Progressive Web Apps...

Dave Rupert / @davatron5000
...on Ruby on Rails...
...at a JavaScript meetup!
WE DESIGN
WE CODE
WE LOVE
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...
Available Today
Web vs Native
Cats vs. Dogs, Right vs. Left, Pants vs. No pants
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.”
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?
“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

Service Workers

- Offline
- Update on reload
- Bypass for network

https://godaytrip.com/

Source: service-worker.js
Received: 8/15/2016, 10:13:26 PM
Status: #973 activated and is stopped (start)
Lighthouse

Progressive Web App
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
- URL responds with a 200 when offline
- Cache contains start_url from manifest

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)
  - First Visual Change: 2089ms
  - Last Visual Change: 2898ms
- Estimated Input Latency (target: 50ms)
  - (36.3ms) 99

92 / 100
The 3 parts of a Progressive Web App

manifest.json
offline.html
service-worker.js
(HTTPS too)
The Web App Manifest

manifest.json

```json
{
    "name": "DayTrip",
    "short_name": "DayTrip",
    "icons": [
        {
            "src": "apple-touch-icon-152x152-precomposed.png",
            "type": "image/png",
            "sizes": "152x152"
        },
        {
            "src": "touch-icon-192x192.png",
            "type": "image/png",
            "sizes": "192x192"
        }
    ],
    "start_url": "/?utm_source=web_app_manifest",
    "display": "standalone",
    "orientation": "portrait",
    "background_color": "#ffffff",
    "theme_color": "#3E8792"
}
```
<head>

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

offline.html
// 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 while
// offline)
function updateStaticCache() {
  return caches.open(version + staticCacheName)
    .then(function (cache) {
      return cache.addAll([  
        '<% url_to_stylesheets "application" %>',
        'https://fonts.googleapis.com/css?family=Source+Sans+Pro',
        '<% url_to_javascript "application" %>',
        '/offline'
      ]);});
}

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

self.addEventListener('activate', function (event) {
Wow. We’re almost done with our quest!?
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

---

**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
- fetch
- push
- sync
- message

**Functional events**
self.addEventListener(‘install’, function( event ) {
  event.waitUntil(
    //
    // Prime the caches
    //
  );
});
“Hey event, wait until the stuff in these parenthesis are done before you tell the other events you fired.”
self.addEventListener('activate', function( event ) {
  event.waitUntil(
    //
    // Clear out old caches
    //
  );
});
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

```javascript
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

```javascript
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

```javascript
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;
}

// 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;
})
// Step 2: Catch Stuff...

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

Looks like your device went offline.
Hiking

1. Enchanted Rock
   Fredericksburg 76 Miles

2. Colorado Bend State Park
   Bend 79 Miles
### Browser Cache

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Type</th>
<th>Initiator</th>
<th>Size</th>
<th>Time</th>
<th>Timeline – Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>categories</td>
<td>200</td>
<td>docu.</td>
<td>Other</td>
<td>49.8 KB</td>
<td>872 ms</td>
<td>1.50s – 1.50s</td>
</tr>
<tr>
<td>application-9518..</td>
<td>304</td>
<td>styles</td>
<td>categories</td>
<td>186 B</td>
<td>124 ms</td>
<td>1.25s – 1.32s</td>
</tr>
<tr>
<td>css?family=Source..</td>
<td>200</td>
<td>styles</td>
<td>categories</td>
<td>670 B</td>
<td>127 ms</td>
<td>1.25s – 1.35s</td>
</tr>
<tr>
<td>36df3011f564f41b..</td>
<td>304</td>
<td>jpeg</td>
<td>categories 5</td>
<td>349 B</td>
<td>123 ms</td>
<td>1.25s – 1.30s</td>
</tr>
<tr>
<td>application-1671f..</td>
<td>304</td>
<td>script</td>
<td>categories</td>
<td>186 B</td>
<td>123 ms</td>
<td>1.25s – 1.28s</td>
</tr>
<tr>
<td>analytics.js</td>
<td>304</td>
<td>script</td>
<td>categories 3</td>
<td>19 B</td>
<td>626 ms</td>
<td>1.25s – 1.35s</td>
</tr>
<tr>
<td>collecttv=1&lt;3_y+j4..</td>
<td>200</td>
<td>gif</td>
<td>Other</td>
<td>63 B</td>
<td>344 ms</td>
<td>1.25s – 1.34s</td>
</tr>
</tbody>
</table>

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

### With Service Worker

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Type</th>
<th>Initiator</th>
<th>Size</th>
<th>Time</th>
<th>Timeline – Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>categories</td>
<td>200</td>
<td>docu.</td>
<td>Other</td>
<td>48.9 KB</td>
<td>294 ms</td>
<td>1.20s – 1.20s</td>
</tr>
<tr>
<td>application-9518..</td>
<td>200</td>
<td>styles</td>
<td>categories</td>
<td>186 B</td>
<td>124 ms</td>
<td>1.20s – 1.27s</td>
</tr>
<tr>
<td>css?family=Source..</td>
<td>200</td>
<td>styles</td>
<td>categories</td>
<td>670 B</td>
<td>127 ms</td>
<td>1.20s – 1.30s</td>
</tr>
<tr>
<td>36df3011f564f41b..</td>
<td>200</td>
<td>jpeg</td>
<td>categories 5</td>
<td>349 B</td>
<td>123 ms</td>
<td>1.20s – 1.25s</td>
</tr>
<tr>
<td>application-1671f..</td>
<td>200</td>
<td>script</td>
<td>categories</td>
<td>186 B</td>
<td>123 ms</td>
<td>1.20s – 1.28s</td>
</tr>
<tr>
<td>analytics.js</td>
<td>200</td>
<td>script</td>
<td>categories 3</td>
<td>19 B</td>
<td>626 ms</td>
<td>1.20s – 1.30s</td>
</tr>
<tr>
<td>collecttv=1&lt;3_y+j4..</td>
<td>200</td>
<td>gif</td>
<td>Other</td>
<td>63 B</td>
<td>343 ms</td>
<td>1.20s – 1.33s</td>
</tr>
</tbody>
</table>

11 requests | 63.8 KB 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.
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”
100% UPTIME, BABY!

Enterprise Ruby on Rails, Finally!
The Near Future
Thanks!

@davatron5000