

Chrome Browser Binding Prototype Installation Instructions (public version)

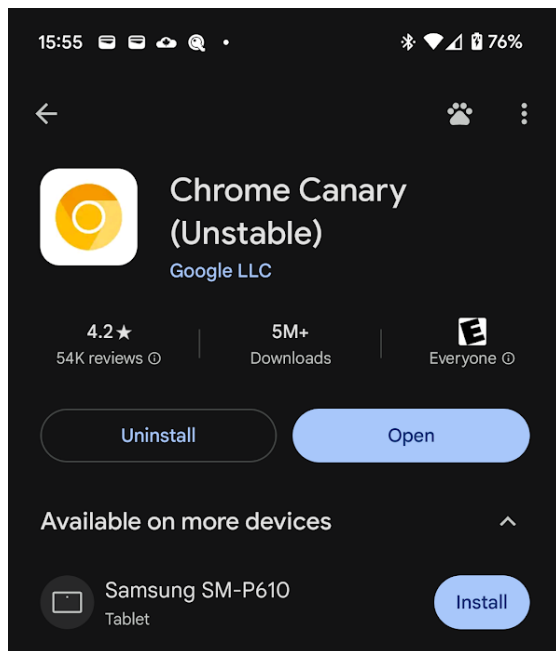
This Document is Public

*Authors: slobodan@chromium.org
January 2025 (Updated February 2025)*

This document describes how to download Chrome Canary and enable the browser binding prototype. At the time of writing the prototype is only implemented on Chrome for Android.

Installing Chrome Canary on Android

1. Open the Play Store app.
2. Search for "Chrome Canary".
3. Install "Chrome Canary (Unstable)"

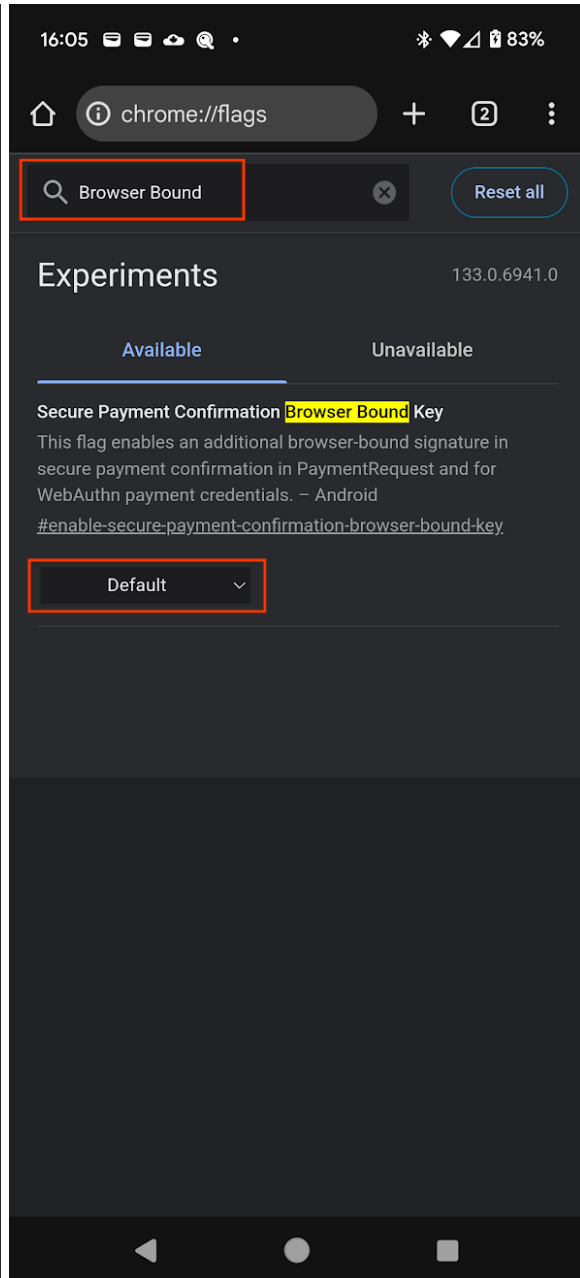
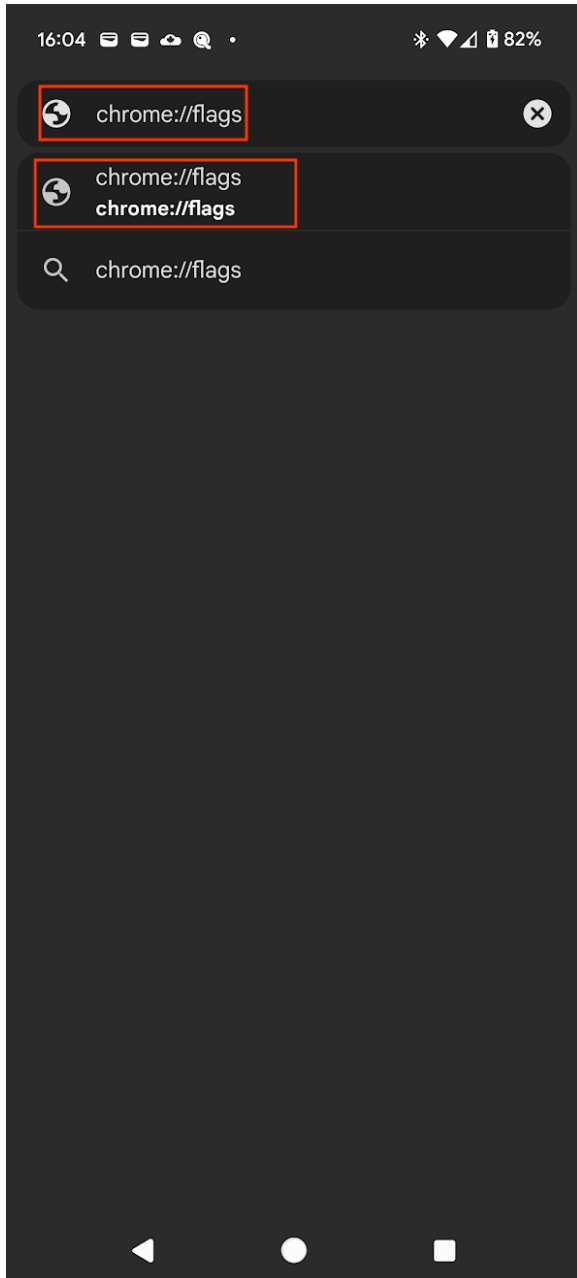


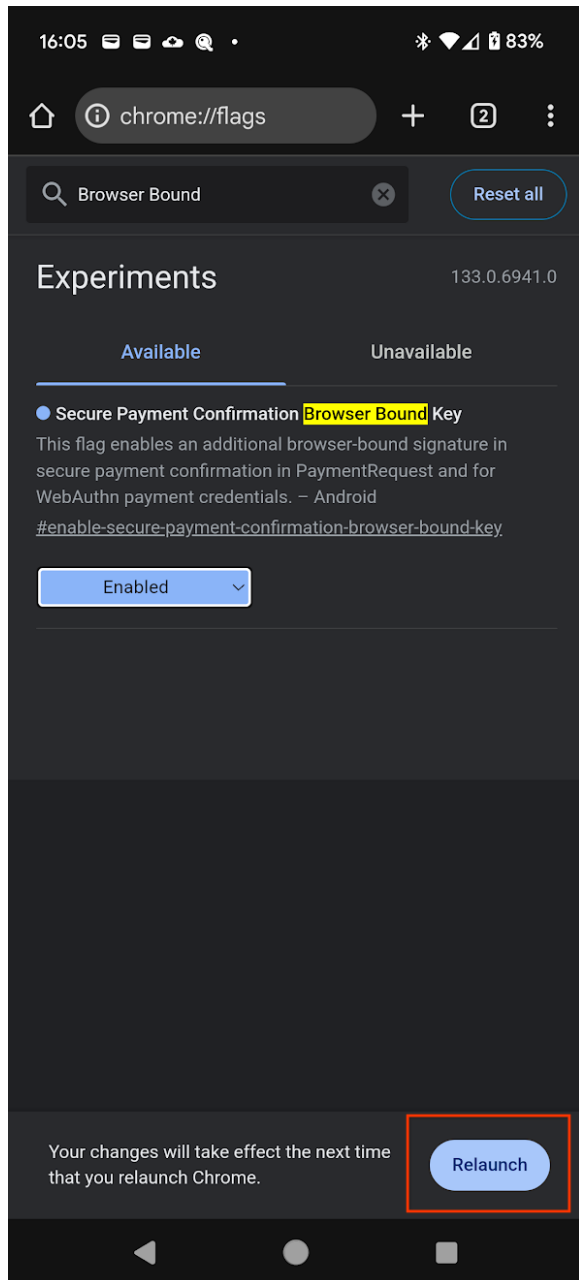
Enable Browser Binding

1. Open the Chrome Canary App

2. Enable the flag:
 - a. Navigate to `chrome://flags`
 - b. Search for "Browser Bound"
(`chrome://flags/#enable-secure-payment-confirmation-browser-bound-key`)
 - c. Switch the flag from "Default" (or "Disabled") to "Enabled"
 - d. Push the "Relaunch" button
 - e. Chrome Canary should relaunch, then show the flags page with browser bound key flag enabled

The browser binding prototype is enabled now. No further steps are needed to enable the prototype.





Notes

February Update: The restriction below no longer apply as of Chrome Canary for Android version 135.0.7017.0.

- At the time of writing the signature output is an array of browser bound signatures. This will be updated shortly to a single browser bound signature to match the most recent version of the proposal (see [github](#)).

- At the time of writing the browser bound key is not recreated at payment assertion if the browser bound key is not present. (For example if the credential was initially created without the flag enabled, or the browser bound key was deleted)
- At the time of writing, an ES256 credential is always generated regardless of the requested algorithm. (Now, only if the requested algorithms list include ES256, an ES256 credential is generated).

Demo

Navigate to <https://pejic.github.io/pr/spc/>.

Observe “isUserVerifyingPlatformAuthenticatorAvailable: true” is shown below the buttons. (A user verifying platform authenticator is required for SPC).

```
isUserVerifyingPlatformAuthenticatorAvailable: true
```

Push “Enroll Credential #1”. You will be prompted to create a passkey for pejic.github.io and will need to verify using either fingerprint or pin. The page will show the relevant output from createCredential

```
isUserVerifyingPlatformAuthenticatorAvailable: true
Credential #1 enrolled: {
  "id": "UC65j1Vbjb5NRuqCL_tz9A",
  "rawId": "UC65j1Vbjb5NRuqCL/tz9A==",
  "response": {
    "attestationObject":
"o2NmbXRkbm9uZWdhbHRTdG10oGhhdXRoRGF0YViU7u3o/yQLfk/deOsDonS6pWkTqfz+
7hMO1+cKFqWv+cBdAAAAAOqbjWZNAR0hPOS2tly1ddQAEFAuuY9VW42+TUbqgi/7c/SIAQ
IDJiABIVgg8esqQw8J6QAS3YSXqaS/2Ycj/cmnA5iv/BchBDe+XfMiWCD9IOuR7Fab6Bc9J22PG
0/otupdoCT4lwh9OQWGctHvBQ==",
    "clientDataJSON":
"{\"type\": \"webauthn.create\", \"challenge\": \"RW5yb2xsbWVudCBjaGFsbGVuZ2U\", \"origin\": \"https://pejic.github.io\", \"crossOrigin\": false, \"payment\": {\"rpld\": \"Rouslan Solomakhin\", \"topOrigin\": \"https://pejic.github.io\", \"total\": {\"value\": \"\", \"currency\": \"\"}, \"instrument\": {\"icon\": \"\", \"displayName\": \"\"}, \"browserBoundPublicKey\": \"pQECAYYgASFYIK--a_kwK5yrFX6JF8F9U-piyLb3isWVB9Th1-Y8v9LNlIgg4A2wYBHsqAJnZrw1QsiyDS8EJ4wpbByj37JJost-zFI\"}}",
  },
  "type": "public-key"
}
Extensions: {
  "payment": {
    "browser_bound_signatures": {
```

```

    "signature_output":
    "MEUCIDfY6y6FoMp9JxPsRfbY1f0+DLp9hLf51wXUGDVATCaQAiEAvJehIncOzYcAhTIIfeARxMmYfKS2kGqQv+0mw9eQP7k="
  }
}
}

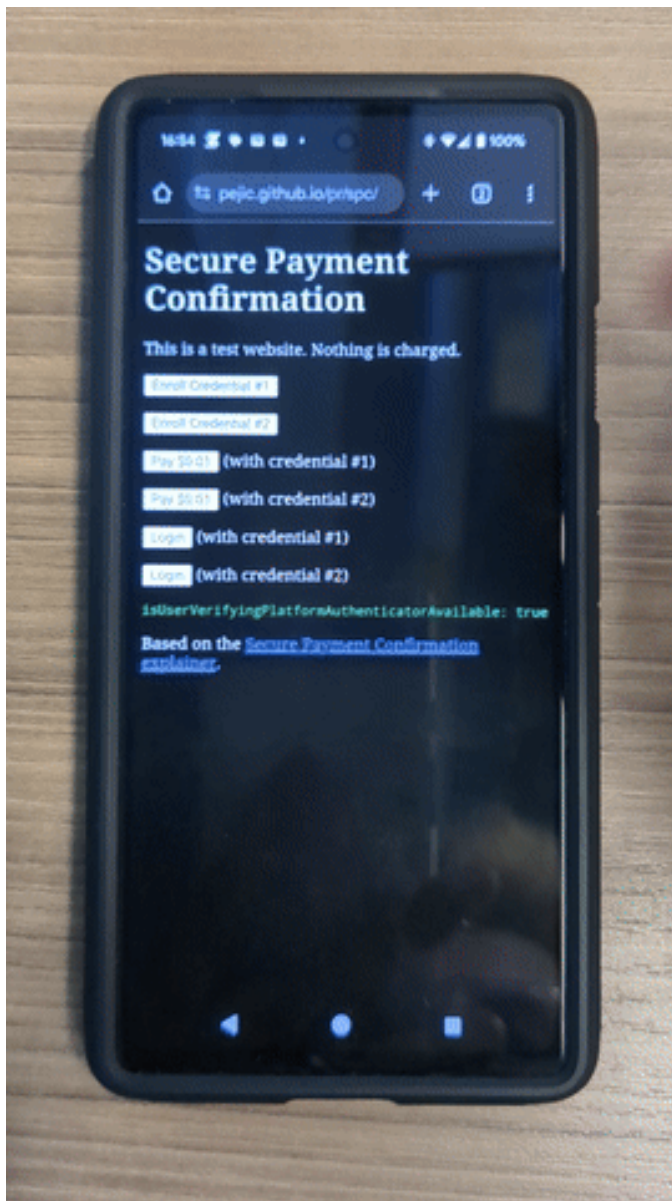
```

Push “Pay \$0.01”. You will be prompted to authenticate the mock payment (no payment will occur). After pushing “Continue” and authenticating with a fingerprint or pin, the page will show the relevant output from the SPC assertion.

```

Credential #1 payment response: {
  "requestId": "4cd077ba-5e01-4c11-becf-df379e82e9d8",
  "methodName": "secure-payment-confirmation",
  "details": {
    "id": "UC65j1Vbjb5NRuqCL_tz9A",
    "rawId": "UC65j1Vbjb5NRuqCL/tz9A==",
    "response": {
      "authenticatorData":
      "7u3o/yQLfk/deOsDonS6pWkTqfz+7hMO1+cKFqWv+cAdAAAAA==",
      "clientDataJSON":
      "{\"type\":\"payment.get\",\"challenge\":\"bmV0d29ya19kYXRh\",\"origin\":\"https://pejic.github.io\",\"crossOrigin\":false,\"payment\":{\"rpld\":\"pejic.github.io\",\"topOrigin\":\"https://pejic.github.io\",\"payeeOrigin\":\"https://pejic.github.io\",\"total\":{\"value\":\"0.01\",\"currency\":\"USD\"},\"instrument\":{\"icon\":\"https://rsolomakhin.github.io/pr/spc/troy-alt-logo.png\",\"displayName\":\"Troy Â·Â·Â·1234\"},\"browserBoundPublicKey\":\"pQECAyYgASFYIK--a_kwK5yrFX6JF8F9U-piyLb3isWVB9Th1-Y8v9LNIlgg4A2wYBHsqAJnZrw1QsiyDS8EJ4wpbByj37JJost-zFI\"}}",
      "signature":
      "MEUCIQDMajmmh6DHkFzYsVAZrABq/X7mBYCyim0YoAtG17tlqwlgOmgY9uu1KC7+SkkP1rBOgrJjRpejwXl0EA5nUqf5RXY=",
      "userHandle": "MjM1NDU0MzE1Ljk5NzI0MjQy"
    },
    "type": "public-key"
  }
}
Extensions: {
  "payment": {
    "browser_bound_signatures": {
      "signature_output":
      "MEUCIQCKn5vbAqGG5eK+KuT1WxfZmNDj29w21fVi86TF2YyXiAlgbIP+IP0E4GQ5yczks7xA+vXQ/hlvmg1zEjxpwg9Ihd0="
    }
  }
}
}

```



February update: The demo page now includes the ability to enter (and show) the passkey credential ids. This can be used to restore the credential id into the demo page (its localStorage) after deleting and reinstalling Chrome clears the demo page's storage.