

Device Access

Like Chromium based browsers, Electron provides access to device hardware through web APIs. For the most part these APIs work like they do in a browser, but there are some differences that need to be taken into account. The primary difference between Electron and browsers is what happens when device access is requested. In a browser, users are presented with a popup where they can grant access to an individual device. In Electron APIs are provided which can be used by a developer to either automatically pick a device or prompt users to pick a device via a developer created interface.

Web Bluetooth API

The [Web Bluetooth API](#) can be used to communicate with bluetooth devices. In order to use this API in Electron, developers will need to handle the [select-bluetooth-device](#) [event on the webContents](#) associated with the device request.

Example

This example demonstrates an Electron application that automatically selects the first available bluetooth device when the `Test Bluetooth` button is clicked.



WebHID API

The [WebHID API](#) can be used to access HID devices such as keyboards and gamepads. Electron provides several APIs for working with the WebHID API:

- The [select-hid-device](#) [event on the Session](#) can be used to select a HID device when a call to `navigator.hid.requestDevice` is made. Additionally the [hid-device-added](#) and [hid-device-removed](#) events on the Session can be used to handle devices being plugged in or unplugged during the `navigator.hid.requestDevice` process.
- [ses.setDevicePermissionHandler\(handler\)](#) can be used to provide default permissioning to devices without first calling for permission to devices via `navigator.hid.requestDevice`. Additionally, the default behavior of Electron is to store granted device permission through the lifetime of the corresponding WebContents. If longer term storage is needed, a developer can store granted device permissions (eg when handling the `select-hid-device` event) and then read from that storage with `setDevicePermissionHandler`.
- [ses.setPermissionCheckHandler\(handler\)](#) can be used to disable HID access for specific origins.

Blocklist

By default Electron employs the same [blocklist](#) used by Chromium. If you wish to override this behavior, you can do so by setting the `disable-hid-blocklist` flag:

```
app.commandLine.appendSwitch('disable-hid-blocklist')
```

Example

This example demonstrates an Electron application that automatically selects HID devices through `ses.setDevicePermissionHandler(handler)` and through `select-hid-device` [event on the Session](#) when the `Test WebHID` button is clicked.

Web Serial API

The [Web Serial API](#) can be used to access serial devices that are connected via serial port, USB, or Bluetooth. In order to use this API in Electron, developers will need to handle the `select-serial-port` [event on the Session](#) associated with the serial port request.

There are several additional APIs for working with the Web Serial API:

- The `serial-port-added` and `serial-port-removed` events on the Session can be used to handle devices being plugged in or unplugged during the `navigator.serial.requestPort` process.
- `ses.setDevicePermissionHandler(handler)` can be used to provide default permissioning to devices without first calling for permission to devices via `navigator.serial.requestPort`. Additionally, the default behavior of Electron is to store granted device permission through the lifetime of the corresponding WebContents. If longer term storage is needed, a developer can store granted device permissions (eg when handling the `select-serial-port` event) and then read from that storage with `setDevicePermissionHandler`.
- `ses.setPermissionCheckHandler(handler)` can be used to disable serial access for specific origins.

Example

This example demonstrates an Electron application that automatically selects serial devices through `ses.setDevicePermissionHandler(handler)` as well as demonstrating selecting the first available Arduino Uno serial device (if connected) through `select-serial-port` [event on the Session](#) when the `Test Web Serial` button is clicked.