

Mozmill



An automated test framework for
Firefox, Thunderbird, and other
Gecko-based products

Anthony Hughes
Seneca College Student, Mozilla QA Contributor, Asst. Test Day Lead

anthony.s.hughes@gmail.com

IRC: ashughes

<http://ashughes.com>

What is Mozmill?

- a test tool/framework
- an extension
- an extensive API
- unit tests
- user interactions
- helps you write, run, and debug tests

Installing Mozmill

- <https://addons.mozilla.org/de/firefox/addon/9018>
- <http://github.com/mikeal/mozmill/downloads>
- Before installing, use a new profile
 - `firefox.exe -P -no-remote`
- For command line instructions:
[https://developer.mozilla.org/en/Mozmill#The Command Line Client](https://developer.mozilla.org/en/Mozmill#The_Command_Line_Client)

Getting the Tests

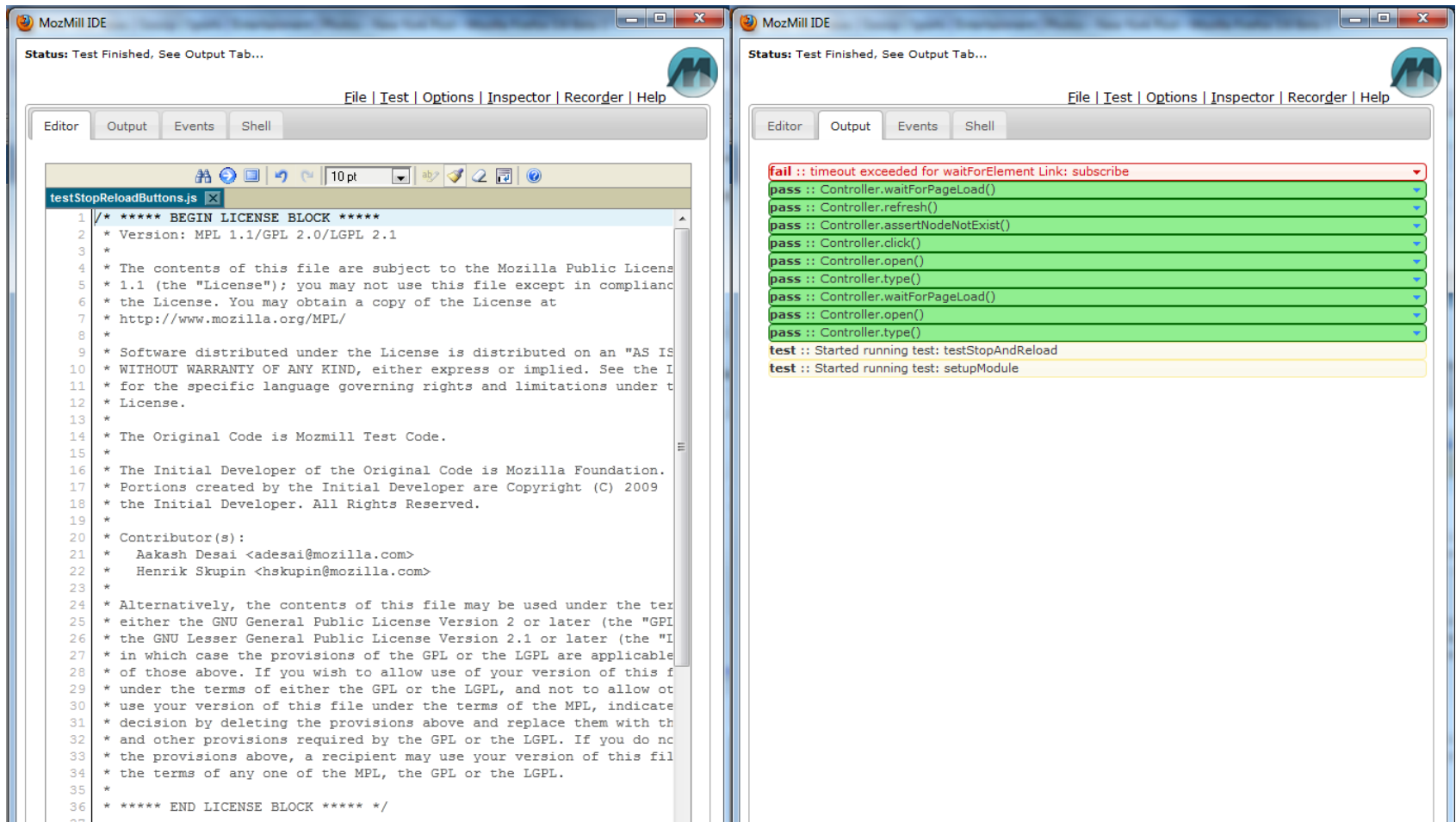
- Tests and shared modules are stored in a repository:

<http://hg.mozilla.org/qa/mozmill-tests>

- Let's go through this guide:

<http://quality.mozilla.org/documents-home/code-docs/mozmill-test-creation/the-test-repository>

Overview of the IDE



Let's look at the IDE in more detail...

Components of a Mozmill Test

- The Controller Object

[https://developer.mozilla.org/en/Mozmill Tests/Mozmill Controller Object](https://developer.mozilla.org/en/Mozmill_Tests/Mozmill_Controller_Object)

- The Elementslib Object

[https://developer.mozilla.org/en/Mozmill Tests/Mozmill Elements Library Object](https://developer.mozilla.org/en/Mozmill_Tests/Mozmill_Elements_Library_Object)

- The Mozmill Object

[https://developer.mozilla.org/en/Mozmill Tests/Mozmill Base Object Interfaces](https://developer.mozilla.org/en/Mozmill_Tests/Mozmill_Base_Object_Interfaces)

Components of a Mozmill Test

- License Block
- Shared Modules
 - RELATIVE_ROOT
 - MODULE_REQUIRES
- Setup Module
- Teardown Module
- Test method
- Dialog Callback Handlers

Writing Tests

Let's walk through an example test:

<http://hg.mozilla.org/qa/mozmill-tests/file/default/firefox/testPopups/testPopupsBlocked.js>

The Review Process

- The review process is fairly straight forward:

1. File a bug
2. Create a working test
3. Create a patch

```
hg add path_to_testscript/file.js
hg diff > patch_name
```

4. Attach it to the bug
5. Wait for review
6. If *r-* make any suggested revisions
 1. Repeat steps 3 to 5 until *r+*
 2. Someone will check in the patch for you

- Let's look at an example:

https://bugzilla.mozilla.org/show_bug.cgi?id=511314

Your First Test

- Let us now create your first test
- This test should test the following functionality:
 - Type “Hello World” into the location bar
 - Click the GO button
 - Wait for the page to load
 - Verify that the resulting page is Wikipedia’s “Hello World Program” article

Helpful Links

- Overall guide to install Mozmill and write tests:
<http://quality.mozilla.org/documents-home/code-docs/mozmill-test-creation>
- Components of a Mozmill test:
https://developer.mozilla.org/en/Mozmill_Tests
- Overview of the Mozmill IDE:
<http://adamchristian.com/archives/185>
- Creating a patch:
<http://quality.mozilla.org/documents-home/code-docs/mozmill-test-creation/review-process>
- A better Inspector:
<https://addons.mozilla.org/en-US/firefox/addon/6622>

Want to Help Out More?

- Take part in our two day test day:

Today through Tomorrow on
irc.mozilla.org #testday

<http://quality.mozilla.org/events/2009/oct/29/testday-test-scripting-mozmill-day-1>

<http://quality.mozilla.org/events/2009/oct/30/testday-testscripting-mozmill>

Questions

Are there any Questions?