this is a text anotation

You've just written a feature and (hopefully!) want to test it. Or you've decided that an existing feature doesn't have enough tests and want to contribute some. ut where do you start! You've loo"ed around and found references to things li"e #xpcshell# or #mo$mill# or #talos#. %hat do they all do! %hat's the overlap! &n short' where should your new tests go!

(his document aims to answer that)uestion. (here's a very short summary of each framewor"' and a bit of *+, to help you pic" your framewor". (his may only narrow down your choices' however' in which case you should read more about the framewor"s and-or hop on .ateam' .)a' or one of the development forums and as")uestions.

/enerally' you should pic" the lowest0level framewor" that you can. &f you are testing 1ava2cript but don't need a window' use xpcshell. &f you're testing page layout' try to use reftest. (he advantage is that you don't drag in a lot of other components that might have their own problems' so you can home in)uic"ly on any bugs in what you are specifically testing.

In production

buildbot automation

(hese tests are found within the mo$illa0central tree' along with the product code. (hey are all run when a changeset is pushed to mo$illa0central' mo$illa0inbound' or try' with the results showing up on tbpl. (hey can also be run on their own.

(he letters in parentheses are the abbreviations used by tbpl.

compiled-code (B)

%ritten in 344' compiled0code tests can test pretty much anything but are difficult to write properly. &n general' this should be your last option for a new test' unless you have to test something that is not exposed to 1ava2cript.

xpcshell (X)

xpcshell are console 1ava2cript tests. (here is no chrome' no content' no window. xpcshell is useful for testing low0level things' such as 563O7 components. &f you don't need a window' use this. xpcshell is particularly useful for testing low0level objects that are exposed to 1ava2cript.

JS shell tests (J)

(ests specifically for the 1ava2cript engine. (hey test every piece of the engine.

crashtest (C)

8eally simple9 open a web page and see if it causes a crash. &f you've found pages that crash :irefox' add a test here to ma"e sure future versions don't experience this crash again.

reftest (R)

, reftest verifies that two web pages are rendered identically to test layout and graphics correctness' ta"ing advantage of the fact that there is generally more than one way to achieve any given visual effect in a browser. :or each test' 8eftest will ta"e two sample pages that try to produce the same effect (normally one with a simple mar"up' and one using more complex mar"up) and verify that they produce the same visual construct.

Mochitest (M)

7ochitest uses 1ava2cript to test features. ,nything piece that has its functionality exposed in 1ava2cript can theoretically be tested with 7ochitest. #6lain# mochitest should be used to test ;O7 ,6&s and other pieces of functionality exposed to web content (i.e. re)uiring no special permissions).

Mochitest-other (Moth)

7ochitest0other are mochitests with higher privileges' logically split into a few sections9

&63 9 tests plugin ,6&s' particularly out0of0process plugins.

a<<y9 tests accessibility interfaces.

chrome9 tests running with high privileges that can test a lot of the browser's functionality. (ests that verify 1ava2cript interactions with chrome0level objects should go here. (hese tests do not exist for mobile :irefox.

browser0chrome9 running in the scope of the browser window' this is a rough =& automation tool testing how the browser interacts with itself and with content. 2ince these are moving away from the rest of mochitest's functionality' they will eventually be split into their own category' #b0c#.

Mochitest-Robocop (Mrc)

7ochitest08obocop tests run on >ative ,ndroid builds only mar"ed with an 'rc' in tbpl. (hese are 1ava based tests which run from the mochitest harness and generated similar log files. (hese are designed for testing the native =& of ,ndroid devices by sending events to the front end.

Talos (T)

(alos is a framewor" for performance testing. &f you're measuring performance' (alos is the place to go.

(alos tests on buildbot are split into a few categories. 2ome test suites are run in several categories but with different configurations or metrics. (he following are the codes as found on tbpl9

tp9 7 easures the load time of a set of test web pages ta"en from the , lexa top ?@@.

s9 2A / rendering performance.

Desktop only

c9 chrome. , set of suites with chrome (i.e. the full =&) enabled.

di9 dirty. =ses a #dirty# places.s)lite that more closely resembles that of an average user.

dr9 ; romaeo. , suite of 1ava2cript performance tests.

n9 nochrome. , set of suites with chrome disabled.

Mobile only

dh9 tdhtml. 7 easures the time to cycle through a set of ; B (7 C test pages.

pn9 tpan. Coads a test page and measures the time to pan to the bottom then bac" up to the top.

sp9 tsspider. 8uns the 2un2pider benchmar" test.

tpn9 (he tp suites with chrome disabled.

ts9 (ests the startup time of :irefox by opening the browser D@ times.

w9 twinopen. (ime to open a new window.

$9 t$oom. Coads a test page and measures the time to $oom in and out.

(hese are 8obocop based tests that are developed and running in either staging-production but have no official names on tbpl9

(tc)9 tchec"erboard. Coads a test page' $ooms and pans' measures the amount of chec"erboarding (delayed painting)

(tcD)9 tchec"erboardD9 2imilar to tchec"erboard' but supports pinch to $oom

(rp)9 trobopan9 Coads a test page and pans to the bottom and bac" to the top. (his measures the lag time in rendering the page.

(pr)9 tprovider9 :ills the awesomebar database (android os db) with thousands of entries and measures the time to perform a series of)ueries.

Mozmill

7 o$mill is an extensive framewor" for browser automation. &t is an extension with an extensive , 6& to help you write functional tests that simulate user interactions' as well as a full unit test , 6&. &t can be used to test configurations that are difficult to simulate in buildbot automation. * , automation uses

7 o$mill to test locali$ed builds' performance over time' and other scenarios.

Speedtests

2peed(ests is a framewor" for executing arbitrary tests' which report results via 1ava2cript calls' in several browsers. Originally this framewor" was designed for modified versions of 7 icrosoft's speed demos but has now been expanded to include conformance tests such as testDED and Fra"en. 2peed(ests are good for cross0browser comparisons when tests don't need to dig too deep into the guts of the browsers.

Peptest

6eptest measures responsiveness' how #snappy# :irefox-(hunderbird feels' by issuing alerts when the event loop is stuc" for more than ?@ ms. &t will soon be available on try' to catch regressions in responsiveness. &f you're helping to improve peppiness' consider writing some peptests to help you measure your improvements and to find future regressions.

Marionette harness

7 arionette is a test automation framewor" used to drive the =& and 12-563O7 layer of remote or local instances.

(he 7 arionette client includes the harness which runs 7 arionette and mochitest0browser0chrome tests. &t will give you similar functionality as 2elenium for :irefox builds' but with built in chrome support as well. %ith it' you can send commands to chrome or content on demand' allowing you to coordinate large scope tests li"e communicating to multiple remote gec"o processes (useful for testing things li"e 2 7 2 messaging for %eb , 6& for example).

(his is currently being used to test D/' but can wor" with any gec"o platform.

So which do I use already?

Bere's a series of)uestions to as" when you want to write some tests. 8emember this is only a rough guide' and it may give you multiple framewor"s. (ry .ateam on irc.mo$illa.org to get some more specific answers.

Is it low-level code?

&f the functionality is exposed to 1ava2cript' consider xpcshell. &f not' you'll probably have to use compiled0code tests.

Does it cause a crash?

&f so' a crashtest could help isolate the problem. >ote that this may lead to more tests once the core

problem is found.

Is it a layout/graphics feature?

8eftest is your best bet' if possible.

Do you need to verify performance?

(ry (alos!

Are you comparing speed or functionality between browsers?

2ee if you can write a 2peedtest.

Want to investigate responsiveness?

6eptest provides an easy way to measure responsiveness.

Testing UI?

&f it's mobile =&' loo" into 8obocop. :or des"top' try browser chrome tests' soon to be split out of 7ochitest' or 7o$mill

Testing Mobile/Android?

7obile =&' loo" at 8obocop. (here are some specific features that 7ochitest or 8eftest can cover. 7ochitest0chrome and browser0chrome do not run on ,ndroid. &f you want to test performance' (alos runs just fine with a few limitations (use 00no3hrome options) and smaller cycles (e.g. <@ iterations instead of D@' etc...)

None of the above?

(o get your tests run through buildbot' try 7ochitest' or' if higher privileges are re)uired and you don't need mobile testing' try 7ochitest chrome tests.

%hile not in buildbot automation' 7o$mill is a bigger framewor" that can test almost anything' on the des"top at least.

:or des"top :irefox or D/' or if you just want to see the future of /ec"o testing' loo" into the on0 going 7arionette project.