**Jabra Browser Integration**

Revision 0.4



**Table of Contents**

[1 SDK overview 3](#_Toc480367364)

[1.1 System requirements 3](#_Toc480367365)

[1.1.1 Jabra devices 3](#_Toc480367366)

[1.1.2 Operating system support 3](#_Toc480367367)

[1.1.3 Browser support 3](#_Toc480367368)

[1.2 Using the library 3](#_Toc480367369)

[1.2.1 SDK web site 4](#_Toc480367370)

[1.2.2 The API 4](#_Toc480367371)

[1.2.3 Sequence diagrams 5](#_Toc480367372)

[1.3 Solution deployment 9](#_Toc480367373)

[1.3.1 Deployment option 1: Use the public Jabra components 9](#_Toc480367374)

[1.3.2 Deployment option 2: Customize the components 10](#_Toc480367375)

# SDK overview

This software package from Jabra helps developers to make solutions, where basic headset call control can be used from within a browser app using JavaScript. Since it is not possible to access USB devices directly from JavaScript, this library provides a solution of getting a route from the JavaScript to the Jabra USB device. The API is a JavaScript library with a facade that hides implementation details. Basic call control is defined by off-hook/on-hook, ringer, mute/unmute and hold/resume. With these features, it is possible to implement a softphone app on a website. Combined with the WebRTC[[1]](#footnote-1) technology it is possible to create a softphone that only requires one small software component installed locally on the computer, while the business logic is implemented in JavaScript.

## System requirements

With current internal implementation of this software package, the following systems are supported.

### Jabra devices

All professional Jabra headsets and Jabra speakerphones are supported. I.e. the Jabra Evolve series[[2]](#footnote-2), the Jabra Pro series, the Jabra Biz series, and the Jabra Speak series[[3]](#footnote-3).

### Operating system support

The following desktop operating systems are supported:

|  |  |
| --- | --- |
| Operating system | Status |
| Windows 64 bit | Windows 7 or newer |
| Windows 32 bit | Windows 7 or newer |
| macOS | Sierra |

### Browser support

Google Chrome web browser - stable channel - 32 bit and 64 bit, if supported by the operating system.

## Using the library

Developers must use the JavaScript library file: **jabra.browser.integration-0.2.js**

The library internally checks for dependencies – and will report this to the app using the library. An example: The Jabra library is initialized and an error callback function is called with this text and a link: **“You need to use this Extension and then reload this page”**. Note that a future version could change the internal implementation and that removes/adds new requirements.

### SDK web site

A web site with documentation, developer tools and a softphone demo:

<https://jabra-browser-sdk.azurewebsites.net/>

### The API

The JavaScript library must be initialized using this function:

**jabra.init(onSuccess, onFailure, onNotify)**

Function used for initialization. This function has three arguments: **onSuccess** – a callback reporting that the library has been successfully initialized and the library is ready for use, **onFailure** – a callback reporting errors during initialization, **onNotify** – a callback used for reporting events.

Example use of the library:

<https://jabra-browser-sdk.azurewebsites.net/development>

Basic functions:

|  |  |
| --- | --- |
|  | Description |
| jabra.ring() | Activate ringer (if supported) on the Jabra Device |
| jabra.offHook() | Change state to in-a-call |
| jabra.onHook() | Change state to idle (not-in-a-call) |
| jabra.mute() | Mutes the microphone (if supported) |
| jabra.unmute() | Unmutes the microphone (if supported) |
| jabra.hold() | Change state to held (if supported) |
| jabra.resume() | Change state from held to OffHook (if supported) |

Callback values from the library (registered during library initialization):

|  |  |
| --- | --- |
|  | Description |
| jabra.requestEnum.mute | Request that the device state should be changed to muted. This must be acknowledged by a Mute command to mute the call. |
| jabra.requestEnum.unmute | Request that the device state should be changed to unmuted. This must be acknowledged by an Unmute command to unmute the call. |
| jabra.requestEnum.endCall | End an active call request. This must be acknowledged by an OnHook command |
| jabra.requestEnum.acceptCall | Accept an incoming call request. This must be acknowledged by an OnHook command. |
| jabra.requestEnum.rejectCall | Reject an incoming call request. This must be acknowledged by an OnHook command to reject the call. |
| jabra.requestEnum.flash | Flash request. This must be acknowledged by a hold or resume command. |

Device management commands:

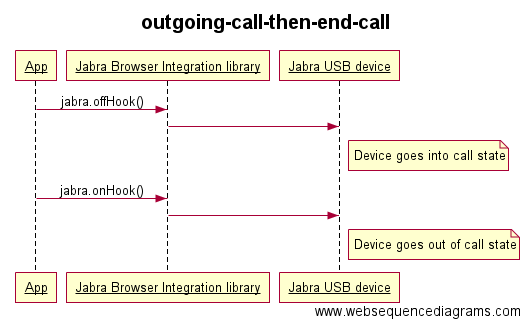
|  |  |
| --- | --- |
|  | Description |
| jabra.getActiveDevice() | Get the current active Jabra Device |
| jabra.getDevices() | List all attached Jabra Devices |
| jabra.setActiveDevice(id) | Select a new active device |

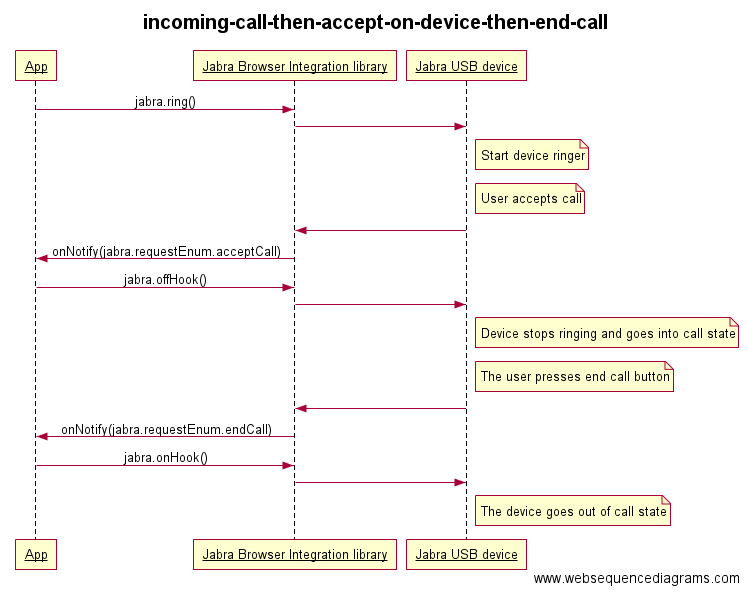
Device management callbacks:

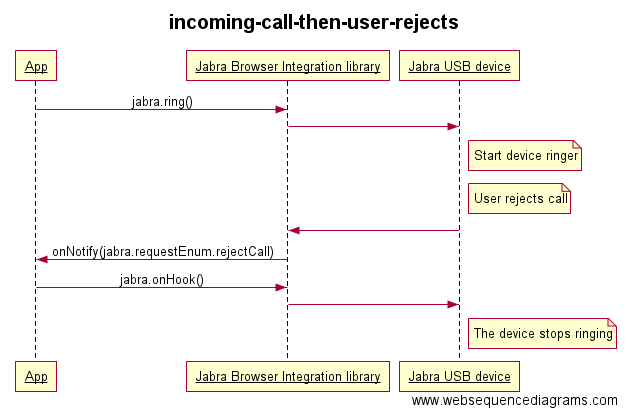
|  |  |
| --- | --- |
|  | Description |
| jabra.requestEnum.deviceAttached | A device has been added |
| jabra.requestEnum.deviceDetached | A device has been removed |

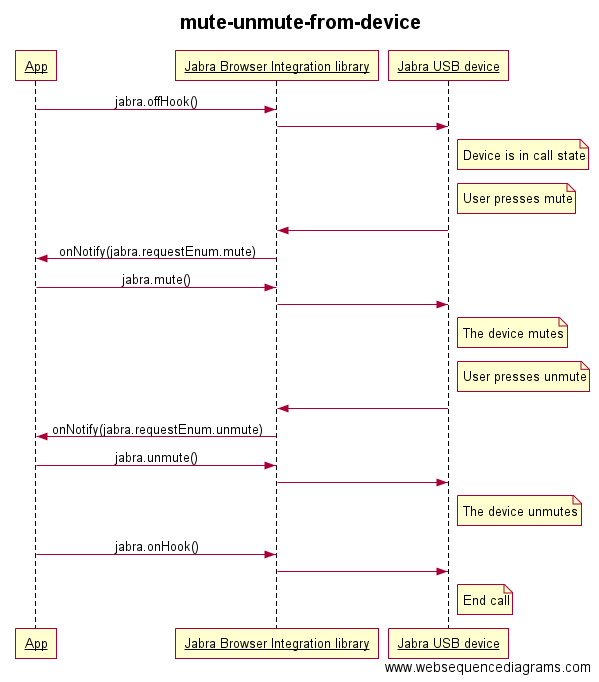
### Sequence diagrams

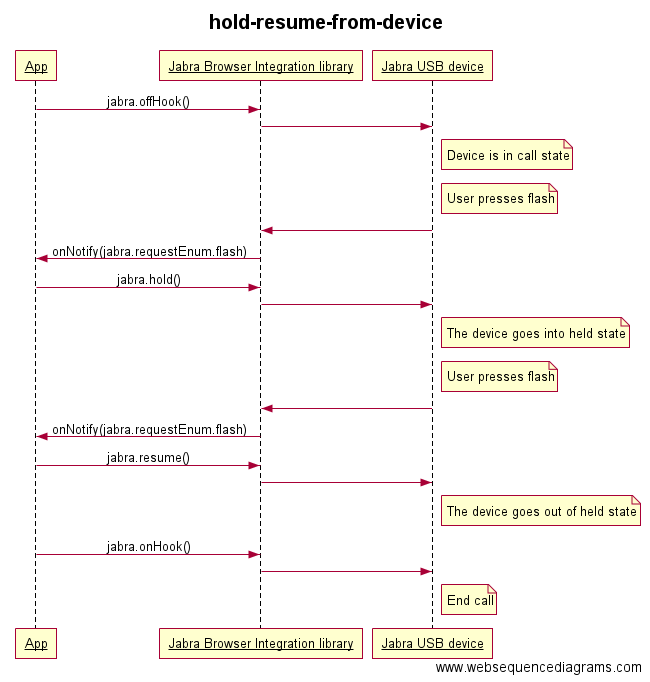
These sequence diagrams shows typical use.











## Solution deployment

When deploying a site that uses this library (with the current implementation), the end-user need to handle two dependencies: 1) A Chrome extension that creates a route out of the browser (non-platform specific) 2) A Host component that needs to be installed on the end-user system that handles the USB communication (platform specific).

### Deployment option 1: Use the public Jabra components

This option is the default used components from the library.

Use this Chrome extension:

<https://chrome.google.com/webstore/detail/okpeabepajdgiepelmhkfhkjlhhmofma>

In addition, these Hosts:

<https://jabra-browser-sdk.azurewebsites.net/download/>

**Security concerns regarding Option 1. The public Chrome extension is configured to allow apps from any sites. This is a potential security issue since any site could access the library and i.e. start the ringer on a headset.**

### Deployment option 2: Customize the components

In this option, a new Chrome extension is made (either by the developer using the library or by Jabra) – and the only change is that this extension is only valid for a specific URL: I.e. <https://www.organization.com/oursoftphone>

This extension must then be published on the Chrome store – and then it gets an Id. This Id must then be added in a white list of allowed extensions in the Host applications. This will be done by Jabra – this will then trigger new releases of all Host implementations. Contact Jabra to get help: <http://developer.jabra.com>

**Security concerns regarding Option 2. This option is more secure than option 1, since only applications on a specific URL will be able to use the library.**

1. <https://en.wikipedia.org/wiki/WebRTC> [↑](#footnote-ref-1)
2. <http://www.jabra.com/business/office-headsets/jabra-evolve> [↑](#footnote-ref-2)
3. <http://www.jabra.com/business/speakerphones/jabra-speak-series> [↑](#footnote-ref-3)