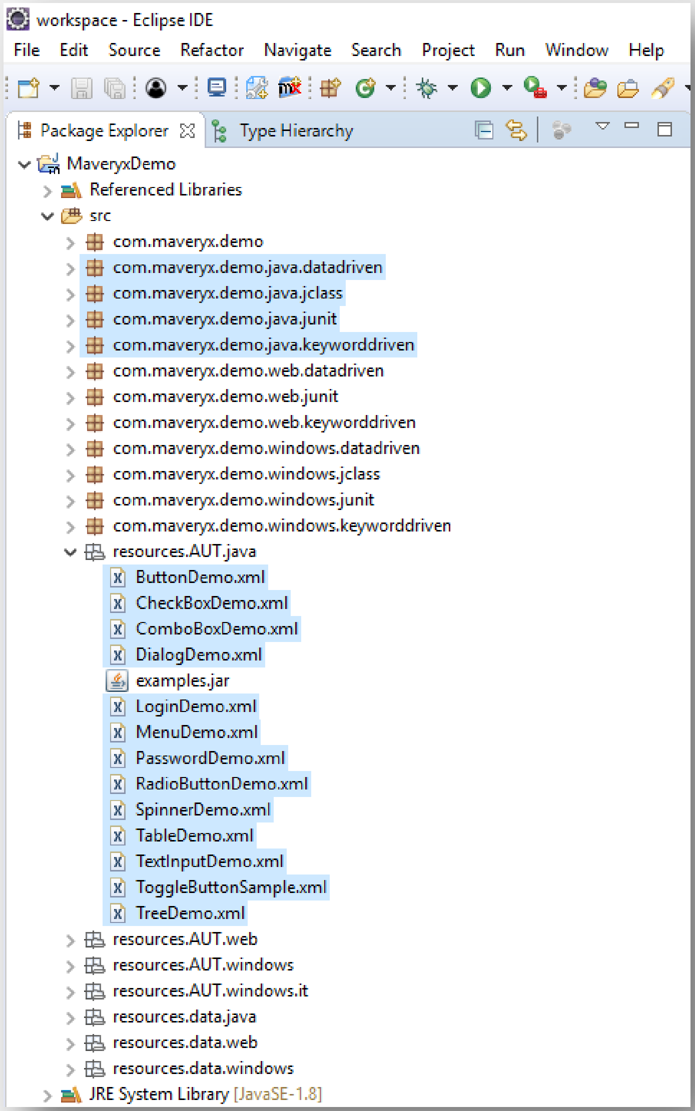
MAVERYX® Test it simple!

The Demo Project (1)

Java

The built-in Demo project has many "ready to be executed" examples for Java Desktop Applications.

Four packages with Java Class, JUnit, Keyword-driven and Data-driven examples are provided, together with and the related AUT Launch files (in *resources.AUT.java*).


workspace - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Type Hierarchy

```
MaveryxDemo
  Referenced Libraries
  src
    com.maveryx.demo
    com.maveryx.demo.java.datadriven
    com.maveryx.demo.java.jclass
    com.maveryx.demo.java.junit
    com.maveryx.demo.java.keyworddriven
    com.maveryx.demo.web.datadriven
    com.maveryx.demo.web.junit
    com.maveryx.demo.web.keyworddriven
    com.maveryx.demo.windows.datadriven
    com.maveryx.demo.windows.jclass
    com.maveryx.demo.windows.junit
    com.maveryx.demo.windows.keyworddriven
    resources.AUT.java
      ButtonDemo.xml
      CheckBoxDemo.xml
      ComboBoxDemo.xml
      DialogDemo.xml
      examples.jar
      LoginDemo.xml
      MenuDemo.xml
      PasswordDemo.xml
      RadioButtonDemo.xml
      SpinnerDemo.xml
      TableDemo.xml
      TextInputDemo.xml
      ToggleButtonSample.xml
      TreeDemo.xml
    resources.AUT.web
    resources.AUT.windows
    resources.AUT.windows.it
    resources.data.java
    resources.data.web
    resources.data.windows
  JRE System Library [JavaSE-1.8]
```

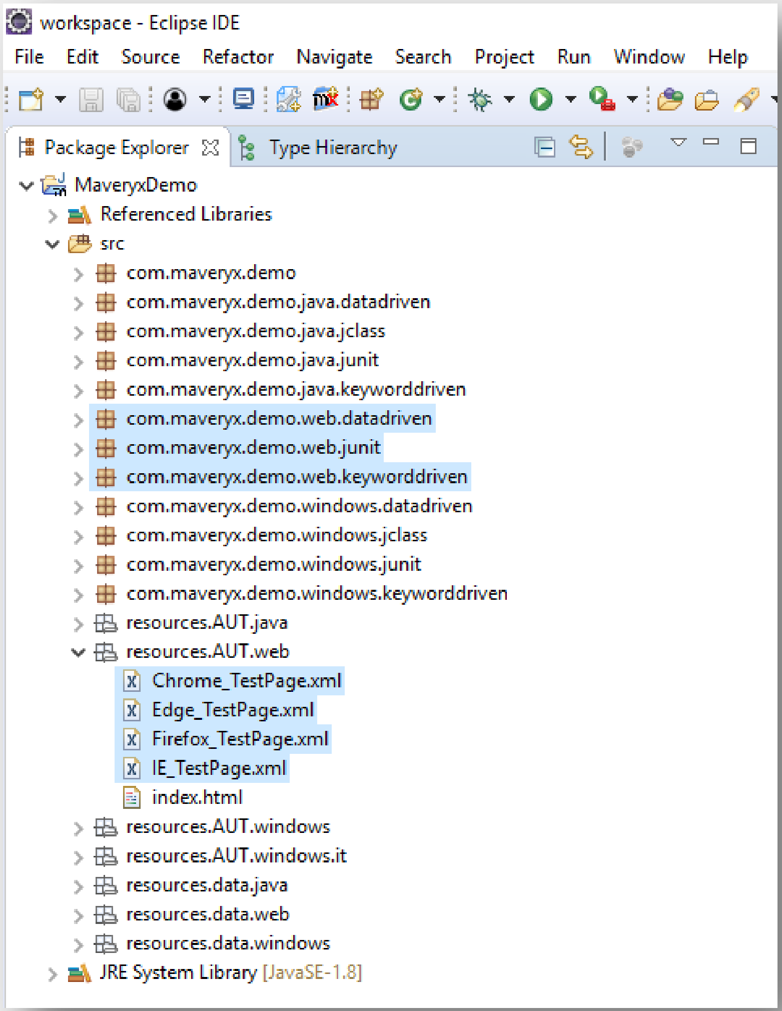
Test it simple!

The Demo Project (2)

Web

The built-in Demo project has many "ready to be executed" examples for Web Applications.

Three packages with JUnit, Keyword-driven and Data-driven examples are provided, together with and the related AUT Launch files (in *resources.AUT.web*).


workspace - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
Type Hierarchy

```
MaveryxDemo
  Referenced Libraries
  src
    com.maveryx.demo
    com.maveryx.demo.java.datadriven
    com.maveryx.demo.java.jclass
    com.maveryx.demo.java.junit
    com.maveryx.demo.java.keyworddriven
    com.maveryx.demo.web.datadriven
    com.maveryx.demo.web.junit
    com.maveryx.demo.web.keyworddriven
    com.maveryx.demo.windows.datadriven
    com.maveryx.demo.windows.jclass
    com.maveryx.demo.windows.junit
    com.maveryx.demo.windows.keyworddriven
    resources.AUT.java
    resources.AUT.web
      Chrome_TestPage.xml
      Edge_TestPage.xml
      Firefox_TestPage.xml
      IE_TestPage.xml
      index.html
    resources.AUT.windows
    resources.AUT.windows.it
    resources.data.java
    resources.data.web
    resources.data.windows
  JRE System Library [JavaSE-1.8]
```

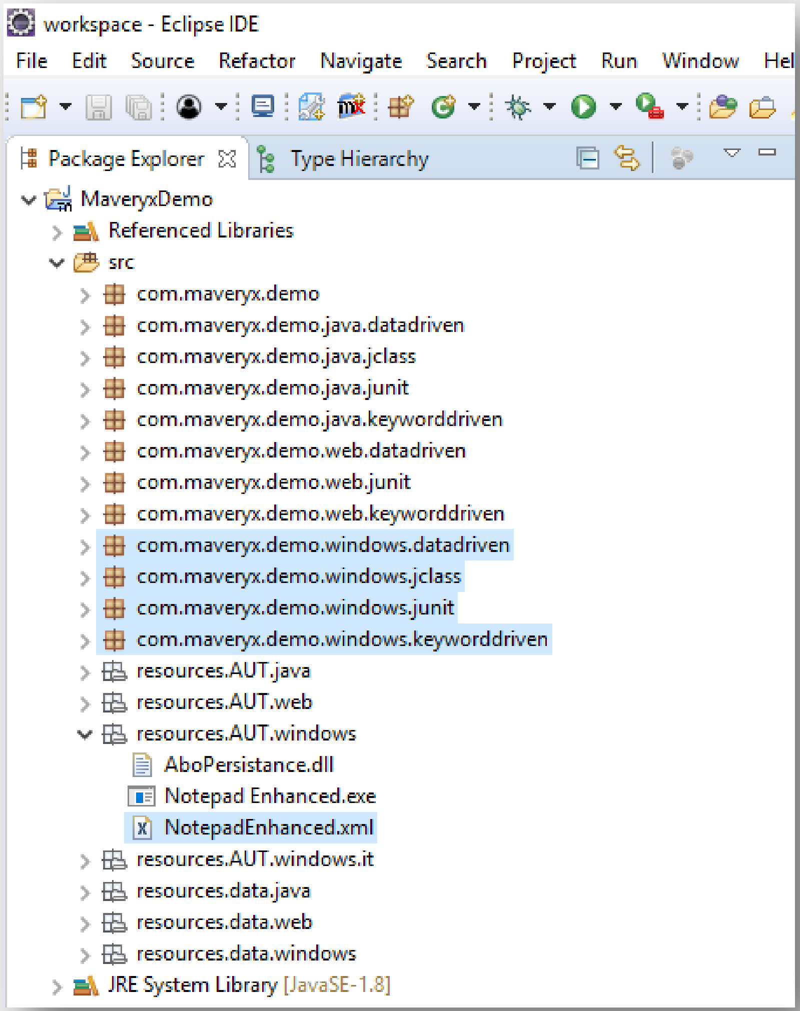
Test it simple!

The Demo Project (3)

.NET

The built-in Demo project has many "ready to be executed" examples for .NET Desktop Applications.

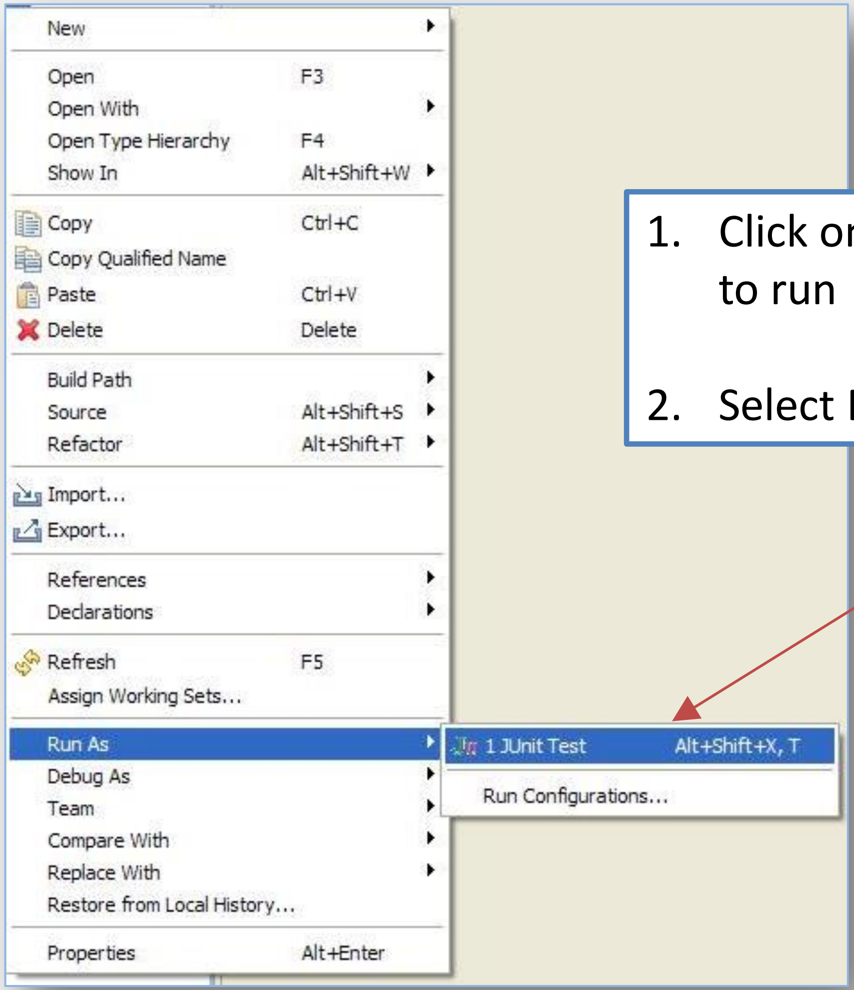
Four packages with Java Class, JUnit, Keyword-driven and Data-driven examples are provided, together with and the related AUT Launch files (in *resources.AUT.windows*).


workspace - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Hel
Package Explorer
Type Hierarchy

```
MaveryxDemo
  Referenced Libraries
  src
    com.maveryx.demo
    com.maveryx.demo.java.datadriven
    com.maveryx.demo.java.jclass
    com.maveryx.demo.java.junit
    com.maveryx.demo.java.keyworddriven
    com.maveryx.demo.web.datadriven
    com.maveryx.demo.web.junit
    com.maveryx.demo.web.keyworddriven
    com.maveryx.demo.windows.datadriven
    com.maveryx.demo.windows.jclass
    com.maveryx.demo.windows.junit
    com.maveryx.demo.windows.keyworddriven
    resources.AUT.java
    resources.AUT.web
    resources.AUT.windows
      AboPersistance.dll
      Notepad Enhanced.exe
      NotepadEnhanced.xml
    resources.AUT.windows.it
    resources.data.java
    resources.data.web
    resources.data.windows
  JRE System Library [JavaSE-1.8]
```

MAVERYX Test it simple!

Run a Test Script



1. Click on the test class or package you want to run

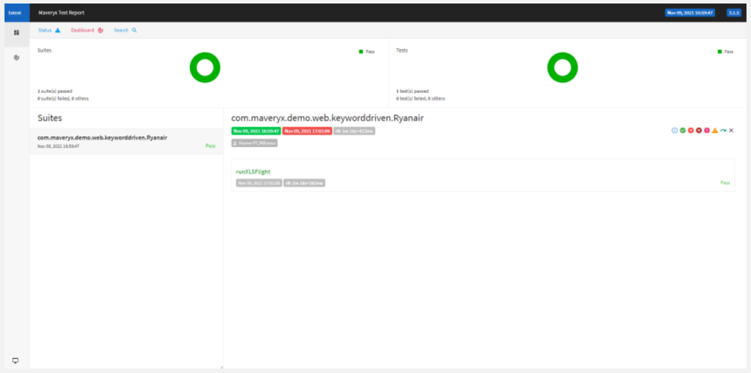
2. Select **File** → **Run As** → **JUnit Test**

Test it simple!

Report & Log

The tests execution will produce

- A Test Report in ***MAVERYX_HOME/bin/report*** folder
- A Test Log(s) in ***MAVERYX_HOME/bin/log*** folder
- A test trace in the Eclipse console
- A JUnit report in the JUnit view



1 suite(s) passed
0 suite(s) failed, 0 others

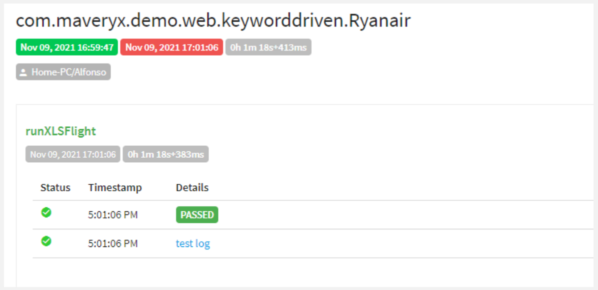
1 test(s) passed
0 test(s) failed, 0 others

Suites

com.maveryx.demo.web.keyworddriven.Ryanair
Nov 09, 2021 16:59:47 Pass

com.maveryx.demo.web.keyworddriven.Ryanair
Nov 09, 2021 16:59:47 Nov 09, 2021 17:01:06 0h 1m 18s+413ms
Home-PC/Alfonso

runXLSFlight
Nov 09, 2021 17:01:06 0h 1m 18s+383ms Pass



com.maveryx.demo.web.keyworddriven.Ryanair

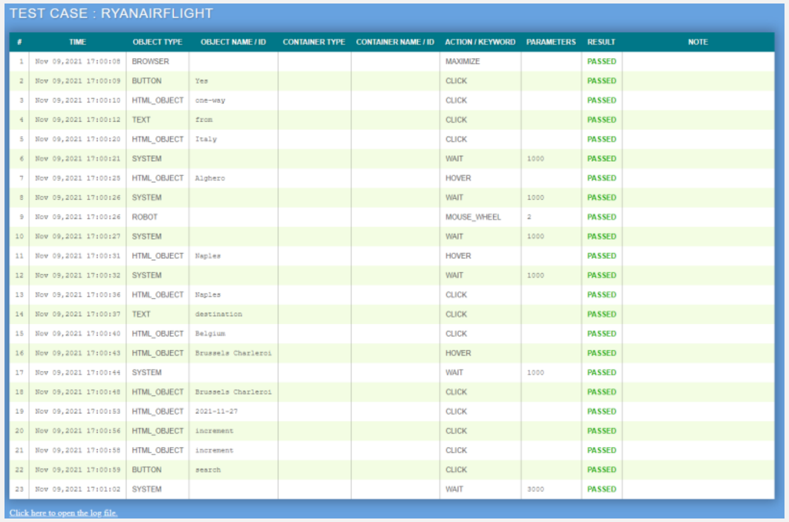
Nov 09, 2021 16:59:47 Nov 09, 2021 17:01:06 0h 1m 18s+413ms

Home-PC/Alfonso

runXLSFlight

Nov 09, 2021 17:01:06 0h 1m 18s+383ms

Status	Timestamp	Details
✓	5:01:06 PM	PASSED
✓	5:01:06 PM	test log



TEST CASE : RYANAIRFLIGHT

#	TIME	OBJECT TYPE	OBJECT NAME / ID	CONTAINER TYPE	CONTAINER NAME / ID	ACTION / KEYWORD	PARAMETERS	RESULT	NOTE
1	Nov 09,2021 17:00:08	BROWSER				MAXIMIZE		PASSED	
2	Nov 09,2021 17:00:09	BUTTON	Yes			CLICK		PASSED	
3	Nov 09,2021 17:00:10	HTML_OBJECT	one-way			CLICK		PASSED	
4	Nov 09,2021 17:00:12	TEXT	from			CLICK		PASSED	
5	Nov 09,2021 17:00:20	HTML_OBJECT	Italy			CLICK		PASSED	
6	Nov 09,2021 17:00:21	SYSTEM				WAIT	1000	PASSED	
7	Nov 09,2021 17:00:25	HTML_OBJECT	Alghero			HOVER		PASSED	
8	Nov 09,2021 17:00:26	SYSTEM				WAIT	1000	PASSED	
9	Nov 09,2021 17:00:26	ROBOT				MOUSE_WHEEL	2	PASSED	
10	Nov 09,2021 17:00:27	SYSTEM				WAIT	1000	PASSED	
11	Nov 09,2021 17:00:31	HTML_OBJECT	Naples			HOVER		PASSED	
12	Nov 09,2021 17:00:32	SYSTEM				WAIT	1000	PASSED	
13	Nov 09,2021 17:00:36	HTML_OBJECT	Naples			CLICK		PASSED	
14	Nov 09,2021 17:00:37	TEXT	destination			CLICK		PASSED	
15	Nov 09,2021 17:00:40	HTML_OBJECT	Belgium			CLICK		PASSED	
16	Nov 09,2021 17:00:43	HTML_OBJECT	Brussels Charleroi			HOVER		PASSED	
17	Nov 09,2021 17:00:44	SYSTEM				WAIT	1000	PASSED	
18	Nov 09,2021 17:00:48	HTML_OBJECT	Brussels Charleroi			CLICK		PASSED	
19	Nov 09,2021 17:00:53	HTML_OBJECT	2021-11-27			CLICK		PASSED	
20	Nov 09,2021 17:00:56	HTML_OBJECT	increment			CLICK		PASSED	
21	Nov 09,2021 17:00:58	HTML_OBJECT	increment			CLICK		PASSED	
22	Nov 09,2021 17:00:59	BUTTON	search			CLICK		PASSED	
23	Nov 09,2021 17:01:02	SYSTEM				WAIT	3000	PASSED	

Click here to open the log file.

MAVERYX® Test it simple!

Cap IV

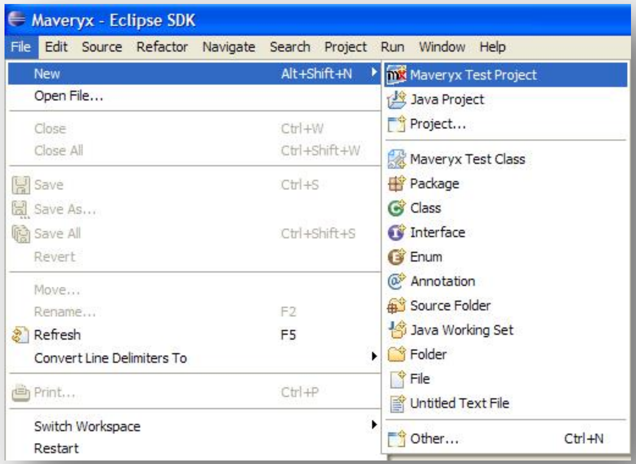
- Install and configure Maveryx
- Get license key
- Run the Demo project
- **Create and run your first test**

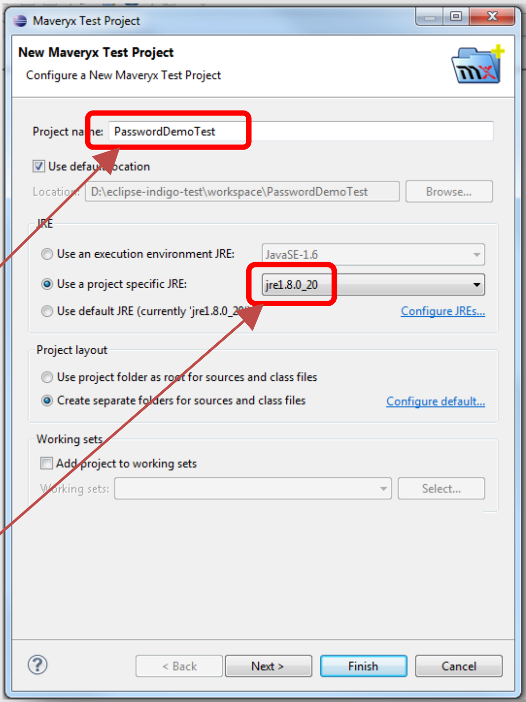
Test it simple!

Create and run your first test

1. **Create a new Maveryx Test Project**
2. Create a new Maveryx Test Class
3. Create the AUT Launch file
4. Write the test case
5. Run the test

Create New Test Project




Maveryx Test Project

New Maveryx Test Project

Configure a New Maveryx Test Project

Project name: PasswordDemoTest

[✓] Use default location

Location: D:\eclipse-indigo-test\workspace\PasswordDemoTest

JRE

- ○ Use an execution environment JRE: JavaSE-1.6
- ● Use a project specific JRE: jre1.8.0_20
- ○ Use default JRE (currently 'jre1.8.0_20')

Project layout

- ○ Use project folder as root for sources and class files
- ● Create separate folders for sources and class files

Working sets

[ ] Add project to working sets

Working sets:

1. Select **File → New → Maveryx Test Project**

In the **Maveryx Test Project** window

   1. enter the Project name (e.g. "*PasswordDemoTest*")
   2. in the **JRE** section make sure that Java/JRE 8 or higher is selected

2. Click **Finish**

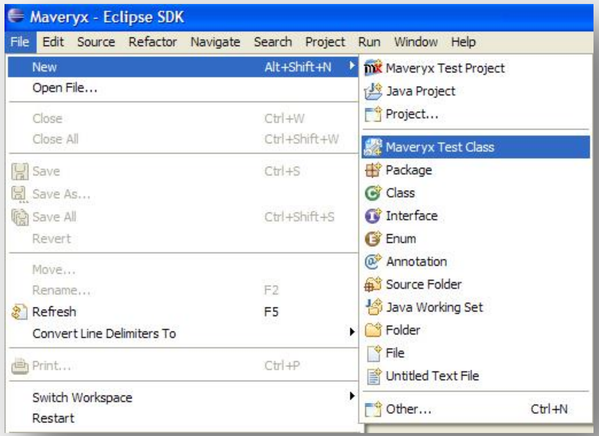
MAVERYX® Test it simple!

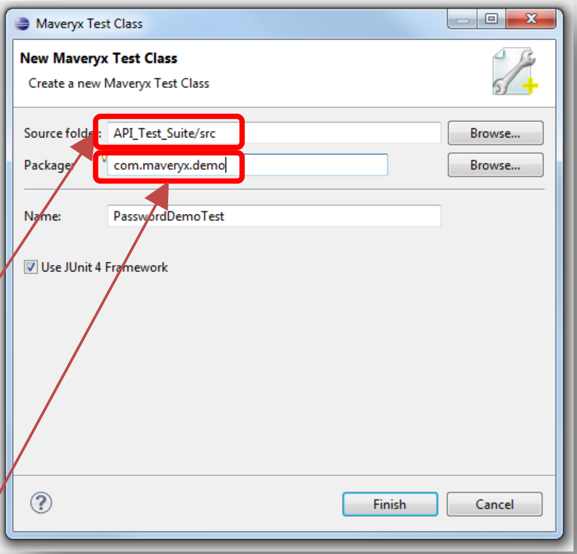
Create and run your first test

1. Create a new Maveryx Test Project
2. **Create a new Maveryx Test Class**
3. Create the AUT Launch file
4. Write the test case
5. Run the test

Test it simple!

Create New Test Script





Source folder: API_Test_Suite/src
Package: com.maveryx.demo
Name: PasswordDemoTest
☑ Use JUnit 4 Framework

1. Select **File → New → Maveryx Test Class**

In the **Maveryx Test Class** window

   1. enter a name for the Package (e.g. "*com.maveryx.demo*")
   2. enter a Name for the test class / script (e.g. "*PasswordDemoTest*")

2. Click **Finish**

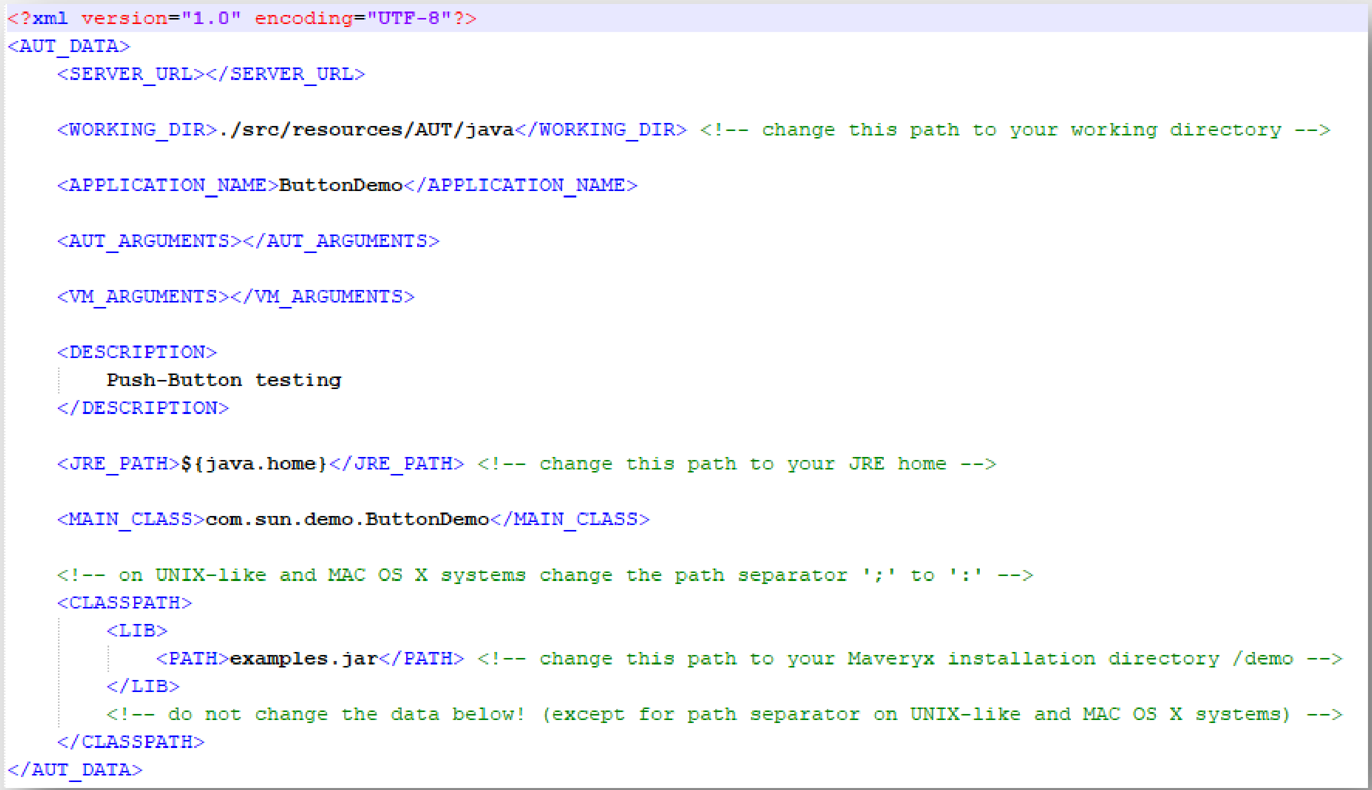
Create and run your first test

1. Create a new Maveryx Test Project
2. Create a new Maveryx Test Class
3. **Create the AUT Launch file**
4. Write the test case
5. Run the test

Test it simple!

Java AUT Launch File

To execute a Java Application-Under-Test it is necessary to create the related AUT launch file.

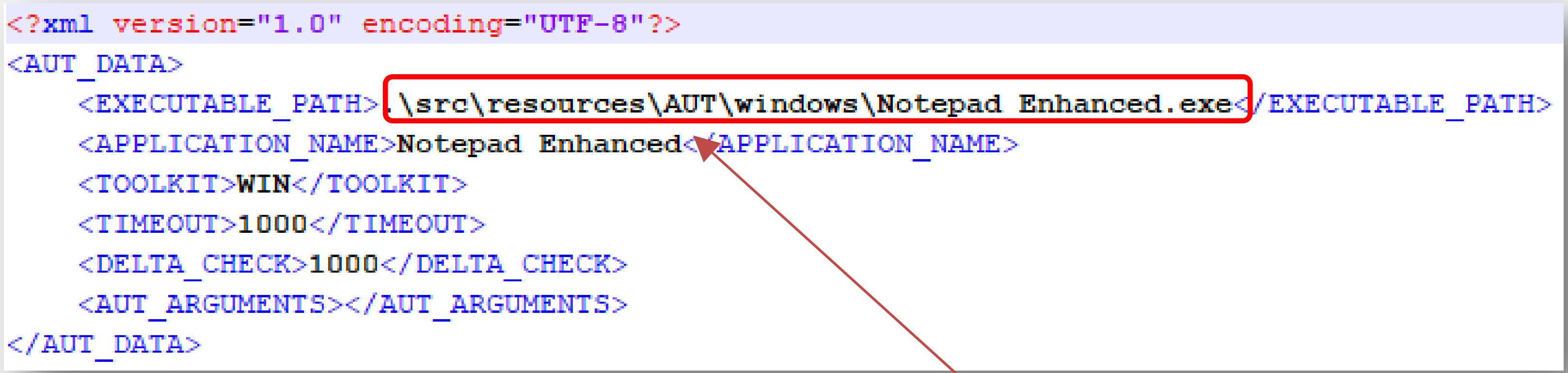


```
<?xml version="1.0" encoding="UTF-8"?>
<AUT_DATA>
    <SERVER_URL></SERVER_URL>

    <WORKING_DIR>./src/resources/AUT/java</WORKING_DIR> <!-- change this path to your working directory -->

    <APPLICATION_NAME>ButtonDemo</APPLICATION_NAME>

    <AUT_ARGUMENTS></AUT_ARGUMENTS>

    <VM_ARGUMENTS></VM_ARGUMENTS>

    <DESCRIPTION>
        Push-Button testing
    </DESCRIPTION>

    <JRE_PATH>${java.home}</JRE_PATH> <!-- change this path to your JRE home -->

    <MAIN_CLASS>com.sun.demo.ButtonDemo</MAIN_CLASS>

    <!-- on UNIX-like and MAC OS X systems change the path separator ';' to ':' -->
    <CLASSPATH>
        <LIB>
            <PATH>examples.jar</PATH> <!-- change this path to your Maveryx installation directory /demo -->
        </LIB>
        <!-- do not change the data below! (except for path separator on UNIX-like and MAC OS X systems) -->
    </CLASSPATH>
</AUT_DATA>
```

MAVERYX®
Test it simple!

MFC & .Net AUT Launch File

To execute a MFC or .NET Application-Under-Test it is necessary to create the related **AUT launch file**.



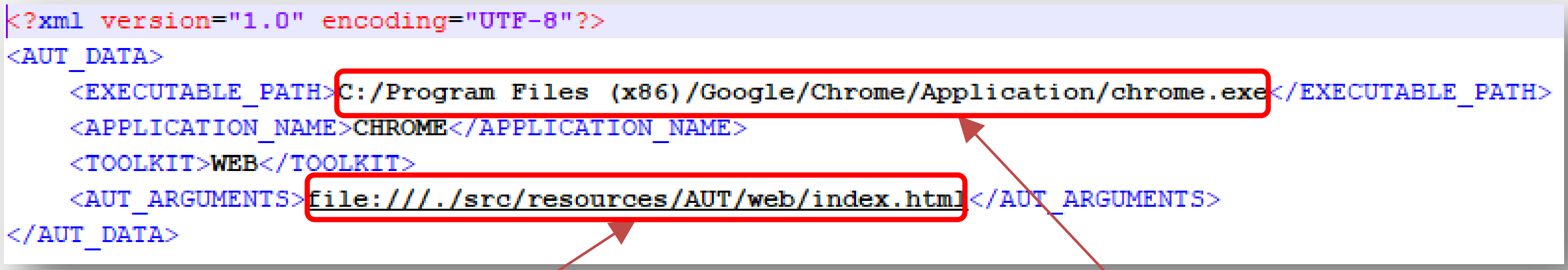
```
<?xml version="1.0" encoding="UTF-8"?>
<AUT_DATA>
    <EXECUTABLE_PATH>.\src\resources\AUT\windows\Notepad Enhanced.exe</EXECUTABLE_PATH>
    <APPLICATION_NAME>Notepad Enhanced</APPLICATION_NAME>
    <TOOLKIT>WIN</TOOLKIT>
    <TIMEOUT>1000</TIMEOUT>
    <DELTA_CHECK>1000</DELTA_CHECK>
    <AUT_ARGUMENTS></AUT_ARGUMENTS>
</AUT_DATA>
```

Set the absolute or relative path to your AUT executable file

Test it simple!

Web AUT Launch File

To execute a Web Application-Under-Test it is necessary to create the related **AUT launch file**.



```
<?xml version="1.0" encoding="UTF-8"?>
<AUT_DATA>
    <EXECUTABLE_PATH>C:/Program Files (x86)/Google/Chrome/Application/chrome.exe</EXECUTABLE_PATH>
    <APPLICATION_NAME>CHROME</APPLICATION_NAME>
    <TOOLKIT>WEB</TOOLKIT>
    <AUT_ARGUMENTS>file:///./src/resources/AUT/web/index.html</AUT_ARGUMENTS>
</AUT_DATA>
```

Set the URL of the AUT

Set the path of the browser **you** want to use for your tests

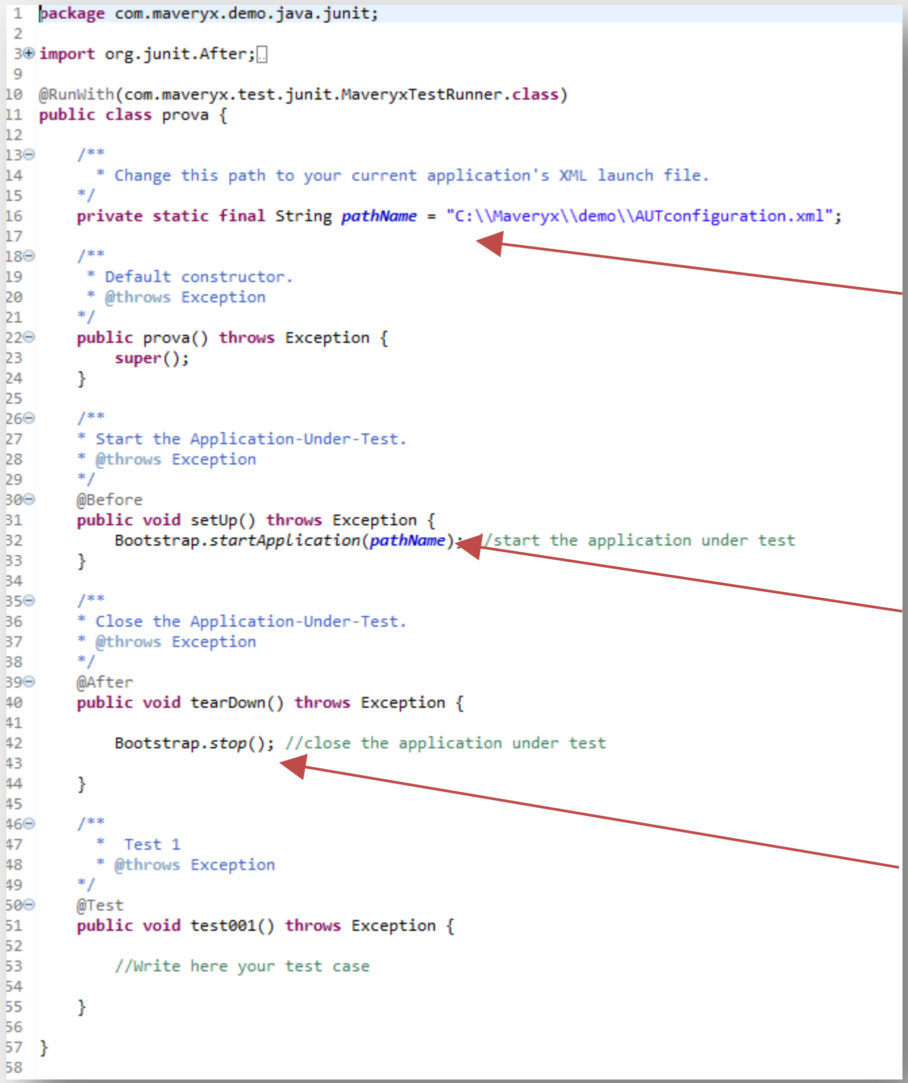
MAVERYX Test it simple!

Create and run your first test

1. Create a new Maveryx Test Project
2. Create a new Maveryx Test Class
3. Create the AUT Launch file
4. **Write the test case**
5. Run the test

Test it simple!

Test Script "stub"



```
package com.maveryx.demo.java.junit;

import org.junit.After;

@RunWith(com.maveryx.test.junit.MaveryxTestRunner.class)
public class prova {

    /**
     * Change this path to your current application's XML launch file.
     */
    private static final String pathName = "C:\\Maveryx\\demo\\AUTconfiguration.xml";

    /**
     * Default constructor.
     * @throws Exception
     */
    public prova() throws Exception {
        super();
    }

    /**
    * Start the Application-Under-Test.
    * @throws Exception
    */
    @Before
    public void setUp() throws Exception {
        Bootstrap.startApplication(pathName); //start the application under test
    }

    /**
    * Close the Application-Under-Test.
    * @throws Exception
    */
    @After
    public void tearDown() throws Exception {

        Bootstrap.stop(); //close the application under test

    }

    /**
     *  Test 1
     * @throws Exception
    */
    @Test
    public void test001() throws Exception {

        //Write here your test case

    }

}
```

Set the full path (*pathName*) to the **AUT *launch*** file.

e.g. **private final** String *pathName* = "C:/Maveryx/demo/AUT/PasswordDemo.xml";

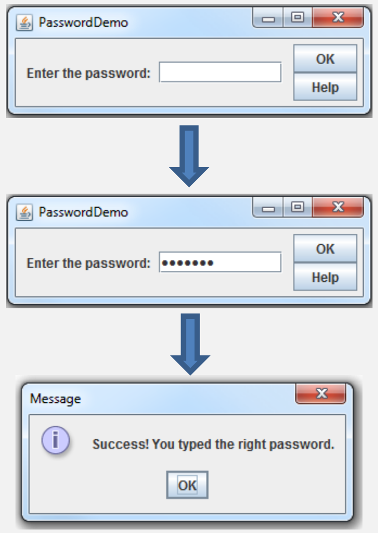
The static method ***startApplication*(*pathName*)** in class *Bootstrap* launches the AUT.

The static method ***stop*()** in class *Bootstrap* closes the AUT.

Test it simple!

Example

```
@Test
public void test001() throws Exception {

    GuiPasswordText t = new GuiPasswordText("Enter the password:");
    assertTrue(t.isEditable()); //check whether the text field is editable
    t.setText("bugaboo");       //enter the password

    GuiButton ok = new GuiButton("OK");
    assertTrue(ok.isEnabled());  //check whether the push button is enabled

    //click the 'OK' button in the main frame to confirm the entered password
    ok.click();

    GuiDialog dialog = new GuiDialog("Message"); //the info message dialog
    GuiLabel message = new GuiLabel("Success!", dialog);

    //check whether the message dialog contains the expected user message
    String expectedMessage = "Success! You typed the right password.";
    assertEquals(expectedMessage, message.getActualId());

    //close the message dialog
    dialog.close();

    //Alternatively,  close the message dialog by clicking the OK button
    //ok.setContainer(dialog);
    //ok.click();
}
```



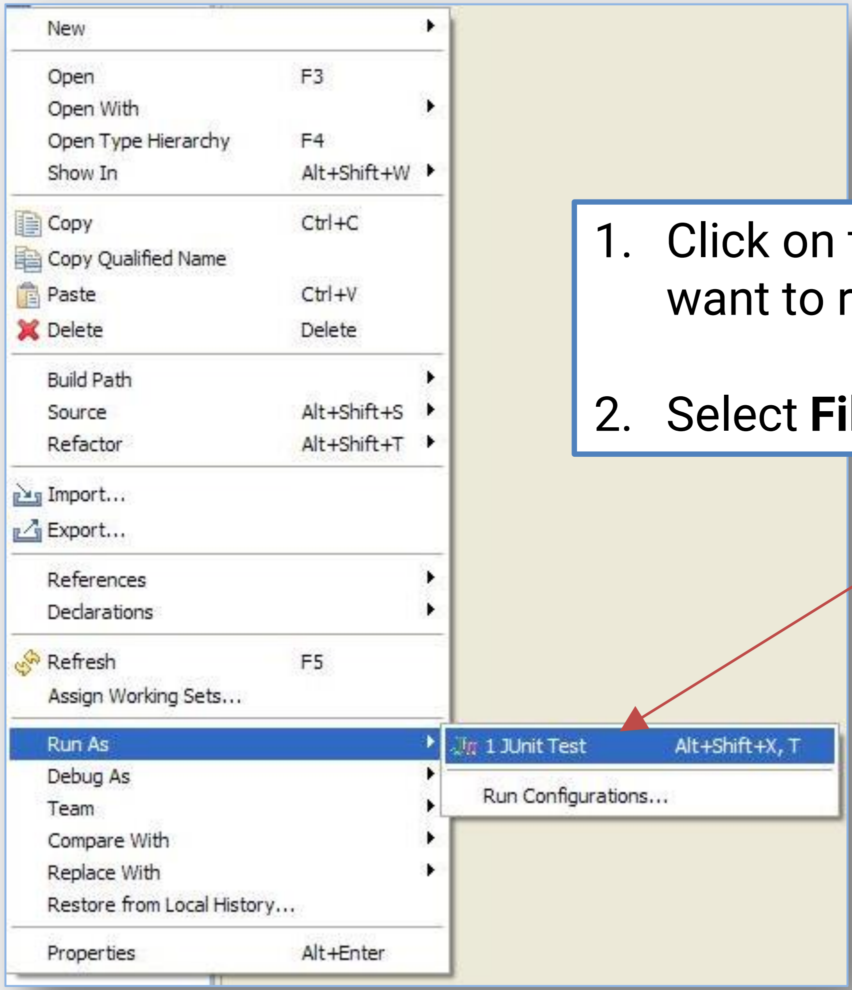
Success! You typed the right password.

Create and run your first test

1. Create a new Maveryx Test Project
2. Create a new Maveryx Test Class
3. Create the AUT Launch file
4. Write the test case
5. **Run the test**

Test it simple!

Run a Test Script



1. Click on the test class or package you want to run

2. Select **File** → **Run As** → **JUnit Test**

MAVERYX® Test it simple!

THANK YOU

sales@maveryx.com

info@maveryx.com

+39 333 30 72 597
+39 351 87 85 706

MAVERYX®
Test it simple!