Spreadsheets

JavaScript, API

ConfrontJS, 2022-03-26

Tomasz Stachewicz

@_tomash

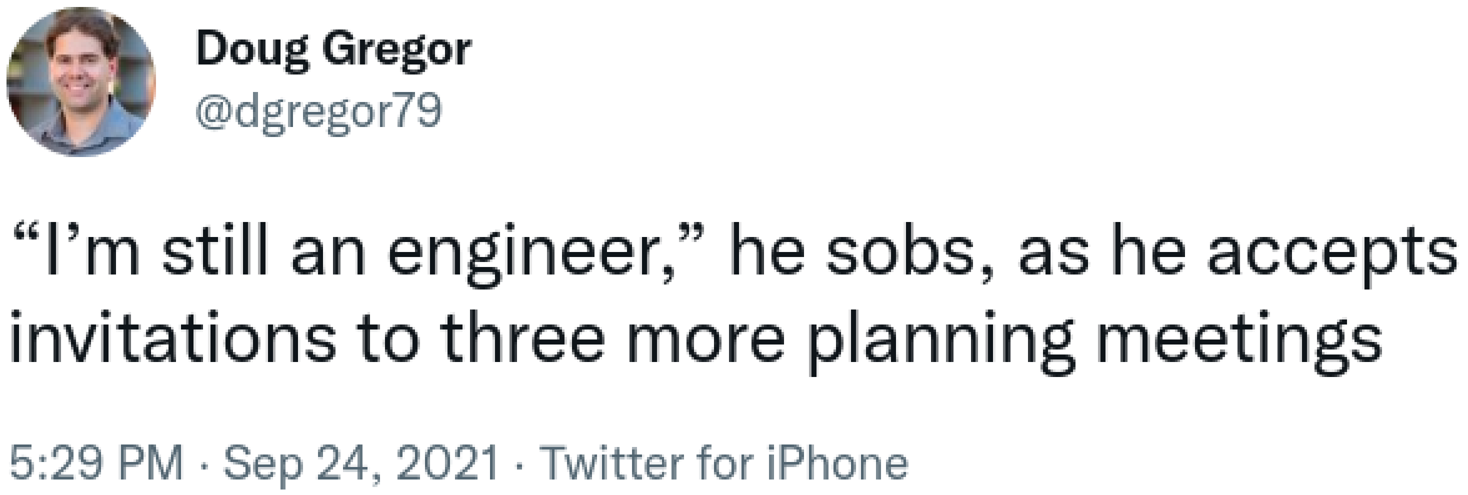
coding since 2005

->

Previously: Founder, CEO @ Rebased

Now: Senior Dev Manager @ Shopify

Still an engineer!



Doug Gregor
@dgregor79

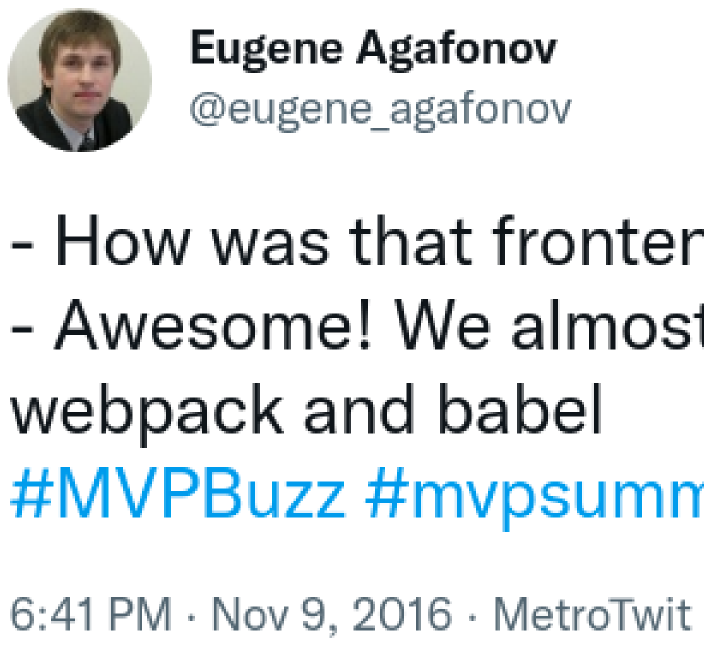
“I’m still an engineer,” he sobs, as he accepts invitations to three more planning meetings

5:29 PM · Sep 24, 2021 · Twitter for iPhone

We are developers

We like building stuff

But sometimes developing a whole application is an overkill.



Eugene Agafonov
@eugene_agafonov

- How was that frontend hackathon?
- Awesome! We almost installed and configured webpack and babel
#MVPBuzz #mvpsummit

6:41 PM · Nov 9, 2016 · MetroTwit

That’s why we have no-code and low-code

But let’s talk spreadsheets.

Spreadsheets do the job™.

”Any dedicated software
must perform substantially better
than Excel at given task”.

(not mine, author unknown)

- familiar
- easy to use and adapt
- data <-> interface

But not very interesting to developers

What if we could script them with JS

… and talk with external API?

Custom Functions in Google Sheets

Google Sheets offers hundreds of built-in functions like `AVERAGE`, `SUM`, and `VLOOKUP`. When these aren't enough for your needs, you can use Google Apps Script to write custom functions — say, to convert meters to miles or fetch live content from the Internet — then use them in Google Sheets just like a built-in function.

Class UrlFetchApp

☆☆☆☆☆

Fetch resources and communicate with other hosts over the Internet.

This service allows scripts to communicate with other applications or access other resources on the web by fetching URLs. A script can use the URL Fetch service to issue HTTP and HTTPS requests and receive responses. The URL Fetch service uses Google's network infrastructure for efficiency and scaling purposes.

Requests made using this service originate from a set pool of IP ranges. You can look up the full list of IP addresses if you need to whitelist or approve these requests.

See also

- `HTTPResponse`

Methods

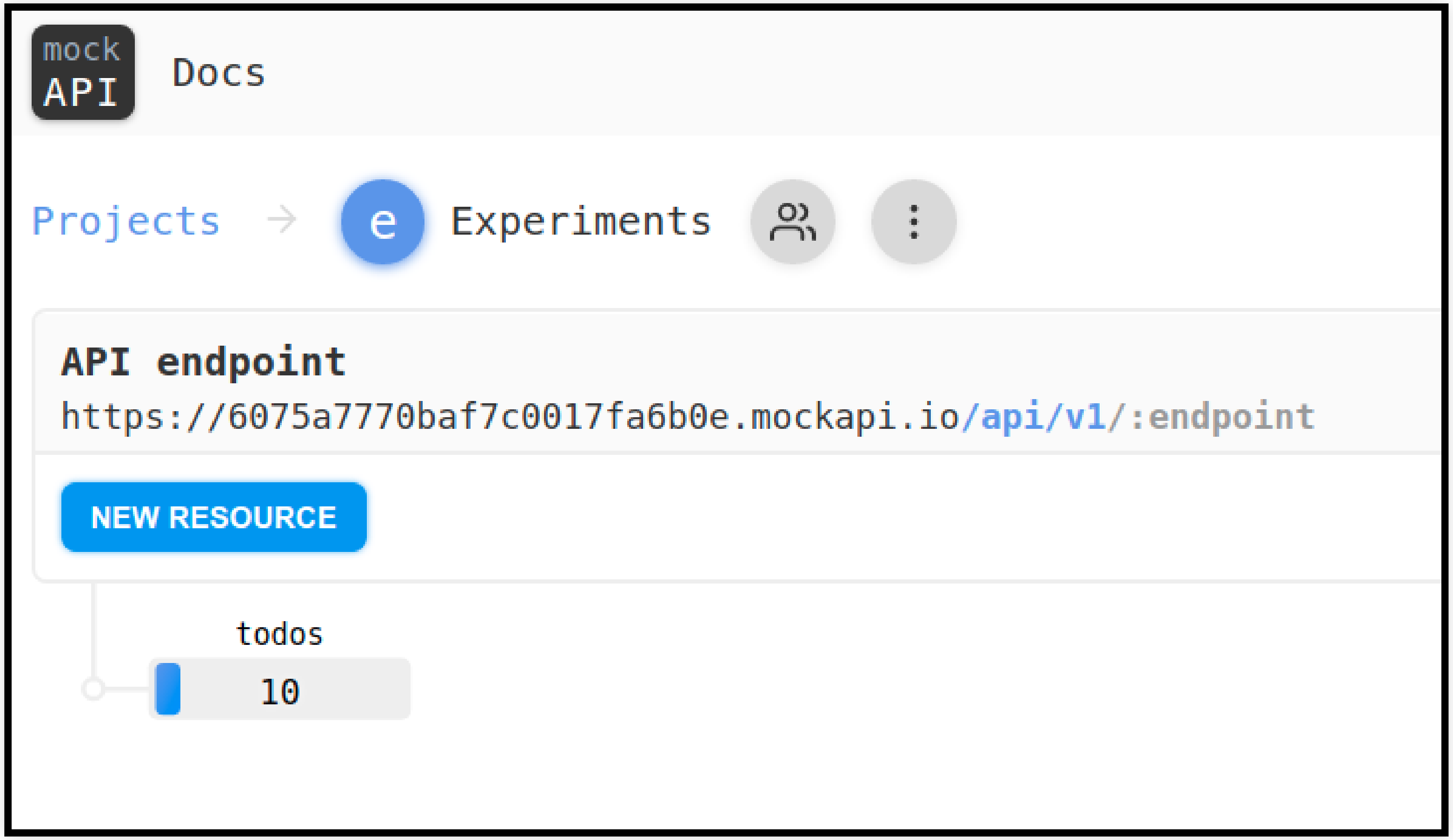
Method	Return type	Brief description
`fetch(url)`	`HTTPResponse`	Makes a request to fetch a URL.
`fetch(url, params)`	`HTTPResponse`	Makes a request to fetch a URL using optional advanced parameters.

Let’s get to the action!

(that concludes the interesting part)

How do we do that?

Let’s generate simplest possible API in MockAPI



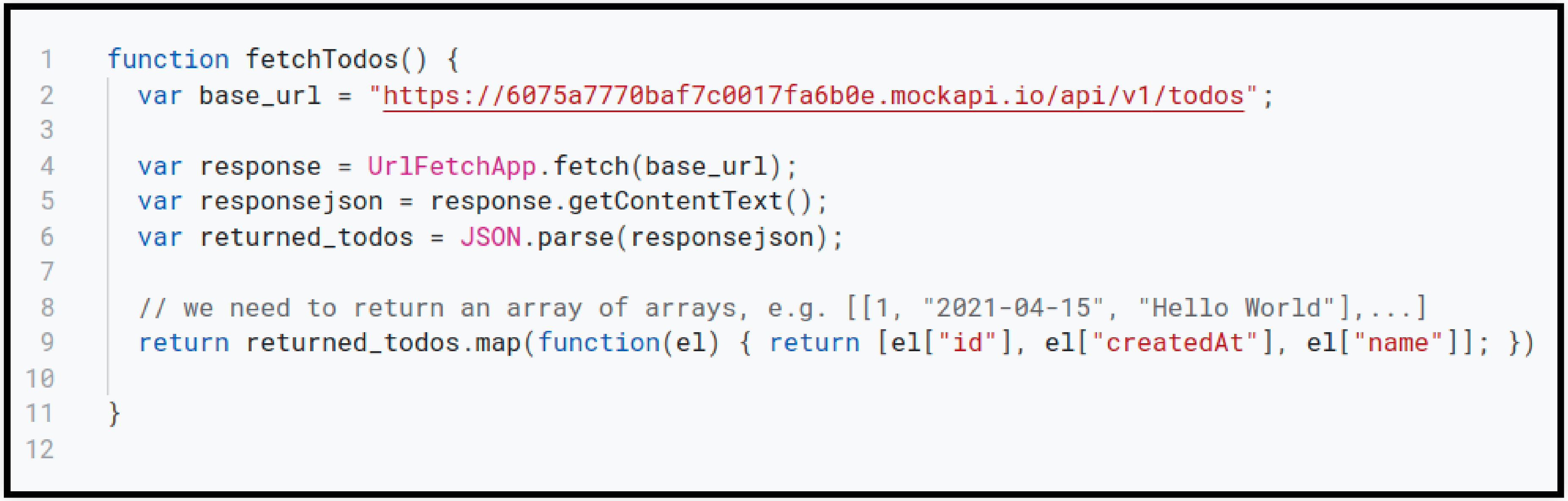
API endpoint

https://6075a7770baf7c0017fa6b0e.mockapi.io/api/v1/:endpoint

todos

10

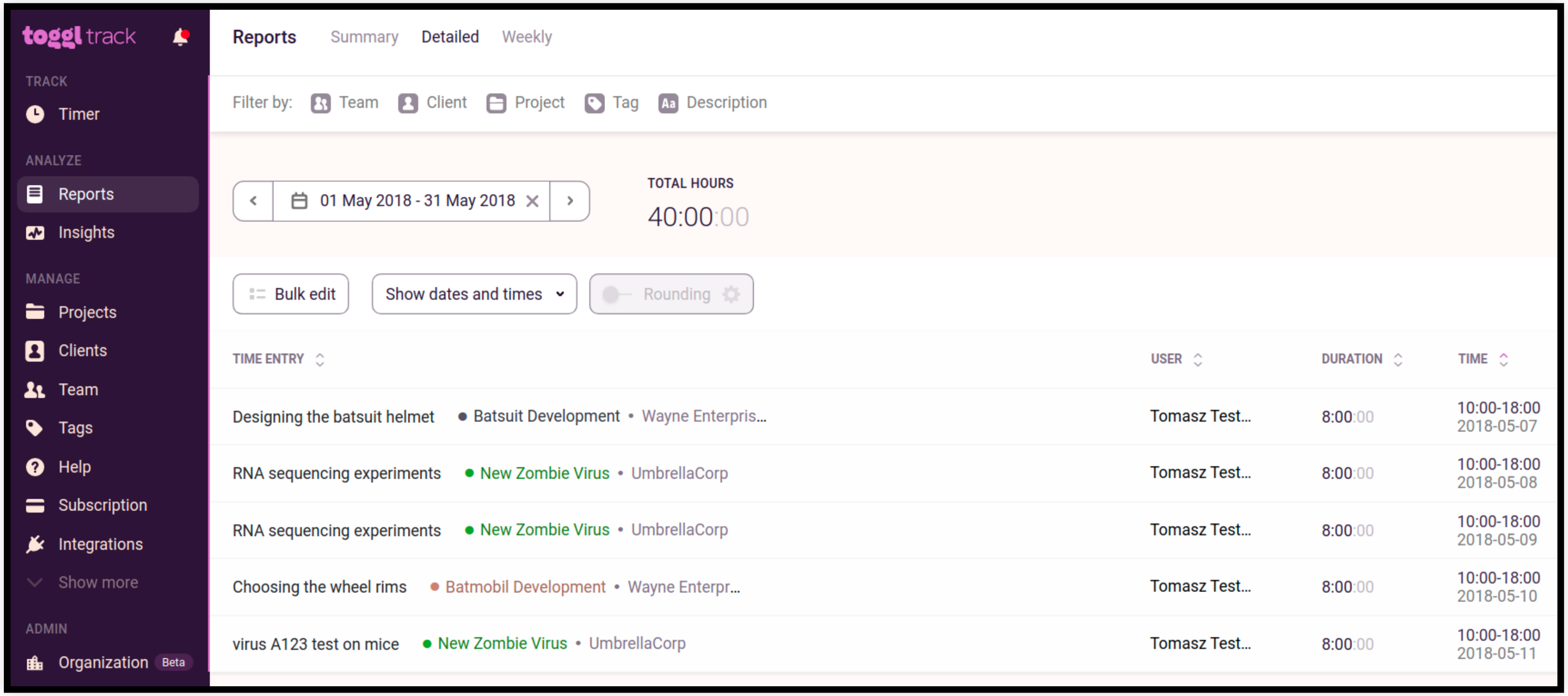
…write a little bit of code…



```
function fetchTodos() {
  var base_url = "https://6075a7770baf7c0017fa6b0e.mockapi.io/api/v1/todos";

  var response = UrlFetchApp.fetch(base_url);
  var responsejson = response.getContentText();
  var returned_todos = JSON.parse(responsejson);

  // we need to return an array of arrays, e.g. [[1, "2021-04-15", "Hello World"],...]
  return returned_todos.map(function(el) { return [el["id"], el["createdAt"], el["name"]]; })

}

```

...and fire it up

Real API is usually a bit more complex

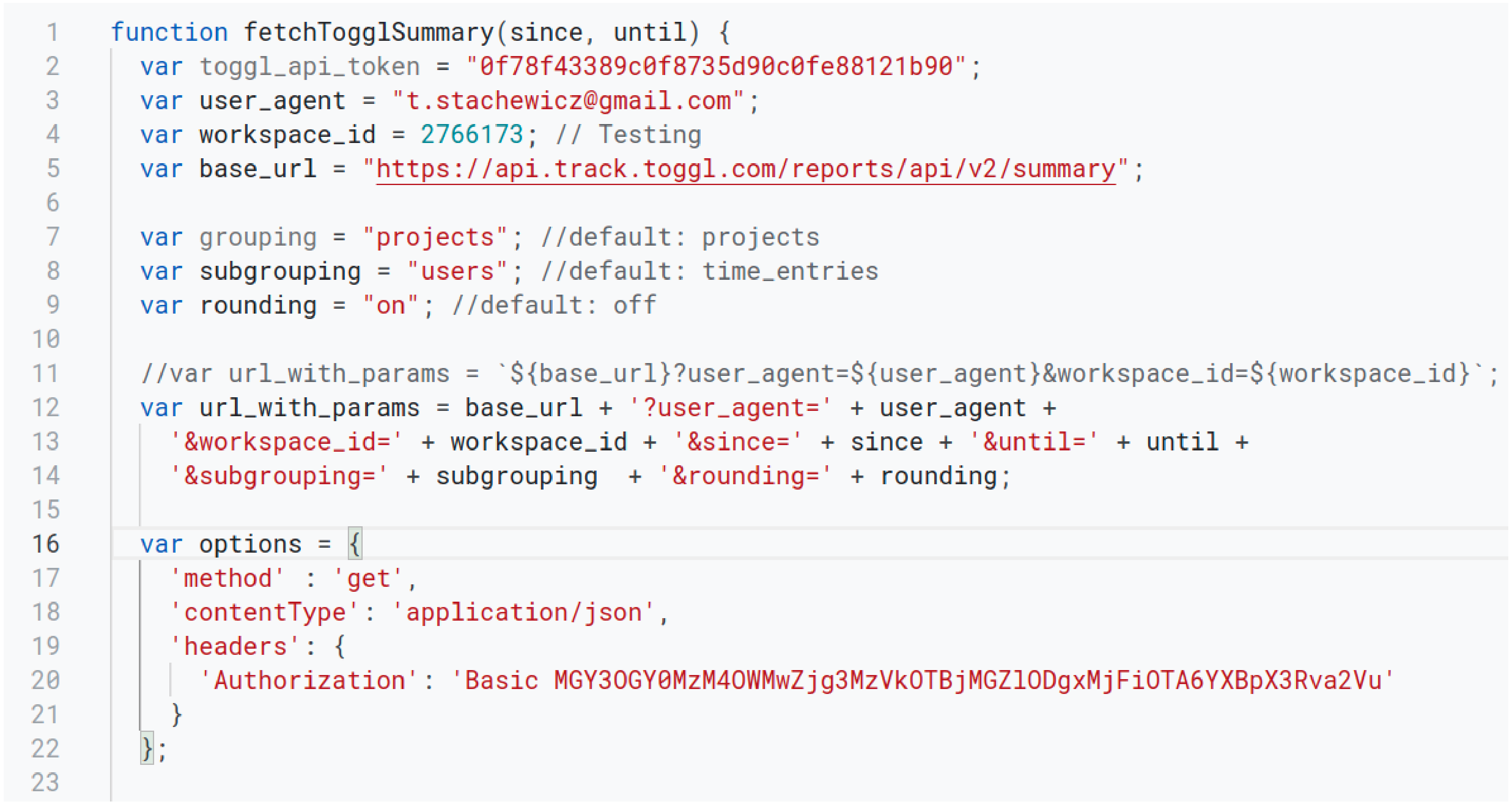


01 May 2018 - 31 May 2018

TOTAL HOURS
40:00:00

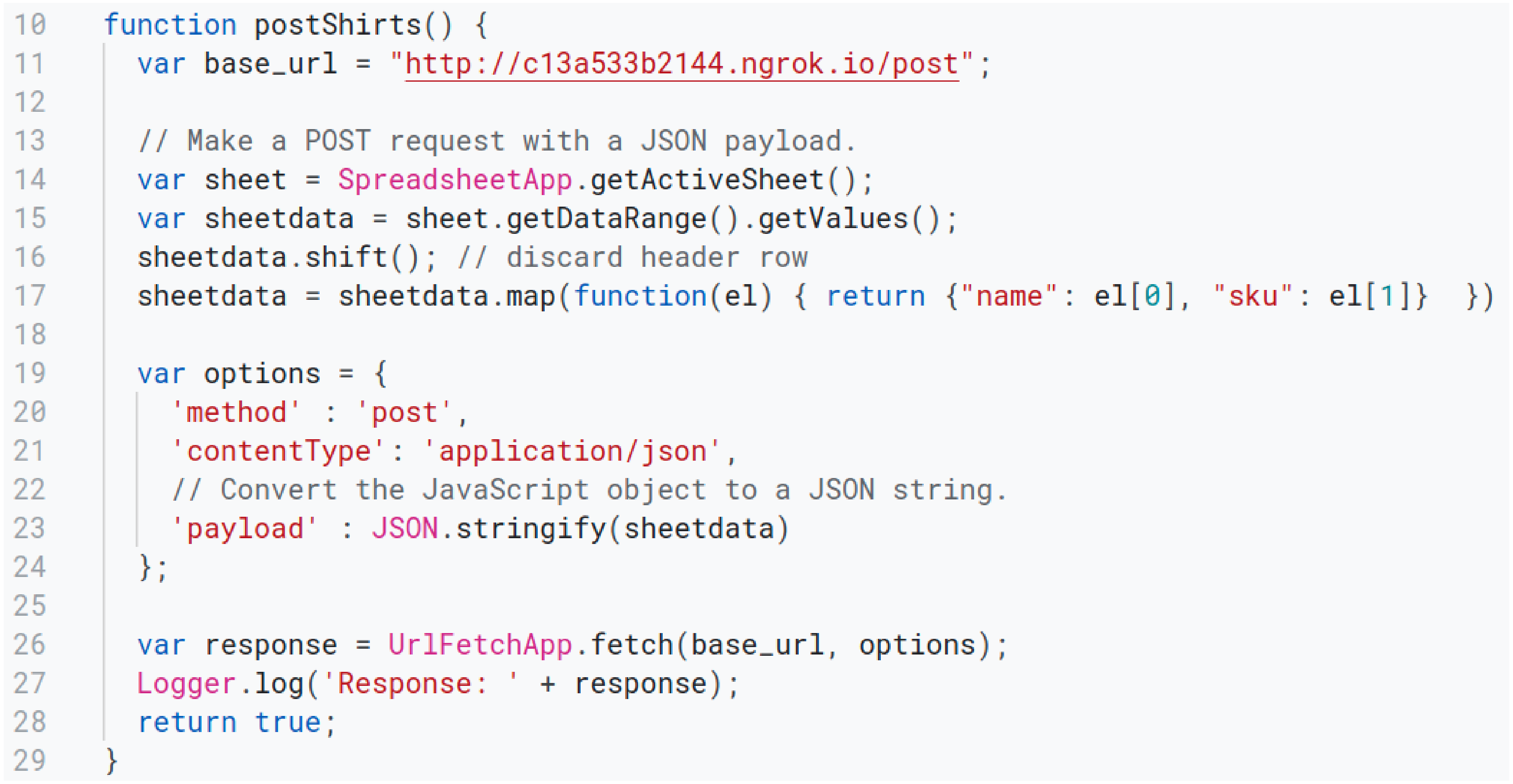
TIME ENTRY		USER	DURATION	TIME
Designing the batsuit helmet	● Batsuit Development • Wayne Enterpris...	Tomasz Test...	8:00:00	10:00-18:00 2018-05-07
RNA sequencing experiments	● New Zombie Virus • UmbrellaCorp	Tomasz Test...	8:00:00	10:00-18:00 2018-05-08
RNA sequencing experiments	● New Zombie Virus • UmbrellaCorp	Tomasz Test...	8:00:00	10:00-18:00 2018-05-09
Choosing the wheel rims	● Batmobil Development • Wayne Enterpr...	Tomasz Test...	8:00:00	10:00-18:00 2018-05-10
virus A123 test on mice	● New Zombie Virus • UmbrellaCorp	Tomasz Test...	8:00:00	10:00-18:00 2018-05-11

But not much more complex!



```
function fetchTogglSummary(since, until) {
  var toggl_api_token = "0f78f43389c0f8735d90c0fe88121b90";
  var user_agent = "t.stachewicz@gmail.com";
  var workspace_id = 2766173; // Testing
  var base_url = "https://api.track.toggl.com/reports/api/v2/summary";

  var grouping = "projects"; //default: projects
  var subgrouping = "users"; //default: time_entries
  var rounding = "on"; //default: off

  //var url_with_params = `${base_url}?user_agent=${user_agent}&workspace_id=${workspace_id}`;
  var url_with_params = base_url + '?user_agent=' + user_agent +
    '&workspace_id=' + workspace_id + '&since=' + since + '&until=' + until +
    '&subgrouping=' + subgrouping  + '&rounding=' + rounding;

  var options = {
    'method' : 'get',
    'contentType': 'application/json',
    'headers': {
      'Authorization': 'Basic MGY3OGY0MzM4OWMwZjg3MzVkOTBjMGZlODgxMjFiOTA6YXBpX3Rva2Vu'
    }
  };

  var response = UrlFetchApp.fetch(url_with_params, options);
  var responsejson = response.getContentText();
  var report_data = JSON.parse(responsejson);

  //we'll be returning a Nx3 array of threes [client-project, user, hours]
  var arr_of_arrs = [];

  report_data["data"].forEach(function(entry) {
    var label = entry["title"]["client"] + " - " + entry["title"]["project"];
    entry["items"].forEach(function(item) {
      var user = item["title"]["user"];
      //time is in miliseconds
      var hours = parseInt(item["time"]) / (1000 * 3600);
      arr_of_arrs.push([label, user, hours]);
    });
  });

  arr_of_arrs.sort(function(el) { el[0]; });

  return arr_of_arrs;
}
```

Can I POST data from spreadsheet?



```
function postShirts() {
  var base_url = "http://c13a533b2144.ngrok.io/post";

  // Make a POST request with a JSON payload.
  var sheet = SpreadsheetApp.getActiveSheet();
  var sheetdata = sheet.getDataRange().getValues();
  sheetdata.shift(); // discard header row
  sheetdata = sheetdata.map(function(el) { return {"name": el[0], "sku": el[1]}  })

  var options = {
    'method' : 'post',
    'contentType': 'application/json',
    // Convert the JavaScript object to a JSON string.
    'payload' : JSON.stringify(sheetdata)
  };

  var response = UrlFetchApp.fetch(base_url, options);
  Logger.log('Response: ' + response);
  return true;
}
```

Is GoogleScript *a* JavaScript?

Drawbacks and risks

- no automated tests
- vendor lock-in
- (but Office365)

There’s more!

- Google services (Gmail, Gmaps, Contacts...)
- JDBC (suggestion: don't)
- Spreadsheet charts 📊

That’s all, folks!