Jayan Karkera **LAB 8** **D15B - 28**

Service Worker Lifecycle

A service worker progresses through three key phases in its lifecycle:

1. Registration
2. Installation
3. Activation

Registration

Registration is the initial step where you inform the browser about your service worker. This process tells the browser where to find your service worker file and initiates its background installation. Here's an implementation example:

```
// app.js
if ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('./service-worker.js')
    .then(function(registration) {
      console.log('Service Worker registered successfully with scope:',
registration.scope);
    })
    .catch(function(error) {
      console.log('Service Worker registration failed:', error);
    });
}
```

This code first checks if the browser supports service workers by testing for `navigator.serviceWorker`. It then registers the service worker using `navigator.serviceWorker.register()`, which returns a promise. On successful registration, it logs the scope, which defines which files the service worker can control. If registration fails, the error is logged.

By default, the service worker's scope is its location and all subdirectories. For example, if your service worker is in the root directory, it controls requests for all files on that domain.

You can also specify a custom scope:

```
// app.js
navigator.serviceWorker.register('./service-worker.js', {
  scope: '/pages/'
});
```

Installation

When a service worker is registered, it triggers an installation event. You can listen for this event to perform tasks during installation, such as precaching resources for offline use:

```
// service-worker.js
self.addEventListener('install', function(event) {
  // Perform installation tasks
});
```

During installation, service workers often cache essential files to enable offline functionality and improve loading performance on subsequent visits.

Activation

After successful installation, the service worker enters the activation phase. If any pages are still controlled by a previous service worker, the new one enters a waiting state. It only activates when all pages using the old service worker are closed. This ensures only one service worker version runs at a time.

```
// service-worker.js
self.addEventListener('activate', function(event) {
  // Perform activation tasks like clearing old caches
});
```

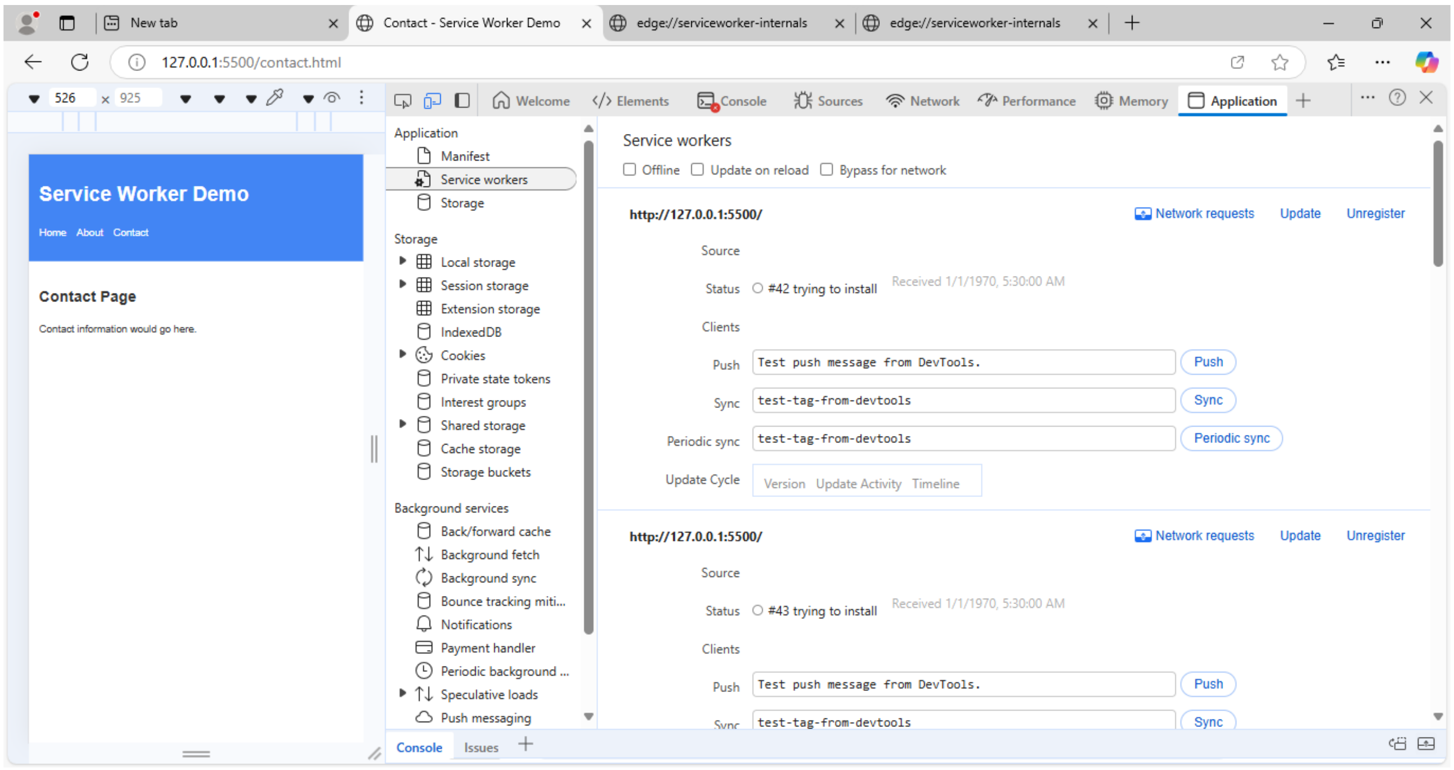
Code

// service-worker.js

```
const cacheName = 'expense-tracker-v1';
const filesToCache = [
  '/',
  '/index.html',
  '/styles.css',
  '/app.js',
  '/manifest.json',
  '/icons/icon-192.png',
  '/icons/icon-512.png'
];

// Install service worker
self.addEventListener('install', (e) => {
  e.waitUntil(
    caches.open(cacheName).then((cache) => {
      return cache.addAll(filesToCache);
    })
  );
});

// Activate service worker
self.addEventListener('activate', (e) => {
  console.log('Service Worker activated');
});

// Fetch data
```

```
self.addEventListener('fetch', (e) => {
  e.respondWith(
    caches.match(e.request).then((response) => {
      return response || fetch(e.request);
    })
  );
});
```

Output



New tab | Contact - Service Worker Demo | edge://serviceworker-internals | edge://serviceworker-internals

127.0.0.1:5500/contact.html

526 x 925

Service Worker Demo

Home About Contact

Contact Page

Contact information would go here.

Welcome | Elements | Console | Sources | Network | Performance | Memory | Application

Application
- Manifest
- Service workers
- Storage

Storage
- Local storage
- Session storage
- Extension storage
- IndexedDB
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage
- Storage buckets

Background services
- Back/forward cache
- Background fetch
- Background sync
- Bounce tracking miti...
- Notifications
- Payment handler
- Periodic background ...
- Speculative loads
- Push messaging

Service workers

☐ Offline ☐ Update on reload ☐ Bypass for network

http://127.0.0.1:5500/ — Network requests | Update | Unregister

Source

Status ○ #42 trying to install — Received 1/1/1970, 5:30:00 AM

Clients

Push `Test push message from DevTools.` [Push]

Sync `test-tag-from-devtools` [Sync]

Periodic sync `test-tag-from-devtools` [Periodic sync]

Update Cycle — Version Update Activity Timeline

http://127.0.0.1:5500/ — Network requests | Update | Unregister

Source

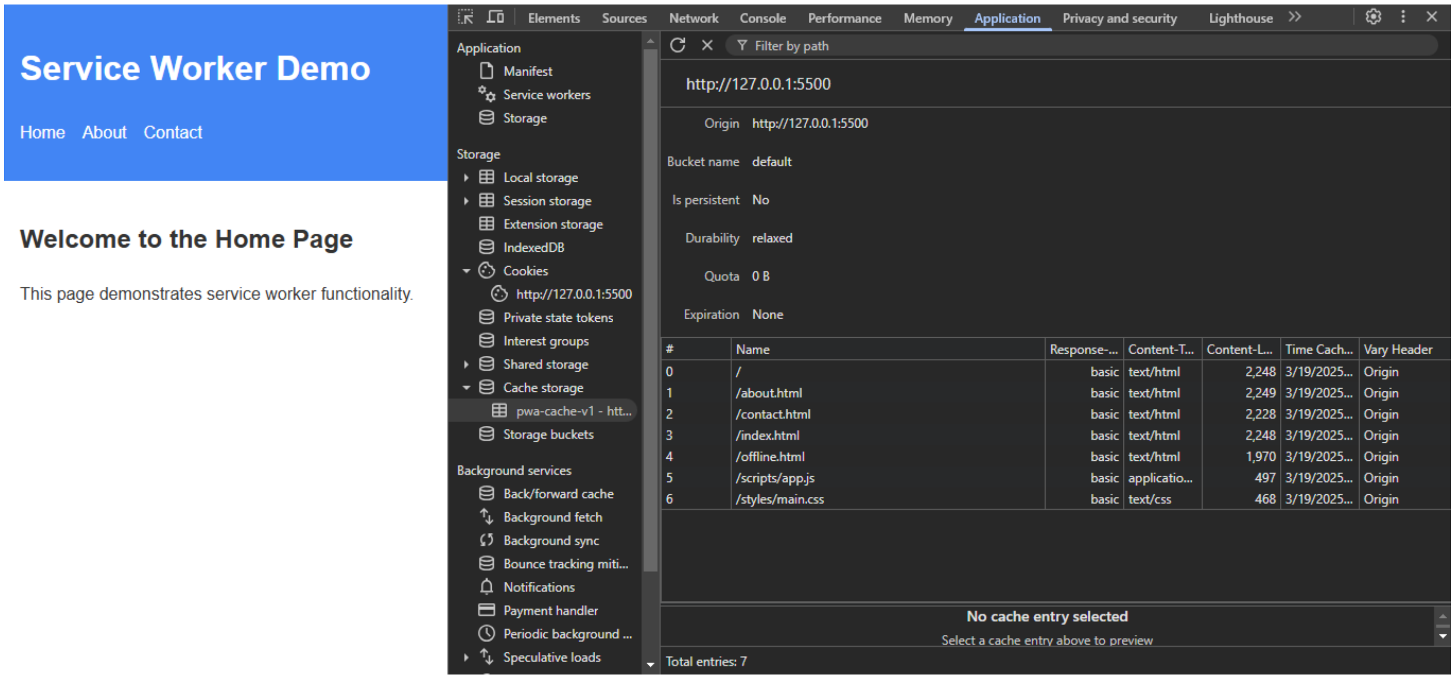
Status ○ #43 trying to install — Received 1/1/1970, 5:30:00 AM

Clients

Push `Test push message from DevTools.` [Push]

Sync `test-tag-from-devtools` [Sync]

Console | Issues



Service Worker Demo

Home About Contact

Welcome to the Home Page

This page demonstrates service worker functionality.

http://127.0.0.1:5500

Origin	http://127.0.0.1:5500
Bucket name	default
Is persistent	No
Durability	relaxed
Quota	0 B
Expiration	None

#	Name	Response-...	Content-T...	Content-L...	Time Cach...	Vary Header
0	/	basic	text/html	2,248	3/19/2025...	Origin
1	/about.html	basic	text/html	2,249	3/19/2025...	Origin
2	/contact.html	basic	text/html	2,228	3/19/2025...	Origin
3	/index.html	basic	text/html	2,248	3/19/2025...	Origin
4	/offline.html	basic	text/html	1,970	3/19/2025...	Origin
5	/scripts/app.js	basic	applicatio...	497	3/19/2025...	Origin
6	/styles/main.css	basic	text/css	468	3/19/2025...	Origin

No cache entry selected

Select a cache entry above to preview

Total entries: 7