**Spotcast**

Home Assistant custom component to start Spotify playback on an idle chromecast device or a Spotify Connect device (thanks to @kleinc80) which means that you can target your automation for chromecast as well as connect devices.

Spotcast implements a cast platform (requires Home Assistant Core 2022.2.0 or later), which enables Google Cast media player entities to play Spotify URI as well as to browse the Spotify library.

This component is not meant to be a full Spotify chromecast media\_player but only serves to start the playback. Controlling the chromecast device and the Spotify playback after the initial start is done in their respective components. Because starting playback using the API requires more powerful token the username and password used for browser login is used.

Used by [Spotify-Card](https://github.com/custom-cards/spotify-card).

**[Community post](https://community.home-assistant.io/t/spotcast-custom-component-to-start-playback-on-an-idle-chromecast-device/114232)**

**Installation**

**HACS**

This component is easiest installed using [HACS](https://github.com/custom-components/hacs).

**Manual installation**

Copy all files from custom\_components/spotcast/ to custom\_components/spotcast/ inside your config Home Assistant directory.

**Configuration**

**Minimum Home Assistant version**

Spotcast is compatible with any version since 2021.12.0.

**Official Spotify Integration**

Note that as of v3.5.2 you must also have the official [Home Assistant Spotify Integration](https://www.home-assistant.io/integrations/spotify/) installed and configured for this custom component to work. This is because it provides the correct device list which has the correct scopes in its token.

**Obtaining sp\_dc and sp\_key cookies**

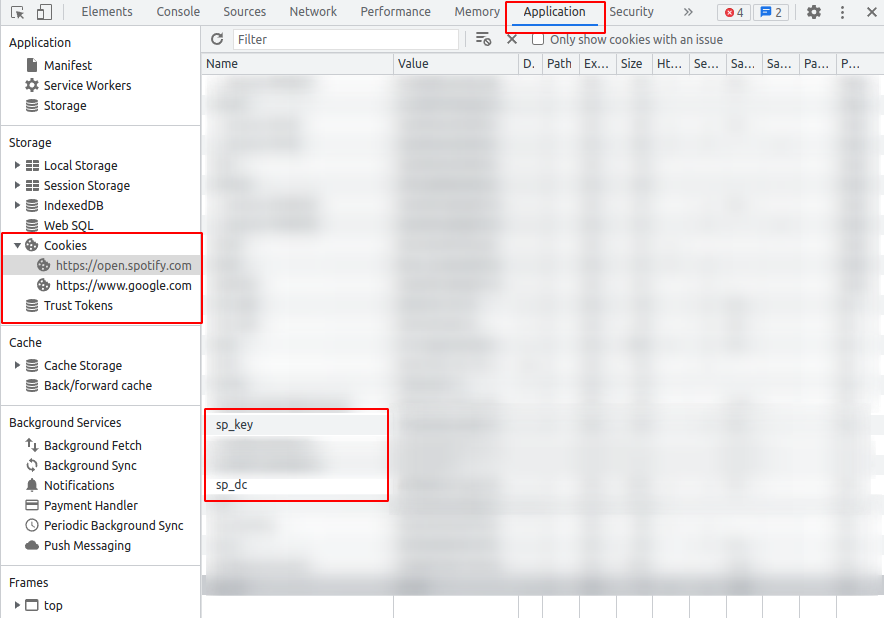
Spotcast uses two cookies to authenticate against Spotify in order to have access to the required services.

To obtain the cookies, these different methods can be used:

**Chrome based browser**

**Chrome web console**

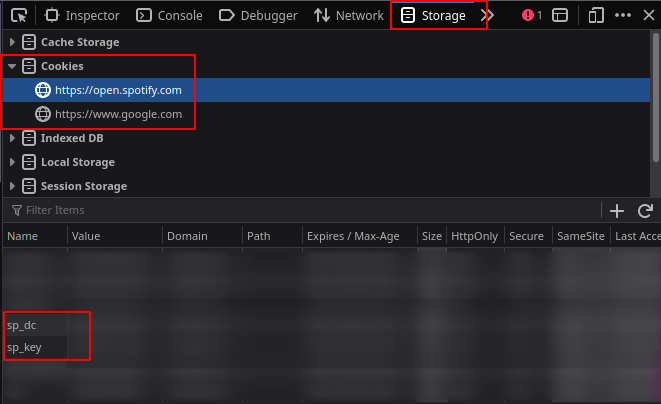
1. Open a new **Incognito window** at [https://open.spotify.com](https://open.spotify.com/) and login to Spotify.
2. Press Command+Option+I (Mac) or Control+Shift+I or F12. This should open the developer tools menu of your browser.
3. Go into the application section.
4. In the menu on the left go int Storage/Cookies/open.spotify.com.
5. Find the sp\_dc and sp\_key and copy the values.
6. Close the window without logging out (Otherwise the cookies are made invalid).

[](https://github.com/fondberg/spotcast/blob/master/images/cookies_chrome_2.png)

**Firefox based browser**

**Firefox web console**

1. Open a new **Incognito window** at [https://open.spotify.com](https://open.spotify.com/) and login to Spotify.
2. Press Command+Option+I (Mac) or Control+Shift+I or F12. This should open the developer tools menu of your browser.
3. Go into the Storage section. (You might have to click on the right arrows to reveal the section).
4. Select the Cookies sub-menu and then https://open.spotify.com.
5. Find the sp\_dc and sp\_key and copy the values.
6. Close the window without logging out (Otherwise the cookies are made invalid).

[](https://github.com/fondberg/spotcast/blob/master/images/cookies_firefox_1.png)

**Single account**

Add the following to your configuration.yaml:

spotcast:

sp\_dc: !secret sp\_dc

sp\_key: !secret sp\_key

country: SE #optional, added in 3.6.24

**Multiple accounts**

Add accounts dict to the configuration and populate with a list of accounts to be able to initiate playback using diffferent accounts than the default.

If you are using v3.5.2 or greater and thus also have the core Spotify Integration installed, then [the additional accounts will also need to be added there as well](https://www.home-assistant.io/integrations/spotify" \l "using-multiple-spotify-accounts) for multiple accounts to work.

spotcast:

sp\_dc: !secret primary\_sp\_dc

sp\_key: !secret primary\_sp\_key

country: SE #optional, added in 3.6.24

accounts:

niklas:

sp\_dc: !secret niklas\_sp\_dc

sp\_key: !secret niklas\_sp\_key

ming:

sp\_dc: !secret ming\_sp\_dc

sp\_key: !secret ming\_sp\_key

**Edit secrets.yaml**

Please note: configuration.yaml is a plain text file and [it is not recommended to store your passwords in this file](https://www.home-assistant.io/docs/configuration/secrets/).

By using the !secret tag you are directing Home Assistant to look inside the secrets.yaml file for your key instead.

For a single account add the following to your secrets.yaml file:

sp\_dc: [your sp\_dc here]

sp\_key: [your sp\_key here]

For multiple accounts add the corresponding entries to your secrets.yaml file. For example, using the same accounts as above:

primary\_sp\_dc: [your sp\_dc here]

primary\_sp\_key: [your sp\_key here]

niklas\_sp\_dc: [niklas sp\_dc here]

niklas\_sp\_key: [niklas sp\_key here]

ming\_sp\_dc: [ming sp\_dc here]

ming\_sp\_key: [ming sp\_key here]

**Call the service**

The spotcast custom component creates a service called 'spotcast.start' in Home Assistant.

**Start playback on Spotify connect device**

- service: spotcast.start

data:

spotify\_device\_id: "ab123c5d7347324c2b1234567890f8d6dc40350"

uri: "spotify:playlist:5xddIVAtLrZKtt4YGLM1SQ"

random\_song: true

**Start playback on a device with default account**

- service: spotcast.start

data:

device\_name: "Kitchen"

uri: "spotify:playlist:5xddIVAtLrZKtt4YGLM1SQ"

random\_song: true

where:

* spotify\_device\_id is the device ID of the Spotify Connect device
* device\_name is the friendly name of the chromecast device
* uri is the Spotify uri, supports all uris including track (limit to one track)
* search is a search query to resolve into a uri. This parameter will be overlooked if a uri is provided
* category let spotify pick a random playlist inside a given [category](https://developer.spotify.com/console/get-browse-categories/)
* country restrict country to use when looking for playlists inside a category
* limit restrict number of playlists to return when looking in a category. Note that only a single playlist will be chosen randomly from them.
* random\_song optional parameter that starts the playback at a random position in the playlist
* repeat optional parameter that repeats the playlist/track (track|context|off)
* shuffle optional parameter to set shuffle mode for playback
* offset optional parameter to set offset mode for playback. 0 is the first song

Optionally you can specify the entity\_id of an existing Home Assistant chromecast media-player like:

- service: spotcast.start

data:

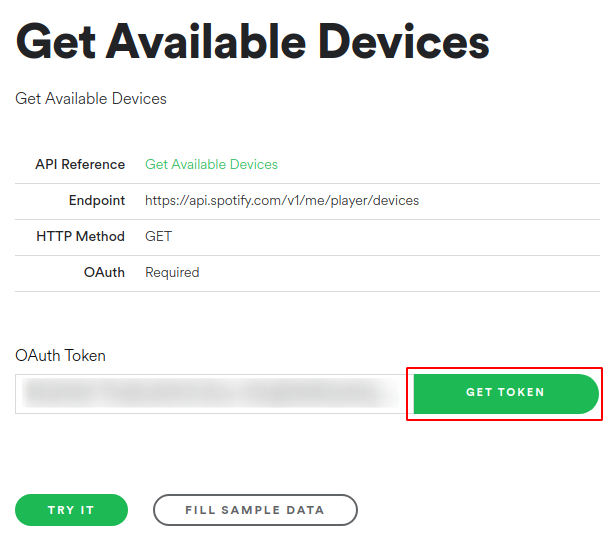
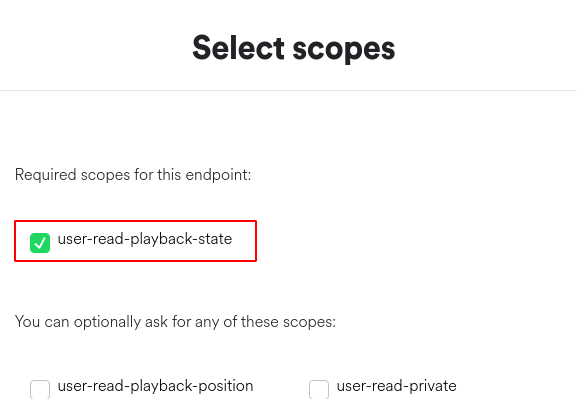
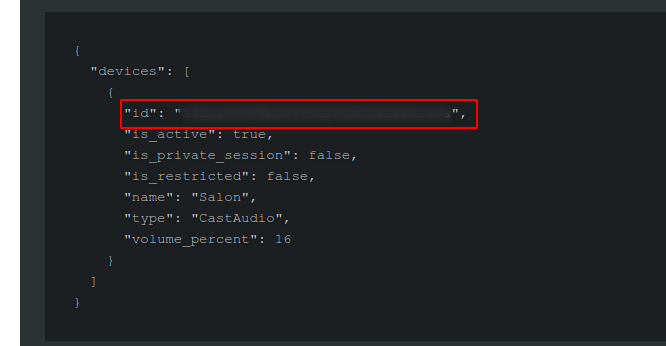
entity\_id: "media\_player.vardagsrum"

uri: "spotify:playlist:5xddIVAtLrZKtt4YGLM1SQ"

**Find Spotify Device ID**

To use the Spotcast service with a Spotify Connect device, you need the spotify\_device\_id. To find the spotify\_device\_id, multiple option are available.

**With Spotify developer portal**

1. Go to [Spotify developer console](https://developer.spotify.com/console/get-users-available-devices/)
2. Click GET TOKEN  
   [](https://github.com/fondberg/spotcast/blob/master/images/get_token.png)
3. Select user-read-playback-state as a scope  
   [](https://github.com/fondberg/spotcast/blob/master/images/select_scope.png)
4. If prompt give permission to your Spotify profile
5. For chromecast devices, make sure to play media on the device prior to checking the logs as they will not show unless active
6. Press the option Try it
7. Read the result in the console in the right.  
   [](https://github.com/fondberg/spotcast/blob/master/images/device_id.png)

**Through Spotcast log**

1. Enable the debug logs (instructions are in section Enabling Debug Log in this README)
2. Reboot Home Assistant
3. Go to Configuration >> Logs >> Load Full Home Assistant Log.
4. For chromecast devices, make sure to play media on the device prior to checking the logs as they will not show unless active
5. Find the log entry get\_spotify\_devices and look for the device ID.

**Sonos Device ID**

1. Open the Spotify Web Player and sign in if needed
2. Make sure you see your Sonos devices in the Connect popup
3. Open your browser's DevTools (F12 for Chrome)
4. Navigate to the Network tab
5. Connect to the desired Sonos device in the Web Player
6. Find the associated request in your DevTools
7. The request URL looks something like this: https://gew1-spclient.spotify.com/connect-state/v1/connect/transfer/from/my\_web\_player\_device\_id/to/my\_sonos\_device\_id
8. The my\_sonos\_device\_id is the spotify\_device\_id you are looking for.

**Log example**

2022-01-13 19:10:35 DEBUG (SyncWorker\_0) [custom\_components.spotcast.helpers] get\_spotify\_devices: media\_player.spotify\_felix: Spotify Félix Cusson: [{'id': '################################', 'is\_active': True, 'is\_private\_session': False, 'is\_restricted': False, 'name': 'Salon', 'type': 'CastAudio', 'volume\_percent': 16}]

2022-01-13 19:10:35 DEBUG (SyncWorker\_0) [custom\_components.spotcast.helpers] get\_spotify\_devices: {'devices': [{'id': '###############################', 'is\_active': False, 'is\_private\_session': False, 'is\_restricted': False, 'name': 'Salon', 'type': 'CastAudio', 'volume\_percent': 16}]}

# Look for

<...> Spotify Félix Cusson: [{'id': '################################', <- This is the device ID

**Automation example**

- id: 'christmas\_play\_christmas\_music'

alias: Christmas play Christmas music

initial\_state: 'on'

trigger:

- event\_data:

id: remote\_windowlamps

event: 5002

platform: event

event\_type: deconz\_event

condition: []

action:

- service: spotcast.start

data:

uri: 'spotify:playlist:56Bor5fbMJlJV7oryb2p3k'

random\_song: true

shuffle: true

start\_volume: 50

entity\_id: media\_player.kitchen

- service: spotcast.start

data:

search: "Brown Bird" # resolves to spotify:artist:5zzbSFZMVpvxSlWAkqqtHP at the time of writing

random\_song: true

shuffle: true

start\_volume: 50

entity\_id: media\_player.kitchen

**Transfer current playback for the account**

Omitting uri will transfer the playback to the specified device.

- service: spotcast.start

data:

device\_name: "Speaker kitchen"

Use the parameter force\_playback to continue the user's playback even if nothing is currently playing.

- service: spotcast.start

data:

device\_name: "Speaker kitchen"

force\_playback: true

where:

* device\_name is the friendly name of the chromecast
* force\_playback (optional) true or false, true to continue the user's playback even if nothing is currently playing

**Start playback on a device with non default account**

- service: spotcast.start

data:

account: "niklas"

device\_name: "Speaker kitchen"

uri: "spotify:playlist:5xddIVAtLrZKtt4YGLM1SQ"

where:

* account is the name of account key in the accounts dictionary in the configuration
* device\_name is the friendly name of the chromecast
* uri is the Spotify uri, supports all uris including track (limit to one track)

**Start podcast playback**

Play the latest episode of a given podcast show.

- service: spotcast.start

data:

account: "niklas"

device\_name: "Speaker kitchen"

uri: "spotify:show:6PeAI9SHRZhghU7NRPXvT3"

ignore\_fully\_played: true

where

* account is the name of account key in the accounts dictionary in the configuration
* device\_name is the friendly name of the Chromecast
* uri is the spotify uri, (podcasts use the 'show' uri)
* ignore\_fully\_played (optional) true or false, true to ignore already fully played episodes (defaults to false and plays the latest released episode)

**Use the sensor**

The sensor has the discovered chromecasts as both json and an array of objects. Since v3.4.0 it does not do its own discovery but relies on data from core cast. Add the following to the sensor section of the configuration:

sensor:

- platform: spotcast

country: SE

The country tag was added in v3.6.24. This tag is optional. If ommited or if you haven't updated the configuration since the update, it will default to "SE" (which it always did before)

Sensor name:

sensor.chromecast\_devices

Attributes

devices\_json: [

{

"name": "Speaker kitchen",

"cast\_type": "audio",

"model\_name": "Google Home",

"uuid": "xxxxx",

"manufacturer": "Google Inc."

},

{

"name": "Speakers upstairs",

"cast\_type": "group",

"model\_name": "Google Cast Group",

"uuid": "xxxx",

"manufacturer": "Google Inc."

},

{

"name": "Living room",

"cast\_type": "cast",

"model\_name": "HK Citation 300",

"uuid": "xxxx",

"manufacturer":"Harman Kardon"

}

]

last\_update: 2019-05-01T15:27:49.828553+02:00

friendly\_name: Chromecast Devices

**Websocket API**

The components websocket api.

Method: spotcast/playlist supporting different playlist\_types.

* user, or default for user chosen saved playlists
* featured for spotify "featured" playlists (not personalized)
* discover-weekly for personalized "Made for \_\_\_\_\_" (includes daily mixes)
* recently-played for "Recently Played"
* ... any other view id as found in the API at <https://api.spotify.com/v1/views/personalized-recommendations>

Example usage:

// Retrieve playlists

const res = await this.props.hass.callWS({

type: 'spotcast/playlists',

playlist\_type: 'featured', // 'user' for saved playlists, 'featured' for spotify featured, or personalized view id

country\_code: 'SV', // Optional country code used by featured playlists

limit: 20, // Optional limit, default is 10

account: 'ming' // optional account name

});

// Retrieve devices

const res = await this.props.hass.callWS({

type: 'spotcast/devices',

account: 'ming' // optional account name

});

// Retrieve player

const res = await this.props.hass.callWS({

type: 'spotcast/player',

account: 'ming' // optional account name

});

**Enabling debug log**

In configuration.yaml for you HA add and attach those the relevant logs. Be sure to disable it later as it is quite noisy.

logger:

default: info

logs:

custom\_components.spotcast: debug

**Contribute**

Please do! Open a Pull Request with your improvements.

This project was made possible by the original creator Niklas Fondberg. All your great work are greatly appreciated.

**License**

Apache 2.0