Progressive Web Apps...

Dave Rupert / @davatron5000

...on Ruby on Rails...

...at a JavaScript meetup!


WE DESIGN
WE CODE
WE LOVE

http://shoptalkshow.com


DAY
TRIP
http://godaytrip.com

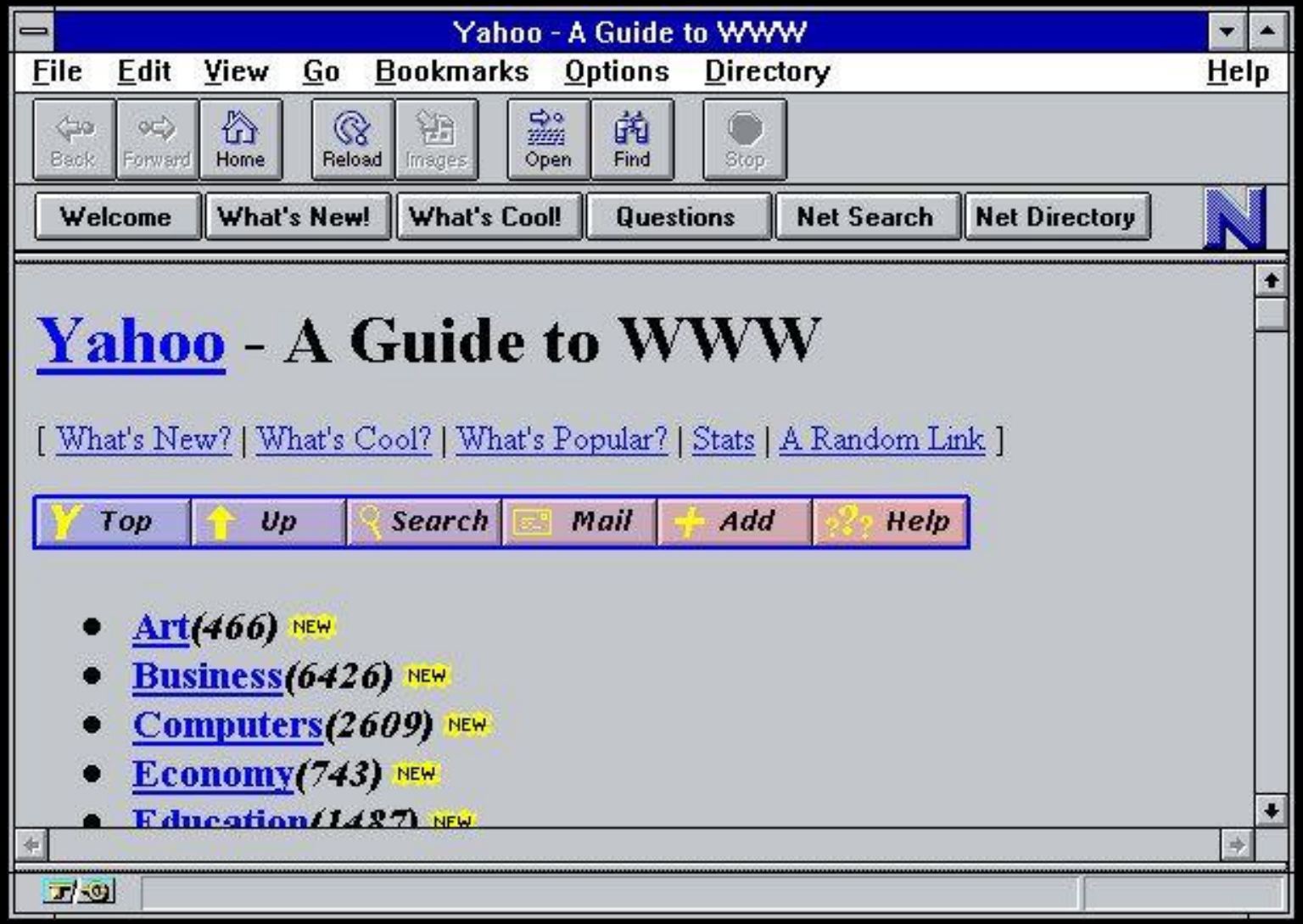
POP QUIZ

POP QUIZ: Native Apps

How many apps does the average mobile user download per month?	0
Percent drop-off for each stage of your mobile app onboarding?	20%
Number of companies in the Top 10 app downloads according to COMSCORE?	4

If only people didn’t have to download…

Introducing... The Web™



Available Today

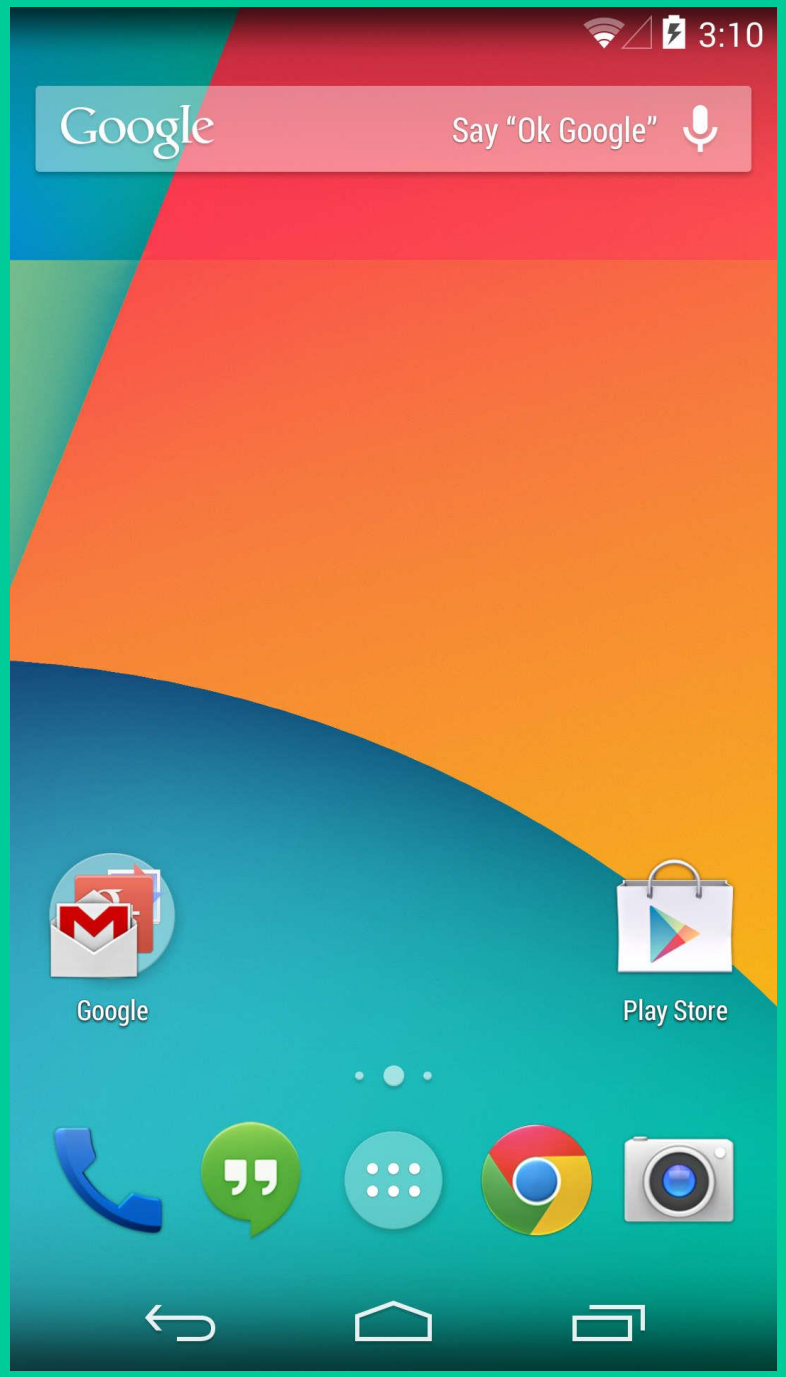
Web vs Native

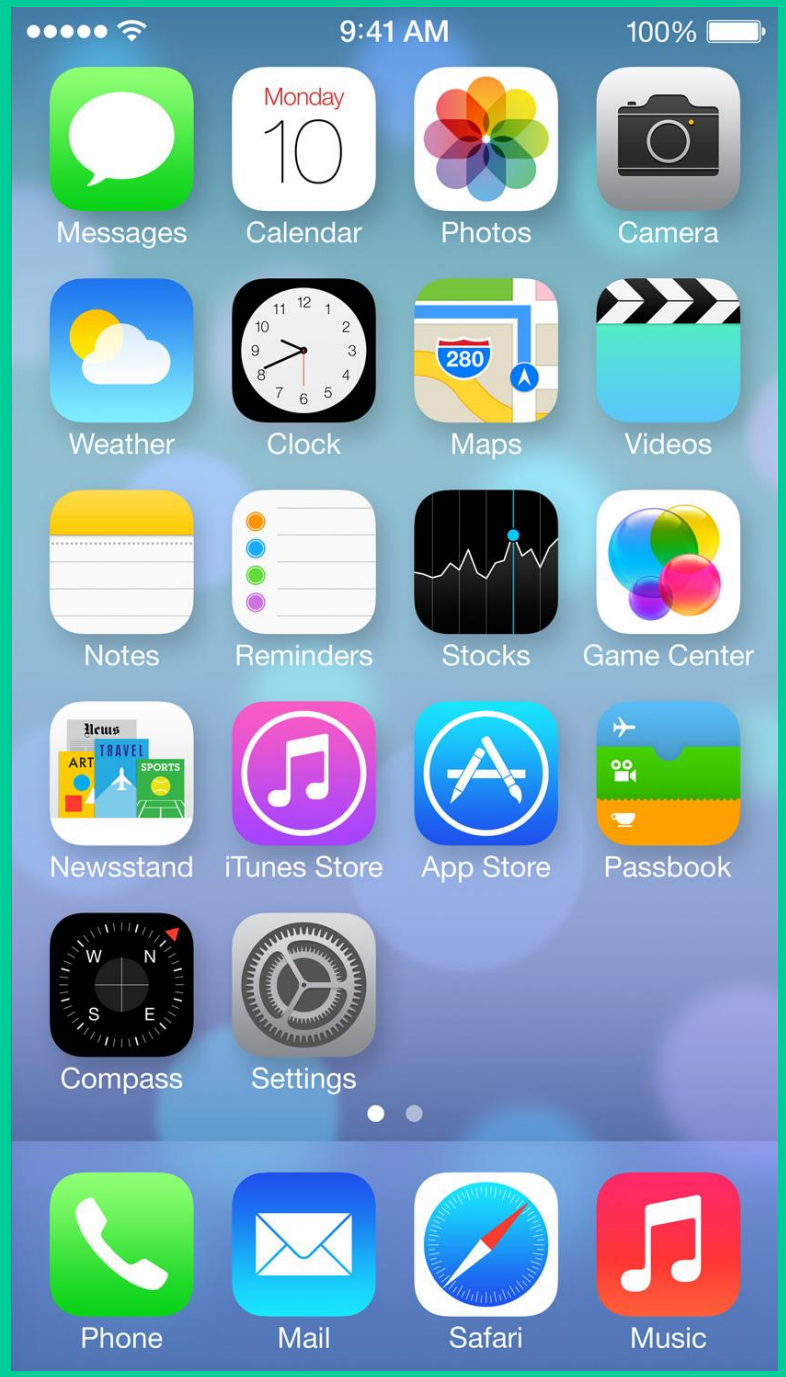
Cats vs. Dogs, Right vs. Left, Pants vs. No pants


DAY
TRIP

How do we get here?

We received a lot of feedback like “Where can I download it?” and “If it’s not on my homescreen I forget about it.”





To native or not to native...

Reasons we'd want to pursue native still

- In the App Store. Easiest way to Install to Homescreen
- Longer Log-ins (we reset every few days, apps would essentially be logged in forever)
- Access to credit cards (not a feature we're using)
- Mobile traffic is 97% iOS

Reasons we'd want to not pursue native

- We can focus on one Universal website, not a bunch of thin clients that all look nearly the same
- Stay in our wheelhouse
- Not have to deal with app stores, accounts, device provisioning, and percentage cuts

How can it work here?

Requirement for an app that takes you deep into the woods, it must work deep in the woods.

“Progressive Web Apps: Escaping Tabs Without Losing Our Soul”

Alex Russell, Google

TL;DR – “The future of the Web hangs on something I invented.”

Progressive Web Apps:

infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul

Infrequently Noted

Alex Russell on browsers, standards, and the process of progress.

« PSA: Service Workers are Coming

A Funny Thing Happened On The Way To The Future... »

Progressive Web Apps: Escaping Tabs Without Losing Our Soul

It happens on the web from time to time that powerful technologies come to exist without the benefit of marketing departments or slick packaging. They linger and grow at the peripheries, becoming old-hat to a tiny group while remaining nearly invisible to everyone else. Until someone names them.

This may be the inevitable consequence of a standards-based process and unsynchronized browser releases. We couldn’t keep a new feature secret if we wanted to, but that doesn’t mean anyone will hear about it. `XMLHTTPRequest` was available broadly since IE 5 and in Gecko-based browsers from as early as 2000. “AJAX” happened 5 years later.

This eventual adding-up of new technologies changes how we build and deliver experiences. They succeed when bringing new capabilities while maintaining shared principles:

- URLs and links as the core organizing system: if you can’t link to it, it isn’t part of the web
- Markup and styling for accessibility, both to humans and search engines
- UI Richness and system capabilities provided as additions to a functional core
- Free to implement without permission or payment, which in practice means standards-based

What’s All This Then?

I'm Alex Russell, a ~~web~~ developer working on Chrome, Blink, and the Web Platform at Google. I'm guil of many JavaScript transgressions.

I help lead the team buildin a new application model for the web, and serve on ECMA TC39 (the standards body fo JavaScript). I'm an elected member of the W3C Technical Architecture Grou and am Tech Lead for Standards inside the Chrom team. I design and advocate for extensible, layered, data driven evolution of the web platform.

My professional aim is to make the web a better platform and to the extent that I can keep politics and economics from creeping in, that's what this blog is abou

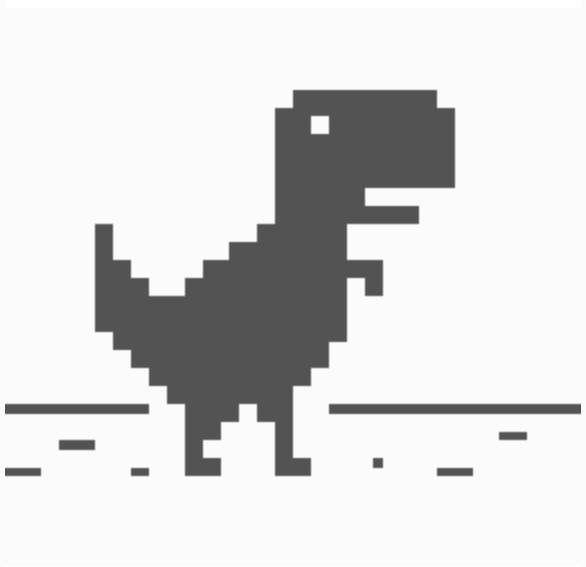
Other facets available upon

What is a Service Worker?

A background web worker with a few superpowers.


LEAVE THE MAIN THREAD ALONE

What is a Service Worker good for?



Push Notifications

Mining bitcoins

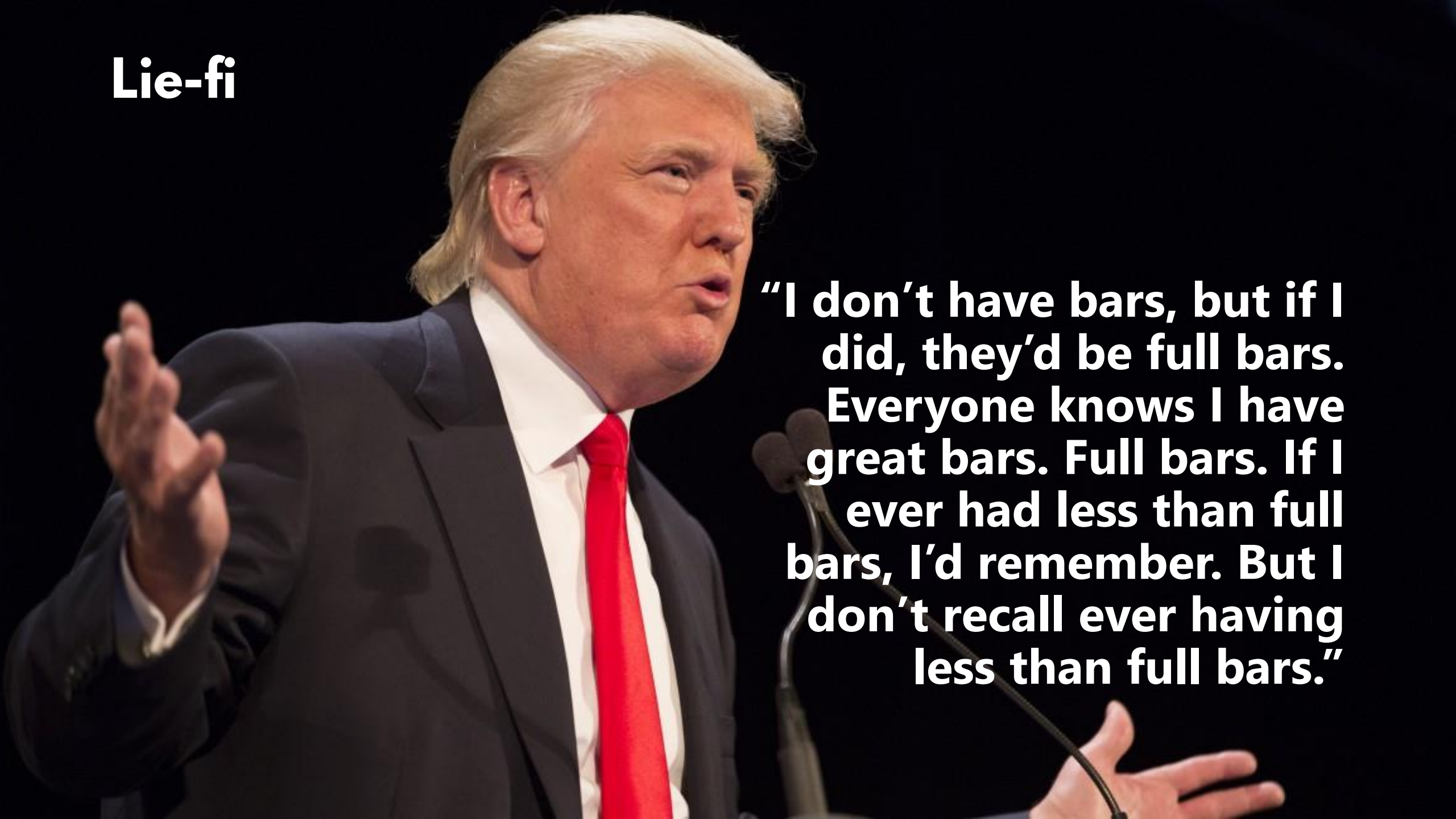
botnets

Offline

Background sync

#webperf webrings

Lie-fi

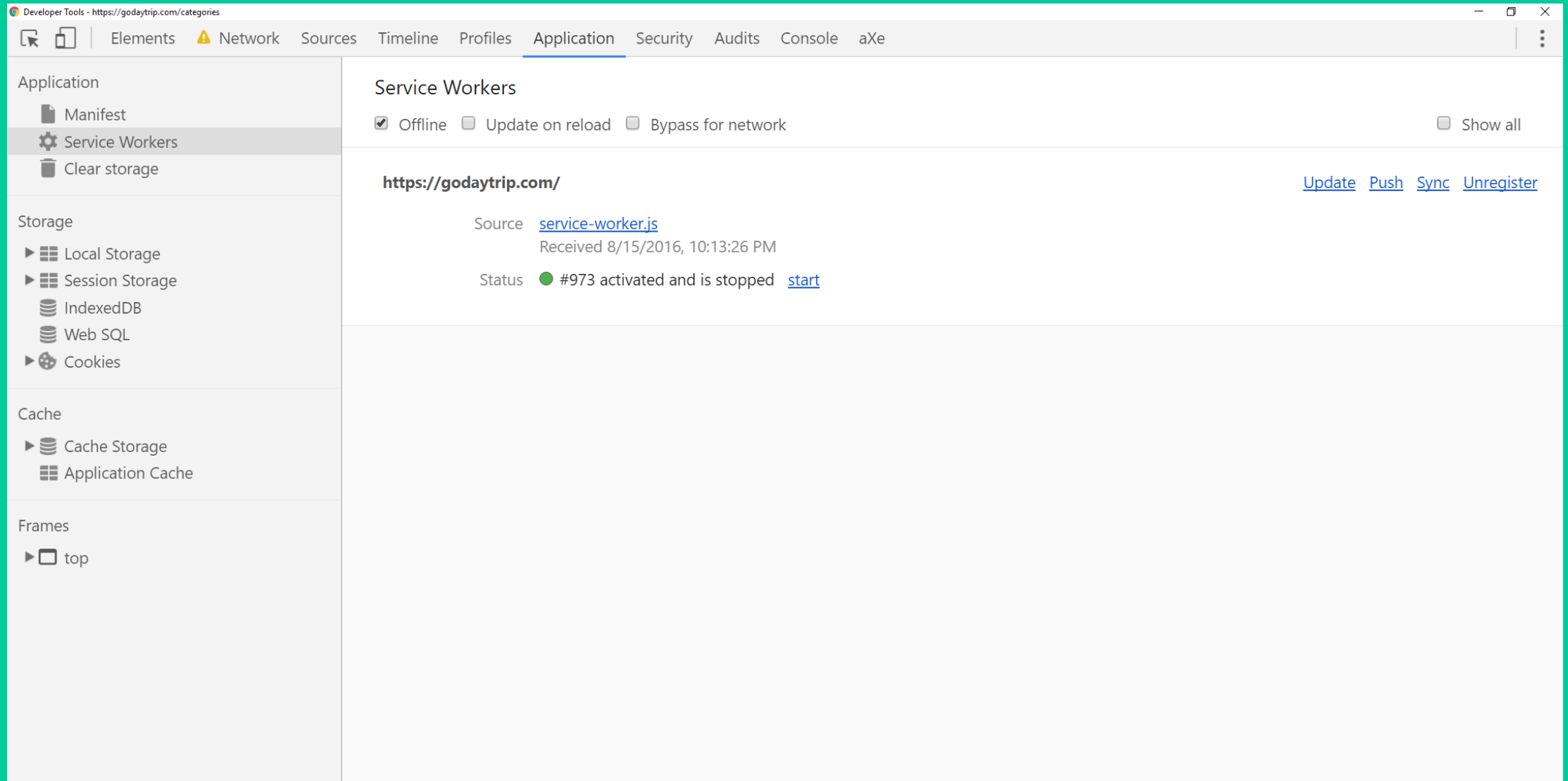

Lie-fi
“I don’t have bars, but if I did, they’d be full bars. Everyone knows I have great bars. Full bars. If I ever had less than full bars, I’d remember. But I don’t recall ever having less than full bars.”

Let’s build a Progressive Web App!

Minimum Viable Product: An Offline Service Worker

The Tools

Application Pane



Lighthouse

Categories on DayTrip | Lighthouse report: https:

chrome-extension://blipmdconlkpinefehnmjammfjpmpbjk/pages/report.html

Lighthouse
https://godaytrip.com/categories

ALPHA

- Progressive Web App
- Best Practices
- Performance Metrics

View: User feature

Progressive Web App

92 / 100

These audits validate the aspects of a Progressive Web App.

App can load on offline/flaky connections

Ensuring your web app can respond when the network connection is unavailable or flaky is critical to providing your users a good experience. This is achieved through use of a Service Worker.

- Has a registered Service Worker — Yes
- URL responds with a 200 when offline — Yes
- Cache contains start_url from manifest — Yes

Page load performance is fast

Users notice if sites and apps don't perform well. These top-level metrics capture the most important perceived performance concerns.

- First meaningful paint (target: 1,600ms) — (1682.1ms) 96
- Speed Index (target: 1,250) — (2330) 89

  First Visual Change: ***2089ms***

  Last Visual Change: ***2898ms***
- Estimated Input Latency (target: 50ms) — (36.3ms) 99

  90% probability of input latency at ***36.3ms*** *or shorter.*

The 3 parts of a Progressive Web App

manifest.json

offline.html

service-worker.js

(HTTPS too)

The Web App Manifest

manifest.json

manifest.json - DayTrip - Visual Studio Code

File Edit View Go Help

manifest.json × offline.html.erb service-worker.js.erb

```
1  {
2    "name": "DayTrip",
3    "short_name": "DayTrip",
4    "icons": [{
5          "src": "apple-touch-icon-152x152-precomposed.png",
6          "type": "image/png",
7          "sizes": "152x152"
8        }, {
9          "src": "touch-icon-192x192.png",
10         "type": "image/png",
11         "sizes": "192x192"
12       }],
13   "start_url": "/?utm_source=web_app_manifest",
14   "display": "standalone",
15   "orientation": "portrait",
16   "background_color": "#ffffff",
17   "theme_color": "#3E8792"
18 }
19
```

master* 0 0

<head>

```
<meta name="theme-color" content="#3E8792">
<link rel="manifest" href="/manifest.json">
```

The Offline Page

offline.html

offline.html.erb - DayTrip - Visual Studio Code

File Edit View Go Help

manifest.json | offline.html.erb | service-worker.js.erb

```
1 <h1>You're offline</h1>
2 <p>Looks like your device went offline.</p>
3
```

master* 0 0

The Service Worker

service-worker.js

service-worker.js.erb - DayTrip - Visual Studio Code

File Edit View Go Help

manifest.json | offline.html.erb | service-worker.js.erb

```
// Update 'version' if you need to refresh the cache
var staticCacheName = 'static';
var version = 'v1::';

// Store core files in a cache (including a page to display w
function updateStaticCache() {
  return caches.open(version + staticCacheName)
    .then(function (cache) {
      return cache.addAll([
        '<%= url_to_stylesheet "application" %>',
        'https://fonts.googleapis.com/css?family=Source+Sans+
        '<%= url_to_javascript "application" %>',
        '/offline'
      ]);
    });
};

self.addEventListener('install', function(event) {
  event.waitUntil(updateStaticCache());
});

self.addEventListener('activate', function (event) {
```

master* 20 0

Wow. We're almost done with our quest!?

HAHAHAhHHHHHahhahH
AhahHAHhHHHHHHHHHHh
ahHAHhAHhAHhAHhHh
HhHHHAHhHaaahHHhH
HHhHhaaHhHhHhHHhHh

Dave Rupert
@davatron5000

Service Worker is my favorite new in-browser JavaScript framework. I haven't had this much fun hitting endless roadblocks since React.

The Service Worker API

10x Faster Than React, Angular, Ember, and jQuery Combined!

If u like Promise(s)...

.then(‘you\’ll love Service Workers’)

.catch(‘my drift?’);

The Service Worker Event Lifecycle

Install

Activate

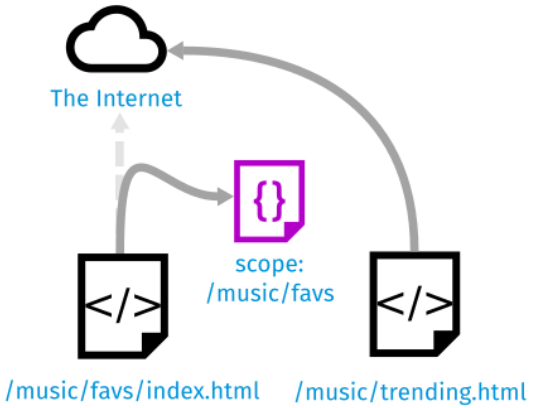
Fetch

Sync

Push

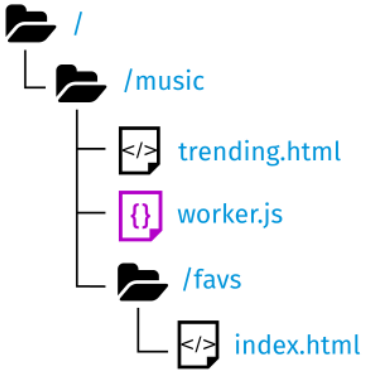
SERVICE WORKERS 101

Important!



The service worker will catch requests from the clients **under** scope only.

The service worker must be served over **https**.



The **max scope** for a service worker is the location of the worker.

Pro tip: if you serve a service worker along with the `Service-Worker-Allowed` header, you can specify here a list of **max scopes** for this worker.

Events

- `install`
- `activate`
- `message`

Functional events

- `fetch`
- `push`
- `sync`

self.addEventListener(‘install’)

```
self.addEventListener(‘install’, function( event ) {
  event.waitUntil(
    //
    // Prime the caches
    //
  )
});
```

event.waitUntil() ?

"Hey event, wait until the stuff in these parenthesis are done before you tell the other events you fired."

self.addEventListener(‘activate’)

```
self.addEventListener(‘activate’, function( event ) {
  event.waitUntil(
    //
    // Clear out old caches
    //
  )
});
```

self.addEventListener(‘fetch’)

```
self.addEventListener(‘fetch’, function( event ) {
  event.respondWith(
    //
    // Instead of fetching stuff from the network,
    // try this stuff instead.
    //
  )
});
```

event.respondWith() ?

"Hey (fetch) event, respond with this instead of what you were gonna do."

Yay! Now we're in the wonderful world of managing cache...

Quite literally one of the hardest problems in Computer Science.
But at least we get to, you know, use JavaScript to do it.

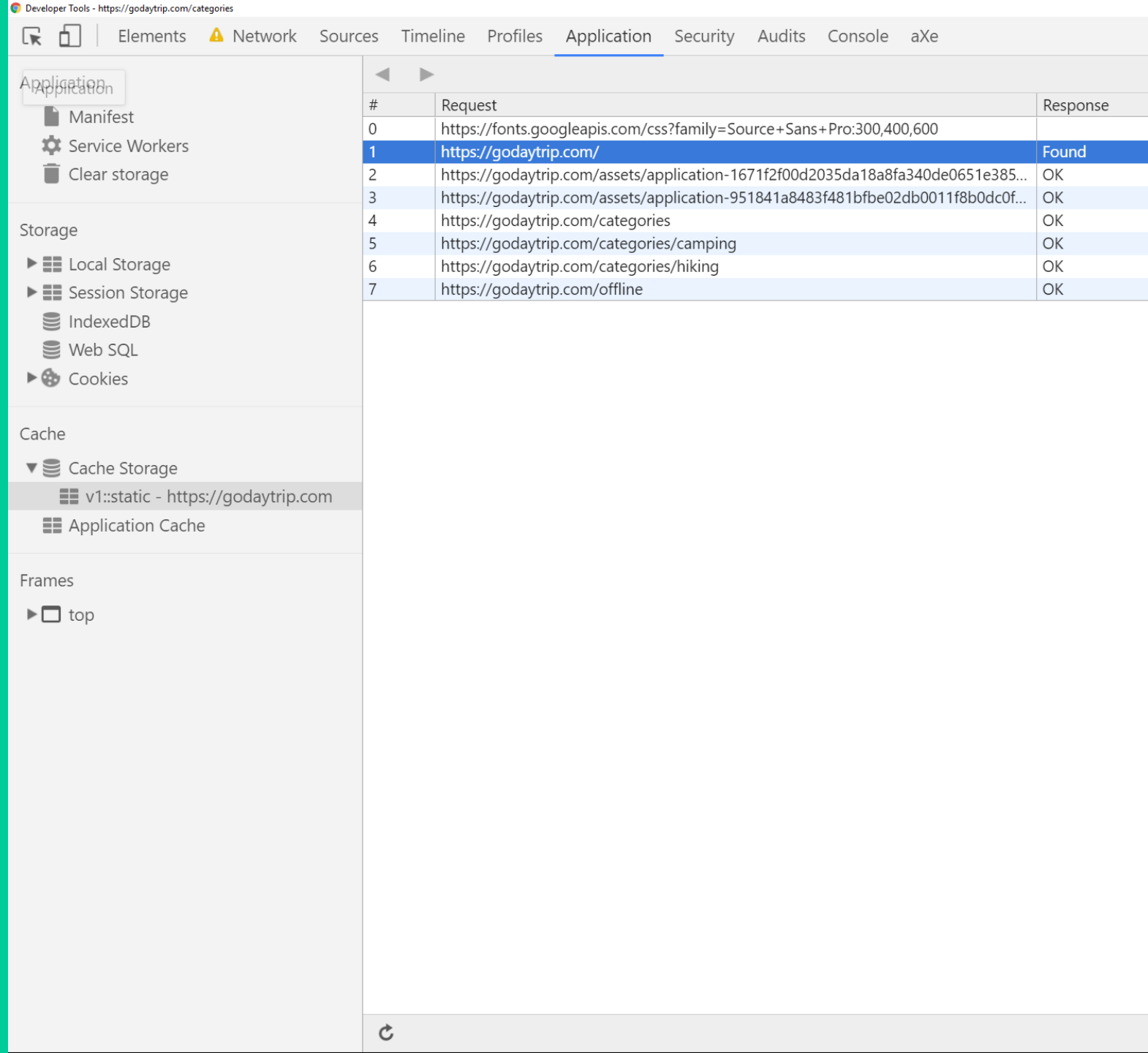
The Cache API

caches.open(cacheName)

caches.keys()

caches.match(request)

caches.delete(key)



Let’s put it all together

~100 Lines of Fun, Stolen from adactio.com/serviceworker.js

Version the cache

```
var staticCacheName = 'static';
var version = 'v1::';
```

How we'll prime the cache

```
function updateStaticCache() {
  return caches.open(version + staticCacheName)
    .then(function (cache) {
      return cache.addAll([
            '/assets/application.css'
            '/assets/application.js',
            '/offline'
    ]);
  });
};
```

Install the Service Worker

```
self.addEventListener('install', function(event) {
  event.waitUntil( updateStaticCache() );
});
```

Once activated, clear old caches

```
self.addEventListener('activate', function (event) {
  event.waitUntil( caches.keys().then(function (keys) {
    // Remove caches whose name is no longer valid
    return Promise.all(
      keys.filter(function (key) {
        return key.indexOf(version) !== 0;
      })
      .map(function (key) {
        return caches.delete(key);
      })
    );
  }));
});
```

The main event: hijacking fetch

```
self.addEventListener('fetch', function (event) {
  // Step 1: Always fetch non-GET requests from the network
  // Step 2: For TEXT/HTML do this:
  //   a) Try the network first
  //   b) If that fails, fallback to the cache
  //   c) If that doesn’t exist, show the offline page
  // Step 3: For non-TEXT/HTML (e.g. Images) do this:
  //   a) Try the cache first
  //   b) If that fails, try the network
  //   c) If that fails, hijack the request
});
```

Fetch, step one

```
// Step 1: Always fetch non-GET requests from the network
var request = event.request;

if (request.method !== 'GET') {
  event.respondWith(
    fetch(request)
      .catch(function () {
        return caches.match('/offline');
      })
  );

  return;
}
```

Fetch, step two

```
// Step 2 For TEXT/HTML do this…
if (request.headers.get('Accept').indexOf('text/html') !== -1) {
  event.respondWith(
    fetch(request)
      // Then Stuff
      // Catch Stuff
  );

  return;
}
```

Fetch, step two

```
// Step 2: Then Stuff…

.then(function (response) {
  // Stash a copy of this page in the cache
  var copy = response.clone();
  caches.open(version + staticCacheName)
    .then(function (cache) {
      cache.put(request, copy);
    });

  return response;
})
```

Fetch, step two

```
// Step 2: Catch Stuff…

.catch(function () {
  return caches.match(request).then(function (response) {
    return response || caches.match('/offline');
  })
})
```

Fetch, step three

```
// Step 3: For non-TEXT/HTML (e.g. Images) do this…
event.respondWith(
  caches.match(request).then(function (response) {
    return response || fetch(request)
      .catch(function () {
        // If the request is for an image, show an offline placeholder
        if (request.headers.get('Accept').indexOf('image') !== -1) {
          return new Response('<svg>…</svg>', {
            headers: { 'Content-Type': 'image/svg+xml' }
          });
        }
      });
  })
);
```

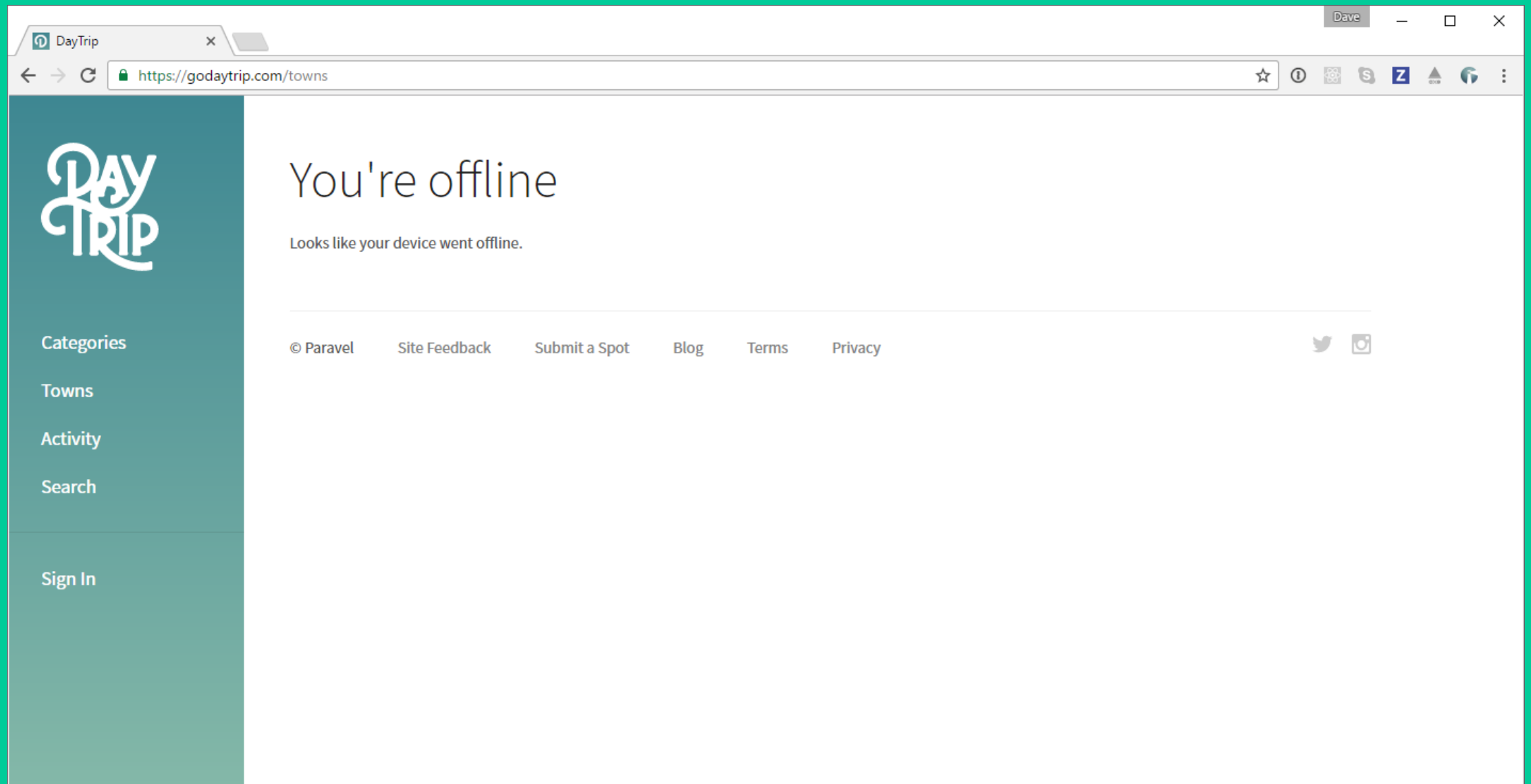
Registering the Service Worker

The Last Step?

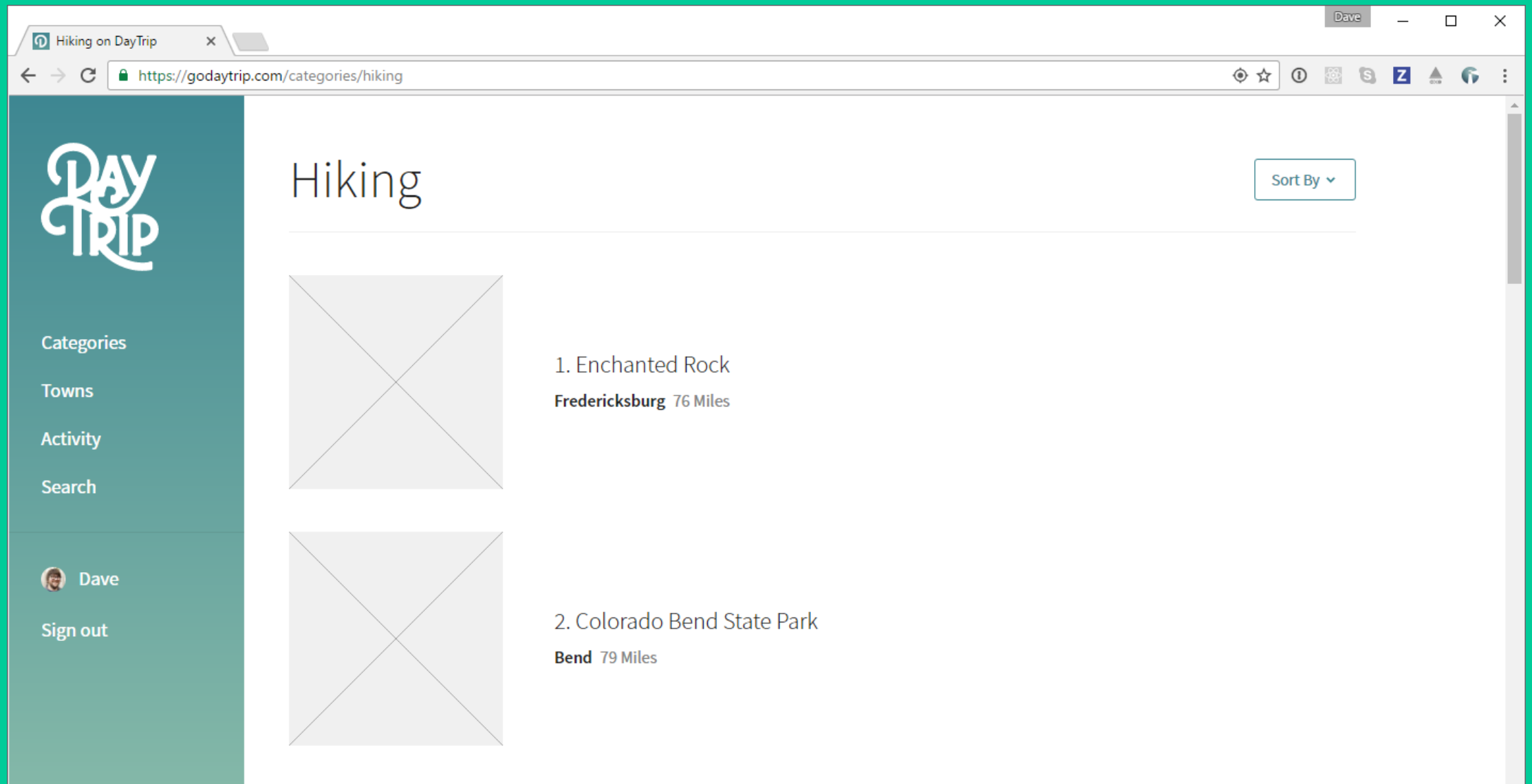
navigator.serviceWorker.register()

```
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register( '/service-worker.js', {
    scope: '/'
  }).then(function(reg) {
    console.log(‘Works! Scope is ' + reg.scope);
  }).catch(function(error) {
    console.log(‘Failed with ' + error);
  });
}
```

Offline page



Cached page with no images



2nd page load (3G)

Browser Cache

Developer Tools - https://godaytrip.com/categories

Elements | Network | Sources | Timeline | Profiles | Application | Security | Audits | Console | aXe

View: | Preserve log | Disable cache | Offline | Regular 3G (100ms, 75

Name	Status	Type	Initiator	Size	Time	Timeline – Start Time
categories	200 OK	docu...	Other	49.8 KB 48.9 KB	872 ms 294 ms	
application-95184... /assets	304 Not ...	styles...	categories:8 Parser	186 B 20.5 KB	124 ms 123 ms	
css?family=Source... fonts.googleapis.c...	200	styles...	categories:9 Parser	670 B 3.4 KB	127 ms 127 ms	
36df3011f564f14b... gravatar.com/avatar	304	jpeg	categories:85 Parser	349 B 4.5 KB	123 ms 122 ms	
application-1671f... /assets	304 Not ...	script	categories:3... Parser	186 B 127 KB	123 ms 120 ms	
analytics.js www.google-anal...	304	script	categories:3... Script	19 B 26.9 KB	626 ms 623 ms	
collect?v=1&_v=j4... www.google-anal...	200	gif	Other	63 B 35 B	344 ms 343 ms	

7 requests | 51.3 KB transferred | Finish: 1.94 s | DOMContentLoaded: 951 ms | Load: 1.94 s

With Service Worker

Developer Tools - https://godaytrip.com/service-worker.js

Elements | Network | Sources | Timeline | Profiles | Application | Security | Audits | Console | aXe

View: | Preserve log | Disable cache | Offline | Regular 3G (100ms, 75

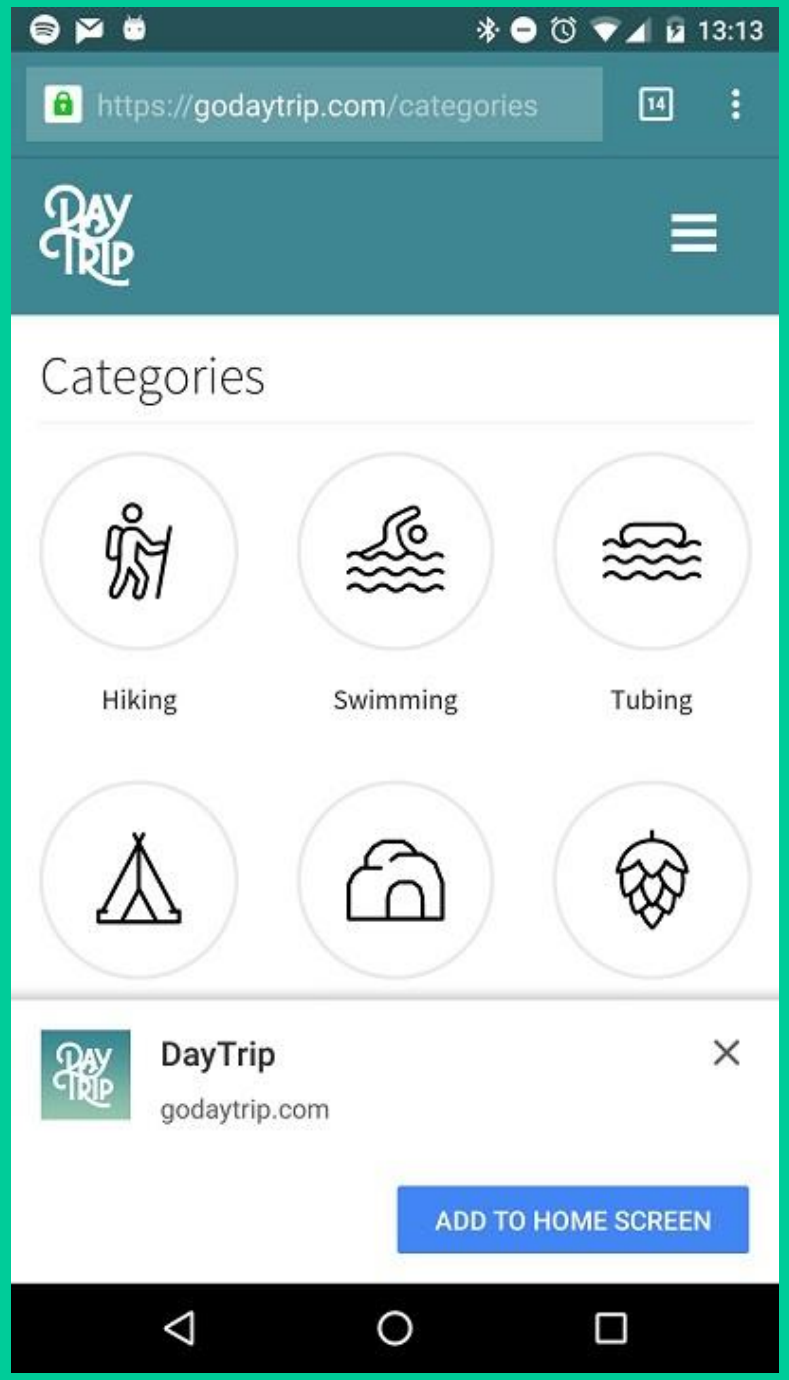
Name	Status	Type	Initiator	Size	Time	Timeline – Start Time
categories	200 OK	docu...	Other	(from...	665 ms 136 ms	
application-9518... /assets	200 OK	styles...	categories:8 Parser	(from...	12 ms 8 ms	
css?family=Sourc... fonts.googleapis....	200	styles...	categories:9 Parser	(from...	9 ms 9 ms	
36df3011f564f14... gravatar.com/ava...	200	jpeg	categories:85 Parser	(from...	8 ms 7 ms	
⚙ 36df3011f564f... gravatar.com/ava...	200	jpeg	Other	(from...	2 ms 1 ms	
application-1671f... /assets	200 OK	script	categories:3... Parser	(from...	17 ms 14 ms	
analytics.js www.google-anal...	200	script	categories:3... Script	(from...	137 ms 131 ms	
⚙ analytics.js	200	text/i...	Other	(from...	3 ms	

11 requests | 63 B transferred | Finish: 2.27 s | DOMContentLoaded: 743 ms | Load: 1.04 s

Add to Homescreen Banner

- Multiple visits
- > 2 minute session time
- ~5 minutes apart

Depends on browser's own heuristics.



The Rails Stuff

Asset pipelines, digest fingerprinting, and scope issues.

serviceworker.js.erb

```
function updateStaticCache() {
  return caches.open(version + staticCacheName)
    .then(function (cache) {
      return cache.addAll([
          '<%= url_to_stylesheet "application" %>'
          '<%= url_to_javascript "application" %>',
          '/offline'
    ]);
  });
};
```

Scope in Service Worker

Nope

- assets/
  - serviceworker.js
  - logo.png
- index.html
- offline.html
- manifest.json

Yep

- assets/
  - logo.png
- index.html
- offline.html
- manifest.json
- serviceworker.js

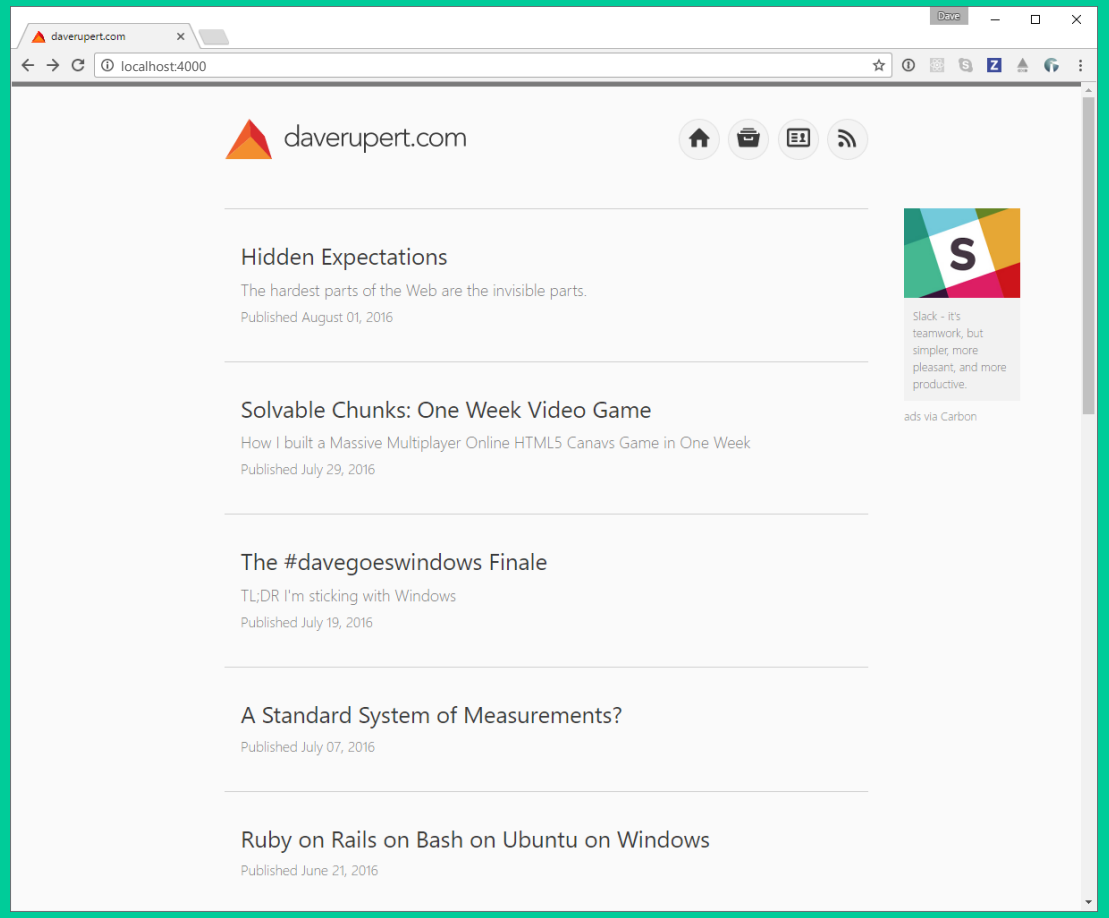
gem install serviceworker-rails

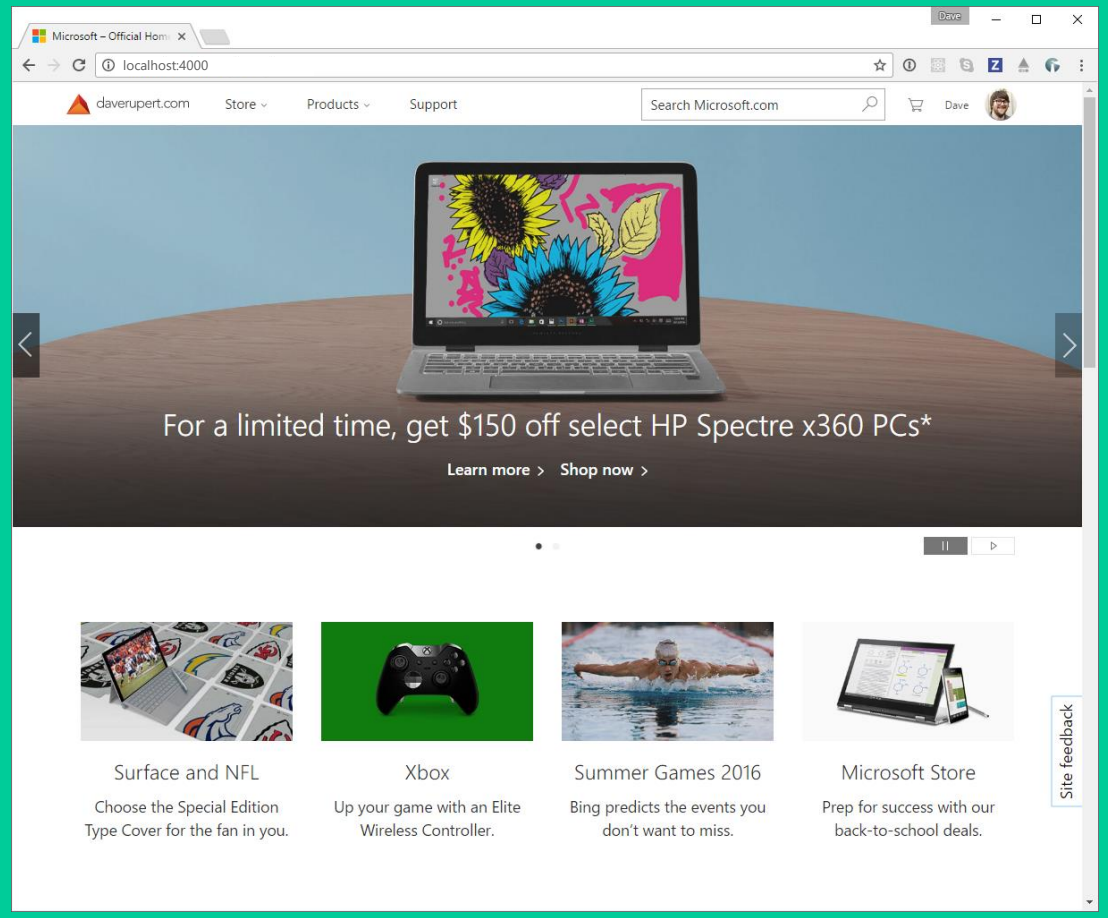
- Fixes Scope Issues
- Sets up route to Dynamic Service Worker in the root
- Sets appropriate Expires headers

Keep Localhost Weird

The… *ahem*… intricacies… of building with Service Workers

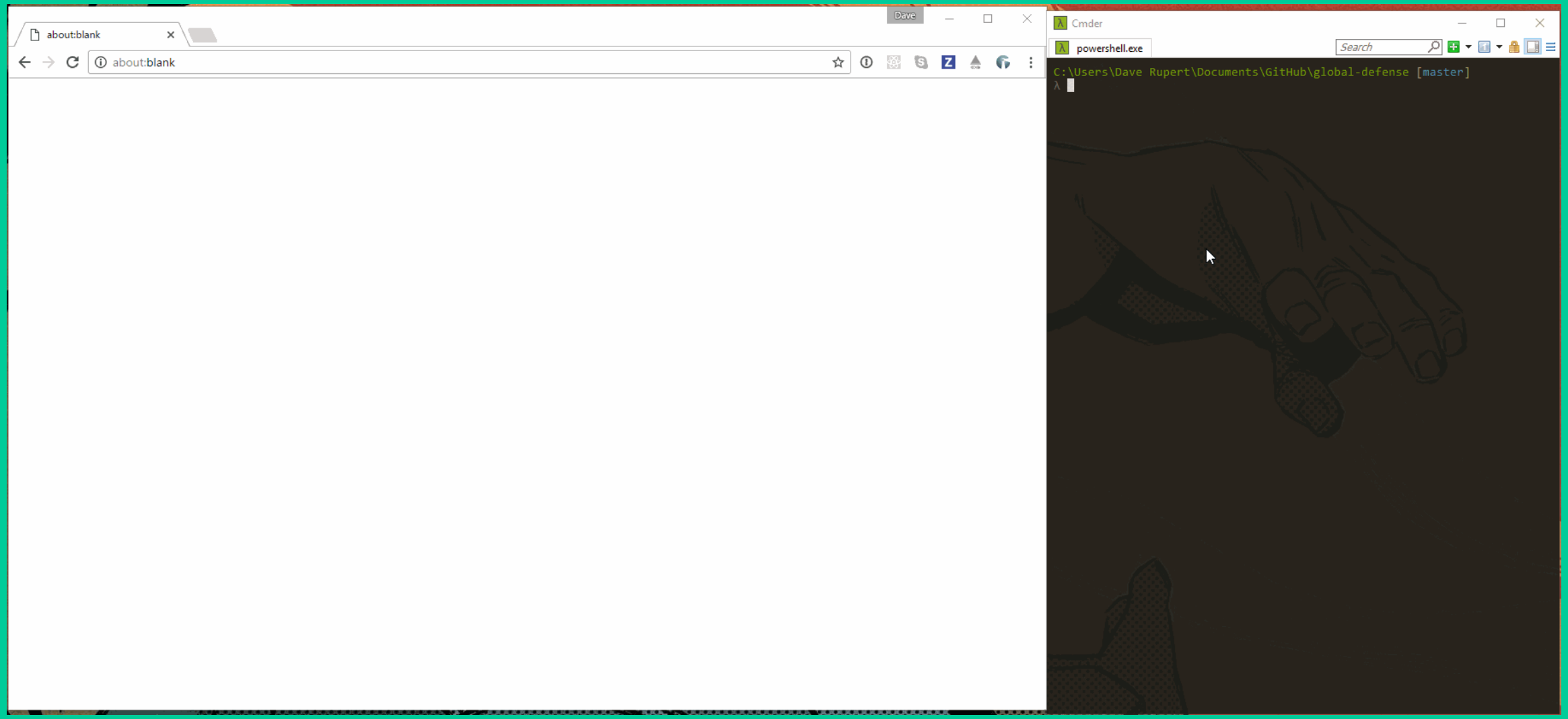
Multiple localhost:4000s





Ghosts...

“Localghost”


Dave
about:blank
about:blank
Cmder
powershell.exe
Search
C:\Users\Dave Rupert\Documents\GitHub\global-defense [master]
λ

100% UPTIME, BABY!

Enterprise Ruby on Rails, Finally!

The Near Future

Thanks!

@davatron5000