CountAPI

Counting should be easy, right?

CountAPI

This API allows you to create simple numeric counters. IaaS, Integer as a Service.

It goes down to:

- Create a counter and restrict its operations
- Reset the value of a counter
- Increment/decrement a counter

All counters are accesible if you know the key and there are not private counters (yet?).

Want to track the number of hits a page had? Sure.
Want to know the number of users that clicked on the button "Feed Cow"? There you go.

So far, ... keys have been created and there have been ... requests served being ... key updates.
This page has been visited ... times.

TL;DR

Each counter is identified inside a **namespace** with a **key**.
The namespace should be unique, so its recommend using your site's domain.
Inside each namespace you can generate all the counters you may need.

The **hit** endpoint provides increment by one counters directly. Each time its requested the counter will increase by one:

```
https://api.countapi.xyz/hit/namespace/key
⇒ 200 { "value": 1234 }
```

Want to start counting right away? Check the examples below!

Example

Lets say you want to display the number of pageviews a site received.

```
<div id="visits">...</div>
```

Remember to change mysite.com with your site's **domain**.

Using countapi-js npm v1.0.2

```
import countapi from 'countapi-js';

countapi.visits().then((result) => {
    console.log(result.value);
});
```

Using JSONP

```
<script>
function cb(response) {
    document.getElementById('visits').innerText = response.value;
}
</script>
<script async src="https://api.countapi.xyz/hit/mysite.com/visits?callback=cb"></scri
```

Using XHR

```
var xhr = new XMLHttpRequest();
xhr.open("GET", "https://api.countapi.xyz/hit/mysite.com/visits");
xhr.responseType = "json";
xhr.onload = function() {
    document.getElementById('visits').innerText = this.response.value;
}
xhr.send();
```

Using jQuery

```
$.getJSON("https://api.countapi.xyz/hit/mysite.com/visits", function(response) {
    $("#visits").text(response.value);
});
```

Multiple pages

If you want to have a counter for each individual page you can replace `visits` with a unique identifier for each page, i.e. `index`, `contact`, `item-1234`. Check the right format a key must have.

Alternatively, you can use some reserved words that are replaced server-side.
For example, if a request is made from `https://mysite.com/example/page`:

- `:HOST:` will be replaced with `mysite.com`
- `:PATHNAME:` will be replaced with `examplepage`

> Note: Reserved words are padded with dots if their length is less than three.

So you could use something like:

```
https://api.countapi.xyz/hit/mysite.com/:PATHNAME:
```

Or even more generic (though not recommended):

```
https://api.countapi.xyz/hit/:HOST:/:PATHNAME:
```

> **Important**: if you want to know the actual key used you can check the `X-Path` header.

Counting events

You can use the API to count any kind of stuff, lets say:

```
<button onclick="clicked()">Press Me!</button>
<script>
function clicked() {
    var xhr = new XMLHttpRequest();
    xhr.open("GET", "https://api.countapi.xyz/hit/mysite.com/awesomeclick");
    xhr.responseType = "json";
    xhr.onload = function() {
        alert(`This button has been clicked ${this.response.value} times!`);
    }
    xhr.send();
}
</script>
```

Press Me!

Libraries

The official javascript promise based wrapper is available in countapi-js npm v1.0.2.

Roadmap

If this API starts getting some traction, I have some ideas in mind:

- Being able to open a SSE stream and receive updates in real time avoiding polling
- Enable batch creating/updating
- Unique counting?
- Float numbers?
- Generate a secret key to update/reset keys

FAQ

Where is the API documentation?

Scroll a bit further.

Is the API free?

Completely free.

Rate Limiting?

Key retrieving and updating has **no** limits whatsoever. Key **creation** is limited to 20/IP/s.

Can I delete a key?

You can't, just let the key expire.
If you created a key with a wrong configuration you can always create another key.

Will I blow up the CountAPI server?

CountAPI is using Redis as database, a very fast key-value solution.
If you are planning to make **tens of thousands** of requests, I'll be glad if you ping me letting me know.

Issues/Contact

If you have issues, suggestions or just want to contact me send me an email.

API

Namespaces

Namespaces are meant to avoid name collisions. You may specify a namespace during the creation of a key. Its recommend use the domain of the application as namespace to avoid collision with other websites.
If the namespace is not specified the key is assigned to the `default` namespace. If your key resides in the default namespace you don't need to specify it.

Endpoints

All requests support cross-origin resource sharing (CORS) and SSL.
You can use JSONP sending the callback parameter. JSONP requests will never fail, they will include the HTTP code in the response.
Also a 1x1 GIF image is supported sending `?img`.

Base API path: https://api.countapi.xyz

In the case of a server failure, the API will send:

```
⇒ 500 { "error": "Error description" }
```

/get/:namespace?/:key

Get the value of a key. Optionally specify the namespace.

```
https://api.countapi.xyz/get/test
⇒ 200 { "value": 42 }

https://api.countapi.xyz/get/mysite.com/test
⇒ 200 { "value": 24 }
```

```
https://api.countapi.xyz/get/nonexisting
⇒ 404 { "value": null }
```

/set/:namespace?/:key?value=:value

Set the value of a key. Optionally specify the namespace. The key **must** be created with `enable_reset` set to `1` (true).

This endpoint will return the previous value before the assignation.

```
https://api.countapi.xyz/set/test?value=69
⇒ 200 { "old_value": 42, "value": 69 }

https://api.countapi.xyz/set/mysite.com/test?value=96
⇒ 200 { "old_value": 24, "value": 96 }
```

```
https://api.countapi.xyz/set/resetdisabled?value=33
⇒ 403 { "old_value": 1234, "value": 1234 }

https://api.countapi.xyz/set/nonexisting?value=33
⇒ 404 { "old_value": null, "value": null }
```

/update/:namespace?/:key?amount=:amount

Updates a key with `+/- amount`. Optionally specify the namespace. The `amount` **must** be within `update_lowerbound` and `update_upperbound` specified during the creation of the key.

```
https://api.countapi.xyz/update/test?amount=5 (value was 42)
⇒ 200 { "value": 47 }

https://api.countapi.xyz/update/mysite.com/test?amount=-7 (value was 53)
⇒ 200 { "value": 46 }
```

```
https://api.countapi.xyz/update/outofrange?amount=3 (value was 47, update_upperbound=2)
⇒ 403 { "value": 47 }

https://api.countapi.xyz/update/nonexisting?amount=1
⇒ 404 { "value": null }
```

/hit/:namespace?/:key

An easier way to track incrementing by one keys. This endpoint will create a key if it doesn't exists and increment it by one on each subsequent request. Optionally specify a namespace.
The key created has the following properties:

- `enable_reset` to `0` (false)
- `update_lowerbound` to `0`
- `update_upperbound` to `1`

Effectively making the key only incrementable by one.

```
https://api.countapi.xyz/hit/mysite.com/visits (value was 35)
⇒ 200 { "value": 36 }

https://api.countapi.xyz/hit/nonexisting (key is created)
⇒ 200 { "value": 1 }
```

/create

Creates a key.
All parameters are optional

name	default	description
key	New UUID	Name of the key
namespace	default	Namespace to store the key
value	0	The initial value stored
enable_reset	0	Allows the key to be resetted with **/set**
update_lowerbound	-1	Restrict update to not subtract more than this number. This number **must** be negative or zero.
update_upperbound	1	Restrict update to not add more than this number. This number **must** be positive or zero.

Note about **expiration**: Every time a key is updated its expiration is set to **6 months**. So don't worry, if you still using it, it won't expire.

Keys and namespaces must have at least 3 characters and less or equal to 64. Keys and namespaces must match: **^[A-Za-z0-9_\-.]{3,64}$**

```
https://api.countapi.xyz/create
⇒ 200 {"namespace":"default", "key":"6d5891ff-ebda-48fb-a760-8549d6a3bf3a", "value":0}

https://api.countapi.xyz/create?namespace=mysite.com&value=42
⇒ 200 {"namespace":"mysite.com", "key":"33606dbe-4800-4228-b042-5c0fb8ec8f08",
"value":42}

https://api.countapi.xyz/create?key=counter&expiration=60
⇒ 200 { "namespace": "default", "key":"counter", "value": 0 }
```

```
https://api.countapi.xyz/create?name=alreadycreated (the key already existed)
⇒ 409 { "namespace": null, "key": null, "value": null }
```

/info/:namespace?/:key

Get information about a key. Optionally specify the namespace.

```
https://api.countapi.xyz/info/test
⇒ 200 {
    "namespace": "default",
    "key": "test",
    "ttl": 321,
    "value": 42,
    "enable_reset": false,
    "update_upperbound": 1,
    "update_lowerbound": 1
}
```

```
https://api.countapi.xyz/info/nonexisting
⇒ 404 {
    "namespace": null,
    "key": null,
    "ttl": null,
    "value": null,
    "enable_reset": null,
    "update_upperbound": null,
    "update_lowerbound": null
}
```

/stats

Get some CountAPI stats

```
https://api.countapi.xyz/stats
⇒ 200 {
    "keys_created": …,
    "keys_updated": …,
    "requests": …,
    "version": "…"
}
```

made by mlomb