# Telerik QA Academy 2014

# Exam #1 Preparation

## Test

Test part I URL: <https://testmoz.com/49193>

Test part II URL: <https://testmoz.com/162181>

Test user: username from student system (<http://telerikacademy.com/>)

Test pass: qaacademy2013

## Bug Reports

You are given a few videos showing bugs in various functionalities. Your task is to watch them, navigate to the URL shown in the video and prepare bug reports.

* Identify the bug
* Reproduce it
* Write bug report

### ASP AJAX control bug

Watch the [Video](http://screencast.com/t/NZ0frnbj).

Go to <http://www.telerik.com/products/aspnet-ajax.aspx> and try to reproduce the bug from it.

Prepare a **bug report** containing all **necessary information** for logging the report in a bug tracking system. Try to **isolate** the minimum steps required for reproducing the bug, removing any steps that are not really influencing the behavior.

The **reports** should be presented in a **table in word** document.

## Automation Tests

### Magento Incorrect Login Test

Navigate to the following URL: <http://demo.nostresscommerce.cz/>

1. Click the **My account** link. Try to login with incorrect e-mail and password. Verify that the login is rejected.
2. **Make another test**, but this time try to login without entering values in the e-mail and password fields. Verify that the user is prompted to fill in the empty fields.

### Random Generator Test

Navigate to the following URL: <http://www.random.org/integers/>. Make a test that generates a random number between 1 and 2. If the number generated is 1 – the test navigates to the Frequently Asked Questions (FAQ) page (you’ll find a link in the **Learn more** menu), verifies that the correct page is loaded and navigates to the home page of the site.  
If the number generated is 2 – the test should navigate directly to the home page.  
Verify that the home page is loaded.

### BugTracker Test

Make functional web tests using any of the three tools you prefer – Visual Studio, Selenium or Telerik Test Studio (as long as the tool provides the functionality you would need for the test). The tests should navigate to the following URL: <http://ifdefined.com/btnet/bugs.aspx>

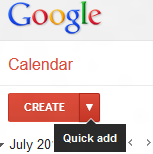
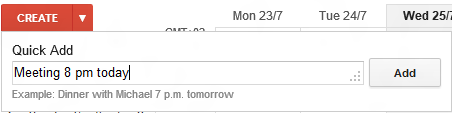
1. Choose a number, so that there could be no existing bug with that ID.   
   Try searching for a bug with that number and verify that the site responds correctly to that search.
2. Go to the page for creating a new bug with the “**add new bug**” link.  
   The form for creating a new bug has a drop-down list named **Project-specific** that is visible only if the “**HasCustomFieldsProject**” option is chosen in the **Project** drop-down list. Make a test that verifies that this functionality is applied correctly.

### Google Calendar

Make functional web tests using Selenium. The tests should navigate to the following URL: <http://www.google.com/> .

Prerequisite: Create an account in google which can be used for tests.

1. Log into a valid google account and verify that the log in was successful and the correct user is logged in.
2. While you are logged and your calendar is shown create an event using the quick add option of the create button. In the text field write the following text “Meeting 8 pm today”. Verify that there is an event today that has such title, and then delete the event.



1. Take the test above and instead of creating the event with title Meeting, create it with unique heading every time using JavaScript.(Tip: You can generate your unique name in separate variable)

**Web driver**

* Convert the tests to C# and test if the backend e-mail validation works properly. Think of an appropriate way to replace your JavaScript functionality (if any). Use different user profiles and run the test with different browsers.

### DataFilterControl

Make functional web tests using Test Studio. The tests should navigate to the following URL: <http://demos.telerik.com/silverlight/> .

* Navigate to all controls, chose DataFilter and verify you are on the correct page.
* Add a filter to the grid and verify it is applied correctly.
  + Apply filter “Quantity is equal to 9”. Each row, shown in the table should have value 9 in column Quantity. Use coded step for that validation. Find parent element of type RadGridView. With that element you can reach rows and cells. In order to have proper Assertion extract the values from the other two filter fields and think how to use them. The assertions should look like that Assert.AreEqual(extectedValue, actualValue). If you need to print something to the Log you can use Log.Writeline() command. Do not forget to use the Add to Element Explorer option.
* Add a data source to the test above. Use the data below. Locate your elements very carefully.

|  |  |  |
| --- | --- | --- |
| **Field** | **Expression** | **Value** |
| **Quantity** | Is equal to | 9 |
| **Name** | Starts with | P |
| **UnitPrice** | Is less than | 50 |