WebExtension Experiments Documentation

Release 0.1

Add-ons team

Sep 25, 2018

Contents

WebExtensions are a cross-browser system for developing browser add-ons.

WebExtensions Experiments allow developers to write experimental WebExtensions APIs for Firefox. They can be used to prototype APIs for landing in Firefox, or for use on Nightly or Developer Edition.

Information about policies for WebExtensions APIs and how to request a new API are available on the Mozilla wiki.

Contents:

CHAPTER 1

Basics

WebExtensions Experiments provide a way for developers to tinker with new APIs for WebExtensions. They allow new WebExtensions APIs to be implemented in an extension. They can be used to prototype APIs for landing in Firefox, or for use on Nightly or Developer Edition.

Note: If you simply want to request a WebExtensions API, please see

this page on the Firefox wiki.

Note: If you'd like to land a WebExtensions API straight into Firefox, are familiar with building mozilla-central, working with Bugzilla, the try server, then please file a bug and follow the usual Firefox development process.

1.1 Overview

An *experiment* refers to all the code that implements some new experimental WebExtension API. An experiment may be placed in its own `.xpi` file, or it may be bundled with an extension that uses it (i.e., have all the experiment code contained in the same `.xpi` file as the extension that uses it).

Note: When experiments were originally created they used a format based on `install.rdf` but this format is deprecated and should not be used.

Experiments allow you to:

- Test and experiment with an API without having to build Firefox at all.
- Write and distribute the API to a set of users without them having to build Firefox.
- Then commit to (or get help committing to) mozilla-central.

Note: We will not add all experiments to Firefox. The goal is to judge each Experiment on its own merit and value.

Some key points:

- The API is implemented in the experiment.
- The API is available in the browser namespace - not in chrome.
- Breaking changes could occur in the experiment, but are discouraged.

1.1.1 How do they work?

For general documentation about designing and developing WebExtension APIs, see the documentation at firefox-source-docs.mozilla.org.

If you prefer working examples, here are a few WebExtensions that use bundled experiments:

- Mike Conley's ohnoreflow
- Rob Helmer's crashme

1.1.2 Non-bundled Experiments

The documentation above explains how to create an experimental API that is bundled with a WebExtension. It is also possible to create a WebExtension that depends on an experimental API contained in a different `.xpi` file, though we are considering removing this capability in a future release. In the mean time, to create an extension that uses an experimental API from a different extension, you must:

- Include an extension ID in the manifest of the extension that contains the implementation of the API. The ID *must* be of the form `name@experiments.addons.mozilla.org` where `name` is a string that identifies the experiment.
- Include the string `"experiments.name"` in the permissions property in the manifest of any extension that will use the API. The string `name` must be replaced by the name in the ID of the extension that provides the API.

In this case, a user must manually install the experiment when they install the WebExtension that uses the experimental API. The WebExtension will be automatically disabled unless/until the experiment it depends on is installed and enabled.

1.1.3 Where do they work?

Currently experiments can only be loaded in Firefox Nightly and Firefox Developer Edition.

Please see the policy page.

CHAPTER 2

Uplifting

Once you've got an experiment the hope is that it gets merged into Firefox and mozilla-central. But before that happens there's a few things that should happen.

2.1 Adding to github

Note: This is optional.

2.2 Tell people

The first step is to file an issue against this repository. In that issue please outline:

- where your experiment is located
- a quick overview of what it does
- any bugzilla bugs that it might address
- if you'd like to move your repository over to this organisation and we can create a repository for you

We'll then include these in our bi-weekly community triage meeting.

There's a few emails and IRC channels that would also like to know about your experiment.

2.3 Updating this documentation

It would be great to add it to this documentation too. If you would like to, please send a pull request to this repository.

2.4 Landing in Firefox

Note: This is optional.

For it to land in mozilla-central a few things are going to have to happen.

- It should follow the code quality guidelines.
- There should be mochi and xpcshell tests at 100% coverage of the API.
- There should be some documentation on the API.
- There should be a review (even if its just cursory) by the following: the security team, the privacy team, a code review by the engineering team and the UX team (where relevant).
- If the API requires a permission, ensure to land the corresponding permissions string.

For more information on what is likely to be accepted, please check out the New API Guidelines.

The best way to do this is by filing a bug in Bugzilla and we can start to work through the steps. File a bug under Toolkit > WebExtensions: Experiments and that process can be started.

2.5 What if it doesn't land in Firefox?

It can still be used by Nightly and Developer Edition.

CHAPTER 3

FAQ

WebExtensions Experiements have many similarities to traditional add-ons, therefore a lot of general documentation is already available on developer.mozilla.org. The following questions and answers are specific to WebExtensions experiments and should help you with common issues.

3.1 Can I include a chrome.manifest in an experiment?

The experiments do not support the 'chrome.manifest', and it will be ignored if included in the experiments add-on source directory. If you need an URI which points to assets packaged within the experiment (e.g. an image, an additional .jsm file, a frame script, etc.), you can register a path with the `resource:` protocol handler.

3.2 Why is my experiment `undefined` on Beta and Release?

On Nightly, Dev Edition or unbranded builds, you can install unsigned experiments if you flip two preferences: `xpinstall.signatures.required` and `extensions.legacy.enabled`. However unsigned experiments cannot be installed on Beta or Release regardless of any preference settings.

If you have any more questions you can't find an answer to please get in touch.

This documentation lives on github, pull requests and issues are welcome.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search