DEBUGGING FRONT-END PERFORMANCE

Tim Kadlec (@tkadlec) & Pat Meenan (@patmeenan)
June 20-21, 2016 at Velocity Conference (#velocityconf)
SERVICE WORKERS
RESOURCE FETCHING (IN CHROME)

DOM
RESOURCE FETCHING (IN CHROME)

DOM

In-memory Resource Cache
RESOURCE FETCHING (IN CHROME)

DOM -> Resource Fetcher

In-memory Resource Cache
RESOURCE FETCHING (IN CHROME)

DOM → Resource Fetcher → Net Stack

In-memory Resource Cache
RESOURCE FETCHING (IN CHROME)

- DOM
- Resource Fetcher
- Net Stack
- In-memory Resource Cache
- Disk Cache
RESOURCE FETCHING (IN CHROME)

DOM → Resource Fetcher → Net Stack → Net

In-memory Resource Cache

Disk Cache
RESOURCE FETCHING (IN CHROME)

DOM → Resource Fetcher → Service Worker → Net Stack → Net

- In-memory Resource Cache
- Disk Cache
• Sees every request for your document
  o Including cross-origin
  o And headers
• Can synthesize responses
• Supports fetch
• Has a programmable cache and Indexed DB
LIMITATIONS

- HTTPS documents only
- Not active for first view
- iFrames are separate documents
- Non-CORS Cross-origin responses are opaque
- No global state
  - or concept of a “page”
- No Sync API’s (localStorage, XHR, etc)
BROWSER SUPPORT

Details on partial support can be found on is ServiceWorker Ready?
MS Edge status: In Development
WebKit status: Under Consideration

http://caniuse.com/#feat=serviceworkers
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('/my-app/sw.js', {
    scope: '/my-app/
  });
}
• Honors caching headers
• Forced re-download every 24 hours
To install immediately:

```javascript
event.waitUntil(self.skipWaiting());
```
self.addEventListener('activate', function(event) {
    // You're good to go!
});

To activate immediately:
    event.waitUntil(self.clients.claim());
navigator.serviceWorker.controller.postMessage(message);

In Service Worker:

self.addEventListener('message', function(event) {
});
self.addEventListener('fetch', function(event) {
  event.respondWith(new Response("Hello world!"));
});
IMPORTS

self.importScripts('foo.js');
self.importScripts('https://ads/offlineads.js');
Some Use Cases
OFFLINE

• Chrome New Tab Page
• Flipkart
CUSTOM ERROR PAGES

• Fetch Request Normally
• For non-200 responses serve a local error page
• Not just server errors:
  o DNS failures
  o CDN/Proxy Errors
  o Intermediaries
SPOF PREVENTION / SLA

• Identify requests of interest
  o 3rd-party javascript
  o Fonts
• Set a timer
• Pass fetch requests through
• If timer expires before fetch completes generate error response
CDN / ORIGIN FAILOVER

- Identify CDN requests by URL
- Set a timer
- Pass fetch request through
- If timer expires, create fetch request to origin
- Respond with first fetch request to complete
MULTI ORIGIN / CDN

- Identify CDN requests by URL
- Replace CDN domain with alternate CDN (or origin)
- Keep track of performance by origin
- Prefer faster origin (keep measuring)

- Could also race the CDNs in parallel
  - Be careful about increased data
STALE-WHILE-REVALIDATE

• Respond with local cached resource
  o Ignoring Expires, max-age, etc.
• Pass fetch request through in parallel
• Update cache with new response

• Works best for known resources (analytics/ads JS, etc)
• Custom response headers with prefetch resources
• When idle, prefetch suggested resources
DELTA COMPRESSION

- Delta compression for build->build updates
  - Include version in URL scheme
  - Get latest version number from cache
  - Request delta from server
  - Apply patch
  - Cache new version

- New compression algorithms
CUSTOM COMPRESSION

- New compression algorithms
  - Brotli
  - Fractal image compression
  - JSON-specific dictionaries
  - Application-specific
- Prove-out new algorithms before standardizing
- …as long as it can be implemented in JS and code size is reasonable
INCREMENTAL PROGRESSIVE IMAGES

- Identify JPEG image requests from known origin
- Synthesize response
- Range request (or smarter) for first few scans
- Stream initial range into synthesized response
- Range request for remaining image (some point later)
- Append remaining data into synthesized response
DRAWING IMAGES LOCALLY

- 930 x 11,362 px WebPageTest waterfall
- 351KB compressed PNG
- 42 MB in-memory (server) to generate
- < 20KB compressed JSON data to describe

- Prefer actual images for WPT waterfalls for easier embedding
  - Otherwise SVG, Canvas or DOM would work
Drewing Images Locally

- Identify appropriate requests (i.e. waterfall.png?...)
- Fetch data necessary to draw image
- Draw to canvas
- Extract as PNG
- Synthesize PNG response
METRICS / DEBUGGING

• Pass all requests through
• Passively observe timings and success
• Report metrics back
• Gives visibility into failures and requests that are in flight
  ○ Unlike resource timing
Background Sync

WebRTC Data Channels?
  - Custom protocol/fetching
  - GeoFencing?
  - Required for:
    - Push Notifications
Debugging
var EXTRA_FILES = [
  '/xjs/_%js/k-xjs.ntp.en_US.aS8.avtrib1.0/m-jsa,ntp,d,csi/r
];
var CHECKSUM = "vm9q2s";
var BLACKLIST = [
  '/gen_20431/",
  '/async/',
];

var FILES = [
  '/images/spr/logo11w.png',
  '/images/spr/chrome_nto_white_logo2.png',
  '/ssl.gstatic.com/chrome/components/doodle-notif
].concat(EXTRA_FILES || []);

var CACHENAME = 'newtab-static-' + CHECKSUM;

self.addEventListener('install', function(event) {
  event.waitUntil(caches.open(CACHENAME).then(function(cache)
    return cache.addAll(FILES);
  ));
});

self.addEventListener('activate', function(event) {
  return event.waitUntil(caches.keys().then(function(keys)
    return Promise.all(keys.map(function(k) {
      if (k.indexOf('newtab-static-') == 0)
        caches.delete(k);
      else {
        return Promise.resolve();
      }
    }));
  ));
});

//
### Service Workers

**www.google.com//chrome/**

- **State**: activated
- **Worker**: running
- **Script URL**: chrome/newtab-service-worker.js
- **Updated**: 2016-06-10 13:45:56.255

**Recent messages**
- Server response time: 2016-06-10 13:45:56.255

**Controlled clients**
- Tab: chrome/newtab/ focus
THANK YOU!

Tim Kadlec (@tkadlec) & Pat Meenan (@patmeenan)
June 20-21, 2016 at Velocity Conference (#velocityconf)

Icons courtesy of The Noun Project