<https://sv443.net/jokeapi/v2/>

v2.3.2 [(Changelog)](https://github.com/Sv443/JokeAPI/blob/master/changelog.md#readme)  
[GitHub](https://github.com/Sv443/JokeAPI) • [Legal Stuff](https://sv443.net/jokeapi/v2/#legal) • [Author](https://github.com/Sv443)

## By using this website and API you are agreeing to the [privacy policy](https://sv443.net/privacypolicy/en)

(Click here to get more information on what data JokeAPI collects)✔ I agree

[🔗](https://sv443.net/jokeapi/v2/#info)Information:

JokeAPI is a REST API that serves uniformly and well formatted jokes.  
It can be used without any API token, membership, registration or payment.  
It supports a variety of filters that can be applied to get just the right jokes you need.  
The usage is very simple and similar to other RESTful APIs and requires only basic knowledge of HTTP requests and JSON, XML, YAML or plain text.  
  
If you come across a term you don't understand, you can try opening the menu and clicking on the section "Terminology".  
JokeAPI currently serves 1369 jokes from 6 different languages. To submit a new joke, [please click here.](https://sv443.net/jokeapi/v2/#submit)  
  
To report a bug or a grammar mistake or to suggest a feature, please [use the GitHub issue tracker.](https://github.com/Sv443/JokeAPI/issues/new/choose)  
If you want to contribute to JokeAPI, please read the [Contributing Guide.](https://github.com/Sv443/JokeAPI/blob/master/.github/Contributing.md)  
  
If you are new to JokeAPI, please start by reading the [Getting Started Guide](https://sv443.net/jokeapi/v2/#getting-started)

[§](https://sv443.net/jokeapi/v2/#note) Note:

If you enjoy using JokeAPI and want to support the development, please consider donating [here ♥](https://github.com/sponsors/Sv443)  
  
I can't speak all languages, so please note that I can't ensure format and validity of jokes that aren't in English or German.  
If you do come across an invalid joke or a joke that is formatted incorrectly, please consider [opening an issue](https://github.com/Sv443/JokeAPI/issues/new?assignees=Sv443&labels=reported+joke&template=3_report_a_joke.md&title=) or [submitting a pull request.](https://github.com/Sv443/JokeAPI/blob/master/.github/Contributing.md#submitting-or-editing-jokes)

[§](https://sv443.net/jokeapi/v2/#uptime) Uptime:

|  |  |
| --- | --- |
| Last day: |  |
| Last week: |  |
| Last month: |  |

[(more stats)](https://status.sv443.net/)

[§](https://sv443.net/jokeapi/v2/#security) Security:

JokeAPI has been a target of DoS attacks in the past, which is why there is now a limit of 120 requests per minute and why joke submissions are manually curated. For more information [click here.](https://sv443.net/jokeapi/v2/#rate-limiting)  
  
All dependencies are tested for vulnerabilities by [Snyk](https://snyk.io/), you can find a summary of the known vulnerabilities [here.](https://snyk.io/test/github/Sv443/JokeAPI)  
Page security score (powered by Mozilla Observatory):

[🔗](https://sv443.net/jokeapi/v2/#news)News:

* **JokeAPI now has its own domain!**  
  You can now replace the base URL https://sv443.net/jokeapi/v2 with https://v2.jokeapi.dev/  
  Note: you should always use the v2 subdomain to call the API but you can omit it if you just want to view the documentation.  
  For backwards compatibility the old URL will still be usable until version 2 will be deprecated.
* **Is JokeAPI serving too many offensive jokes?**  
  Try using version 2.3.0's new [safe mode](https://sv443.net/jokeapi/v2/#safe-mode) and the [explicit flag.](https://sv443.net/jokeapi/v2/#blacklist-flags)

[🔗](https://sv443.net/jokeapi/v2/#try-it)Try it out here:

|  |  |
| --- | --- |
| Select [category / categories](https://sv443.net/jokeapi/v2/#categories): | Any  Custom: Programming Misc Dark Pun Spooky Christmas |
| Select [language](https://sv443.net/jokeapi/v2/#languages-endpoint): |  |
| Select [flags](https://sv443.net/jokeapi/v2/#blacklist-flags) to blacklist: | (optional)    nsfw religious political racist sexist explicit |
| Select [response format](https://sv443.net/jokeapi/v2/#response-formats): | default (json) xml yaml plain text |
| Select at least one [joke type](https://sv443.net/jokeapi/v2/#joke-type): | single twopart |
| Search for a joke that contains this [search string](https://sv443.net/jokeapi/v2/#search-string): |  |
| Search for a joke in this [ID range](https://sv443.net/jokeapi/v2/#id-range): | (optional)    From:  To: |
| [Amount](https://sv443.net/jokeapi/v2/#amount) of jokes: |  |

URL: https://v2.jokeapi.dev/joke/Any  
Reset Form Send Request >

Result:

(Set parameters and click "Send Request" above)

[🔗](https://sv443.net/jokeapi/v2/#getting-started)Getting Started:

JokeAPI doesn't require any API token, authorization or payment to work, although there is a hard limit of 120 requests per minute due to some DoS attacks I have been getting.  
To start using JokeAPI, you can do one of these two things:

* You can see if your language has a wrapper for JokeAPI (to see a list of officially supported wrappers, [click here](https://sv443.net/jokeapi/v2/#wrappers))  
  Then, follow its documentation to communicate with the API (Disclaimer: I don't take any responsibility for their code).
* You can directly communicate with JokeAPI with HTTP requests (XHR) from within your code.  
  Here's a few resources to get you started: ([Python XHR Crash Course](https://medium.com/better-programming/crash-course-in-http-requests-using-python-59246373a187), [JavaScript XHR Documentation](https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest), [.NET HttpClient Class Documentation](https://docs.microsoft.com/en-us/dotnet/api/system.net.http.httpclient?view=netcore-3.1))  
  Once you know how to send HTTP requests, please take a look at [all the endpoints JokeAPI has.](https://sv443.net/jokeapi/v2/#endpoints)  
  After you've finished reading that, you can start testing out the different endpoints with a client such as [Postman](https://www.getpostman.com/) and look at some examples by [going to the "Examples" section.](https://sv443.net/jokeapi/v2/#examples)

If your project really can't contain any explicit jokes, consider enabling the [Safe Mode.](https://sv443.net/jokeapi/v2/#safe-mode)  
  
If you know any good jokes and want to add them to JokeAPI, please [click here.](https://sv443.net/jokeapi/v2/#submit)  
If you need any help, feel free to [join my Discord server.](https://sv443.net/discord)  
  
I struggled at explaining some features so consult the provided examples if you don't understand them.

[🔗](https://sv443.net/jokeapi/v2/#wrappers)Official API Wrappers:

A wrapper library translates some API into an interface that is easier to use for a certain language or environment.  
In the case of RESTful APIs like JokeAPI, a wrapper library makes it so you can use your language's features to communicate with the API while the wrapper will do all the HTTP requests for you in the background.  
  
Disclaimer: I will not take any responsibility for the wrappers, they are community-made and I can't ensure security, validity and compatibility.  
Also, some of these might be outdated or not finished yet. Please consider supporting the creators or contributing to the libraries :)  
  
The following is a list of officially supported JokeAPI wrappers:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Language** | **Wrapper** | **Author** |
|  | Rust | [Sv443\_JokeAPI Rust Crate](https://github.com/canarado/sv443_jokeapi_wrapper#readme) | [canarado](https://github.com/canarado) |
|  | Python | [Sv443s JokeAPI Python Wrapper](https://github.com/thenamesweretakenalready/Sv443s-JokeAPI-Python-Wrapper#readme) | [thenamesweretakenalready](https://github.com/thenamesweretakenalready) |
|  | Node.js | [JokeAPI - JavaScript wrapper](https://github.com/sahithyandev/sv443-joke-api-js-wrapper#readme) | [sahithyandev](https://github.com/sahithyandev) |
|  | Go | [jokeapi-go](https://github.com/Icelain/jokeapi#readme) | [Icelain](https://github.com/Icelain) |

[(Click here to see the icon attributions)](https://sv443.net/jokeapi/v2/#other-attributions)

[🔗](https://sv443.net/jokeapi/v2/#submit)Submit a Joke:

|  |  |
| --- | --- |
| Category: |  |
| Type: |  |
| Language: |  |
| Flags: | NSFW Religious Political Racist Sexist Explicit |
| Joke: |  |

Submit >

Payload Preview:

{  
 "formatVersion": 3,  
 "category": "Misc",  
 "type": "single",  
 "joke": "",  
 "flags": {  
 "nsfw": false,  
 "religious": false,  
 "political": false,  
 "racist": false,  
 "sexist": false,  
 "explicit": false  
 },  
 "lang": "en"  
}

Note: Only Unicode characters in the range U+0000 to U+0FFF are allowed [(more info)](https://unicode-table.com/en)

[🔗](https://sv443.net/jokeapi/v2/#rate-limiting)Rate Limiting:

Due to JokeAPI being target of DoS attacks in the past, every client has a request budget of 120 requests they can send per minute.  
Sending any requests after this limit will make the API temporarily give you a HTTP 429 error until the time runs out.  
When submitting jokes, your budget is only 5 requests per minute.  
  
Almost everywhere when using JokeAPI, you will be getting the following headers which tell you all you need to know about your current request budget:

|  |  |
| --- | --- |
| **Name** | **Description** |
| Retry-After | Tells you in how many seconds your request budget gets reset back to 120 |
| RateLimit-Limit | How many requests in total you can send to JokeAPI per minute |
| RateLimit-Remaining | How many requests you have left in your budget |
| RateLimit-Reset | Contains an [IMF-fixdate](https://tools.ietf.org/html/rfc7231#section-7.1.1.1) timestamp of when your request budget gets reset back to 120 |

[🔗](https://sv443.net/jokeapi/v2/#url-parameters)URL Parameters:

URL parameters are like the settings or the configuration of an HTTP request.  
The parameters need to be prefixed by a single question mark (?) and separate key/value pairs need to be delimited from another by an ampersand (&). Keys are separated from values with an equals sign (=).  
Example: https://example.org/some/page?key1=foo&key2=bar&key3=baz ... (yellow = key/value pair)

[§](https://sv443.net/jokeapi/v2/#format-param) Format:

?format=format  
  
This parameter is global - it can be used on every single endpoint of JokeAPI (excluding the documentation and static content).  
It is used to change the response format from the default (JSON) to XML and YAML and TXT, depending on what you need.  
If the format parameter is invalid or not specified at all, the default value of JSON will be used.  
Available formats are json, xml, yaml, txt  
  
Example 1 (click to view)URL: https://v2.jokeapi.dev/joke/Any?format=xml

<data>

<category>Programming</category>

<type>single</type>

<joke>// This line doesn't actually do anything, but the code stops working when I delete it.</joke>

<flags>

<nsfw>false</nsfw>

<religious>false</religious>

<political>false</political>

<racist>false</racist>

<sexist>false</sexist>

<explicit>false</explicit>

</flags>

<id>12</id>

<lang>en</lang>

</data>

Example 2 (click to view)URL: https://v2.jokeapi.dev/joke/Any?format=txt

How many programmers does it take to screw in a light bulb?

None. It's a hardware problem.

[§](https://sv443.net/jokeapi/v2/#flags-param) Blacklist Flags:

?blacklistFlags=flag1[,flag2,...]  
  
This parameter can only be used on the "joke" endpoint.  
If it is used, jokes that match the specified flag(s) will not be served ([what is a flag?](https://sv443.net/jokeapi/v2/#blacklist-flags)).  
If you want to use multiple flags, separate them with a comma (,) or a plus sign (+).  
These are all the available flags: nsfw, religious, political, racist, sexist, explicit  
  
Example 1 (click to view)URL: https://v2.jokeapi.dev/joke/Any?blacklistFlags=nsfw  
Using the above URL makes sure you don't get any jokes that are not safe for work.  
Example 2 (click to view)URL: https://v2.jokeapi.dev/joke/Any?blacklistFlags=nsfw,racist,sexist,explicit  
Using the above URL makes sure you don't get any jokes that are not safe for work, racist and sexist and that doesn't contain explicit language (swear words, etc.).

[§](https://sv443.net/jokeapi/v2/#lang) Language:

?lang=langcode  
  
This parameter can be used on every endpoint, but it is possible that no translations or jokes exist yet for your language.  
  
There are two types of languages; system languages and joke languages. Both are separate from each other.  
All system messages like errors can have a certain system language, while jokes can only have a joke language.  
It is possible, that system languages don't yet exist for your language while jokes already do.  
If no suitable system language is found, JokeAPI will default to English.  
The language codes in JokeAPI follow the [ISO 639-1](https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes) / Alpha-2 specification.  
  
If you are interested in providing translations, please read the [Contributing Guide](https://github.com/Sv443/JokeAPI/blob/master/.github/Contributing.md#submitting-translations); it will explain it.  
You can also submit jokes in different languages. Please understand that I will not be able to ensure their proper formatting or legitimacy.  
  
Currently available system languages: cs, de, en, it, ru  
Currently available joke languages: cs, de, en, es, fr, pt  
  
Example 1 (click to view)URL: https://v2.jokeapi.dev/joke/Any?lang=ru  
Using the above URL will fetch only jokes that are Russian. If an error gets returned, it will also be in Russian, provided a translation was found.

[§](https://sv443.net/jokeapi/v2/#type-param) Joke Type:

?type=type  
  
This parameter can only be used on the "joke" endpoint.  
If it is set, you will only receive jokes with the specified joke type ([what is a joke type?](https://sv443.net/jokeapi/v2/#joke-type)).  
If it is not set or set to an invalid value, you will receive jokes of both types.  
Available types are: single, twopart  
  
Example (click to view)URL: https://v2.jokeapi.dev/joke/Any?type=twopart

{

"category": "Programming",

"type": "twopart",

"setup": "What is a dying programmer's last program?",

"delivery": "Goodbye, world!",

"flags": {

"nsfw": false,

"religious": false,

"political": false,

"racist": false,

"sexist": false,

"explicit": false

},

"id": 58,

"lang": "en"

}

(no jokes with type "single" will be served)

[§](https://sv443.net/jokeapi/v2/#contains-param) Contains:

?contains=string  
  
This parameter can only be used on the "joke" endpoint.  
If this parameter is specified, only jokes that contain the value of this parameter will be served (case insensitive).  
IMPORTANT: If the value of this parameter contains special characters, it needs to be [percent-encoded](https://en.wikipedia.org/wiki/Percent-encoding) according to the [URI syntax standard](https://tools.ietf.org/html/rfc3986#section-2.1), otherwise you might get an error or a wrong response.  
  
Example (click to view)URL: https://v2.jokeapi.dev/joke/Any?contains=C%23 (%23 = pound character / hashtag / # - so the thing searched for here is "C#")

{

"category": "Programming",

"type": "twopart",

"setup": "Why do programmers wear glasses?",

"delivery": "Because they need to C#",

"flags": {

"nsfw": false,

"religious": false,

"political": false,

"racist": false,

"sexist": false,

"explicit": false

},

"id": 51,

"lang": "en"

}

[§](https://sv443.net/jokeapi/v2/#idrange-param) ID Range:

?idRange=number[-number]  
  
This parameter can only be used on the "joke" endpoint.  
If it is used, you will only get jokes that are inside the specified range of IDs.  
The lower and upper boundary of the ID range needs to be delimited with a minus (-), comma (,) or a plus sign (+).  
If you only want to get a single joke by its ID, you can also ignore the delimiter and just use the single ID as the value for this parameter.  
As there are currently 1369 jokes, the ID range can be any range from 0 - 1368  
  
Example 1 (click to view)URL: https://v2.jokeapi.dev/joke/Any?idRange=0-55  
Using the above URL will make JokeAPI only serve you jokes that are inside the ID range of 0 to 55.  
Example 2 (click to view)URL: https://v2.jokeapi.dev/joke/Any?idRange=33  
Using the above URL will make JokeAPI only serve you the joke with the ID 33 (provided there are no other filters that are blacklisting that joke).

[§](https://sv443.net/jokeapi/v2/#amount-param) Joke Amount:

?amount=number  
  
This parameter can only be used on the "joke" endpoint.  
If it is used, you will get a certain number of jokes in a single request.  
If JokeAPI has less jokes than demanded, it will not error out but instead send the maximum amount of jokes it can serve.  
Maximum possible number is 10 and using an invalid value will make JokeAPI default to 1.  
The joke format will change from the default when you set the amount parameter to 2 or more.  
  
Click to view the format when fetching multiple jokes

{

"error": false,

"jokes": [

{

"category": "Misc",

"type": "single",

...

},

...

],

"amount": 10

}

Example (click to view)URL: https://v2.jokeapi.dev/joke/Any?amount=5  
Using the above URL will make JokeAPI serve you five jokes at once.

[§](https://sv443.net/jokeapi/v2/#dryrun-param) Submission Dry Run:

?dry-run  
  
This parameter can only be used on the "submit" endpoint. It doesn't need a value (part to the right of the equals character).  
  
If present, JokeAPI will respond just like the usual "submit" endpoint except the joke isn't actually saved to the API.  
This functionality can be used to check if a joke is valid / what is wrong with it.  
  
A successful dry run will have its error property set to false.  
It will also have a message property with a short status message and a timestamp property with a 13-character UNIX timestamp.

[🔗](https://sv443.net/jokeapi/v2/#safe-mode)Safe Mode:

Safe Mode was added in version 2.3.0. If enabled, JokeAPI will try its best to serve only jokes that are considered safe for everyone.  
Unsafe jokes are those who can be considered explicit in any way, either through the used language, its references or its [blacklist flags.](https://sv443.net/jokeapi/v2/#blacklist-flags)  
Jokes from the category Dark are also generally marked as unsafe.  
  
Note: this filter is really fine but that doesn't mean human error is out of the question.  
If you find any jokes that you consider unsafe but are served even when safe mode is activated, please [submit an issue here.](https://github.com/Sv443/JokeAPI/issues/new/choose)  
  
The Safe Mode can be enabled by adding the value-less URL parameter safe-mode to the URL.  
  
Example 1 (click to view)URL: https://v2.jokeapi.dev/joke/Any?safe-mode  
Using the above URL will make sure you won't get any unsafe jokes.  
Example 2 (click to view)URL: https://v2.jokeapi.dev/joke/Misc,Programming?format=xml&safe-mode&type=single  
This URL will yield a safe joke from the Misc or Programming category in the file format XML that's also of type single.

[🔗](https://sv443.net/jokeapi/v2/#errors)Errors:

In JokeAPI all response data, no matter from which endpoint will have a boolean "error" parameter.  
Usually, it is set to false, which means there was no error in your request or on the server.  
If the "error" parameter is set to true though, there was an error and JokeAPI will add an "internalError", "message", "causedBy", "additionalInfo" and "timestamp" parameter.  
  
The "internalError" parameter will be set to false, if the error was due to a malformed or failed request and it will be true, if the error is JokeAPI's fault.  
The "message" parameter will contain a short version of the error message and the "timestamp" parameter will contain a 13-character Unix timestamp.  
The "causedBy" parameter is an array of possible causes of this error and the "addidionalInfo" parameter contains a more descriptive error message.  
  
If possible, JokeAPI will try to translate the error messages but keep in mind there might not exist a translation for your language yet or it is not 100% perfect.  
You can view an example of an error, where an invalid category was used, just below this paragraph.  
Together with the [HTTP status code](https://sv443.net/jokeapi/v2/#status-codes), which will also be set according to the type of error, these parameters can be used to improve error handling massively from what was the error system in previous versions of JokeAPI.  
  
  
Example Error:

{

"error": true,

"internalError": false,

"code": 106,

"message": "No matching joke found",

"causedBy": [

"No jokes were found that match your provided filter(s)"

],

"additionalInfo": "The specified category is invalid - Got: \"foo\" - Possible categories are: \"Any, Misc, Programming, Dark, Pun, Spooky, Christmas\" (case insensitive)",

"timestamp": 1579170794412

}

[§](https://sv443.net/jokeapi/v2/#status-codes) Status Codes:

|  |  |  |
| --- | --- | --- |
| **Code** | **Name** | **Description** |
| 200 | Ok | The request was accepted and processed successfully and the response text contains the requested data or a status message |
| 201 | Created | The joke submission is formatted correctly and was accepted and successfully saved (unless [?dry-run](https://sv443.net/jokeapi/v2/#dryrun-param) was used) |
| 400 | Bad Request | The request you have sent to JokeAPI is formatted incorrectly and cannot be processed |
| 403 | Forbidden | You have been added to the blacklist due to malicious behavior and are not allowed to send requests to JokeAPI anymore |
| 404 | Not Found | The URL you have requested couldn't be found |
| 413 | Payload Too Large | The payload data sent to the server exceeds the maximum size of 5120 bytes |
| 414 | URI Too Long | The URL exceeds the maximum length of 250 characters |
| 429 | Too Many Requests | You have exceeded the limit of 120 requests per minute and have to wait a bit until you are allowed to send requests again |
| 500 | Internal Server Error | There was a general internal error within JokeAPI. You can get more info from the properties in the response text |
| 523 | Origin Unreachable | My server is temporarily offline due to maintenance or a dynamic IP update. Please be patient in this case. |

Success • Client Error • Server Error

[🔗](https://sv443.net/jokeapi/v2/#troubleshooting)Troubleshooting:

HTTP Error 403: First, please check if you can access all endpoints with your web browser. If not, your IP address has most likely been acting malicious and was permanently blacklisted.  
If you keep getting HTTP 403 (Forbidden) response codes in your program but not in your browser and have never shown malicious behavior before, the issue is most likely that Cloudflare is blocking your request.  
This might be due to inconsistent request data or headers that are often used in malicious requests or cyber attacks (most of the time it's gonna be the fault of the library you are using).  
A fix to this issue would be to explicitly set a User-Agent header in each of your requests.  
This is the user agent of your browser which you should be able to use in the fix described above:  
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36  
  
HTTP Error 429: You will get 429 errors occasionally if you exceed the maximum of 120 requests per minute.  
In this case, you will need to wait up to one minute to be able to talk to the API again.  
If you want to index the API or fetch all jokes, are better off going to the [GitHub repository](https://github.com/Sv443/JokeAPI), where you will find everything you need and you don't need to spam JokeAPI.

[🔗](https://sv443.net/jokeapi/v2/#endpoints)Endpoints:

An endpoint is an access point to send the HTTP requests to and get your response.  
JokeAPI offers these following endpoints:

[§](https://sv443.net/jokeapi/v2/#joke-endpoint) Get Joke:

GET https://v2.jokeapi.dev/joke/[Category/-ies]  
  
This endpoint is the one you want to call to get a joke.  
A valid [joke category](https://sv443.net/jokeapi/v2/#categories) or multiple joke categories, delimited with a comma, plus or minus character (, + -) must be provided in the URL.  
These are all the available categories: Any, Misc, Programming, Dark, Pun, Spooky, Christmas  
To see how this URL is built with certain parameters set, [visit the "Try It" section.](https://sv443.net/jokeapi/v2/#try-it)  
Full example - click to view  
  
  
  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?blacklistFlags](https://sv443.net/jokeapi/v2/#flags-param)
* [?lang](https://sv443.net/jokeapi/v2/#lang)
* [?idRange](https://sv443.net/jokeapi/v2/#id-range)
* [?contains](https://sv443.net/jokeapi/v2/#search-string)
* [?type](https://sv443.net/jokeapi/v2/#joke-type)
* [?amount](https://sv443.net/jokeapi/v2/#amount)

[(More info on filtering parameters)](https://sv443.net/jokeapi/v2/#url-parameters)

[§](https://sv443.net/jokeapi/v2/#info-endpoint) Info:

GET https://v2.jokeapi.dev/info  
  
This endpoint provides a lot of information about JokeAPI and its jokes:  
- The version number  
- The amount of jokes  
- All the available categories, flags, types and formats  
- A 13-character UNIX timestamp  
- The URL to a joke submission form  
- A list of languages (code and name) JokeAPI currently supports  
- The minimum and maximum values of an ID range per each language  
- The amount of safe jokes there are per language  
- A string with some information, like a message of the day  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#categories-endpoint) Categories:

GET https://v2.jokeapi.dev/categories  
  
This endpoint returns a list / an array of all available [joke categories](https://sv443.net/jokeapi/v2/#categories), all available [category aliases](https://sv443.net/jokeapi/v2/#category-aliases) and a 13-character UNIX timestamp.  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#langcodes-endpoint) Language Code:

GET https://v2.jokeapi.dev/langcode/[Language]  
  
This endpoint returns the ISO 639-1 language code of a provided language. It is searched with a fuzzy search, so you just have to provide the approximate language name.  
The resulting language code is to be used in [fetching](https://sv443.net/jokeapi/v2/#joke-endpoint) and [submitting](https://sv443.net/jokeapi/v2/#submit-endpoint) jokes in different languages.  
  
Example 1 - click to view  
https://v2.jokeapi.dev/langcode/german

{

"error": false,

"code": "de"

}

Example 2 - click to view  
https://v2.jokeapi.dev/langcode/sw3d1sh

{

"error": false,

"code": "sv"

}

Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#languages-endpoint) Supported Languages:

GET https://v2.jokeapi.dev/languages  
  
This endpoint returns lists of supported languages in jokes and supported languages in system messages (error messages).  
Also, it returns a list of possible ISO 639-1 language codes you can use to submit a joke or add a translation.  
Example - click to view

Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#flags-endpoint) Flags:

GET https://v2.jokeapi.dev/flags  
  
This endpoint returns a list / an array of all available [blacklist flags](https://sv443.net/jokeapi/v2/#blacklist-flags) and a 13-character UNIX timestamp.  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#formats-endpoint) Formats:

GET https://v2.jokeapi.dev/formats  
  
This endpoint returns a list / an array of all available [response formats](https://sv443.net/jokeapi/v2/#response-formats) and a 13-character UNIX timestamp.  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#ping-endpoint) Ping:

GET https://v2.jokeapi.dev/ping  
  
This endpoint returns a parameter named "ping" that contains the word "Pong!" and a 13-character UNIX timestamp.  
It is intended for external uptime monitoring but you can also use it if you want to.  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)
* [?lang](https://sv443.net/jokeapi/v2/#lang)

[§](https://sv443.net/jokeapi/v2/#endpoints-endpoint) Endpoints:

GET https://v2.jokeapi.dev/endpoints  
  
This endpoint returns a list / an array of all available endpoints, their usage (method, url and supported parameters) and a short description each.  
Note: the lang parameter will not work here.  
  
Supported URL parameters:

* [?format](https://sv443.net/jokeapi/v2/#response-formats)

[§](https://sv443.net/jokeapi/v2/#submit-endpoint) Submit Joke:

POST https://v2.jokeapi.dev/submit  
  
This endpoint is used to programatically submit a joke to be reviewed and added to JokeAPI's official jokes.  
If you instead just want to manually submit a joke, please [click here.](https://sv443.net/jokeapi/v2/#submit)  
  
To ensure backwards compatibility you can use the PUT method even though it is discouraged.  
Using the PUT method might be deprecated in the future.  
  
The payload needs to be sent as JSON. Different [formats](https://sv443.net/jokeapi/v2/#response-formats) are not allowed here.  
The request payload has to follow the same object structure that the jokes are served in when using the "joke" endpoint.  
  
Make sure to add a "formatVersion" property, though, which just contains the current joke format version (3).  
This is needed since there were a few changes to the object format since the previous versions and this serves as a kind of acknowledgement.  
  
The joke category has to be any valid [category](https://sv443.net/jokeapi/v2/#categories), excluding "Any".  
If the [joke type](https://sv443.net/jokeapi/v2/#joke-type) is set to "single", the properties "setup" and "delivery" will need to be omitted and the joke is to be put in the "joke" property.  
If the [joke type](https://sv443.net/jokeapi/v2/#joke-type) is set to "twopart", the "joke" property is to be replaced with the properties "setup" and "delivery".  
The ["flags"](https://sv443.net/jokeapi/v2/#blacklist-flags) property needs to be an object that contains the boolean properties "nsfw", "religious", "political", "racist", "sexist" and "explicit". These need to be set accordingly.  
The joke ID doesn't need to be provided as it will be automatically assigned to the joke.  
  
This endpoint has a unique URL parameter; [?dry-run](https://sv443.net/jokeapi/v2/#dryrun-param), which if present will make the API respond just like the usual "submit" endpoint except the joke isn't actually saved to the API.  
This functionality can be used to check if a joke is valid / what is wrong with it without having the joke actually submitted to the API.  
  
If the joke is valid and the server accepted the submission, you will get a response with the [status code 201.](https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/201)  
  
Example Payload:

{

"formatVersion": 3,

"category": "Misc",

"type": "single",

"joke": "A horse walks into a bar...",

"flags": {

"nsfw": true,

"religious": false,

"political": true,

"racist": false,

"sexist": false,

"explicit": false

},

"lang": "en"

}

Supported URL parameters:

* [?dry-run](https://sv443.net/jokeapi/v2/#dryrun-param)

Click here to see how the server responds to a joke submissionServer Response:

{

"error": false,

"message": "Joke submission was successfully saved. It will soon be checked out by the author.",

"submission": {

"formatVersion": 2,

"category": "Misc",

"type": "single",

"joke": "A horse walks into a bar...",

"flags": {

"nsfw": true,

"religious": false,

"political": true,

"racist": false,

"sexist": false,

"explicit": false

},

"lang": "en"

},

"timestamp": 1579685839560

}

Click here to see how the server responds to a submission dry runServer Response:

{

"error": false,

"message": "Dry Run complete! No errors were found.",

"timestamp": 1604627398681

}

[🔗](https://sv443.net/jokeapi/v2/#filtering)Filtering Jokes:

I created JokeAPI with the main purpose of it having many versatile filtering options as other APIs didn't offer this amount of customization.  
In this section you will learn how the filtering works.  
  
There are six different filtering methods in JokeAPI:

* [Joke Category](https://sv443.net/jokeapi/v2/#categories)
* [Blacklist Flags](https://sv443.net/jokeapi/v2/#blacklist-flags)
* [Response Format](https://sv443.net/jokeapi/v2/#response-formats)
* [Joke Type](https://sv443.net/jokeapi/v2/#joke-type)
* [Search String](https://sv443.net/jokeapi/v2/#search-string)
* [ID Range](https://sv443.net/jokeapi/v2/#id-range)
* [Language](https://sv443.net/jokeapi/v2/#lang)

[🔗](https://sv443.net/jokeapi/v2/#api-tokens)API Tokens:

JokeAPI has a way of whitelisting certain clients. This is achieved through an API token.  
At the moment, you will only receive one of these tokens temporarily if something breaks or if you are a business and need more than 120 requests per minute.  
If you do end up receiving a token, using it is as easy as adding an Authorization header with the actual token as its value.  
You will receive a response header called Token-Valid which will contain the number 1 if the token is valid.  
If the token is invalid, the Token-Valid header will be set to 0 and JokeAPI will treat your request as if you didn't supply a token in the first place.

[🔗](https://sv443.net/jokeapi/v2/#examples)Examples:

These are some examples in some commonly used languages to show you how you could implement JokeAPI:  
  
Node.js (click to show)Node.js:

const https = require("https"); *// Native module, no need to explicitly install*

const baseURL = "https://v2.jokeapi.dev";

const categories = ["Programming", "Misc", "Pun", "Spooky", "Christmas"];

const params = [

"blacklistFlags=nsfw,religious,racist",

"idRange=0-100"

];

https.get(`${baseURL}/joke/${categories.join(",")}?${params.join("&")}`, res => {

console.log("\n");

res.on("data", chunk => {

*// On data received, convert it to a JSON object*

let randomJoke = JSON.parse(chunk.toString());

if(randomJoke.type == "single")

{

*// If type == "single", the joke only has the "joke" property*

console.log(randomJoke.joke);

console.log("\n");

}

else

{

*// If type == "twopart", the joke has the "setup" and "delivery" properties*

console.log(randomJoke.setup);

setTimeout(() => {

console.log(randomJoke.delivery);

console.log("\n");

}, 3000);

}

});

res.on("error", err => {

*// On error, log to console*

console.error(`Error: ${err}`);

});

});

Web-JavaScript (click to show)Web-JS:

var baseURL = "https://v2.jokeapi.dev";

var categories = ["Programming", "Misc", "Pun", "Spooky", "Christmas"];

var params = [

"blacklistFlags=nsfw,religious,racist",

"idRange=0-100"

];

var xhr = new XMLHttpRequest();

xhr.open("GET", baseURL + "/joke/" + categories.join(",") + "?" + params.join("&"));

xhr.onreadystatechange = function() {

if(xhr.readyState == 4 && xhr.status < 300) *// readyState 4 means request has finished + we only want to parse the joke if the request was successful (status code lower than 300)*

{

var randomJoke = JSON.parse(xhr.responseText);

if(randomJoke.type == "single")

{

*// If type == "single", the joke only has the "joke" property*

alert(randomJoke.joke);

}

else

{

*// If type == "single", the joke only has the "joke" property*

alert(randomJoke.setup);

alert(randomJoke.delivery);

}

}

else if(xhr.readyState == 4)

{

alert("Error while requesting joke.\n\nStatus code: " + xhr.status + "\nServer response: " + xhr.responseText);

}

};

xhr.send();

C# (click to show)C#:

using System;

using System.Threading.Tasks;

using System.Net.Http;

namespace JokeAPIExample

{

class Program

{

static async Task Main(string[] args)

{

string joke = await GetJoke();

Console.WriteLine("Joke = " + joke);

}

public async static Task GetJoke()

{

var client = new HttpClient();

var response = await client.GetAsync("https://jokeapi.dev/joke/Any?format=txt&type=single&blacklistFlags=nsfw,racist,sexist&lang=en");

var joke = await response.Content.ReadAsStringAsync();

return joke;

}

}

}

C# - Advanced (click to show)C#:

using System;

using System.Threading.Tasks;

using System.Net.Http;

using Newtonsoft.Json; *// Install with NuGet Package Manager: https://www.nuget.org/packages/Newtonsoft.Json/12.0.3?\_src=template*

namespace TestHttpRequest

{

class Program

{

static async Task Main(string[] args)

{

Joke randomJoke = await GetJokeAdvanced();

if (randomJoke.type == "single")

{

Console.WriteLine(randomJoke.joke);

}

else

{

Console.WriteLine(randomJoke.setup);

System.Threading.Thread.Sleep(3000);

Console.WriteLine(randomJoke.delivery);

}

}

public async static Task GetJokeAdvanced()

{

const string baseUrl = "https://jokeapi.dev";

string[] categories = { "Programming", "Miscellaneous", "Pun", "Spooky", "Christmas" };

string[] parameters = {

"blacklistFlags=nsfw,religious,racist,sexist",

"idRange=0-100"

};

string requestUrl = $"{baseUrl}/joke/{string.Join(",", categories)}?{string.Join("&", parameters)}";

Joke randomJoke;

using (var httpClient = new HttpClient())

{

var json = await httpClient.GetStringAsync(requestUrl);

randomJoke = JsonConvert.DeserializeObject(json);

}

return randomJoke;

}

}

public class Joke

{

public string type { get; set; }

public string joke { get; set; }

public string setup { get; set; }

public string delivery { get; set; }

public int id { get; set; }

public Flags flags { get; set; }

}

public class Flags

{

public bool nsfw { get; set; }

public bool religious { get; set; }

public bool political { get; set; }

public bool racist { get; set; }

public bool sexist { get; set; }

}

}

[🔗](https://sv443.net/jokeapi/v2/#terminology)Terminology:

You might come across some terms in this documentation you don't understand. This section teaches you what those terms mean.

[§](https://sv443.net/jokeapi/v2/#categories) Categories:

JokeAPI has a first, coarse filter that just categorizes the jokes depending on what the joke is about or who the joke is directed at.  
A joke about programming will be in the "Programming" category, dark humor will be in the "Dark" category and so on.  
If you want jokes from all categories, you can instead use "Any", which will make JokeAPI randomly choose a category.  
Since version 2.3.0, there are now aliases to these categories. [This section](https://sv443.net/jokeapi/v2/#category-aliases) explains how they work.  
  
These are all the available categories: Any, Misc, Programming, Dark, Pun, Spooky, Christmas  
To set the category you want, you need to add it to the URL path, just after the "/joke" endpoint, like this: https://v2.jokeapi.dev/joke/Programming

[§](https://sv443.net/jokeapi/v2/#category-aliases) Category Aliases:

Since version 2.3.0, there are now category aliases. You can use these as a shorter or just different version of the normal set of [categories.](https://sv443.net/jokeapi/v2/#categories)  
Internally, these aliases will be resolved to one of the normal set of [categories](https://sv443.net/jokeapi/v2/#categories) and your request (including the API's response) will be treated as if you used the resolved value.  
  
The following is a list of aliases and what they will be resolved to internally:

|  |  |
| --- | --- |
| **Alias** | **Resolved Category** |
| Miscellaneous | Misc |
| Coding | Programming |
| Development | Programming |
| Halloween | Spooky |

Example - click to view

[§](https://sv443.net/jokeapi/v2/#blacklist-flags) (Blacklist-) Flags:

Blacklist Flags (or just "Flags") are a more fine layer of filtering.  
Multiple flags can be set on each joke and they tell you something about the offensiveness of each joke.  
You can tell JokeAPI to not give you jokes that have the specified flags by using the [?blacklistFlags](https://sv443.net/jokeapi/v2/#flags-param) URL parameter.  
  
These are all the available flags: nsfw, religious, political, racist, sexist, explicit

[§](https://sv443.net/jokeapi/v2/#response-formats) (Response-) Formats:

Response Formats (or just "Formats") are a way to get your data in a different file format.  
Maybe your environment or language doesn't support JSON natively. In that case, JokeAPI is able to convert the JSON-formatted joke to a different format for you.  
You can tell JokeAPI which file format to respond with by using the [?format](https://sv443.net/jokeapi/v2/#format-param) URL parameter.  
  
These are all the available formats: json, xml, yaml, txt

[§](https://sv443.net/jokeapi/v2/#joke-type) Joke Type:

Each joke comes with one of two types: single or twopart.  
If a joke is of type "twopart", it has a setup string and a delivery string, which are both part of the joke.  
They are separated because you might want to present the users the delivery after a timeout or in a different section of the UI.  
A joke of type "single" only has a single string, which is the entire joke.  
If you only want one type of joke, you can use the [?type](https://sv443.net/jokeapi/v2/#type-param) URL parameter.  
  
These are the available types: single, twopart

[§](https://sv443.net/jokeapi/v2/#search-string) Search String:

If the search string filter is used, only jokes that contain the specified string will be returned.  
You can specify this search string by using the [?contains](https://sv443.net/jokeapi/v2/#contains-param) URL parameter.  
IMPORTANT: If the value of this parameter contains special characters, it needs to be [percent-encoded](https://tools.ietf.org/html/rfc3986#section-2.1) according to the URI standard ([MDN Reference](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/encodeURIComponent)), otherwise you might get an error or a wrong response.

[§](https://sv443.net/jokeapi/v2/#id-range) ID Range:

If this filter is used, you will only get jokes that are within the provided range of IDs.  
You don't necessarily need to provide an ID *range* though, a single ID will work just fine as well.  
Example: an ID range of 0-9 will mean you will only get one of the first 10 jokes, while an ID range of 5 will mean you will only get the 6th joke.  
You can set the ID range by using the [?idRange](https://sv443.net/jokeapi/v2/#idrange-param) URL parameter.  
As there are currently 1369 jokes, the ID range can be anywhere in the range of 0 to 1368  
This range varies from language to language though, so I recommend taking a look at the [/info endpoint](https://sv443.net/jokeapi/v2/#info-endpoint), which contains all possible ID ranges.

[§](https://sv443.net/jokeapi/v2/#amount) Joke Amount:

This filter allows you to set a certain amount of jokes to receive in a single call to the ["Get Joke" endpoint.](https://sv443.net/jokeapi/v2/#joke-endpoint)  
You can set it using the [?amount](https://sv443.net/jokeapi/v2/#amount-param) URL parameter.  
Setting the filter to an invalid number will result in the API defaulting to serving a single joke.  
Setting it to a number larger than 10 will make JokeAPI default to the maximum (10).  
If you request more than 5 jokes, the API will try to serve them encoded with Brotli, Gzip or Deflate, depending on whether or not you set a [Accept-Encoding](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Accept-Encoding) header.

[🔗](https://sv443.net/jokeapi/v2/#contributors)Contributors:

The following is a list of people that have directly contributed to JokeAPI's development.  
I am very grateful for this voluntary effort to improve JokeAPI and would like to thank anyone who contributed :)

Liam Townsley

[contact@liamtownsley.me](mailto:contact@liamtownsley.me?subject=JokeAPI%20Contribution)[https://liamtownsley.me](https://liamtownsley.me/)

* Fixed grammar mistakes
* Removed dead dependent

ThatCopy

<https://github.com/ThatCopy>

* Added Czech translation
* Fixed grammar mistakes

Sahithyan Kandathasan

[sahithyan2701@gmail.com](mailto:sahithyan2701@gmail.com?subject=JokeAPI%20Contribution)

* Added changelog in Markdown format

Afarfeel

<https://github.com/Afarfeel>

* Added Italian translation

sournachos

<https://github.com/sournachos>

* Added Spanish jokes

If you want to contribute to JokeAPI, please [read this guide.](https://github.com/Sv443/JokeAPI/blob/master/.github/Contributing.md)

[🔗](https://sv443.net/jokeapi/v2/#dependents)Dependents:

These are some of the projects that depend on JokeAPI:

* [dozens Advices](https://github.com/ZephyrVentum/dozens-Advices) by [ZephyrVentum](https://github.com/ZephyrVentum)
* [shadow-bot](https://github.com/QGIsK/Shadow-bot) by [QGIsK](https://github.com/QGIsK)
* [Jokepy](https://github.com/aksty/Jokepy) by [aksty](https://github.com/aksty/Jokepy)
* [Dark1](https://github.com/whiteadi/Dark1) by [whiteadi](https://github.com/whiteadi)
* [Prejudice-Networks](https://github.com/LiamTownsley/Prejudice-Networks) by [LiamTownsley](https://github.com/LiamTownsley)
* <https://irshad.ml/humour.html> by [draco-malfoy](https://github.com/draco-malfoy)
* [Random-Joke-Generator-with-Flutter](https://github.com/variousnabil/Random-Joke-Generator-with-Flutter) by [variousnabil](https://github.com/variousnabil)
* [comma](https://thatcopy.pw/comma) by [ThatCopy](https://github.com/ThatCopy)
* [Joke Teller](https://github.com/AlHood77/Joke_Teller) by [AlHood77](https://github.com/AlHood77)
* [Jokes plugin for Craft CMS 3.x](https://github.com/remcoov/jokes) by [remcoov](https://github.com/remcoov)
* [PoshBot.Joker](https://github.com/ToastIT-dev/PoshBot.Joker) by [ToastIT-dev](https://github.com/ToastIT-dev)
* [JokeAPI\_ComputerCraft](https://github.com/Sv443/JokeAPI_ComputerCraft) by [Sv443](https://github.com/Sv443)
* [Dev Dad Jokes](https://github.com/jonathanbossenger/devdadjokes) by [Jonathan Bossenger](https://github.com/jonathanbossenger)

([Submit an issue](https://github.com/Sv443/JokeAPI/issues/new/choose) to get your project added here)

[🔗](https://sv443.net/jokeapi/v2/#legal)Legal Stuff:

[§](https://sv443.net/jokeapi/v2/#license) License:

JokeAPI is licensed under the MIT License.  
Before modifying and / or redistributing the source code, please take two minutes to read the license text.  
[You can view the full license text by clicking here.](https://sv443.net/LICENSE)

[§](https://sv443.net/jokeapi/v2/#privacy) Privacy:

JokeAPI has to collect some anonymized data to work.  
To view a list with details, please click here.  
To view the privacy policy, please [click here.](https://sv443.net/privacypolicy/en)  
To view the site notice / imprint, [click here.](https://sv443.net/imprint/en)

[§](https://sv443.net/jokeapi/v2/#disclaimer) Disclaimer:

I will hereby not claim any legal responsibility or liability for JokeAPI and the jokes it serves (especially those from the "Dark" category).  
Whether it is used maliciously or breaks something in your project or someone gets offended by a joke, I can't be held accountable.  
Additionally, I will only be able to provide security updates for a small selection of versions, a list of which you can find [here.](https://github.com/Sv443/JokeAPI/security/policy#supported-versions)  
I am doing my best to ensure security and stability but there's only so much a single developer can do.  
Please report any issue that may arise to [the GitHub issue tracker](https://github.com/Sv443/JokeAPI/issues/new/choose) and I will try my best to fix it as soon as possible.  
If you want to contact me, you can [join my Discord server](https://sv443.net/discord) (fastest way to contact me) or send me an E-Mail: (click to show)

[🔗](https://sv443.net/jokeapi/v2/#attributions)Attributions:

These are the attributions to everything made by third parties that JokeAPI is based on or needs to function.

[§](https://sv443.net/jokeapi/v2/#fonts) Fonts:

JokeAPI uses these third party fonts on its documentation page:

* [Cascadia Code](https://github.com/Microsoft/Cascadia-Code) by [Microsoft](https://github.com/Microsoft)
* [Roboto](https://github.com/Google/Roboto) by [Google](https://github.com/Google)

[§](https://sv443.net/jokeapi/v2/#other-attributions) Other Attributions:

Third party icons used on this page:

* The Rust icon is owned by [Mozilla.](https://mozilla.org/)
* The Python icon is owned by the [Python Software Foundation.](https://www.python.org/psf/)
* The Node.js icon is a trademark of the [OpenJS Foundation.](https://openjsf.org/)
* The Go icon is owned by [Google.](https://about.google/)

The Sv443 Network is not endorsed by or affiliated with any of the above mentioned entities.  
All of these icons are protected from being indexed by the majority of web crawlers.  
  
  
The rectangular badges, like the ones found in the [uptime](https://sv443.net/jokeapi/v2/#uptime) and [security](https://sv443.net/jokeapi/v2/#security) sections, are powered by [Shields.io](https://shields.io/)

[§](https://sv443.net/jokeapi/v2/#libraries) Libraries:

JokeAPI uses these libraries:

* [Code-Prettify](https://github.com/google/code-prettify) by [Google](https://github.com/Google)
* [JSLib](https://github.com/Sv443/JSLib) by [Sv443](https://github.com/Sv443)

These libraries are served using [jsDelivr.](https://www.jsdelivr.com/)

[§](https://sv443.net/jokeapi/v2/#dependencies) Dependencies:

These are the packages JokeAPI depends on (excluding dependencies of dependencies and devDependencies):

* [@pm2/io](https://npmjs.com/package/@pm2/io)
* [dotenv](https://npmjs.com/package/dotenv)
* [farmhash](https://npmjs.com/package/farmhash)
* [fs-extra](https://npmjs.com/package/fs-extra)
* [fuse.js](https://npmjs.com/package/fuse.js)
* [http-ratelimit](https://npmjs.com/package/http-ratelimit)
* [js2xmlparser](https://npmjs.com/package/js2xmlparser)
* [json-to-pretty-yaml](https://npmjs.com/package/json-to-pretty-yaml)
* [mysql](https://npmjs.com/package/mysql)
* [node-wrap](https://npmjs.com/package/node-wrap)
* [promise-all-sequential](https://npmjs.com/package/promise-all-sequential)
* [rate-limiter-flexible](https://npmjs.com/package/rate-limiter-flexible)
* [request-ip](https://npmjs.com/package/request-ip)
* [snyk](https://npmjs.com/package/snyk)
* [svjsl](https://npmjs.com/package/svjsl)
* [url-parse](https://npmjs.com/package/url-parse)
* [xss](https://npmjs.com/package/xss)

© Copyright 2018-2021 Sv443 - licensed under the [MIT license](https://sv443.net/LICENSE)  
This project is a part of the [Sv443 Network](https://sv443.net/)

<https://www.voicerss.org/sdk/javascript.aspx>

# **Text-to-speech JavaScript SDK**

The Voice RSS **Text-to-Speech JavaScript SDK** wraps Voice RSS [Text-to-Speech API](https://www.voicerss.org/api).

The Voice RSS **Text-to-Speech JavaScript SDKs** will help to make integration with our [Text-to-Speech API](https://www.voicerss.org/api) faster and easer.

If you have any questions or suggestions please feel free to [contact us via e-mail](https://www.voicerss.org/contact.aspx).

## Documentation

To integrate an application with the Voice RSS **Text-to-Speech JavaScript SDK** it needs to add reference to the file voicerss-tts.min.js.

### **Convert text-to-speech**

The following example demonstrates asynchronous converting text-to-speech and plays it in an internet browser:

<!DOCTYPE html>

<html>

<head>

<title></title>

<meta charset='utf-8' />

<script src='voicerss-tts.min.js'></script>

</head>

<body>

<script>

VoiceRSS.speech({

key: '<API key>',

src: 'Hello, world!',

hl: 'en-us',

v: 'Linda',

r: 0,

c: 'mp3',

f: '44khz\_16bit\_stereo',

ssml: false

});

</script>

</body>

</html>