Web-based payment apps overview

- Benefits of web-based payment apps
- How does a web-based payment app work?
- How merchants discover your payment app
- APIs you can use inside the payment handler window
  - WebAuthn support
  - Credential Management API support
  - WebOTP support
- Next steps

Web Payments brings to the web a browser's built-in interface that allows users to enter required payment information easier than ever before. The APIs can invoke web-based payment apps, as well as Android payment apps.

Benefits of web-based payment apps

Checkout flow with a web-based payment app.

- Payments are made in modals, in the context of the merchant website, which provides better user experience than typical payment app techniques that use redirects or pop-ups.
- Web Payments APIs can be integrated into established websites allowing you to leverage the existing user base.
- Unlike platform-specific apps, web-based payment apps don't need to be installed in advance.

How does a web-based payment app work?

Web-based payment apps are built using the standard web technologies. Every web-based payment app must include a service worker.

A Service worker is an event-driven script that runs in the background even if the registering website is not open in the browser. Service workers enable websites to work offline and send push notifications, because they can respond to requests with a cache that is stored locally in advance.

In a web-based payment app, a service worker can act as a mediator for payment requests by:

- Opening a modal window and displaying the payment app's interface.
- Bridging the communication between the payment app and the merchant.
- Getting an authorization from the customer and passing the payment credential to the merchant.

Learn how a payment app works on a merchant in Life of a payment transaction.

How merchants discover your payment app

In order for a merchant to use your payment app, they need to use the Payment Request API and specify the payment method you support using the payment method identifier.

If you have a payment method identifier that is unique to your payment app, you can set up your own payment method manifest and let browsers discover your app.

Learn how it works and how you can set up a new payment method in Setting up a payment method.

APIs you can use inside the payment handler window

A "payment handler window" is a window in which payment apps are launched. In Chrome, since it's a regular Chrome browser window, most web APIs should work as if used in a top-level document, with only a few exceptions:

- Resizing the viewport is disabled.
- `window.open()` is disabled.

Payment Handler API is only supported in Chrome as of January 2023. However, since Chromium based browsers already have the implementation, some of them may expose the API in the future.

WebAuthn support

WebAuthn is an authentication mechanism based on the public key cryptography. You can let users sign-in through a biometric verification. WebAuthn is already supported in the payment handler window on Chrome, and the standard body is looking into creating an even-tighter connection between Web Payments and WebAuthn.

Credential Management API support

The Credential Management API provides a programmatic interface between the site and the browser for seamless sign-in across devices. You can let users sign-in to your website automatically based on the information stored to the browser's password manager. It's planned to be enabled in Chrome, but still under development.

WebOTP support

The WebOTP API helps you programmatically obtain an OTP from an SMS message and verify a phone number for the user more easily. It's planned to be enabled in Chrome, but still under development.

You can check out the list of known issues and features planned to be added to the payment handler window in the Chromium bug tracker.

Next steps

To start building a web-based payment app, you have three distinct parts to implement:

- Registering a web-based payment app
- Orchestrating payment transactions with a service worker
- Handling optional payment information with a service worker

---

□□□ #3
□ □□□ □□□ 2026-10-31 23:08:28 UTC
□ cocobook-sync □□□ 2026-11-04 17:39:46 UTC