INTELLIGENT WEBSITES WITH SERVICE WORKERS

Bron Thulke

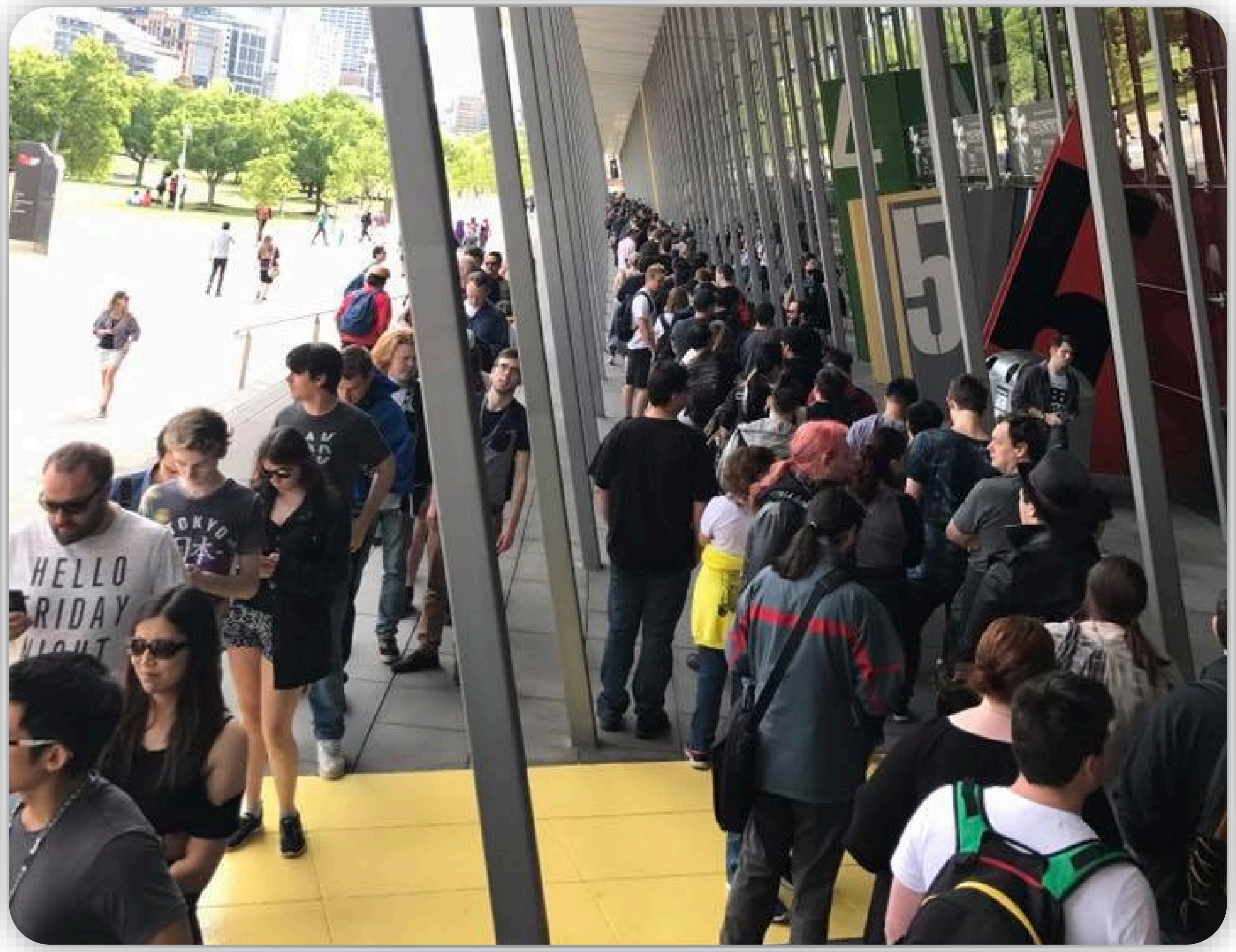
WHO AM I?

Bron Thulke

- Web Developer at Angel Web Designs
- Mostly Microsoft stack
- Never enough time in the day to learn all the things
- Service workers are my new "ooooh" thing

Twitter: @_bron_

Blog: http://blog.angelwebdesigns.com.au



A TYPICAL SITUATION

Information Portal style website

Accessed where no internet connectivity

Need to keep in touch with users over time

Can't assume users are on the website at the time

Example: an event website!

LET'S BUILD A BETTER WEB

CACHED!

- ✓ Increased performance
- ✓ Offline access

SMART!

- ✓ Aware of current connectivity
- ✓ Working in the background

INVOLVING!

- ✓ Notifying users of information
- ✓ Giving users control over cache and script updates

INTRODUCING SERVICE WORKERS

What is a service worker?

- Basically, a JavaScript script that can run without the web page

What can we do with service workers?

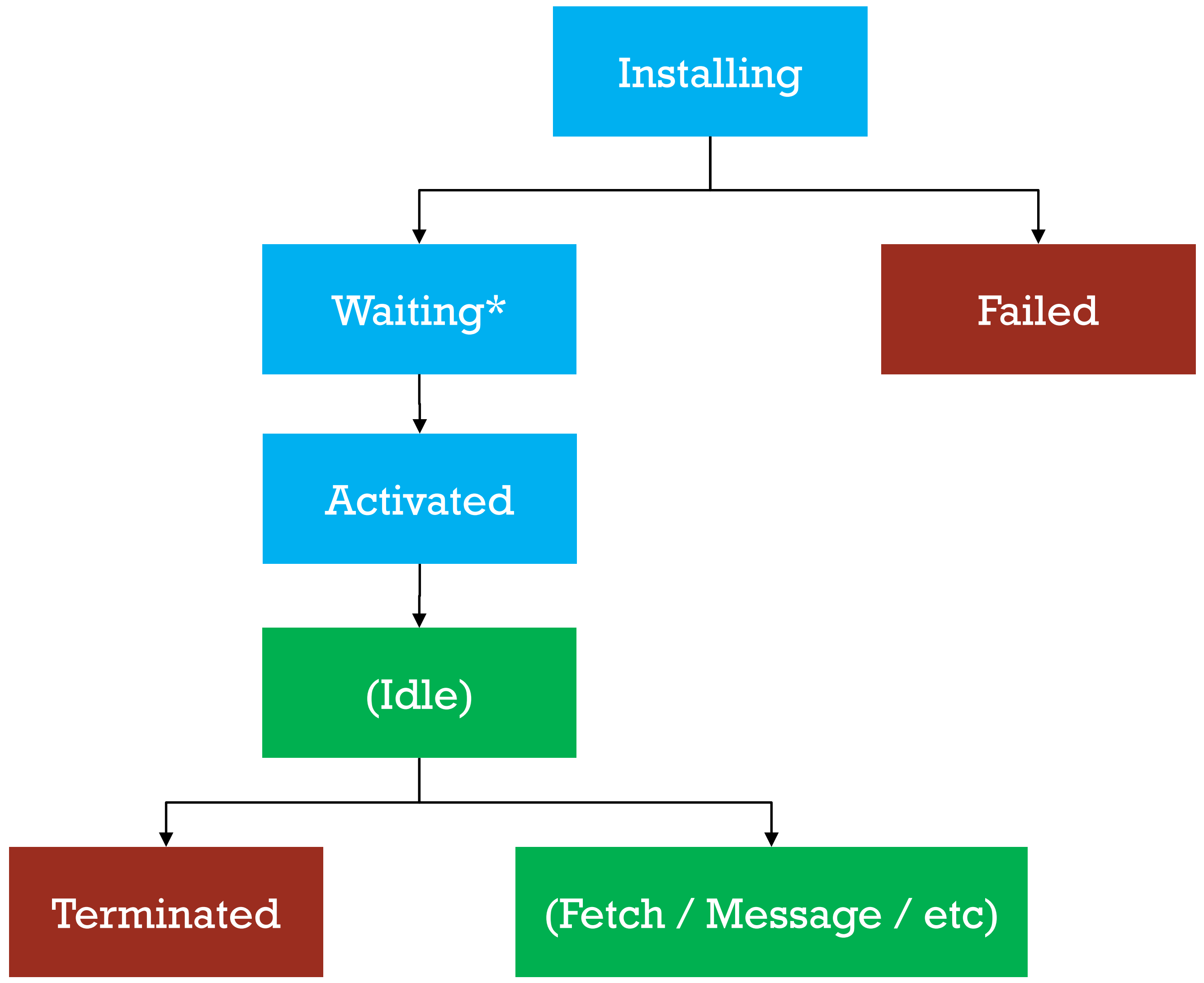
- Build progressive websites
- Cache resources
- Background sync
- Push notifications to our users

REGISTERING A SERVICE WORKER

In your main JS code

```
if ('serviceWorker' in navigator) {
    navigator.serviceWorker.register('/sw.js').then(function(registration) {
       console.log('ServiceWorker registration successful');
    }).catch(function(err) {
       console.log('ServiceWorker registration failed: ', err);
    });
}
```

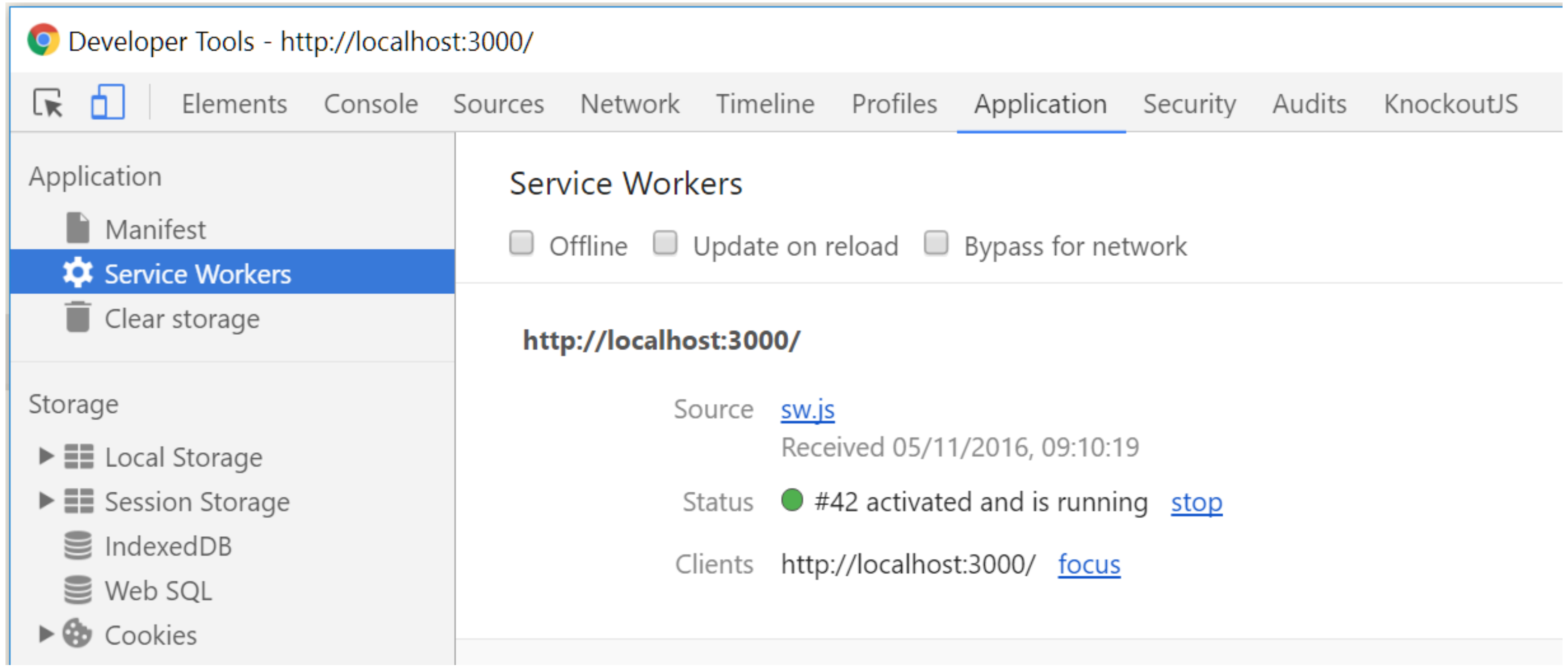
- Check for browser compatibility
- Register the service worker
- Use the Promise API to handle the result



LIFECYCLE OF A SERVICE WORKER

Service workers get installed into the browser.

By default, they do not start "controlling" a page until the next restart.


Developer Tools - http://localhost:3000/

Service Workers

☐ Offline ☐ Update on reload ☐ Bypass for network

http://localhost:3000/

Source	sw.js Received 05/11/2016, 09:10:19
Status	● #42 activated and is running stop
Clients	http://localhost:3000/ focus

MANAGING USING CHROME DEV TOOLS

Chrome has strong development tools for managing Service Workers.

Firefox tools are also getting better.

If in doubt, delete and start again!

Cache API

The Cache API offers simple methods such as:

caches.open ()

caches.addAll ()

caches.keys()

caches.match()

Set up cached files

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

  event.waitUntil(
    caches.open(CACHE_NAME)
      .then(function(cache) {
        return cache.addAll(urlsToCache);
      })
  );
});
```

Clear old cache files

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

  var cacheWhitelist = ['pages-cache-v1', 'blog-posts-cache-v1'];

  event.waitUntil(
    caches.keys().then(function(cacheNames) {
      return Promise.all(
        cacheNames.map(function(cacheName) {
          if (cacheWhitelist.indexOf(cacheName) === -1) {
            return caches.delete(cacheName);
          }
        })
      );
    })
  );
});
```

FETCH EVENTS

Simple handling of request using cache

```
self.addEventListener('fetch', function(event) {
  event.respondWith(
    caches.match(event.request)
      .then(function(response) {
        // Cache hit - return response
        if (response) {
          return response;
        }
        return fetch(event.request);
      }
    )
  );
});
```

Handle the "fetch" event to return an item from the cache or call out to the network.

You could return anything you like here.

To also cache new requests, be aware of the need to clone your requests and responses, so not consumed by your extra uses.

Good to also handle errors (404 etc).

PUSH AND NOTIFICATIONS API'S

Offers handling of push messages and notifications when site is in foreground OR background

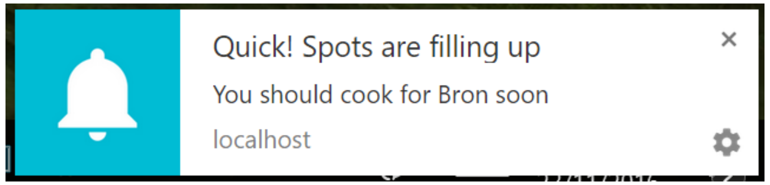
Requires the use of a messaging service for the client to pick it up (e.g. Google Cloud Messaging with Android/Chrome)

Requires HTTPS (or localhost)

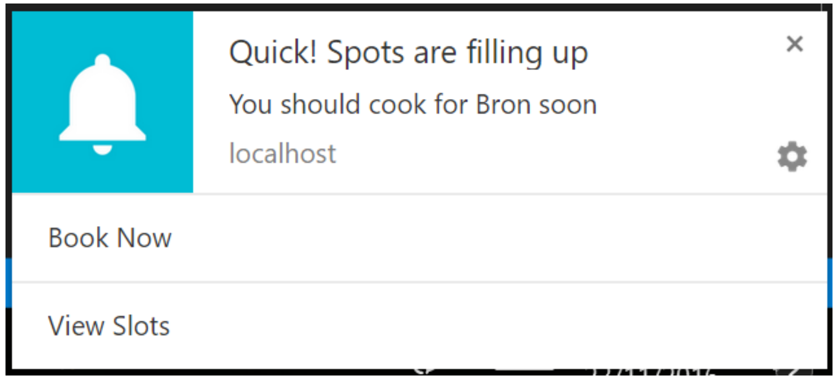
Handle push messages and notify user

```
self.addEventListener("push", function(event) {

  var title = 'Quick! Spots are filling up';
  const options = {
    body: 'You should cook for Bron soon',
    icon: '/content/images/icon.png',
    badge: '/content/images/badge.png'
  };

  event.waitUntil(self.registration.showNotification(title, options));
});
```

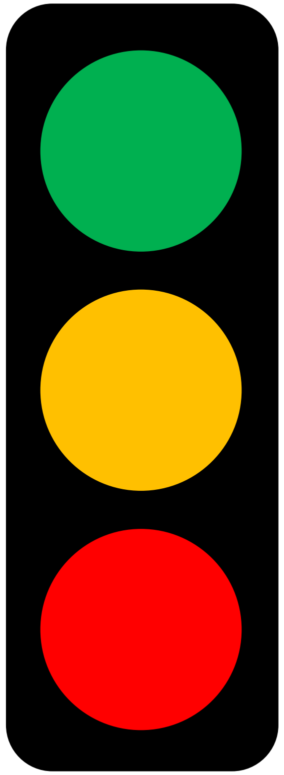
Good


Quick! Spots are filling up
You should cook for Bron soon
localhost

BETTER!


Quick! Spots are filling up
You should cook for Bron soon
localhost

BROWSER COMPATIBILITY



Chrome
Firefox
Opera

Microsoft Edge

Safari
Internet Explorer

Check out: https://jakearchibald.github.io/isserviceworkerready/

KEEP IN MIND…

- This is cutting edge!
- Dev tools break
- HTTPS required
- If in doubt: debug your service worker

THANKS!

Read more about Service Workers:

- https://developers.google.com/web/fundamentals/getting-started/primers/service-workers
- https://developers.google.com/web/fundamentals/getting-started/codelabs/
- https://jakearchibald.github.io/isserviceworkerready/
- And everything else by Jake Archibald

Demo site:

- https://bronthulke.github.io/service-workers-demo

Feel free to connect with me:

Twitter:	@_bron_
LinkedIn:	https://au.linkedin.com/in/bronsquire