Zen and the Art of Telemetry:

An Inquiry into Performance

ANTHONY ZHANG

What?

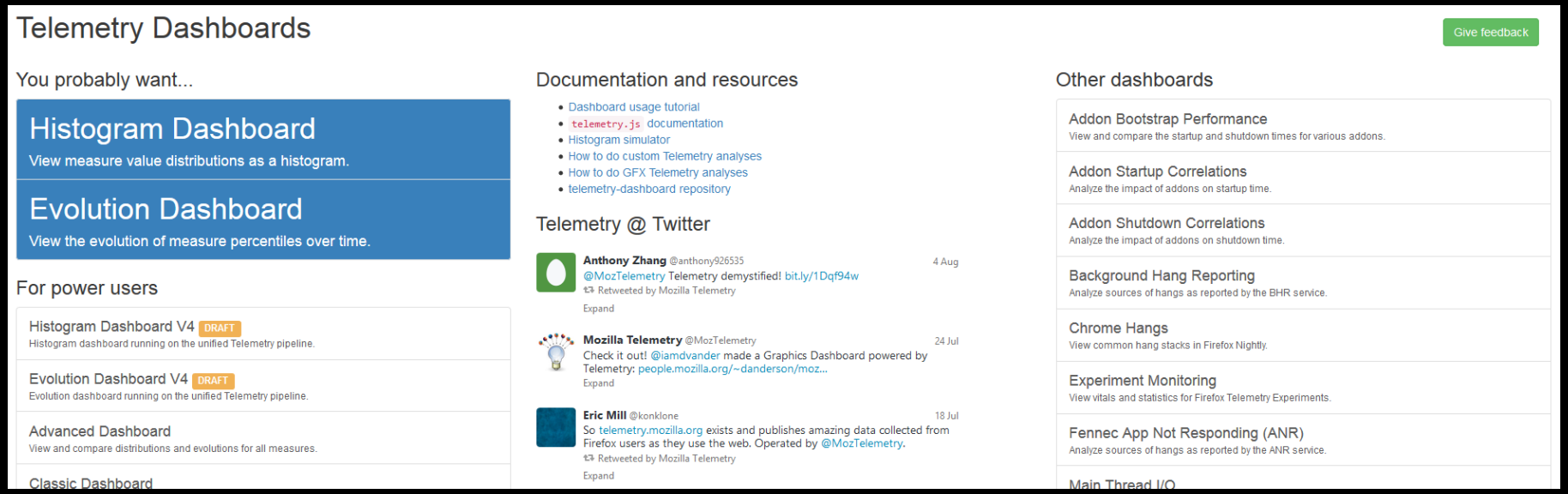
I work with the Desktop Performance team.
We find slow things and turn them into fast things.

Performance <3 data.
That's where Telemetry comes in.

Enter Telemetry

Firefox obtains (anonymous) metrics while browsing.
We receive several hundred gigabytes of these per day.

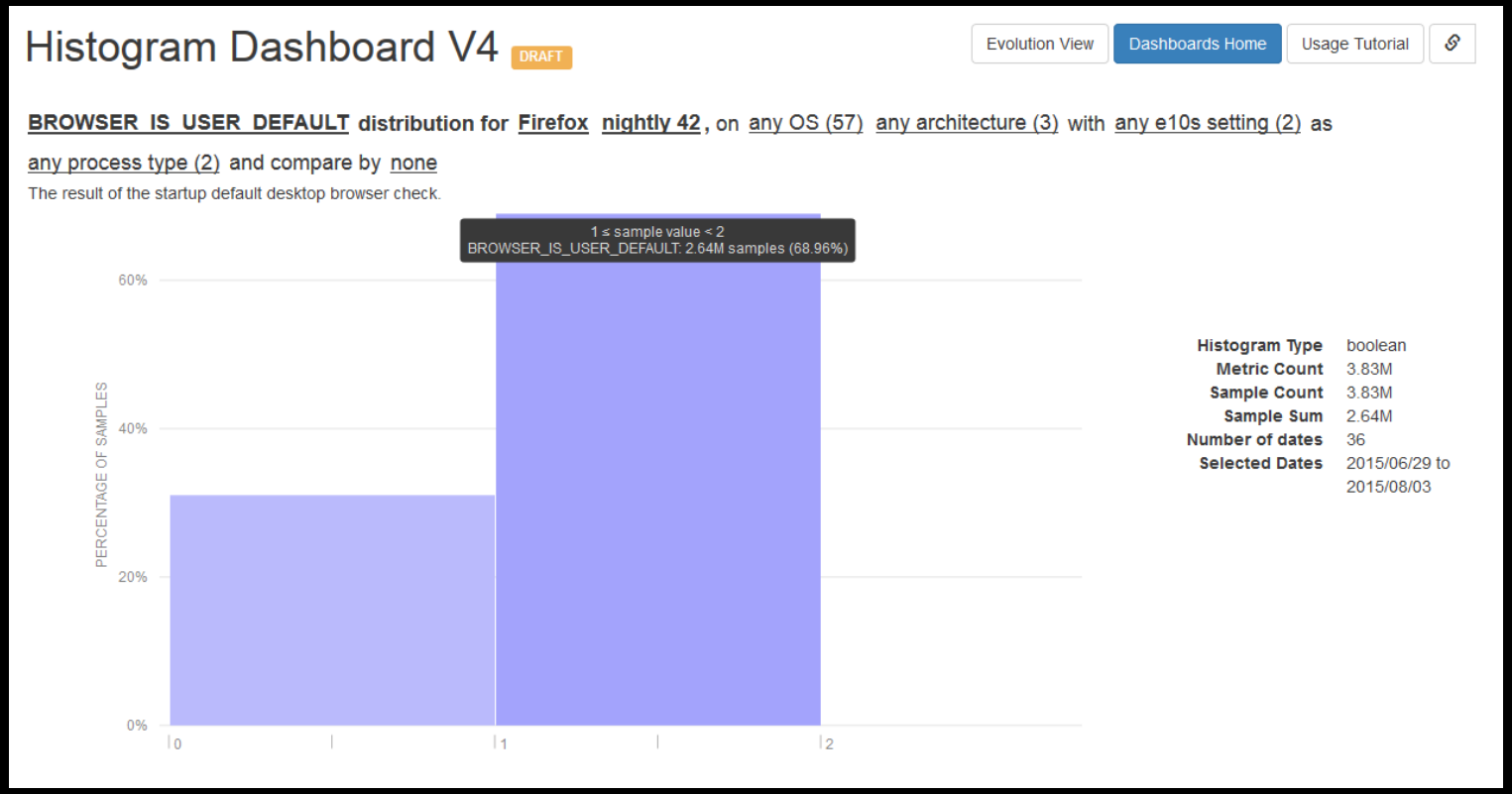
All that data is put together and made publicly available.



Check out telemetry.mozilla.org!

Q. How often is Firefox the default browser?

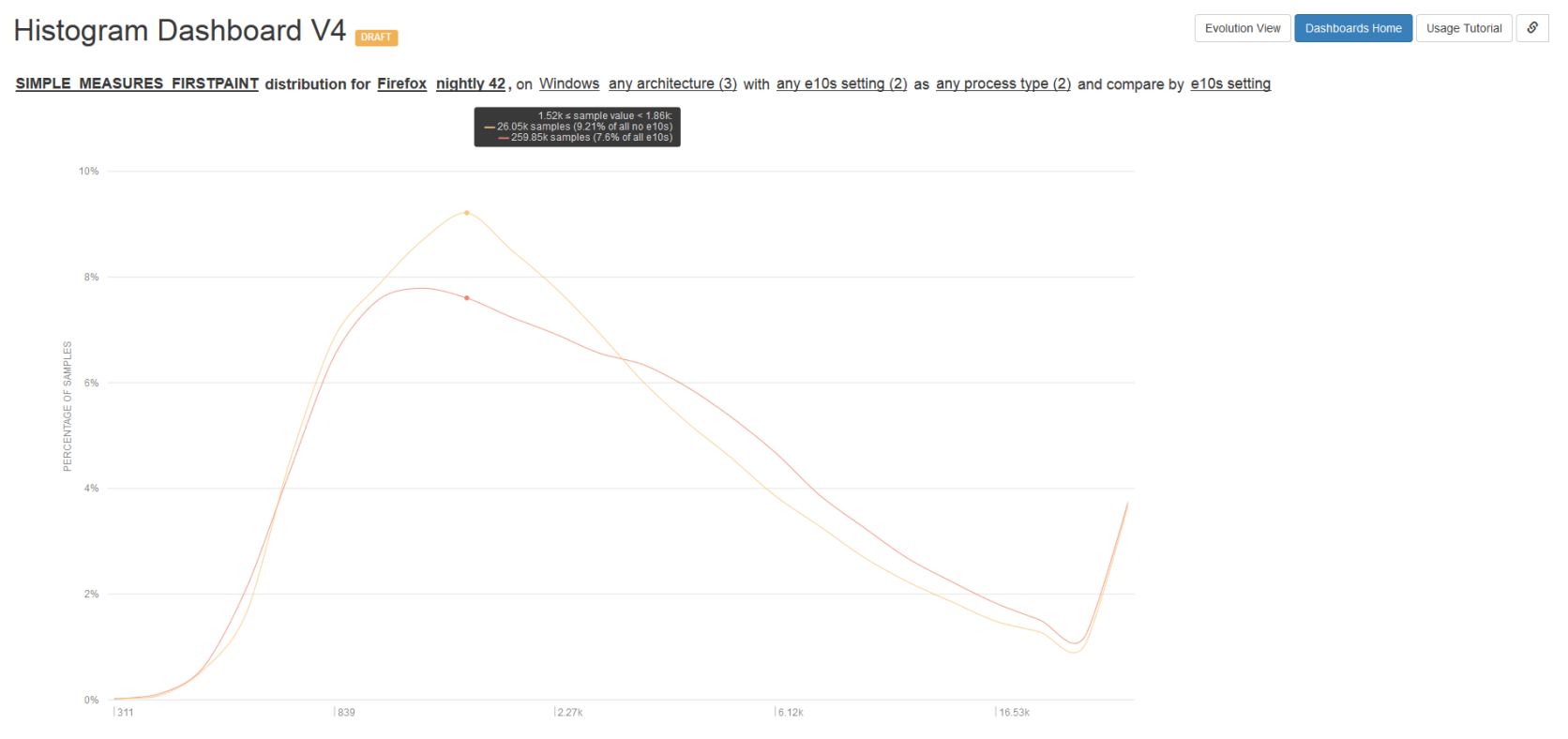
The purpose of Telemetry is to answer questions.



A. 68.96% of our nightly users have Firefox as their default browser.

Q. Does e10s make startup faster?

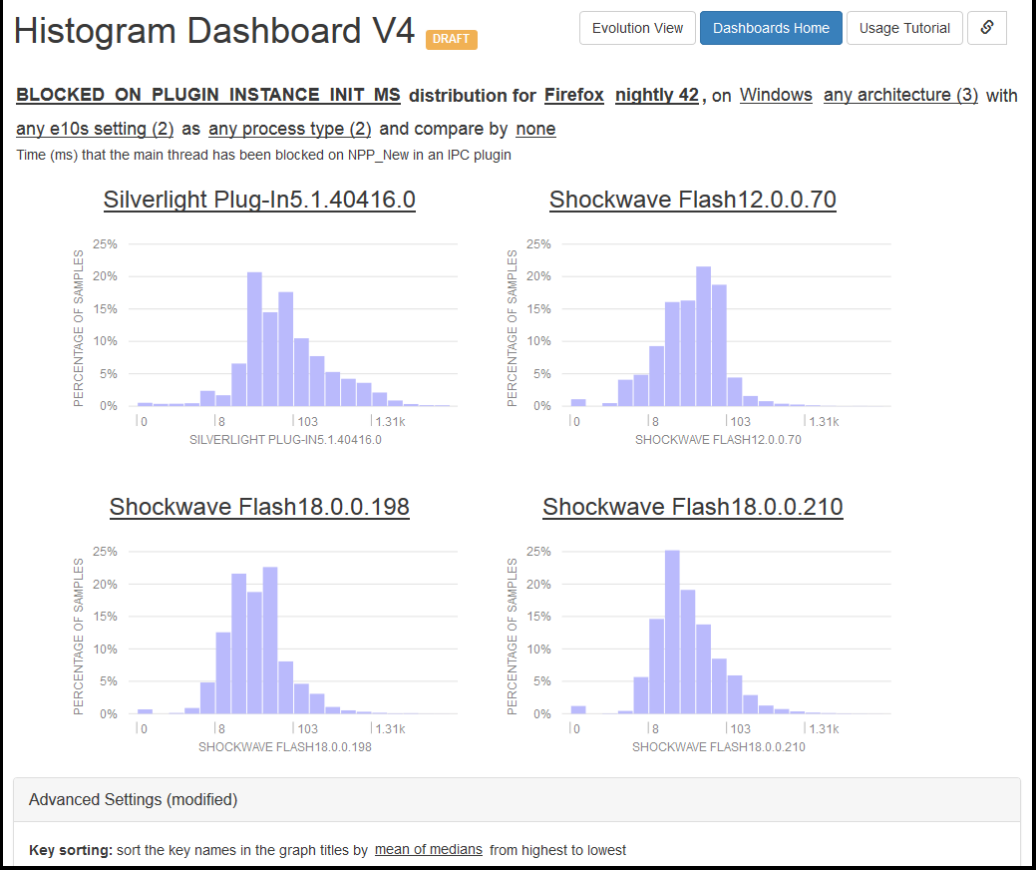
The dashboards on telemetry.mozilla.org cover many common use cases.



A. No, it's slightly slower.

Q. Which plugins tend to freeze the browser on load?

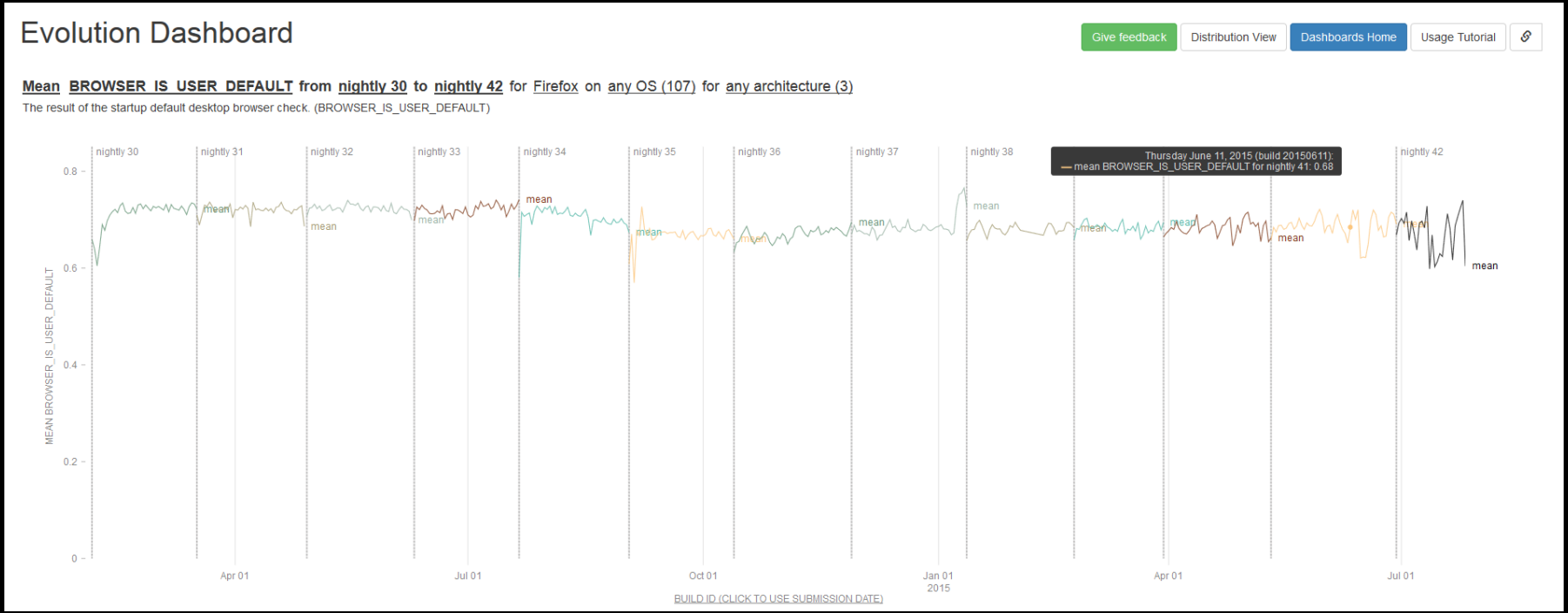
Better dashboards make it easier to make data-driven decisions.



A. Silverlight and Flash (is anyone surprised?)

Q. How has default-browser-ness changed over time?

There are also lots of other, more specialized views available on telemetry.mozilla.org.



A. The fraction of nightly users with Firefox as their default browser is consistently around 70%.

Telemetry.js

Can't answer your question with the dashboards?
Make your own with Telemetry.js!

```
        console.log("Available dates:\n" + evolutionMap[""].dates().join("\n"));
    });
});
```

Getting the overall median value of GC_MS on nightly 42 on Windows from July 19, 2015 to August 1, 2015:

```
Telemetry.init(function() {
    Telemetry.getEvolution("nightly", "42", "GC_MS", {os: "Windows_NT"}, true, function(evolutionMap) {
        var evolution = evolutionMap[""].dateRange(new Date("2015-07-19"), new Date("2015-08-01"));
        var histogram = evolution.histogram();
        console.log("Median GC_MS: " + histogram.percentile(50));
    });
});
```

API Reference

```
Telemetry.init(function callback() { ... })
```

Initializes Telemetry.js with various pieces of metadata that are used by a lot of the other functionality in the library. When

The library runs in the browser or in Node.js.
We also have an API!

I still have questions!

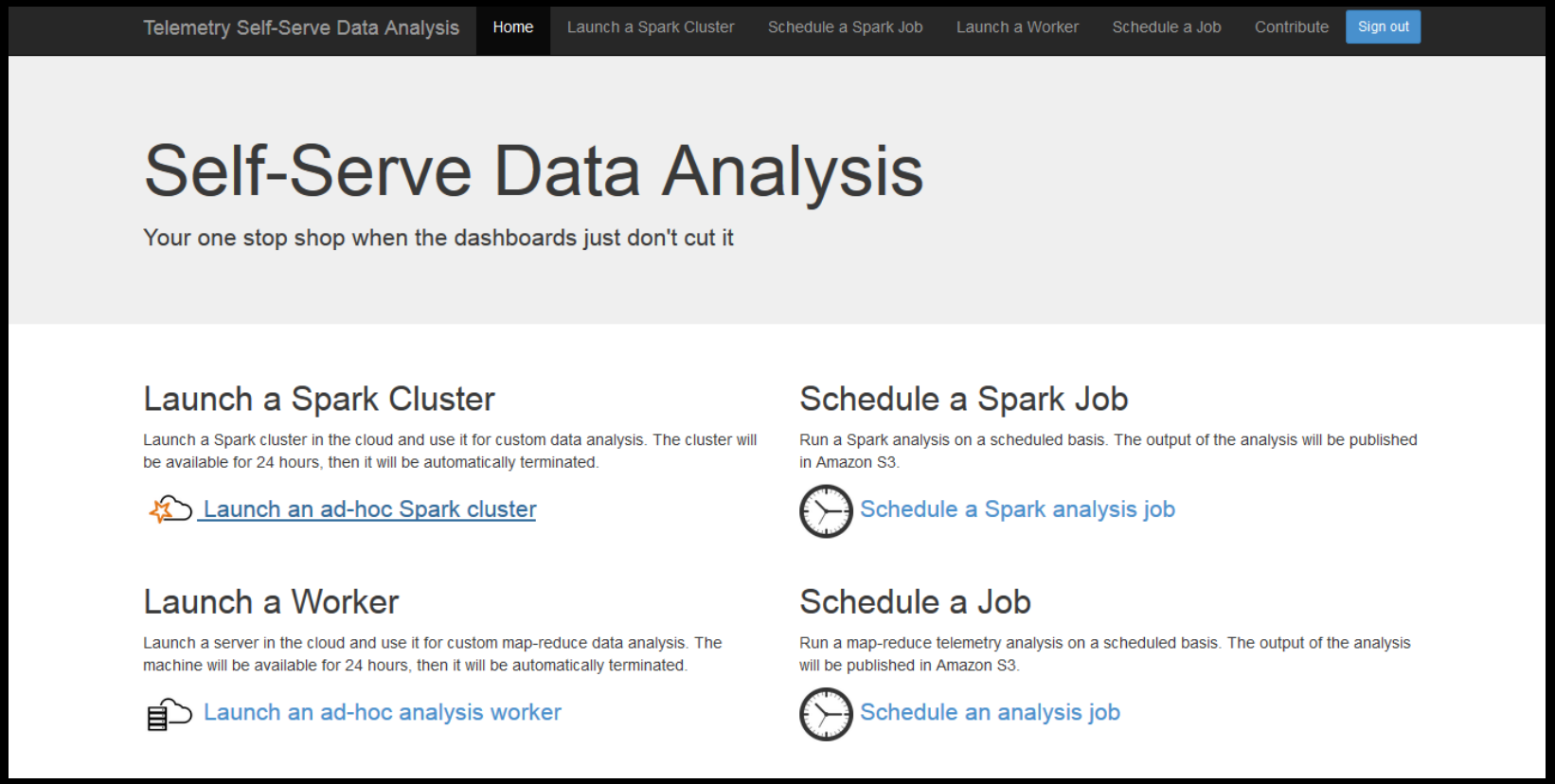
We've got you covered.
Let's say we have a more unusual request.

To what extent is start-up time correlated to the current phase of the moon?

Follow along with the code on git.io/vOhTI.

Start your browsers...

For everything else, there's custom Telemetry analyses.



Head on over to telemetry-dash.mozilla.org!

Titles are hard

Let's write code instead:

Moon Phase Correlation Analysis

In [ ]:
```
from moztelemetry import get_pings, get_pings_properties, get_one_ping_per_client
```

This Wikipedia article has a nice description of how to calculate the current phase of the moon. In code, that looks like this:

In [2]:
```
def approximate_moon_visibility(current_date):
    days_per_synodic_month = 29.530588853 # change this if the moon gets towed away
    days_since_known_new_moon = (current_date - dt.date(2015, 7, 16)).days
    phase_fraction = (days_since_known_new_moon % days_per_synodic_month) / days_per_synodic_month
    return (1 - phase_fraction if phase_fraction > 0.5 else phase_fraction) * 2

def date_string_to_date(date_string):
    return dt.datetime.strptime(date_string, "%Y%m%d").date()
```

So far, so good.

Let's get messy

Time to get ourselves some data!

Let's randomly sample 10% of pings for nightly submissions made from 2015-07-05 to 2015-08-05:

In [4]:
```
pings = get_pings(sc, app="Firefox", channel="nightly", submission_date=("20150705", "20150805"), fraction=0.1, schema="v4")
```

Extract the startup time metrics with their submission date and make sure we only consider one submission per user:

In [5]:
```
subset = get_pings_properties(pings, ["clientId", "meta/submissionDate", "payload/simpleMeasurements/firstPaint"])
subset = get_one_ping_per_client(subset)
cached = subset.cache()
```

Obtain an array of pairs, each containing the moon visibility and the startup time:

In [16]:
```
pairs = cached.map(lambda p: (approximate_moon_visibility(date_string_to_date(p["meta/submissionDate"])), p["payload/simpleMeasur
pairs = np.asarray(pairs.filter(lambda p: p[1] != None and p[1] < 100000000).collect())
```

This part takes about 10 minutes to run.

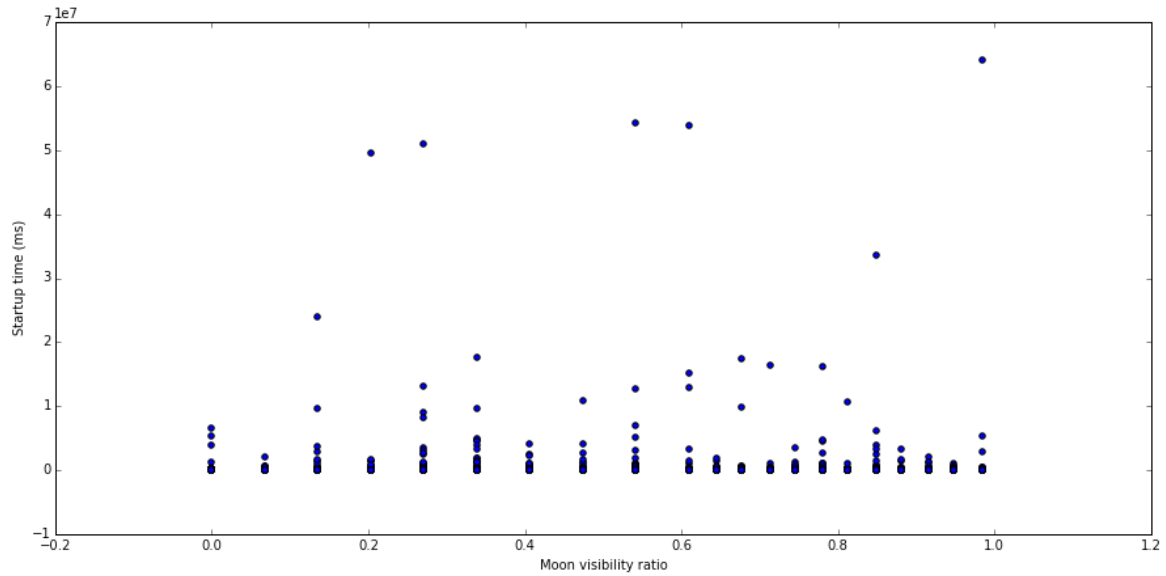
There's about 900k total unique users submitting start-up time metrics in the specified period.

Show me the numbers!

Plots are a great way to check your answers:

Let's see what this data looks like:

In [18]:
```
plt.figure(figsize=(15, 7))
plt.scatter(pairs.T[0], pairs.T[1])
plt.xlabel("Moon visibility ratio")
plt.ylabel("Startup time (ms)")
plt.show()
```



The correlation coefficient is now easy to calculate:

In [19]:
```
np.corrcoef(pairs.T)[0, 1]
```

Out[19]: 0.00048989909014640089

The verdict

We see that the correlation is roughly 0.0005.

The moon does not have a significant effect on Firefox start-up time.

(in the last month)

(on nightly)

(for now)

Links and stuff

Telemetry Dashboards/Documentation
telemetry.mozilla.org

Telemetry Demystified
anthony-zhang.me/blog/telemetry-demystified

Custom Telemetry Analyses
telemetry-dash.mozilla.org

Performance Team @ Mozilla
wiki.mozilla.org/Performance

Example Analysis: Moon Phase Correlation
git.io/vOhTI

We're on Twitter too! @MozTelemetry

Also, #perf @ irc.mozilla.org.