Mozilla, Firefox, etc

Mozilla | Octobre 2021

Bonjour !

Je suis Sylvestre Ledru

Je parle de Firefox

Twitter @SylvestreLedru

Who Am I? At Mozilla

➔ Mozilla for ~8 years – Director of engineering

➔ Org - OS Integration & Engineering Effectiveness - 40+ persons

Two missions:

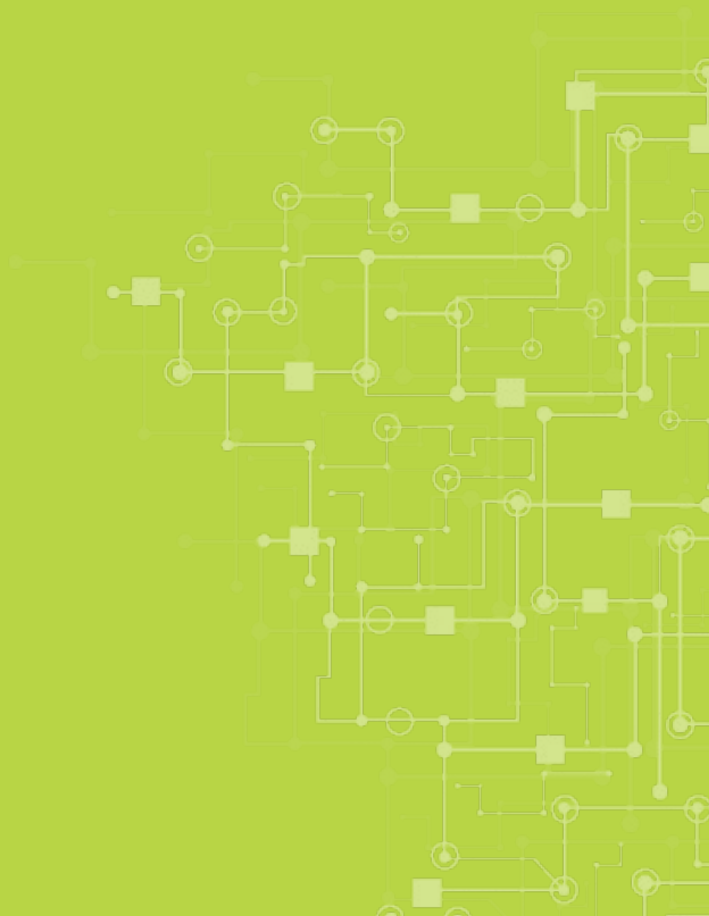
We make it easy to develop and release Mozilla software securely at scale

We hack on Firefox to build solid foundations on top of which differentiating features can be built.

➔ Head of the Mozilla French branch
Spokesperson

About:

moz://a



About:Mozilla

- ➔ Adventure started by Netscape (~1994)
- ➔ Failed against Microsoft (Internet Explorer)
- ➔ Decided to open the sources of Netscape (20 years ago)

- ➔ Documentary about this period:
  Code rush:
  https://www.youtube.com/watch?v=u404SLJj7ig

About:Mozilla

- ➔ Brought by AOL in 1998
- ➔ AOL gave some money to the Mozilla Foundation in 2003
- ➔ Mozilla was nothing
- ➔ Massive refactorings
- ➔ Firefox 1.0 released 18 years ago
  A game changer (popup blocker, tab, etc)

mozilla
FOUNDATION

About: Firefox

moz://a

About:Firefox

- Web browser with ~500 million users
  - Only (major) browser developed by a non-profit
  - The last major not on blink/webkit

- Support 4 operating systems:
  - Microsoft Windows XP => 1 (32 & 64 bit)
  - GNU/Linux (32/64)
  - Mac OS X (64 + arm)
  - Android (various arm + intel)

- iOS – not based on Gecko

About:Firefox

- We release every 4 to 6 weeks
- 11 major releases - 2021
  - About 61 minor releases since Jan 1st

About:Firefox:Releases

➔ Other versions:

- In parallel, 2 other feedback branches :
- Nightly - updated daily with recent code changes
- Beta – 2 per week Desktop – 1 for Mobile
- Devedition

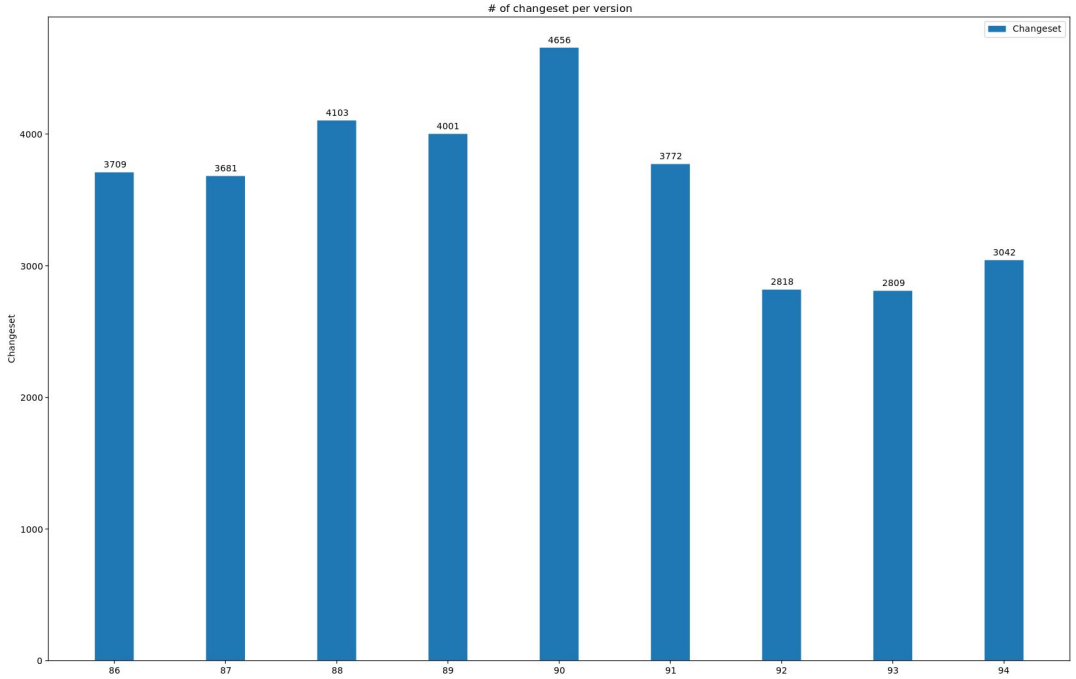
About:Firefox:code

- Gecko is the based of Firefox
  - And ... Thunderbird, Seamonkey and Firefox OS (rip)

- ... has had 772 990 commits made by +8 400 contributors representing ~23.9M lines of code

- About 400 developers / month
- About 3700 commits per month (about 180 commits per working day)

About:Firefox:Code



Patches landed per nightly cycle

About:Firefox:code

Language	Code Lines	Comment Lines	Comment Ratio	Blank Lines	Total Lines	Total Percentage
C++	6,282,197	1,400,026	18.2%	1,224,755	8,906,978	27.2%
JavaScript	6,034,290	1,736,360	22.3%	1,164,450	8,935,100	27.3%
HTML	3,387,763	99,180	2.8%	360,787	3,847,730	11.7%
C	3,154,230	824,939	20.7%	469,231	4,448,400	13.6%
Rust	2,393,849	449,824	15.8%	245,211	3,088,884	9.4%
Python	898,780	254,735	22.1%	241,219	1,394,734	4.3%
XML	598,883	9,712	1.6%	20,720	629,315	1.9%
Assembly	332,696	28,042	7.8%	36,657	397,395	1.2%
CSS	224,778	13,486	5.7%	32,120	270,384	0.8%
Java	158,485	61,306	27.9%	24,334	244,125	0.7%
Autoconf	99,172	1,875	1.9%	13,499	114,546	0.3%
shell script	82,312	16,742	16.9%	13,106	112,160	0.3%
Objective-C	55,895	9,476	14.5%	12,233	77,604	0.2%
Make	43,021	13,030	23.2%	11,367	67,418	0.2%

About:Firefox:Continuous Integration

- ➔ We run a few tests... with a few different platforms and options

- ➔ 1 506 hours for the average full CI run
- ➔ A full run is 5000 tasks

- ➔ From 100 to 300 machine years/month

Why quality?

moz://a

Why quality?

- ➔ Besides the obvious answers...
- ➔ Browsers, just like OS, are the most common applications but even harder to secure

Chinese Hackers Using Firefox Extension to Spy On Tibetan Organizations

February 25, 2021 Ravie Lakshmanan

Le FBI a démasqué des pédophiles grâce à une faille 'non publique'

Une vaste action à l'encontre des visiteurs d'un site web de pédopornographie a été menée en exploitant une faille non encore connue dans un navigateur. De cette manière, le FBI a réussi à pirater avec succès des milliers d'utilisateurs.

Russian hackers modify Chrome and Firefox to track secure web traffic

The perpetrators may have Russian government support.

ACTIVE EXPLOIT Will send your data to Ukraine if run

6 years ago **contact@fukusa.nl**

21.60 KB, text/plain

How to ship quality?

moz://a

Quality?

- Three types of QA:
  - 1) Pre release channel (nightly, beta, etc)
  - 2) Catch issues during development phase
  - 3) Automated tests & testsuites when the code land

1 - Pre release testing

moz://a

Pre release testing

- The Web is a crazy platform
- All possible combinations of
  - HTML
  - CSS
  - Javascript (+ asm.js & WebAssembly)
  - Media format (Images, Audio, Video, etc)
  - Network
  - Server
  - OS
  - ...

Pre release testing ...

➔ And we cannot trust user input.

Pre release testing

- We rely a lot on users on prerelease channel
  - Experiments (A/B testing) on pre-release channels
- Nightly - two nightlies per day
  - Hundred thousand of users
- Beta - 2 per week Desktop – 1 for Mobile
  - Millions of users

Manual testing

- Teams which test manually the new features

- Three colors
  - Green - Let’s ship it
  - Orange - We have to fix a few bugs
  - Red - Won’t be able to ship in this cycle

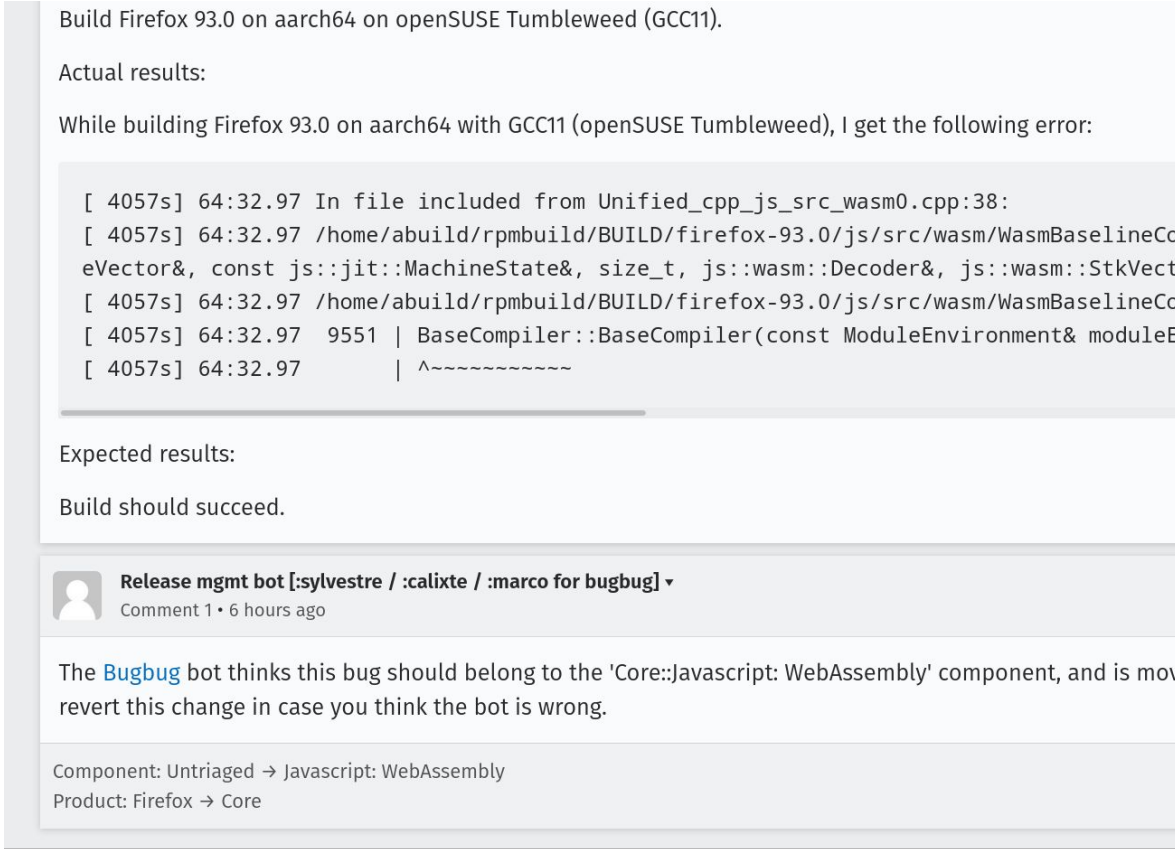
Digression about bug management

- About 5500 bugs reported per month on Firefox
  Some are tasks, defects or enhancements

- Because of the scale, we need help… from Machine
  Leveraging machine learning
  https://github.com/mozilla/bugbug
  Can run on github projects

Digression about bug management

Moving bug from untriage to the right component
Example:



Build Firefox 93.0 on aarch64 on openSUSE Tumbleweed (GCC11).

Actual results:

While building Firefox 93.0 on aarch64 with GCC11 (openSUSE Tumbleweed), I get the following error:

```
[ 4057s] 64:32.97 In file included from Unified_cpp_js_src_wasm0.cpp:38:
[ 4057s] 64:32.97 /home/abuild/rpmbuild/BUILD/firefox-93.0/js/src/wasm/WasmBaselineCo
eVector&, const js::jit::MachineState&, size_t, js::wasm::Decoder&, js::wasm::StkVect
[ 4057s] 64:32.97 /home/abuild/rpmbuild/BUILD/firefox-93.0/js/src/wasm/WasmBaselineCo
[ 4057s] 64:32.97  9551 | BaseCompiler::BaseCompiler(const ModuleEnvironment& moduleE
[ 4057s] 64:32.97       | ^~~~~~~~~~~~
```

Expected results:

Build should succeed.

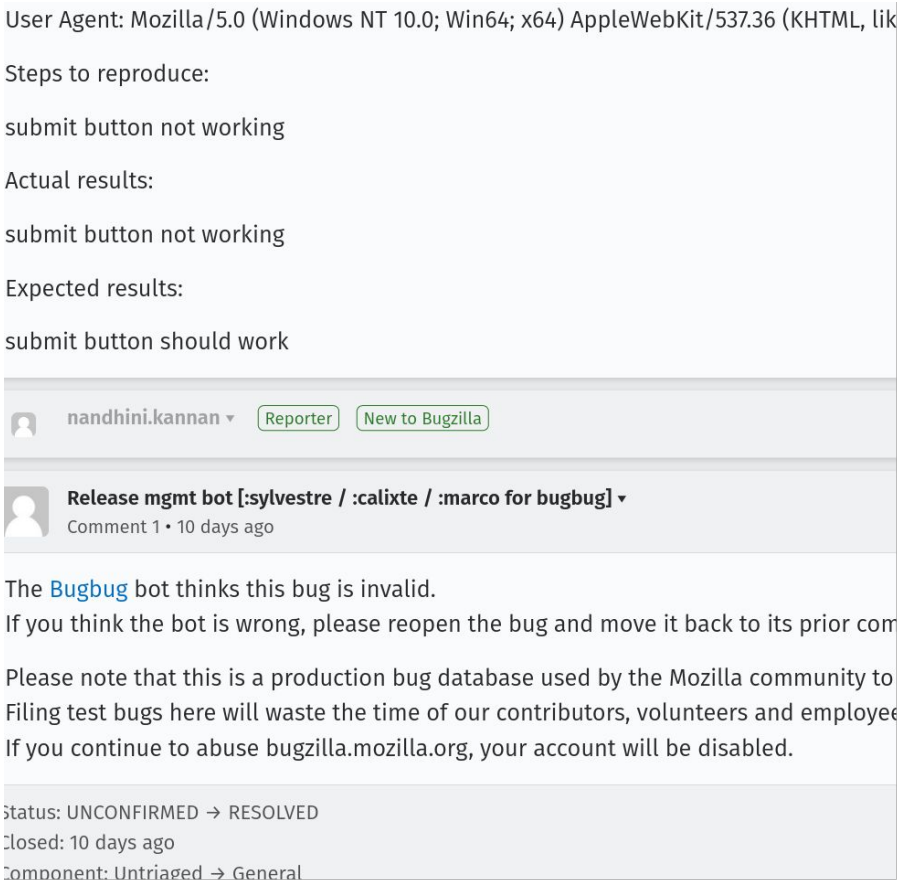
Release mgmt bot [:sylvestre / :calixte / :marco for bugbug] ▾
Comment 1 • 6 hours ago

The Bugbug bot thinks this bug should belong to the 'Core::Javascript: WebAssembly' component, and is mov
revert this change in case you think the bot is wrong.

Component: Untriaged → Javascript: WebAssembly
Product: Firefox → Core

Digression about bug management

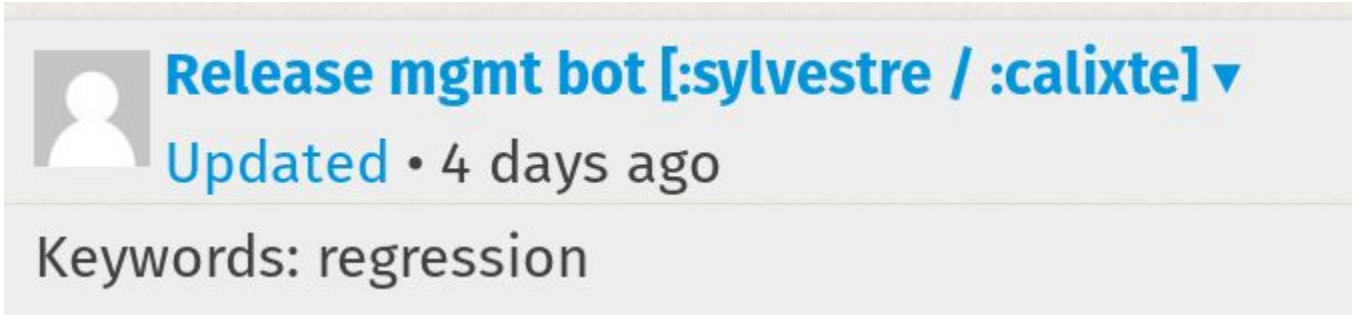
Detect spam



Digression about bug management

➔ Adding keywords on bugs

Example:



Pre release testing - web compat

- ➔ Platform to report Web compatibility issues
- ➔ Different of behavior between browsers leading to rendering issues or JS errors

julienw commented on 19 Jun 2017

URL: https://mobile.twitter.com/
Browser / Version: Firefox 56.0
Operating System: Android
Problem type: Something else - I'll add details below

Steps to Reproduce

1. Navigate to https://mobile.twitter.com/ with a valid account
2. Click the "retweet" icon on any tweet, and choose "Quote Tweet"
3. Start typing something

Expected Behavior:
What you type is what gets written.

Actual Behavior:
The first letter is *also* inserted at the end, you have to manually delete it.

Note: this doesn't happen on Mobile Twitter on Desktop. So could be something about how the virtual keyboard works.

From webcompat.com with ❤

webcompat-bot added `browser-firefox` `status-needstriage` labels on 19 Jun 2017

2 - Code quality?

moz://a

Numerous way to detect issues

- ➔ Static analysis
- ➔ Linting
- ➔ Crash reporting
- ➔ Code coverage
- ➔ Fuzzy
- ➔ etc

Static analysis / linting

- ➔ C & C++ are hard languages like really really hard!
  (did I say that Rust is the future and C/C++ the past?)
- ➔ How to detect programming mistakes
  - ◆ Related to the language designs
  - ◆ Usage of our APIs
  - ◆ Limit the code legacy

- ➔ Exam

```
/* !!! Should move this into its own .c and un-static it. */
static char *errStrings[] = {
    "Operation completed successfully.\n",
    "ERROR: NSS_Initialize() failed.\n"
    "ERROR: Unable to set initial password on the database.\n"
};
```

Static analysis / linting

- Clang analyzer: 23 checkers
  - Dead code, insecure functions, etc
- clang-tidy : 65+ checkers
  - Best practices, coding style, performances, C++ 11, 14 or 17 upgrade
- Mozilla's: 28 checkers
  - Security issues, bad usages of API, best practices

SA tools that we use

- ➔ We use other tools for other languages
  - Javascript - Eslint
  - Python - flake8/pylint
  - Rust - clippy
  - Java (android) - findbug
  - Bash - shellcheck
  - Typos - codespell

  For every review – average of 12 minutes analysis

Crash analysis

- When a crash occurs
  - Handled by breakpad
  - Sent to https://crash-stats.mozilla.com/
  - Doing some voodoo magic on them

Operating System

Operating System	Count	Percentage
Windows 7	247	98.0%
Windows 10	3	1.2%
Windows 8.1	2	0.8%

Product *

Product	Version	Count	Percentage	Installations
Firefox	58.0.1	252	27.7%	232
Firefox	57.0.4	203	22.3%	133

Crash analysis

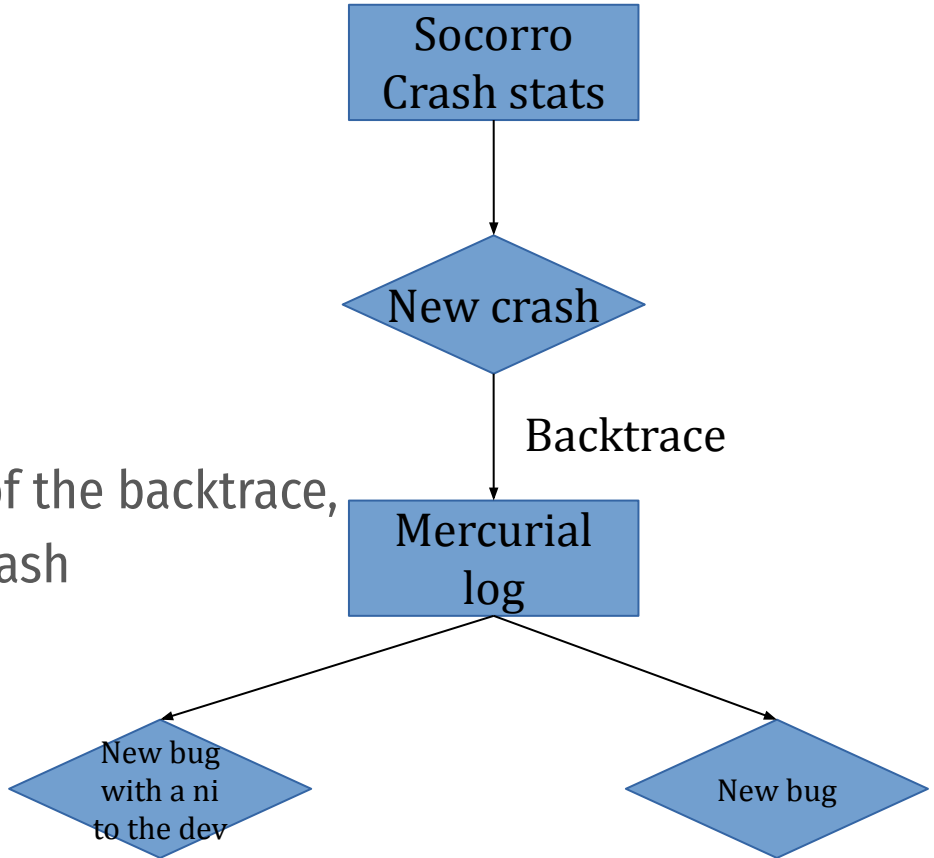
➔ Data mining on the results

Correlations for Firefox Release

```
(100.0% in signature vs 05.85% overall) reason = EXCEPTION_ACCESS_VIOLATION_WRITE
(95.03% in signature vs 01.32% overall) GFX_ERROR "[D3D11] failed to get compositor device." = true
(95.03% in signature vs 01.40% overall) GFX_ERROR "[D3D11] Failed to init compositor with reason: " = true
(72.79% in signature vs 00.21% overall) address = 0x198
(100.0% in signature vs 31.34% overall) Module "winsta.dll" = true
```

Crash analysis - clouseau

- ➔ Look at new crash signatures
- ➔ Extract the backtrace
- ➔ Look at the recent VCS history
- ➔ If a change touched one level of the backtrace, it might be the source of the crash
- ➔ 212 bugs reported



Fuzzing

- ➔ Send invalid, unexpected, or random data as inputs
- ➔ We are testing:
  - ◆ JavaScript features, DOM, Layout, CSS, Stylo, etc
  - ◆ Media file formats (images, audio, video)
  - ◆ API level
- ➔ Last 2 y, over 600 security bugs

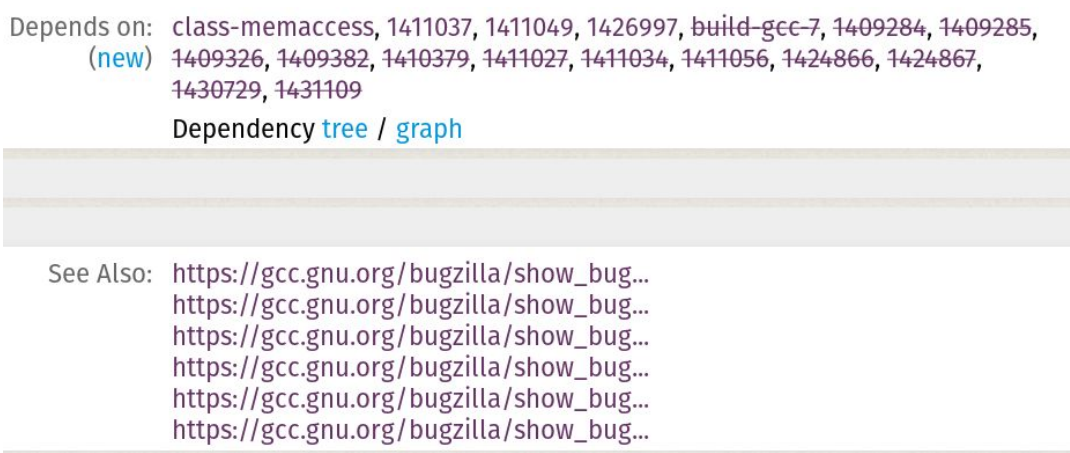
```
RegExp.prototype[Symbol.split].call({})


Backtrace:

Program received signal SIGSEGV, Segmentation fault.
0x0000000000a531a0 in JSVAL_TO_STRING_IMPL (l=<error reading variable: Canno
/dist/include/js/Value.h:778
#0  0x0000000000a531a0 in JSVAL_TO_STRING_IMPL (l=<error reading variable: C
/dist/include/js/Value.h:778
#1  toString (this=0x8) at js/src/opt64/dist/include/js/Value.h:1272
#2  getSource (this=0x7ffff7e6e140) at js/src/vm/RegExpObject.h:451
#3  js::regexp_construct_no_sticky (cx=0x7ffff6907800, argc=<optimized out>,
#4  0x0000000000879fd1 in CallJSNative (args=..., native=0xa53130 <js::regex
JS::Value*)>, cx=0x7ffff6907800) at js/src/jscntxtinlines.h:235
[...]
#28 main (argc=<optimized out>, argv=<optimized out>, envp=<optimized out>)
rax     0x8     8
rbx     0x7ffff6907800  140737330051072
rcx     0x0     0
```

Other best practices

- Once or twice a day, compiler Firefox trunk with -Werror on:
  - Build with gcc snapshot packages from Debian experimental (currently version 11)
  - Clang trunk (currently version 14)
- Find new issues in our code
- Find bugs in the compiler



3 - Automation

moz://a

20 to 30 years of tests

- ➔ 85,000 unique test files
- ➔ By almost 8500 developers
- ➔ Executed on all our supported platforms against a variety of build configs with different runtimes parameters

- ➔ Would represent the execution of 2.3 billion of tests per day

Crazy CI

CI

- ➔ Launched on every commit
- ➔ But quite complex and *stupid* algorithm to schedule them - back in early 2020

Leveraging ML

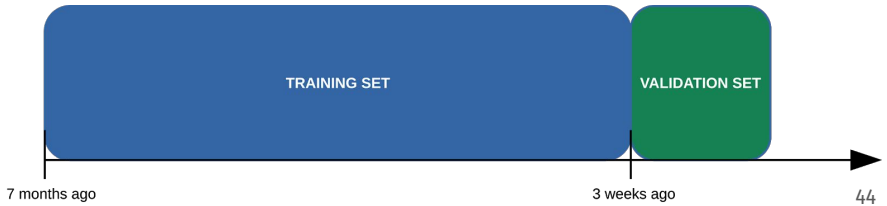
- Be smarter with test scheduling.
- Examples:
  - If we touch a gtk file, is it interesting to run all Javascript unit tests?
  - From the time where C++ compilers had different behavior, we have unit test to verify hashmap or string classes behavior. Is it interesting to run it for every build?

Leveraging ML

- We have very well qualified data set
- Why not delegating to a machine

  According to the past, when we touched file X, we broke the testsuite Y

  This testsuite didn't break once for the last X months.
- We used 7 months of data for training using a XGBoost model



Leveraging ML - results

- Cost divided by 3
- Previous solution: reduced the number of test tasks by 70%!
  CI system with no test selection: by almost 99%!
- Exposed to developers with a single command: *mach try auto*
- Fully documented - with sources and data: https://hacks.mozilla.org/2020/07/testing-firefox-more-efficiently-with-machine-learning/

Any questions?

moz://a


moz://a