This site can’t be reached

OFFLINE FIRST

web applications

Nir Kaufman

Nir Kaufman

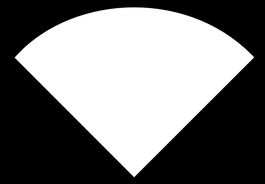
Google Developer Expert

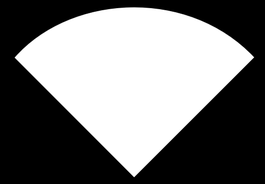
Community leader

Principle Consultant @ **500Tech**

Easy to get started

WHY OFFLINE ?





Above Ground

Under Ground

On Ground

WE NEED **SPACE.**

Elements
Console
Sources
Network
Performance
Memory
Application
Security
Audits
Augury
AdBlock
Application
Manifest
Service Workers
Clear storage
Storage
Local Storage
Session Storage
IndexedDB
Web SQL
Cookies
Cache
Cache Storage
Application Cache
Frames
top
Clear storage
https://jakearchibald.com
Usage
112 KB used out of 15278 MB storage quota
112 KB
108 KB Cache Storage
4.2 KB Service Workers
Application
Unregister service workers
Storage
Local and session storage
IndexedDB
Web SQL
Cookies
Cache
Cache storage
Application cache
Chrome devtools

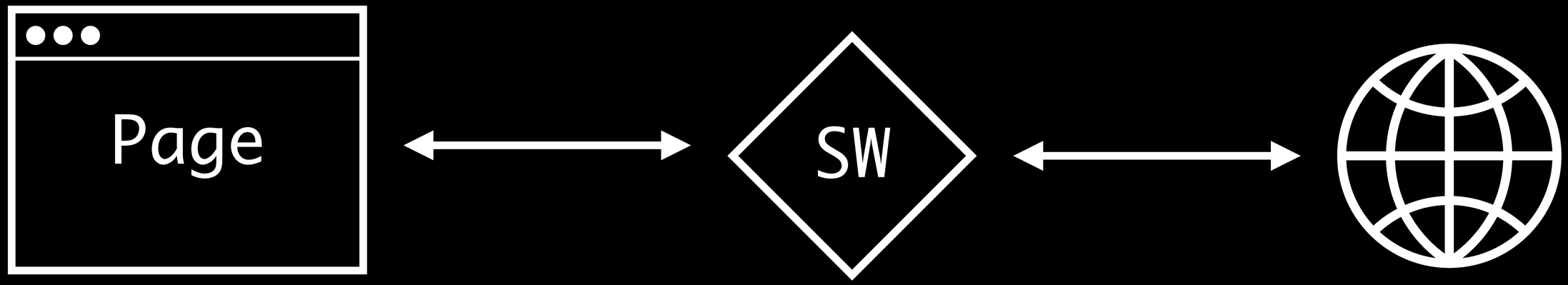
```
navigator.storage.estimate()
```

SERVICE **WORKER**

script that your browser runs in the background

http://caniuse.com/#search=ServiceWorkers

Service worker as a proxy server



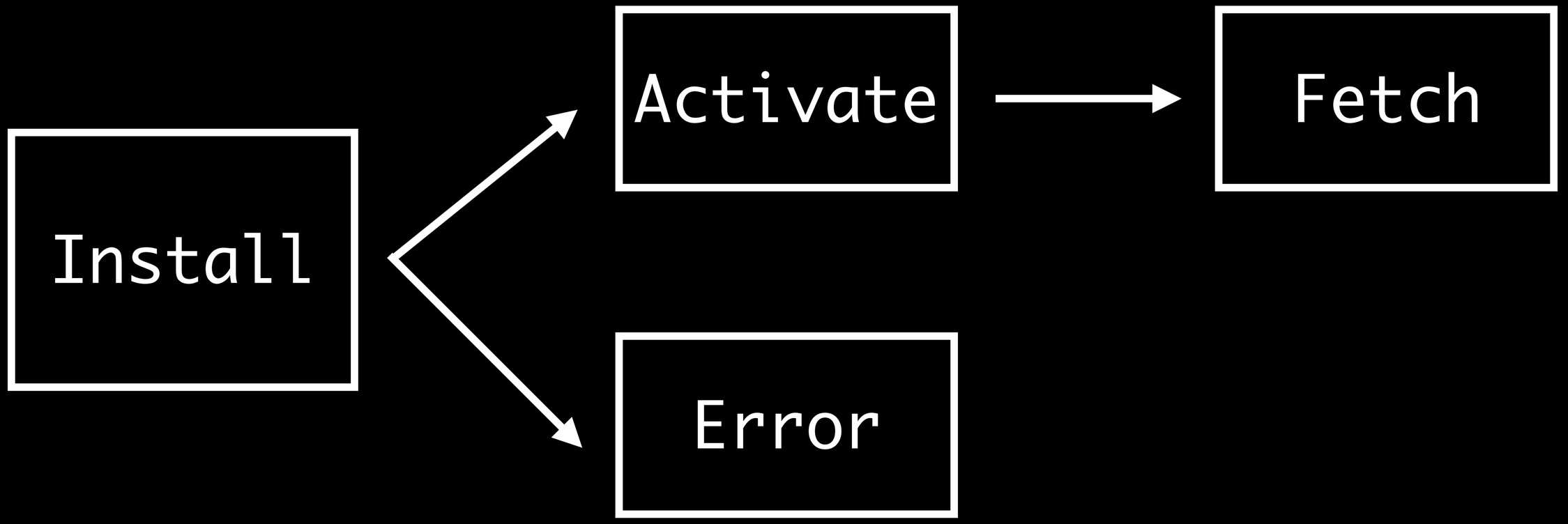
Create new file

```
self.addEventListener('fetch', function (event) {
  console.log(event.request);
});
```

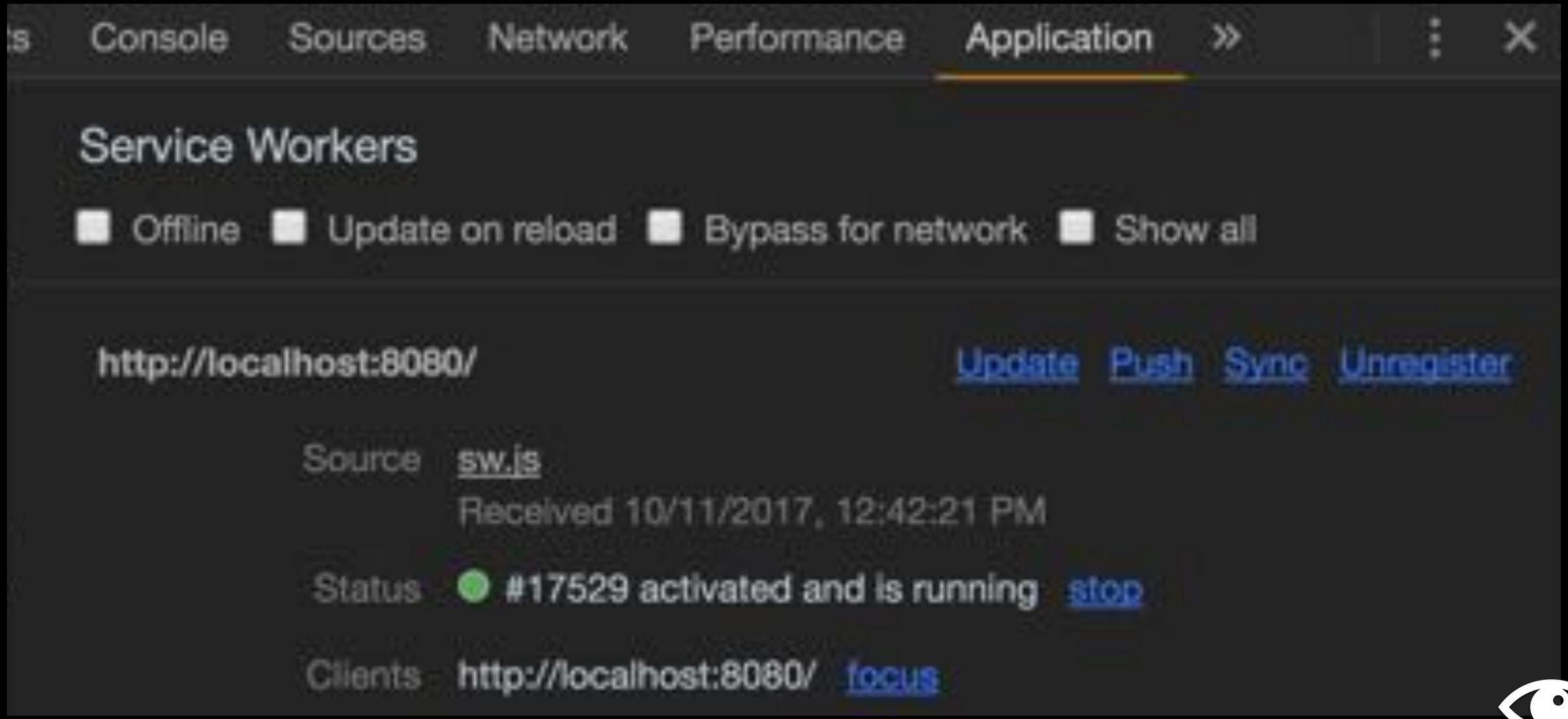
Register it

```
navigator.serviceWorker.register('/sw.js')
```

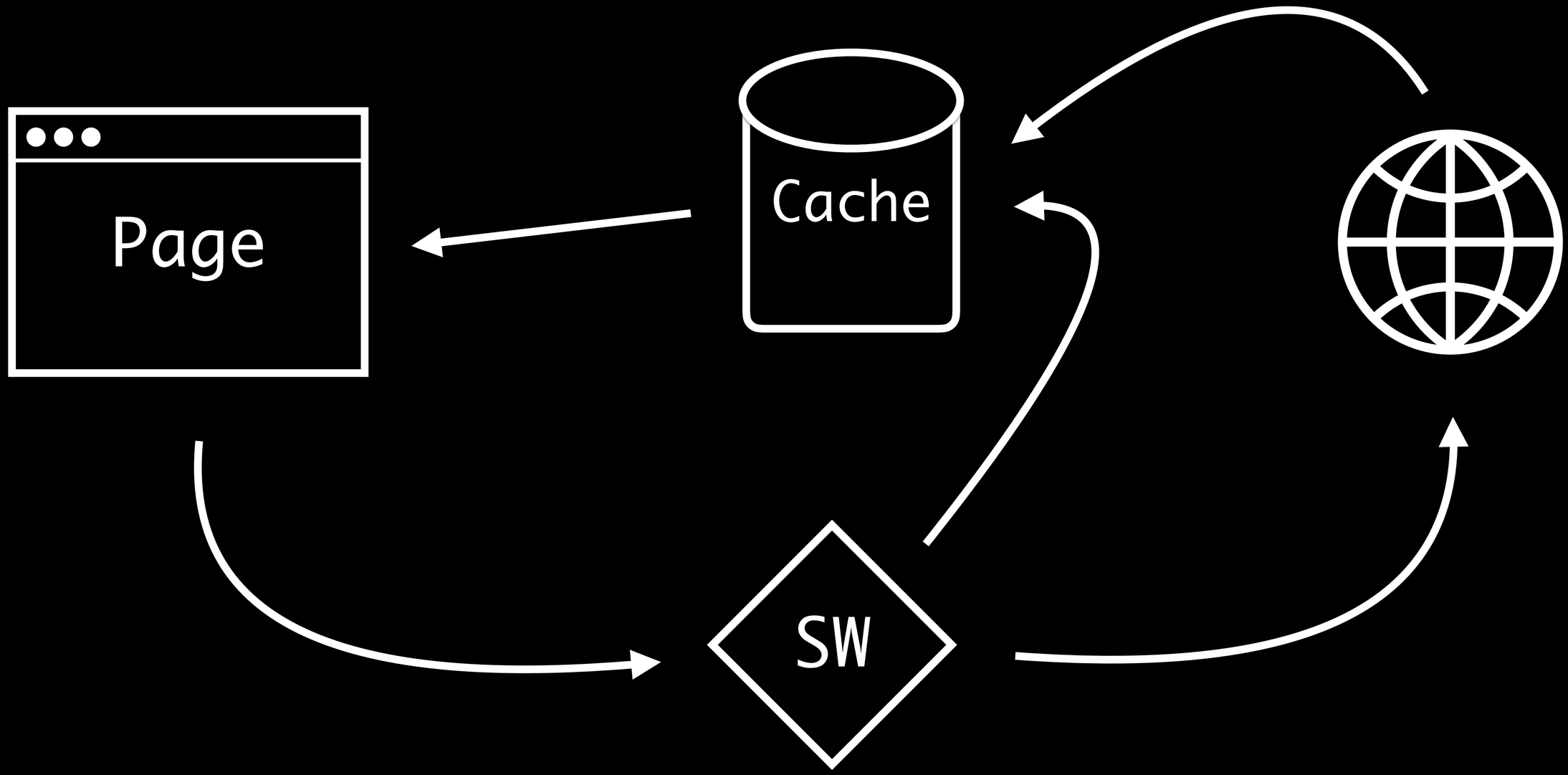
Service worker life cycle



Chrome devtools



Cache machine



Gaining control

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    new Response('SW control your page!’)
);
```

Hijacking Requests

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    new Response('SW control your page!’)
);
```

Hijacking Requests

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    new Response('SW control your page!’)
);
```

Basic proxy

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    fetch(event.request)
      .then(function (response) {
        return response;
      })
      .catch(function () {
        return new Response("Something went wrong...");
      })
  )
});
```

Basic proxy

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    fetch(event.request)
      .then(function (response) {
        return response;
      })
      .catch(function () {
        return new Response("Something went wrong...");
      })
  )
});
```

Basic proxy

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    fetch(event.request)
      .then(function (response) {
        return response;
      })
      .catch(function () {
        return new Response("Something went wrong...");
      })
  )
});
```

CACHE
INTREFACE

provides a storage mechanism for Request / Response object pairs

provides a **storage** mechanism for **Request / Response** object pairs

Use a cache

```
self.addEventListener('install',(event) => {
  event.waitUntil(
    caches.open(‘STATIC_TODO')
        .then((cache) => {
         return cache.addAll([
           // any url you want to cache
         ])
    })
  )
})
```

Use the cache

```
self.addEventListener('install',(event) => {
  event.waitUntil(
    caches.open(‘STATIC_TODO')
        .then((cache) => {
          return cache.addAll([
            // any url you want to cache
          ])
    })
  )
})
```

Use a cache

```
self.addEventListener('install',(event) => {
  event.waitUntil(
    caches.open(‘STATIC_TODO')
        .then((cache) => {
          return cache.addAll([
            // any url you want to cache
          ])
    })
  )
})
```

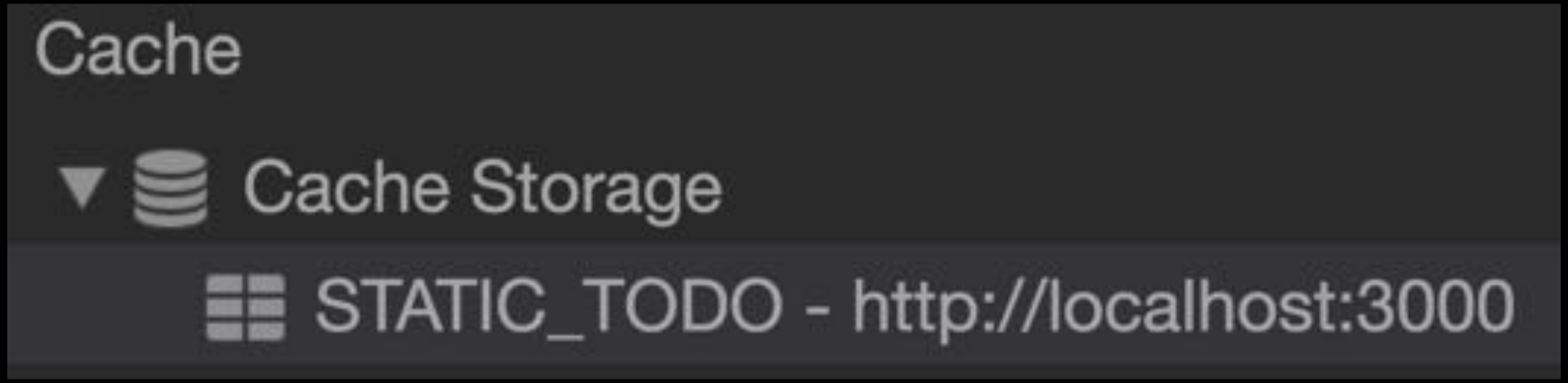
Use a cache

```
self.addEventListener('install',(event) => {
  event.waitUntil(
    caches.open(‘STATIC_TODO')
        .then((cache) => {
          return cache.addAll([
            // any url you want to cache
          ])
    })
  )
})
```

Use a cache

```
self.addEventListener('install',(event) => {
  event.waitUntil(
    caches.open(‘STATIC_TODO')
        .then((cache) => {
          return cache.addAll([
            // any url you want to cache
          ])
    })
  )
})
```

Check the devtools



Refactoring our fetch function

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    caches.match(event.request)
      .then(function (response) {
        return response ? response : fetch(event.request);
      })
  )
});
```

Refactoring our fetch function

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    caches.match(event.request)
      .then(function (response) {
        return response ? response : fetch(event.request);
      })
  )
});
```

Refactoring our fetch function

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    caches.match(event.request)
      .then(function (response) {
        return response ? response : fetch(event.request);
      })
  )
});
```

Refactoring our fetch function

```
self.addEventListener('fetch', (event) => {
  event.respondWith(
    caches.match(event.request)
      .then(function (response) {
        return response ? response : fetch(event.request);
      })
  )
});
```

INDEXED **DB**

low-level API for client-side storage

```
> npm install idb
```

Create a database

```
const db = idb.open('MVC_DB', 1, (upgradeDb) => {
  const todosStore =
        upgradeDb.createObjectStore('todos');
})
```

Create and init a database

```
const db = idb.open('MVC_DB', 1, (upgradeDb) => {
  const todosStore =
        upgradeDb.createObjectStore('todos');
})
```

Create and init a database

```
const db = idb.open('MVC_DB', 1, (upgradeDb) => {
  const todosStore =
        upgradeDb.createObjectStore('todos');
})
```

Check the devtools

Adding data

```
db.then(function (db) {
  const transaction = db.transaction('todos', 'readwrite');
  const todoStore = transaction.objectStore('todos');

  todoStore.put('world', 'hello');
  return transaction.complete;
})
  .then(function () {
    console.log('transaction completed')
  });
```

Adding data

```
db.then(function (db) {
  const transaction = db.transaction('todos', 'readwrite');
  const todoStore = transaction.objectStore('todos');

  todoStore.put('world', 'hello');
  return transaction.complete;
})
  .then(function () {
    console.log('transaction completed')
  });
```

Adding data

```
db.then(function (db) {
  const transaction = db.transaction('todos', 'readwrite');
  const todoStore = transaction.objectStore('todos');

  todoStore.put('world', 'hello');
  return transaction.complete;
})
  .then(function () {
    console.log('transaction completed')
  });
```

Adding data

```
db.then(function (db) {
  const transaction = db.transaction('todos', 'readwrite');
  const todoStore = transaction.objectStore('todos');

  todoStore.put('world', 'hello');
  return transaction.complete;
})
  .then(function () {
    console.log('transaction completed')
  });
```

Adding data

```
db.then(function (db) {
  const transaction = db.transaction('todos', 'readwrite');
  const todoStore = transaction.objectStore('todos');

  todoStore.put('world', 'hello');
  return transaction.complete;
})
  .then(function () {
    console.log('transaction completed')
  });
```

Adding data

```
db.then(function (db) {
  const transaction = db.transaction('todos', 'readwrite');
  const todoStore = transaction.objectStore('todos');

  todoStore.put('world', 'hello');
  return transaction.complete;
})
  .then(function () {
    console.log('transaction completed')
  });
```

Reading data

```
db.then(function (db) {
  const transaction = db.transaction('todos');
  const todoStore = transaction.objectStore('todos');

  return todoStore.get('hello');
})
  .then(function (value) {
    console.log('the value for hello is: ' + value);
  });
```

Reading data

```
db.then(function (db) {
  const transaction = db.transaction('todos');
  const todoStore = transaction.objectStore('todos');

  return todoStore.get('hello');
})
  .then(function (value) {
    console.log('the value for hello is: ' + value);
  });
```

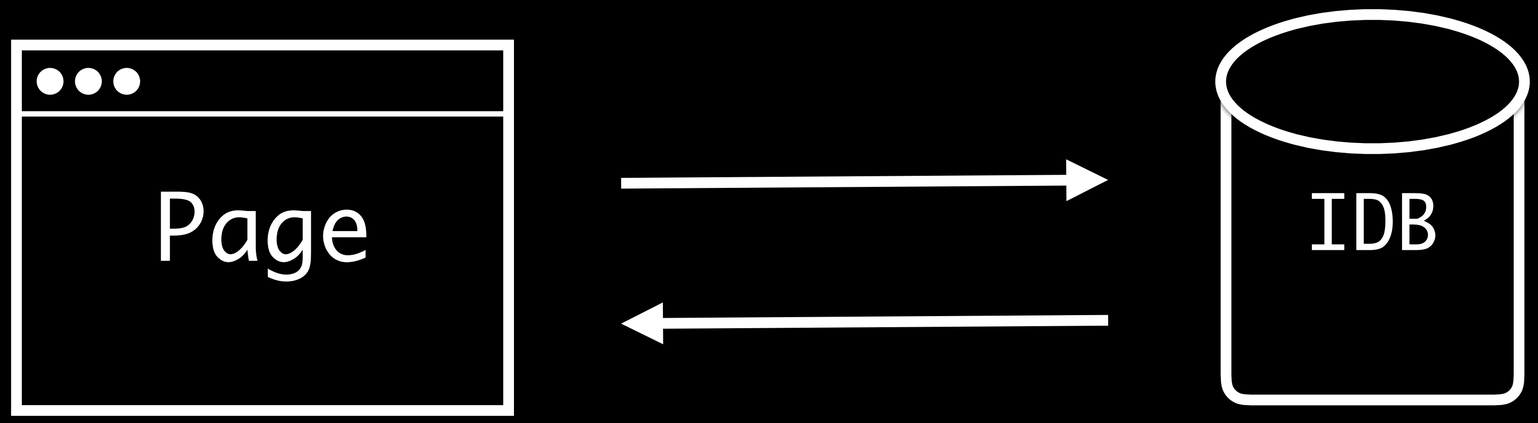
Reading data

```
db.then(function (db) {
  const transaction = db.transaction('todos');
  const todoStore = transaction.objectStore('todos');

  return todoStore.get('hello');
})
  .then(function (value) {
    console.log('the value for hello is: ' + value);
  });
```

Reading data

```
db.then(function (db) {
  const transaction = db.transaction('todos');
  const todoStore = transaction.objectStore('todos');

  return todoStore.get('hello');
})
  .then(function (value) {
    console.log('the value for hello is: ' + value);
  });
```

Using a local db



NEXT **STEPS**

THE CODE.

github.com/demo-projects/offline-todo

UDACITY

FREE COURSE

Offline Web Applications by Google

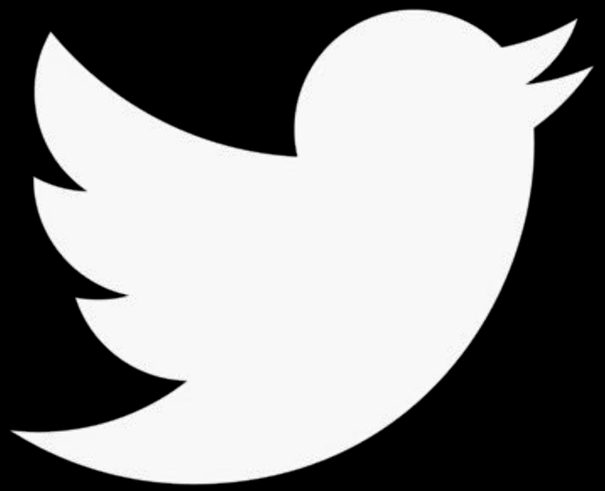
Using IndexedDB & Service Worker

START FREE COURSE

Workbox

JavaScript Libraries for Progressive Web Apps

https://workboxjs.org/



ME.
@nirkaufman

nirkaufman@gmail.com