# Managing to find browser bugs

Paul Theriault, platsec 2016

## Workshop Materials

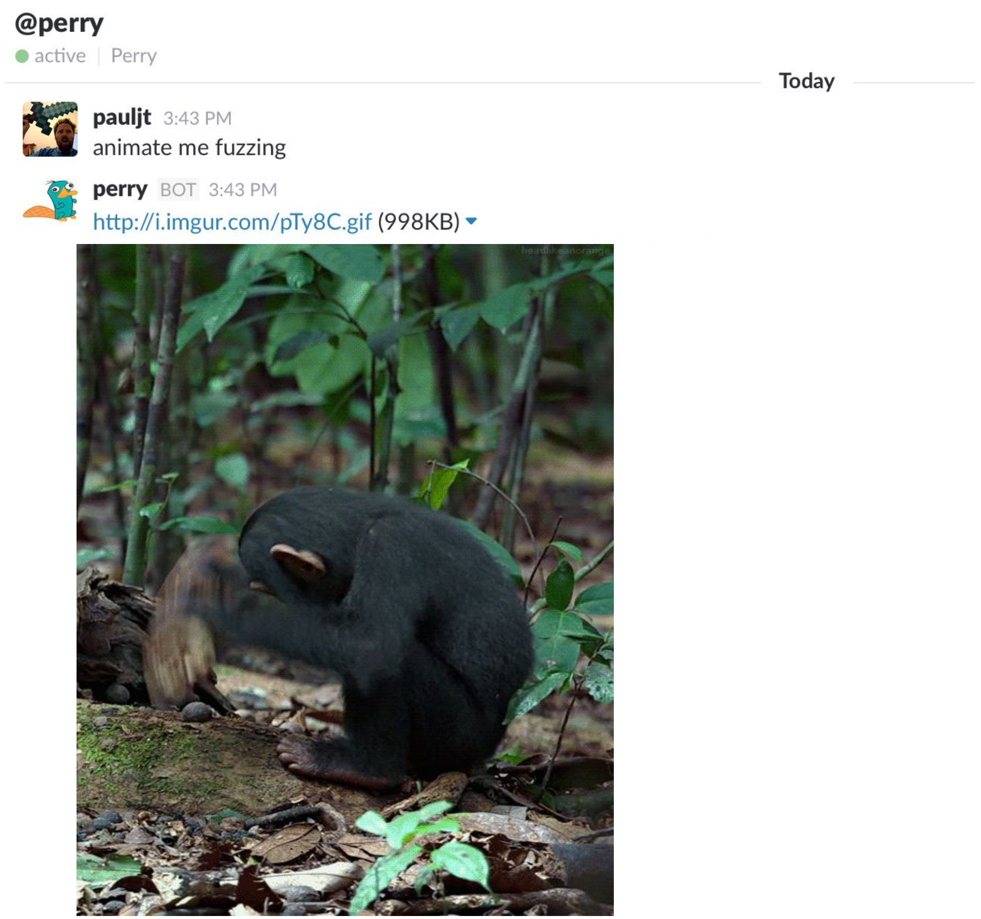
1. Python (3, preferably)
2. Dharma & Framboise
3. ~~Firefox Asan/Debug build~~

## TLDR

- Let’s finds some browser* bugs
- Because reasons
- Learn a little about browser internals
- Learn to use two fuzzing tools to test Web APIs
  - Dharma
  - Framboise
- Today we focus on test case generation, everything else left as homework
- Introduction to fuzzing, if you are an expert, please help others. Also consider using this hour to join our team: here

  You could get paid to ...

## Do fuzzing?



@perry
active | Perry

Today

**paulјt** 3:43 PM
animate me fuzzing

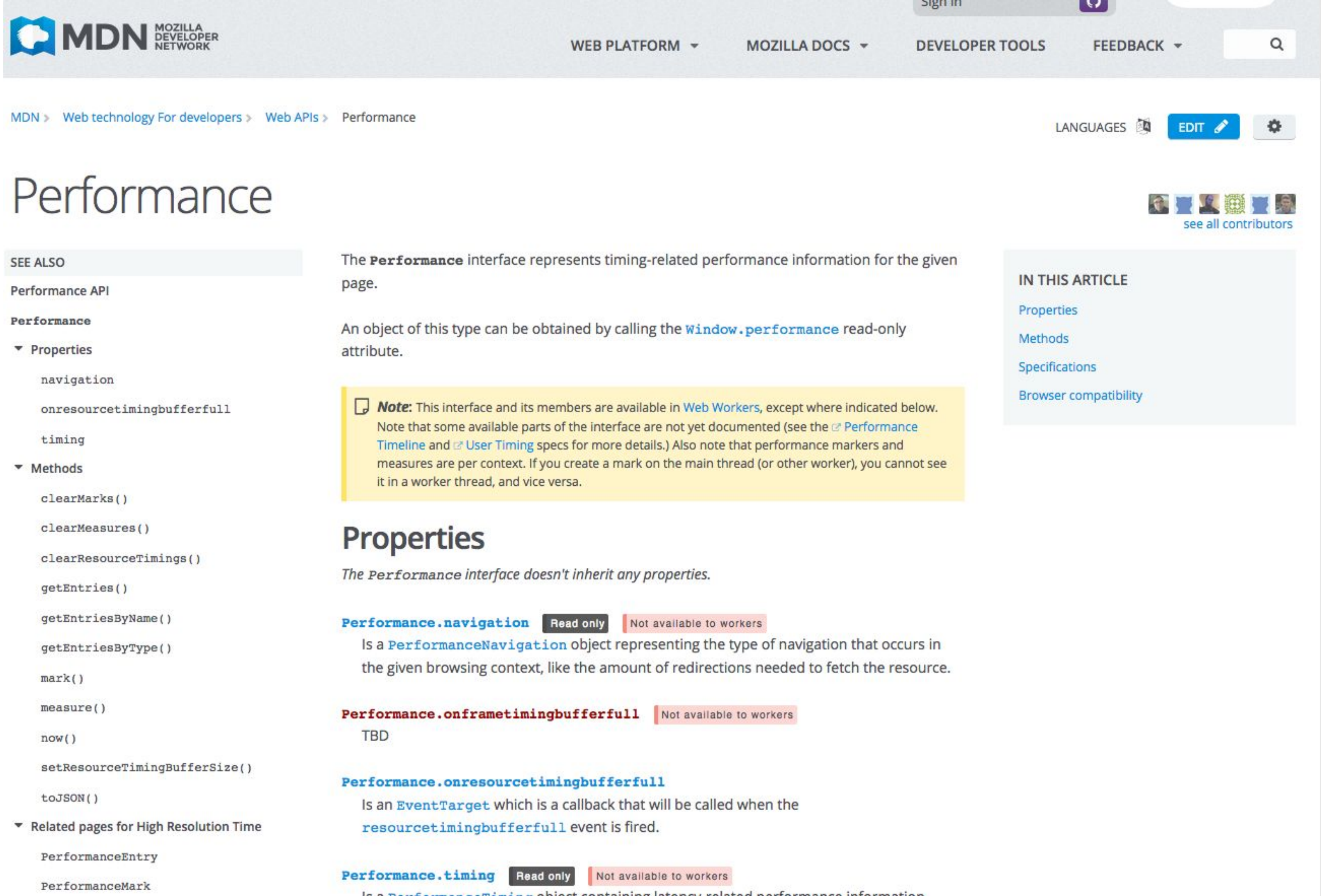
**perry** BOT 3:43 PM
http://i.imgur.com/pTy8C.gif (998KB)

## Browsers - how do they work?

- Who even knows? I'm just a manager.
- Let's focus on one part: Web APIs
- Let's get some help: thanks Posidron!
- E.g. Web Timing API
- FlyWeb API (time permitting)

## Web Timing API

```
>> performance.now()

2122137.495  //milliseconds from

>> performance.measure("foo","connectStart","connectEnd")

>> performance.getEntriesByName("foo")[0]

PerformanceMeasure { name: "connect", entryType: "measure",
startTime: 24, duration: 0 }
```



MDN > Web technology For developers > Web APIs > Performance

# Performance

see all contributors

SEE ALSO

Performance API

**Performance**

- Properties
  - `navigation`
  - `onresourcetimingbufferfull`
  - `timing`
- Methods
  - `clearMarks()`
  - `clearMeasures()`
  - `clearResourceTimings()`
  - `getEntries()`
  - `getEntriesByName()`
  - `getEntriesByType()`
  - `mark()`
  - `measure()`
  - `now()`
  - `setResourceTimingBufferSize()`
  - `toJSON()`
- Related pages for High Resolution Time
  - `PerformanceEntry`
  - `PerformanceMark`

The **`Performance`** interface represents timing-related performance information for the given page.

An object of this type can be obtained by calling the `Window.performance` read-only attribute.

***Note***: This interface and its members are available in Web Workers, except where indicated below. Note that some available parts of the interface are not yet documented (see the Performance Timeline and User Timing specs for more details.) Also note that performance markers and measures are per context. If you create a mark on the main thread (or other worker), you cannot see it in a worker thread, and vice versa.

## Properties

*The `Performance` interface doesn't inherit any properties.*

**`Performance.navigation`** Read only Not available to workers
Is a `PerformanceNavigation` object representing the type of navigation that occurs in the given browsing context, like the amount of redirections needed to fetch the resource.

**`Performance.onframetimingbufferfull`** Not available to workers
TBD

**`Performance.onresourcetimingbufferfull`**
Is an `EventTarget` which is a callback that will be called when the `resourcetimingbufferfull` event is fired.

**`Performance.timing`** Read only Not available to workers
Is a `PerformanceTiming` object containing latency-related performance information

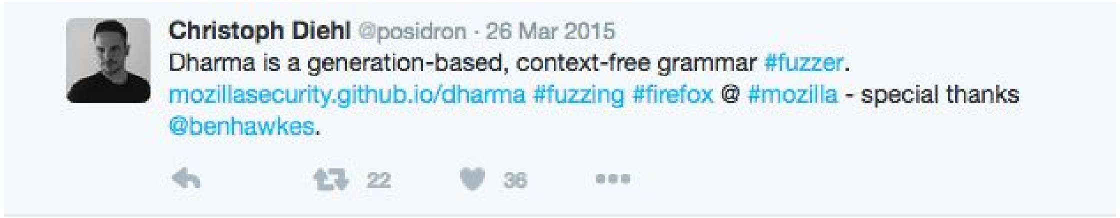
IN THIS ARTICLE

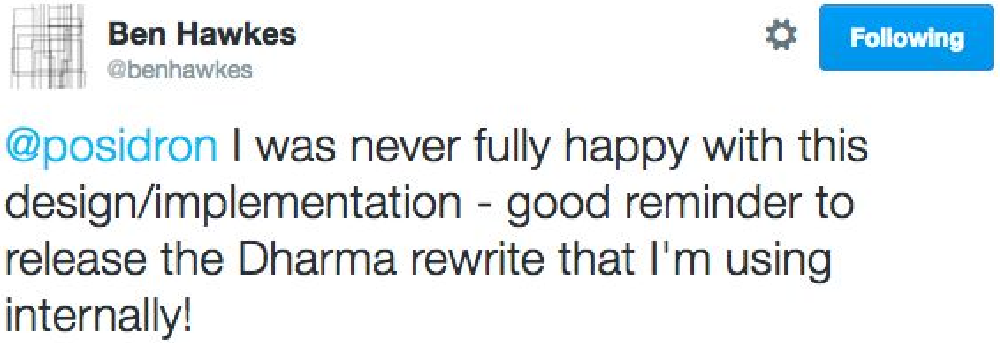
- Properties
- Methods
- Specifications
- Browser compatibility

## WebIDL: the source of truth

```
//http://searchfox.org/mozilla-central/source/dom/webidl/Performance.webidl

typedef double DOMHighResTimeStamp;
typedef sequence <PerformanceEntry> PerformanceEntryList;

[Exposed=(Window,Worker)]
interface Performance {
  [DependsOn=DeviceState, Affects=Nothing]
  DOMHighResTimeStamp now();

};
```

## Dharma


Christoph Diehl @posidron · 26 Mar 2015
Dharma is a generation-based, context-free grammar #fuzzer.
mozillasecurity.github.io/dharma #fuzzing #firefox @ #mozilla - special thanks @benhawkes.


Ben Hawkes
@benhawkes

@posidron I was never fully happy with this design/implementation - good reminder to release the Dharma rewrite that I'm using internally!

https://github.com/MozillaSecurity/dharma

## Dharma Basics

```
%const% VARIANCE_MAX := 50
%const% VARIANCE_TEMPLATE := "try { %s } catch (e) { }"
%const% MAX_REPEAT_POWER := 5

%section% := value
methods :=
	performance.now()
%section% := variable
%section% := variance
main :=
	+methods+

%%% COMMENT LIKE THIS. USE TABS NOT SPACES!
```

## Running Dharma

```
dharma$ ./dharma.py -grammars grammars/template.dg
[Dharma] 2016-09-22 22:25:01,001 INFO: Machine random seed:
-232680778323537235
[Dharma] 2016-09-22 22:25:01,002 DEBUG: Using configuration from:
/Users/ptheriault/git/fuzz/dharma/dharma/settings.py
[Dharma] 2016-09-22 22:25:01,004 DEBUG: Processing grammar content of
grammars/common.dg
[Dharma] 2016-09-22 22:25:01,007 DEBUG: Processing grammar content of
grammars/performance/example.dg
try { performance.now() } catch (e) { }
try { performance.now() } catch (e) { }
try { performance.now() } catch (e) { }
try { performance.now() } catch (e) { }
```

## Activity 1: Fuzz performance.mark(...)

Usage: performance.mark("some_string")
Dharma Syntax:

```
alert:=
    alert("+someString+")

someStrings:=
    foo
    bar
    Baz
```

What does to `+common:text+` do?

## Activity 2: valid input for performance.getEntriesByName

USAGE:

```
performance.getEntriesByName(“random”) //returns nothing

fetch(“http://some.url”)
performance.getEntriesByName(“http://some.url”)  //returns one entry
```

Dharma Syntax ( Hint: "+url:url+" gives you a random URL)

```
%section% := value
alertTime:=
    alert(!time!)

%section% := variable
time:=
    @time@="Time was " + performance.now()
```

## Activity 3: Extending the template

Performance API is available in workers

Use an alternative template:

grammars/var/templates/html5/default.html

## Want more? Do the rest!

Finish the rest of the API ! Or see the spoilers folder for example.

## Framboise

- Fuzzer for in-depth testing of WebAPIs
- https://github.com/MozillaSecurity/framboise
- Is a fuzzer in its own right (see modules directory)
- Can also be used to run test cases for other fuzzers

## Setup

Change the following in your settings file:

- FilesystemLogger: &FilesystemLogger
- Set Default targetapp

## Running Framboise

```
framboise$ ./framboise.py -debug -fuzzer 1:WebAudio

framboise$ ./framboise.py -debug -fuzzer 1:WebAudio
-with-set-timeout -with-set-interval -with-events

framboise$ ./framboise.py -debug -fuzzer 1:WebAudio
-with-set-interval -with-set-timeout -with-events
-max-commands 200
```

## Feeding Dharma test cases to Framboise

Run Dharma as a server

```
dharma$ python dharma.py -grammars grammars/canvas2d.dg -server -template
grammars/var/templates/html5/default.html
```

Launch the dharma launch file with framboise (instead of just opening in browser)

```
framboise$ ./framboise.py -testcase
/Users/ptheriault/git/fuzz/dharma/dharma/grammars/var/index.html -debug
```

## Pro-tips

- Performance
- Automation-friendly
- Defect Oracles
  - ASAN
  - Assertions (debug builds)
  - Differential testing
  - Well-defined behavior
- Test Samples