Lecture 2: Moving to the server side…

Olivier Liechti
TWEB

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Test alert

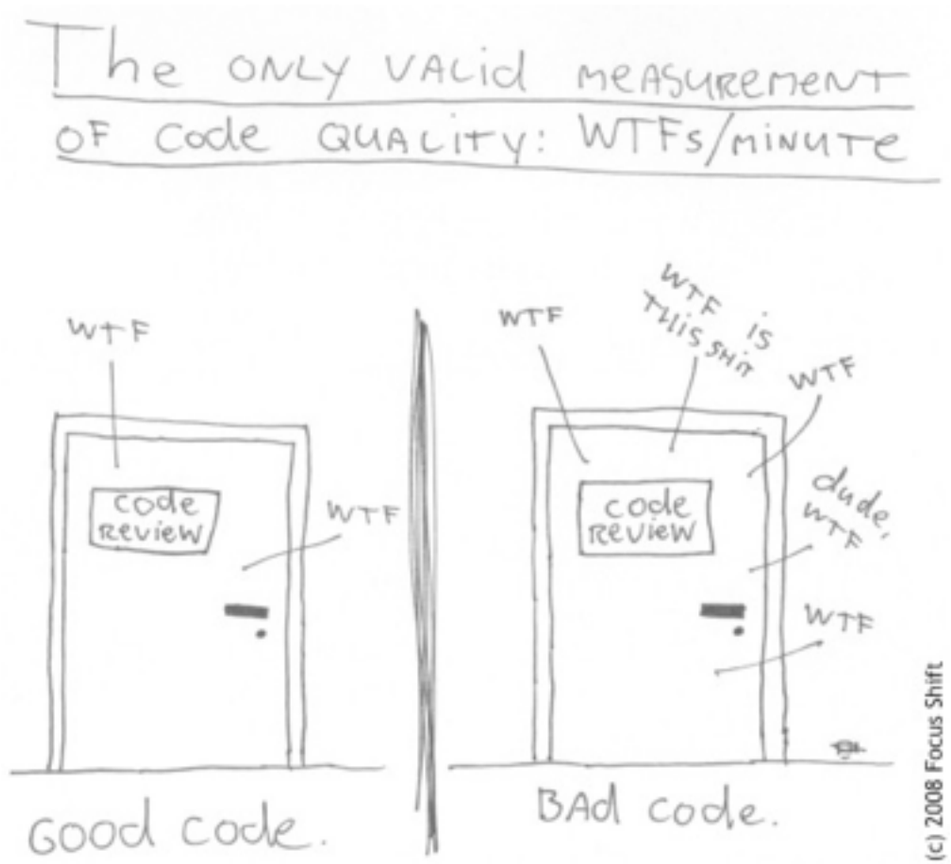
Manipulation

Warning

Agenda

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

13h00 - 13h15	15’	**Code review** Feedback on first labs
13h15 - 13h45	30’	**Lecture** Intro to JavaScript, part II
13h45 - 14h15	30’	**Lecture** Intro to Node.js
14h15 - 15h00	45’	**Environment setup** Install node, configure the debugger
15h00 - 15h30		Break
15h30 - 18h00	150’	**Lab** Implement and deploy a Github front-end on Heroku


The ONLY VALID MEASUREMENT OF CODE QUALITY: WTFs/MINUTE
WTF
code review
WTF
Good code.
WTF
WTF IS THIS SHIT
WTF
code review
dude, WTF
WTF
BAD code.
(c) 2008 Focus Shift

Code review

```
<script type="text/javascript">

    function Personne(n, p, s){
        this.nom = n;
        this.prenom = p,
        this.sexe = s;
    }

    var source   = $("#pers").html();
    var template = Handlebars.compile(source);
    var content = document.getElementById(“pers”);

    …

</script>
```

in index.htm

Script **does not work** in Chrome, because the document JQuery **ready handler** is not used.

Script should be enclosed in **$()** or **$(document).ready()**

Inline scripts are not recommended. Move your code to an external file.

To implement the "**mouse move with shift key pressed**", you don't need to handle key events. The state of the shift key is available in the mouse event.

```
var shiftIsDown = false;

$(document).on('keyup keydown', function(e){shiftIsDown = e.shiftKey} );
```

Making a synchronous ajax call is a **very bad practice**, because it will block the UI during the call!!

```
var rooms = jQuery.parseJSON(
    jQuery.ajax({
        url: "./rooms.json",
        async: false,
        dataType: 'json'
    }).responseText
).rooms;

$(rooms).each(function(index, value){
    $('#chambres').append("<p>Room: " + value.name + ", " + value.uselessNumber + "</p>");
});
```

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Separate data structure logic from presentation logic (you may want to render your object graph with another visualization library).

```
function Edge(from, to, type) {
    this.from = from;
    this.to = to;
    switch(type) {
        case "inherit":
            this.style = "arrow-center"
            this.arrowScaleFactor = 1;
            this.color = {"color":"red",
                          "highlight": "red"}
            break;
        case "property":
            this.color = {"color":"green",
                          "highlight": "green"}
            break;
    }
}
```

Be **consistent**: apply the same pattern in Edge and in Vertex!

```
function Vertex(id, label, type, source) {
    this.id = id;
    this.label = label;
    this.type = type;
    this.source = source;
    this.edges = [];
    this.addEdge = function(to, type) {
        this.edges[this.edges.length] = new
Edge(this.id, to, type);
    }
}
```

```
sVer.color = {};
sVer.color.border = "green";

if(current.hasOwnProperty(key))
sVer.color.background = "yellow";
else sVer.color.background = "gray";

ver.addEdge(sVer.id, "property");
this.addVertex(sVer);
```

Adding properties dynamically is possible. Defining them in the constructor makes your code **easier to use** (think contract).


```
function Vertex(id, label, type, source) {
    this.id = id;
    this.label = label;
    this.type = type;
    this.source = source;
    this.edges = [];
    this.addEdge = function(to, type) {
        this.edges[this.edges.length] = new Edge(this.id, to, type);
    }
}
```

```
function Vertex(id, label, type, source) {
    this.id = id;
    this.label = label;
    this.type = type;
    this.source = source;
    this.edges = [];
}

Vertex.prototype.addEdge = function(to, type) {
    this.edges[this.edges.length] = new Edge(this.id, to, type);
}
```

Do you **understand the implications** of declaring the function here (memory, performance)?

Do you **understand the difference** with the other code, which attaches the function to the prototype?

When I read this line (which is way below the constructor, I am lead to believe that a Person takes a zip code… **BAD**.

```
function Person(n, g, r) {
        this.name   =    n;
        this.gender = g;
        this.room = r;
};

...

//génération d'une personne
var gen = chance.gender();
var personne = new  Person(chance.name({ gender: gen }),gen,chance.zip());
```

Be consistent in your code. Why not write edges: jsGraph.edgesList() ?

```
var eList = jsGraph.edgesList();
var physMode = (confPhysMode)?true:false;

var data = {
  nodes: jsGraph.vertexes,
  edges: eList
};
```

```
function Personne(n, p, s){
    this.nom = n;
    this.prenom = p,
    this.sexe = s;
}
```

```
        // NE FONCTIONNE PAS
if (isKeyPressed(event).shiftKey == 16){
    $("#student").append(p2.greet()+"<br/>");
} else {
   $( "#student" ).html(p2.greet());
}
```

CODE IN ENGLISH.
COMMENT IN ENGLISH.

```
//affichage de l'heure
setInterval(function(){$("#clock").text(new Date())},1000);
```

```
if(personne.sexe == "Female") content.className = "bg-danger";
```

```
//Génère un nouveau graph de type network
    var construct = function () {
```

Clean up legacy code!

heig-vd
Haute Ecole d'Ingénierie et de Gestion

```
// load json
//getRooms = function() {
//    $.ajax({
//        url:"rooms.json",
//        type:"GET",
//        dataType:"json",
//        success: function(json) {
//            $.each(json, function(item, value) {
//                if (item == "rooms") {
//                    $.each(value, function(i, v) {
//                        $("#rooms").append(v.name + ' ' + v.space + '</br>');
//                    })
//                }
//            })
//        }
//    });
//};
getRooms = functi
    $.getJSON("ro
        $.each(js
            if (i
                $
                }
            }
        });
    });
};
```

```
/*
var nodes = [
        {id: 1, label: 'black node', group: 'objectNodes'},
    {id: 2, label: 'black node', group: 'ownPropertyNodes'},
    {id: 3, label: 'black node', group: 'inheritedPropertyNodes'},
      ];

          edges = [
  //prototypeEdges
    {from: 1, to: 2, width: 3, color: 'red', style: 'arrow-center', width:2},
  //propertyEdges
    {from: 1, to: 3, width: 1, color: 'green', style: 'line'},
  ];
*/
```

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

JavaScript 101 (Part 2)

JavaScript 101, Part 2

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Arrays are objects
- Functions are objects
- `Function.prototype` vs `myFunction.prototype`
- Closures
- Module patterns
- `this`

Rule #7
Arrays are objects

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

```
var fruits = ["apple", "pear"];

fruits.push("banana");
console.log(Object.getPrototypeOf(fruits));

for (var i=0; i<fruits.length; i++) {
  console.log("fruits[" + i + "] = " + fruits[i]);
}
```

Make sure that you understand how **map** and **reduce** work

```
var transformedFruits = fruits.map( function(fruit) {
  return fruit.toUpperCase();
});
transformedFruits.forEach( function(fruit) {
  console.log(fruit);
});

var count = fruits.reduce( function(val, fruit) {
  console.log("reducer invoked with " + val);
  return val+1;
}, 0);
console.log("There are " + count + " fruits in the array");
```

http://codepen.io/wasadigi/full/LtnAp/

Rule #8
Functions are objects

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

```
function aFunction() {
}

var f = function() {
}

var g = function g() {
  g(); // recursive call
}

var h = function(functionParam) {
  functionParam();
}

h(f);
h(g);
```

Rule #9

Whenever a function is defined, an object is created.

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Its prototype is Function.prototype.
Its prototype property is the object that will be the prototype of instances created with the function used as a constructor with the **new** keyword.

```
function Robot() {
}
var r2d2 = new Robot();


var t1 = Object.getPrototypeOf(Robot) === Function.prototype;
var t2 =  Robot.__proto__ === Function.prototype; //
deprecated

var t3 =  Object.getPrototypeOf(r2d2) === Robot.prototype;
var t4 =  Object.getPrototypeOf(Robot.prototype) ===
Object.prototype;
```

Rule #10
"new" makes constructors behave in a special way, but constructors are normal functions

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

```
function Animal(name) {
  this.name = name;
}
var cat = new Animal("Félix");
```

1. A **new object** is created.
2. Its **prototype** is set to `Animal.prototype`
3. The constructor function is called; **`this`** is bound to the newly created object.
4. In general, the constructor does not return any object. In this case, the result of the new expression is the newly created object.

Rule #11
Functions can be nested

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

```
function f1(p1) {
  console.log("f1 can see " + p1);
  function f2(p2) {
    console.log("f2 can see " + p2 + " " + p1);
    function f3(p3) {
      console.log("f3 can see " + p3 + " " + p2 + " " + p1);
    }
    f3(3);
  }
  f2(2);
}
f1(1);
```

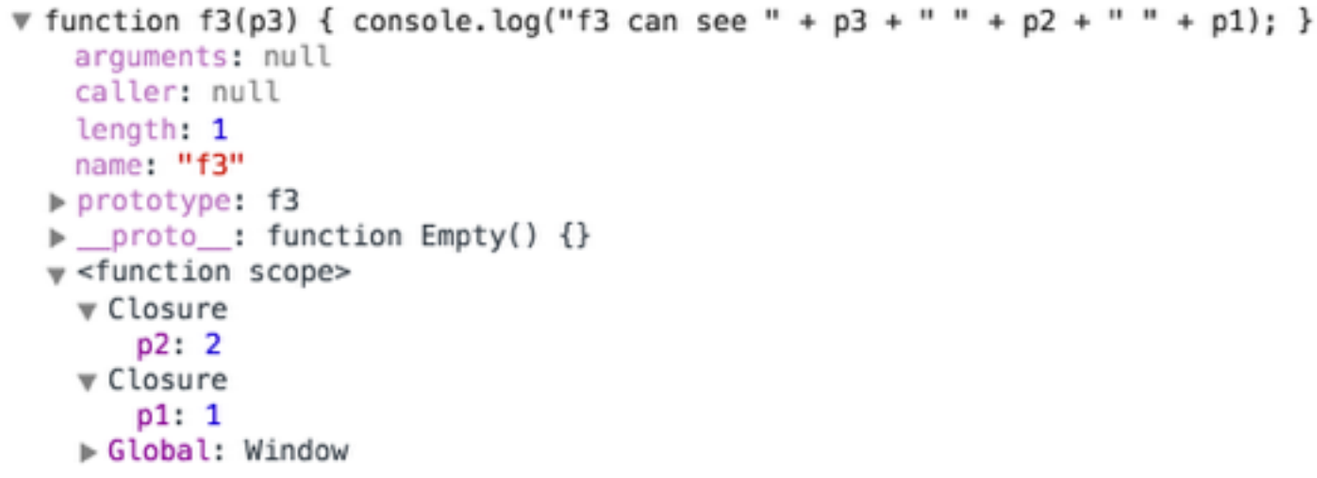
- An **object** is created for every function.
- Each function has access to variables defined in the **parent** functions (an in the **global scope**).

Rule #12
A closure is formed when a nested function accesses a *free* variable

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

```
function f1(p1) {
  console.log("f1 can see " + p1);
  function f2(p2) {
    console.log("f2 can see " + p2 + " " + p1);
    function f3(p3) {
      console.log("f3 can see " + p3 + " " + p2 + " " + p1);
    }
    f3(3);
  }
  f2(2);
}
f1(1);
```

- In a function, a **free variable** is a variable that is neither a local variable, nor a parameter of the function.
- A **closure** is the combination of a code block (the function code) and saved parent environments.



Rule #13
When 2 functions are created in the same scope, closures are formed on the same environment.

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

```
function parent() {

  var sharedVariable;

  function child1() {
    sharedVariable = 1;
  }

  function child2() {
    sharedVariable = 2;
  }

  function getValue() {
    return sharedVariable;
  }

  return {
    toggleToOne: child1,
    toggleToTwo: child2,
    getValue: getValue
  }
}

var myToggle1 = parent();
var myToggle2 = parent();
myToggle1.toggleToTwo();
myToggle2.toggleToOne();
console.log(myToggle1.getValue());
console.log(myToggle2.getValue());
```

- This pattern can be used to implement **private variables** in Javascript.
- But remember that in this case, the code of the functions is **cloned** in all instances (unlike when you use `Class.prototype.method`).
- Be aware of a **well known issue** when closures are formed within a loop.

```
function test() {
  var functions = [];

  for (var i=0; i<10; i++) {
    functions.push(function() {
      console.log(i);
    });
  };

  return functions;
}

var fs = test();
fs.forEach( function(f) { f(); } );
```

- **What is happening?**
- **Why?**
- **How do you fix this?**

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Closures

Rule #14
Patterns are applied to create modules

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

```
var myModule = (function() {

  var aPrivateVar;
  var privateFunction_1 = function() {}
  var privateFunction_2 = function() {}

  return {
    publicFunction: privateFunction_1
  }

})();

myModule.publicFunction();
```

http://codepen.io/wasadigi/full/GxsfC/

- The function is **immediately invoked**.

- When `privateFunction_1` accesses `aPrivateVar`, a **closure** is formed.
- is **available even after** the immediately invoked function has returned.
- `privateFunction_1` and `privateFunction_2` share the same parent scope.

```
Hello world
> dir(myModule)
▼ Object
  ▼ publicFunction: function () {
      arguments: null
      caller: null
      length: 0
      name: ""
    ▶ prototype: Object
    ▶ __proto__: function Empty() {}
    ▼ <function scope>
      ▼ Closure
          aPrivateVar: "world"
      ▶ Global: Window
  ▶ __proto__: Object
< undefined
>
```

Rule #15
The value of `this` depends on the way a function has been called

heig-vd
Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

```
function Robot(name) {
  this.name = name;
}

Robot.prototype.greet = function greet() {
  debugger;
  console.log("hello " + this.name);
  if (this.name === undefined) {
    return; // oops...
  }
  greet(); // no dot notation!
}

var r2d2 = new Robot("r2d2");
r2d2.greet();
```

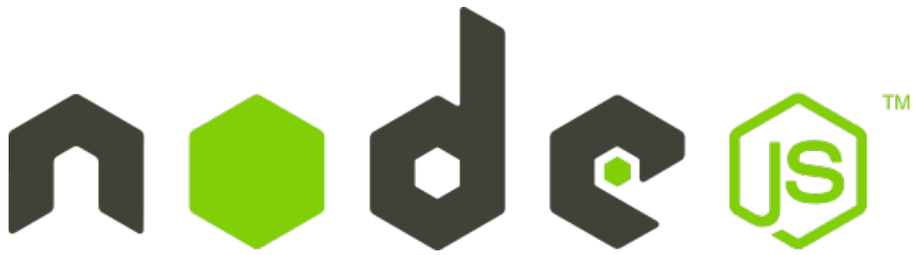
- When a function is called on an object (i.e. **as a method**), then this refers to the object.
- When a function is called **as a function** (no dot notation), then this refers to the **global object**).
- When a function is is called as a constructor (with new), then this **refers** to the **newly created instance**.
- There are methods defined on `Function.prototype` to control the value of this: `apply` and `call`.

http://codepen.io/wasadigi/full/stkum/

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Node.js 101

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



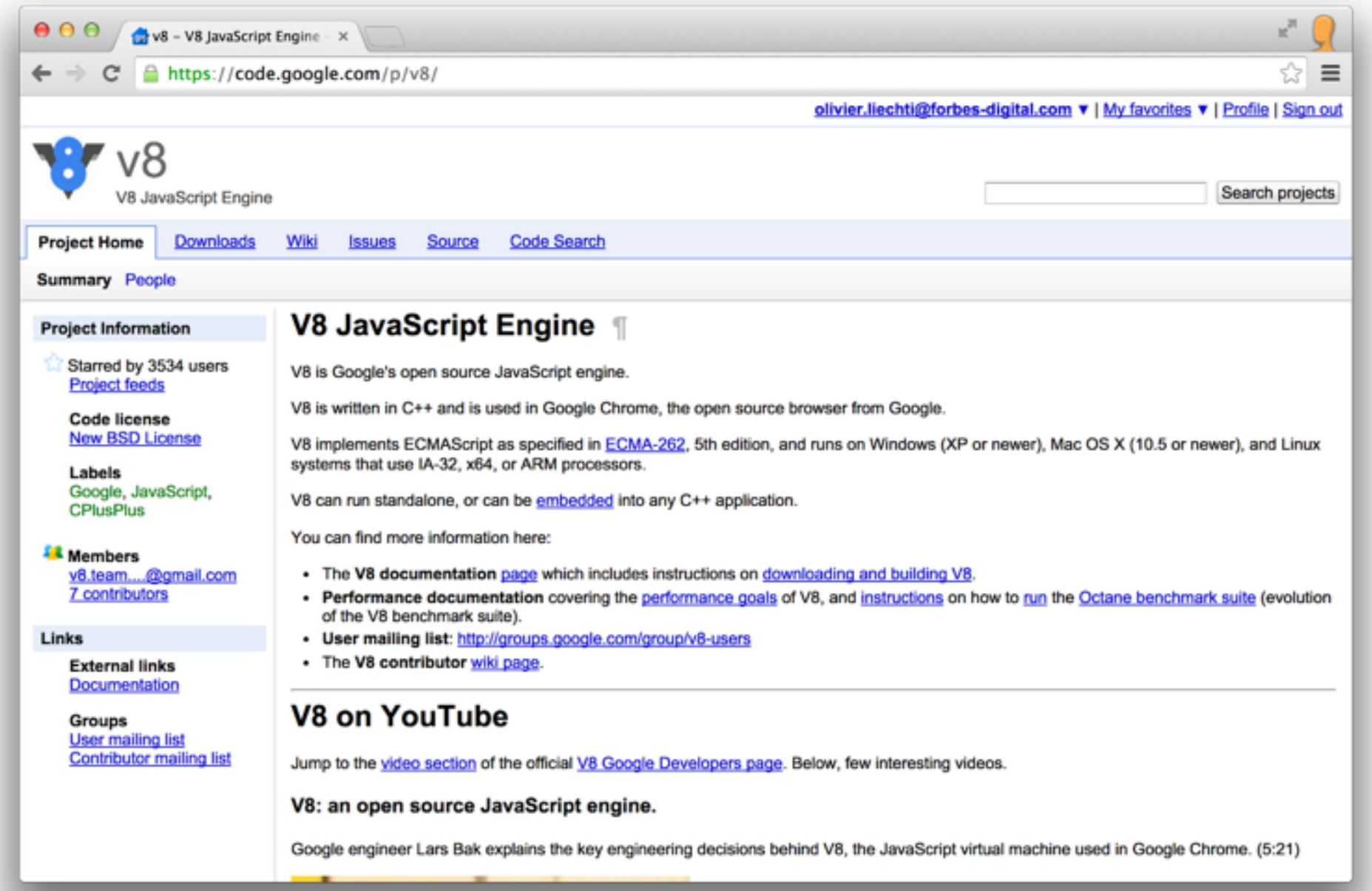
“Node.js is a **platform** built on Chrome's JavaScript runtime for easily building **fast**, **scalable network applications**.

not only!

Node.js uses an **event-driven**, **non-blocking I/O model** that makes it lightweight and efficient, perfect for **data-intensive real-time applications that run across distributed devices**.”

- About these Docs
- Synopsis
- Assertion Testing
- Buffer
- C/C++ Addons
- Child Processes
- Cluster
- Console
- Crypto
- Debugger
- DNS
- Domain
- Events
- File System
- Globals
- HTTP
- HTTPS
- Modules
- Net
- OS
- Path
- Process
- Punycode
- Query Strings
- Readline
- REPL
- Stream
- String Decoder
- Timers
- TLS/SSL
- TTY
- UDP/Datagram
- URL
- Utilities
- VM
- ZLIB

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



Google

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Trends

Topics

Subscribe

Node.js
Software

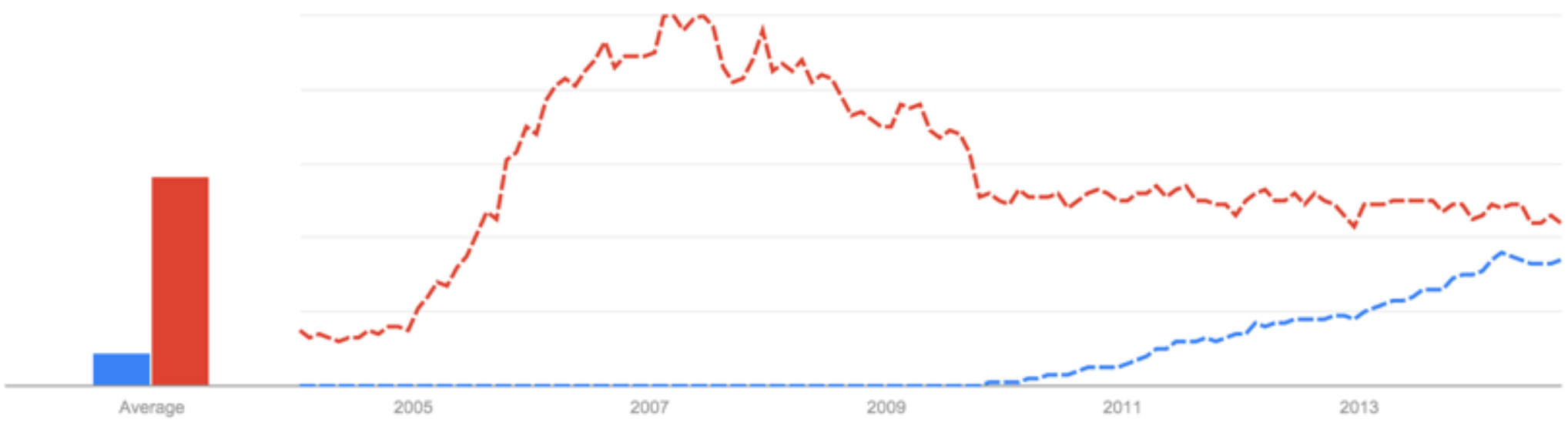
Ruby on Rails
Web Framework

+ Add term

Beta: Measuring search interest in *topics* is a beta feature which quickly provides accurate measurements of overall search interest. To measure search interest for a specific *query*, select the "search term" option.

Interest over time

News headlines Forecast



heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Let's look at a **first example**

```
/*global require */

var fs = require("fs");

/**
 * Simple function to test the synchronous readFileSync function provided by
 * Node.js
 * @param {string} filename - the name of the file we want to read in the test
 */
function testSyncRead(filename) {
  console.log("I am about to read a file: " + filename);
  var data = fs.readFileSync(filename);
  console.log("I have read " + data.length + " bytes (synchronously).");
  console.log("I am done.");
}

// We get the file name from the argument passed on the command line
var filename = process.argv[2];

console.log("\nTesting the synchronous call");
testSyncRead(filename);
```

This is for the Brackets editor

We use a standard **Node module** for accessing the file system

fs.readFileSync is synchronous: it blocks the main thread until the data is available.

process is a global object provided by Node.js

```
$ node sample2.js medium.txt

Testing the synchronous call
I am about to read a file: medium.txt
I have read 1024 bytes (synchronously).
I am done.
```

*Synchronous functions are easier to use, but they have **severe** performance implications!!*

Let's look at a **second example**

```
/*global require */

var fs = require("fs");

/**
 * Simple function to test the asynchronous readFile function provided by Node.js
 * @param {string} filename - the name of the file we want to read in the test
 */
function testAsyncRead(filename) {
  console.log("I am about to read a file: " + filename);

  fs.readFile(filename, function (err, data) {
    console.log("Nodes just told me that I have read the file.");
  });

  console.log("I am done. Am I?");
}
// We get the file name from the argument passed on the command line
var filename = process.argv[2];

console.log("\nTesting the asynchronous call");
testAsyncRead(filename);
```

```
$ node sample2.js medium.txt

Testing the asynchronous call
I am about to read a file: medium.txt
I am done. Am I?
Nodes just told me that I have read the file.
```

fs.readFile is asynchronous: it does not block the main thread until the data is available.

We must provide a **callback function**, which Node.js will invoke when the data is available.

Problems can happen when an (asynchronous) function is called.

*Node.js developers **have to** learn the asynchronous programming style.*

Node.js v0.10.32 Manual & Documentation

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Assertion Testing
- Buffer
- C/C++ Addons
- Child Processes
- Cluster
- Console
- Crypto
- Debugger
- DNS
- Domain
- Events
- File System
- Globals
- HTTP
- HTTPS
- Modules
- Net
- OS
- Path
- Process
- Punycode
- Query Strings
- Readline
- REPL
- Stream
- String Decoder
- Timers
- TLS/SSL
- TTY
- UDP/Datagram
- URL
- Utilities
- VM
- ZLIB

Let's look at a **third example**

```
/*global require */

var http = require("http");

/**
 * This function starts a http daemon on port 9000. It also
 * registers a callback handler, to handle incoming HTTP
 * requests (a simple message is sent back to clients).
 */
function runHttpServer() {
  var daemon = http.createServer();

  daemon.on("request", function (req, res) {
    console.log("A request has arrived: URL=" + req.url);
    res.writeHead(200, {
      'Content-Type': 'text/plain'
    });
    res.end('Hello World\n');
  });

  console.log("Starting http daemon...");
  daemon.listen(9000);
}

runHttpServer();
```

We use a standard **Node module** that takes care of the HTTP protocol.

Node can provide us with a **ready-to-use** server.

We can attach **event handlers** to the server. Node will notify us asynchronously, and give us access to the request and response.

We can **send back** data to the client.

We have wired everything, let's **welcome** clients!

HTTP Node.js v0.10.32 Ma ×

nodejs.org/api/http.html#http_event_request

NPM REGISTRY
DOCS
BLOG
COMMUNITY
LOGOS
JOBS

@nodejs

Node.js v0.10.32 Manual & Documentation

Index | View on single page | View as JSON

Table of Contents

- HTTP
  - http.STATUS_CODES
  - http.createServer([requestListener])
  - http.createClient([port], [host])
  - Class: http.Server
    - Event: 'request'
    - Event: 'connection'
    - Event: 'close'
    - Event: 'checkContinue'
    - Event: 'connect'
    - Event: 'upgrade'
    - Event: 'clientError'
    - server.listen(port, [hostname], [backlog], [callback])
    - server.listen(path, [callback])
    - server.listen(handle, [callback])
    - server.close([callback])
    - server.maxHeadersCount
    - server.setTimeout(msecs, callback)
    - server.timeout
  - Class: http.ServerResponse
    - Event: 'close'
    - Event: 'finish'
    - response.writeContinue()

These are the events that are **emitted** by the class. You can write callbacks and **react** to these events.

How does Node.js use an **event loop** to offer an asynchronous programming model?

```
on('request', function(req, res) { // my code});
```

```
on('data', function(data) { // my code});
```

Callback functions that **you** have written and registered

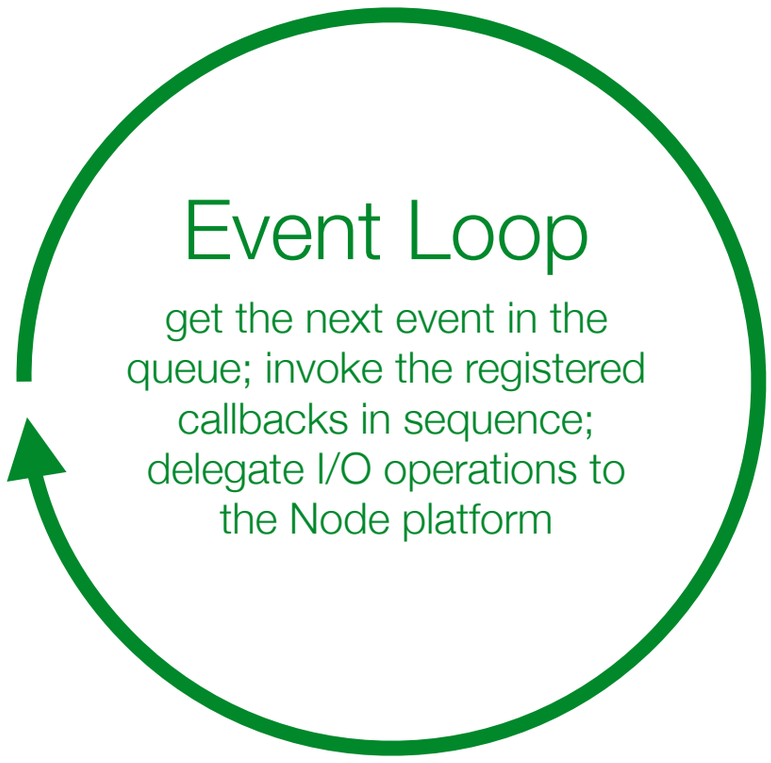
'request' **event**

'request' **event**

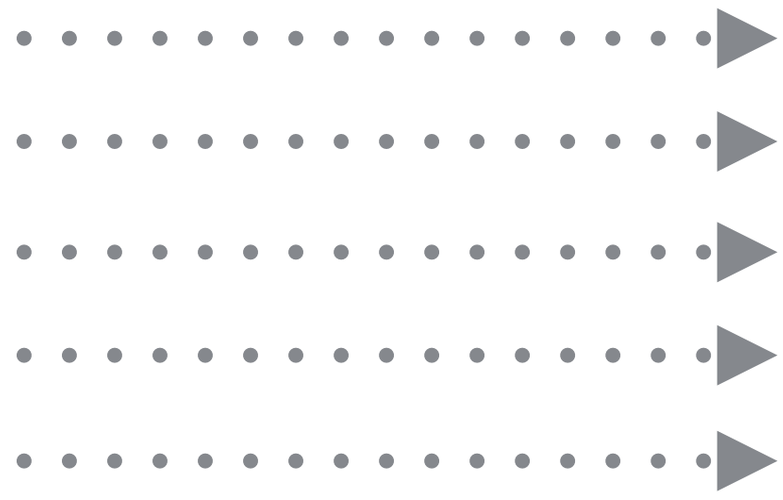
'data' **event**

'request' **event**

Queue of events that have been **emitted**

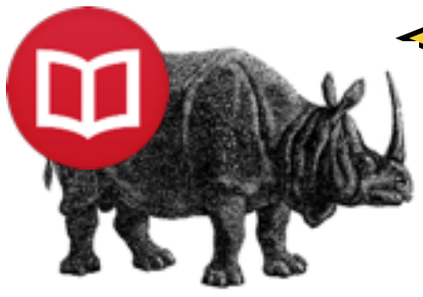


All the code that **you** write runs on a **single thread**



The **long-running tasks** (I/Os) are executed by Node in parallel; Node emits events to report progress (which triggers your callbacks).

Another pattern is to provide a callback to node when invoking an asynchronous function.



What is **npm**?

---

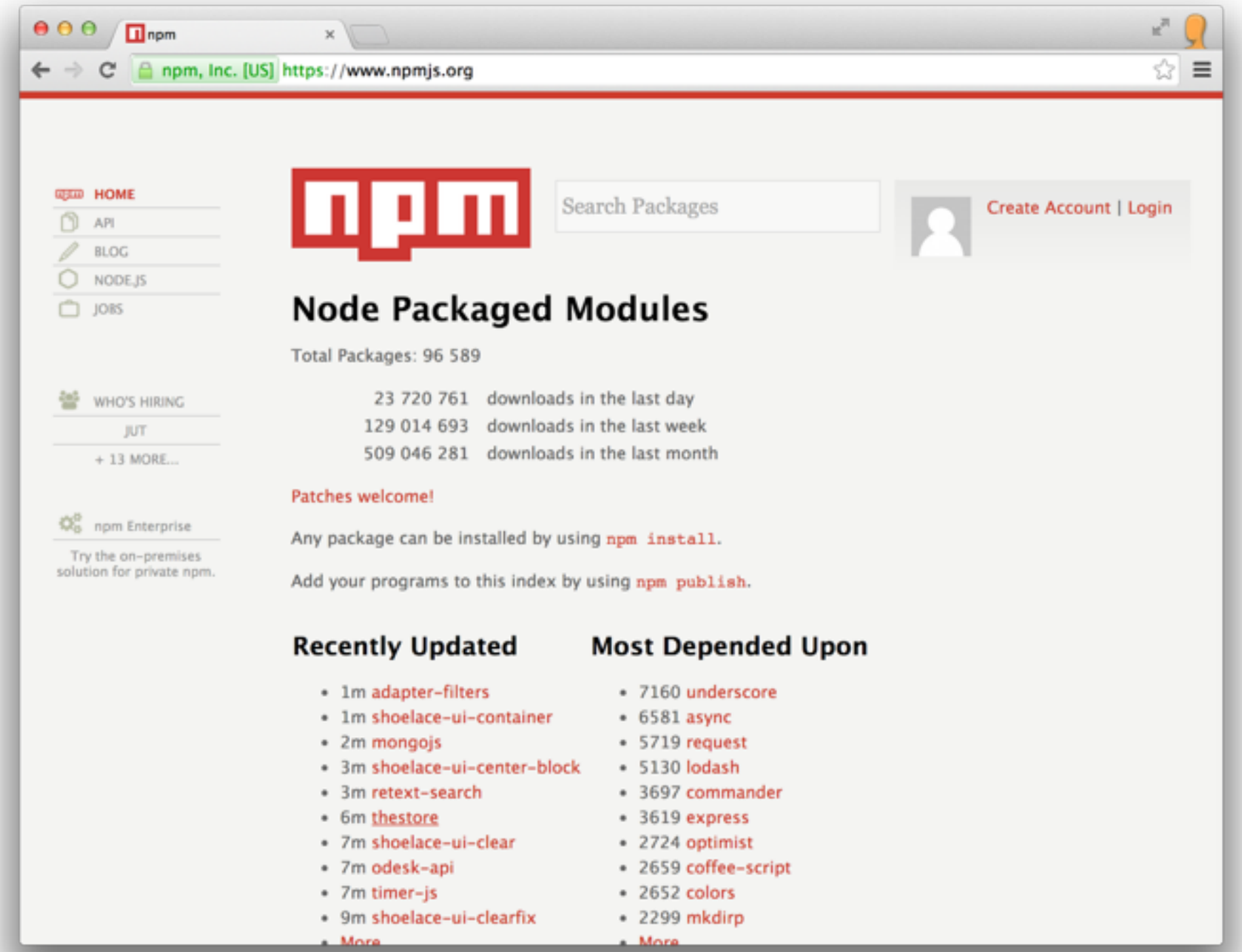
*”**npm** is the **package manager** for the Node JavaScript platform. It puts **modules** in place so that node can find them, and manages **dependency** conflicts intelligently.*

*It is extremely configurable to support a wide variety of use cases. Most commonly, it is used to **publish**, **discover**, **install**, and **develop** node programs.”*

https://www.npmjs.org/doc/cli/npm.html

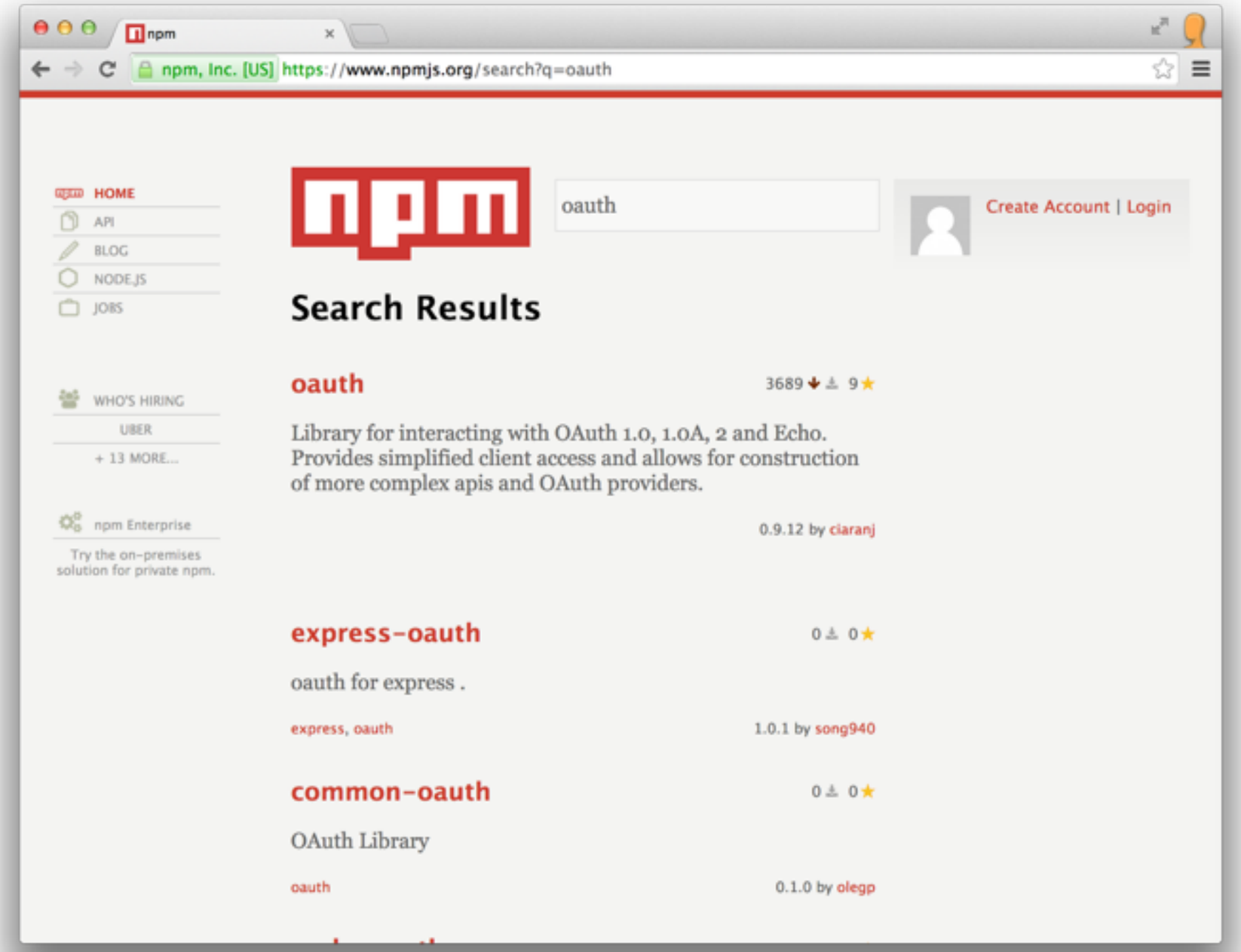
You **have to** read this:
https://www.npmjs.org/doc/misc/npm-faq.html

npm is a set of command line tools that work together with the **node registry**



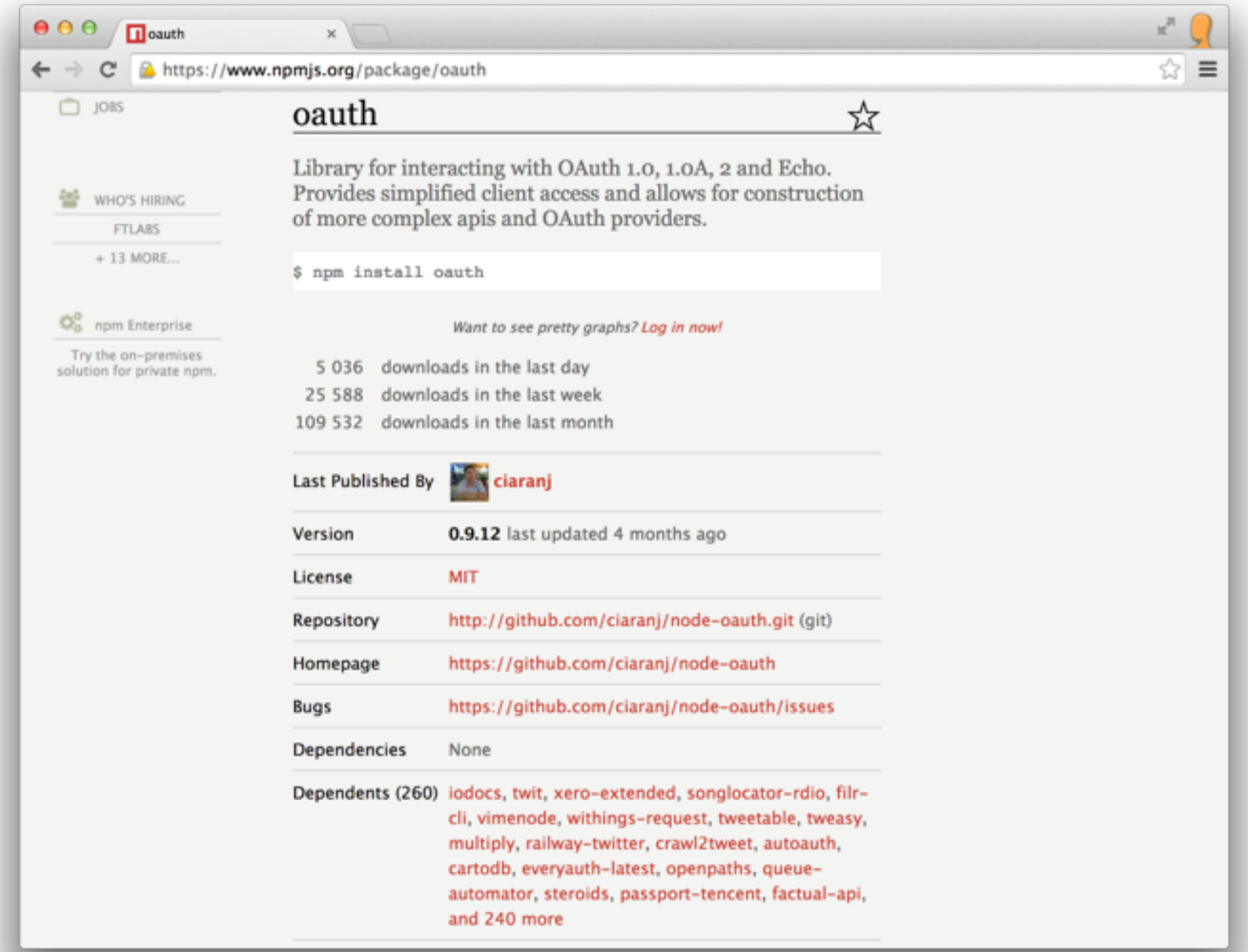
https://www.npmjs.org

npm is a set of command line tools that work together with the **node registry**



https://www.npmjs.org

npm is a set of command line tools that work together with the **node registry**



https://www.npmjs.org

express is one of the most popular Node modules. It is a web framework.

express 55273 ⭳ 531 ★

Fast, unopinionated, minimalist web framework

express, framework, sinatra, web, rest, restful, router, app, api

4.9.5 by dougwilson

express

web application framework for node

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

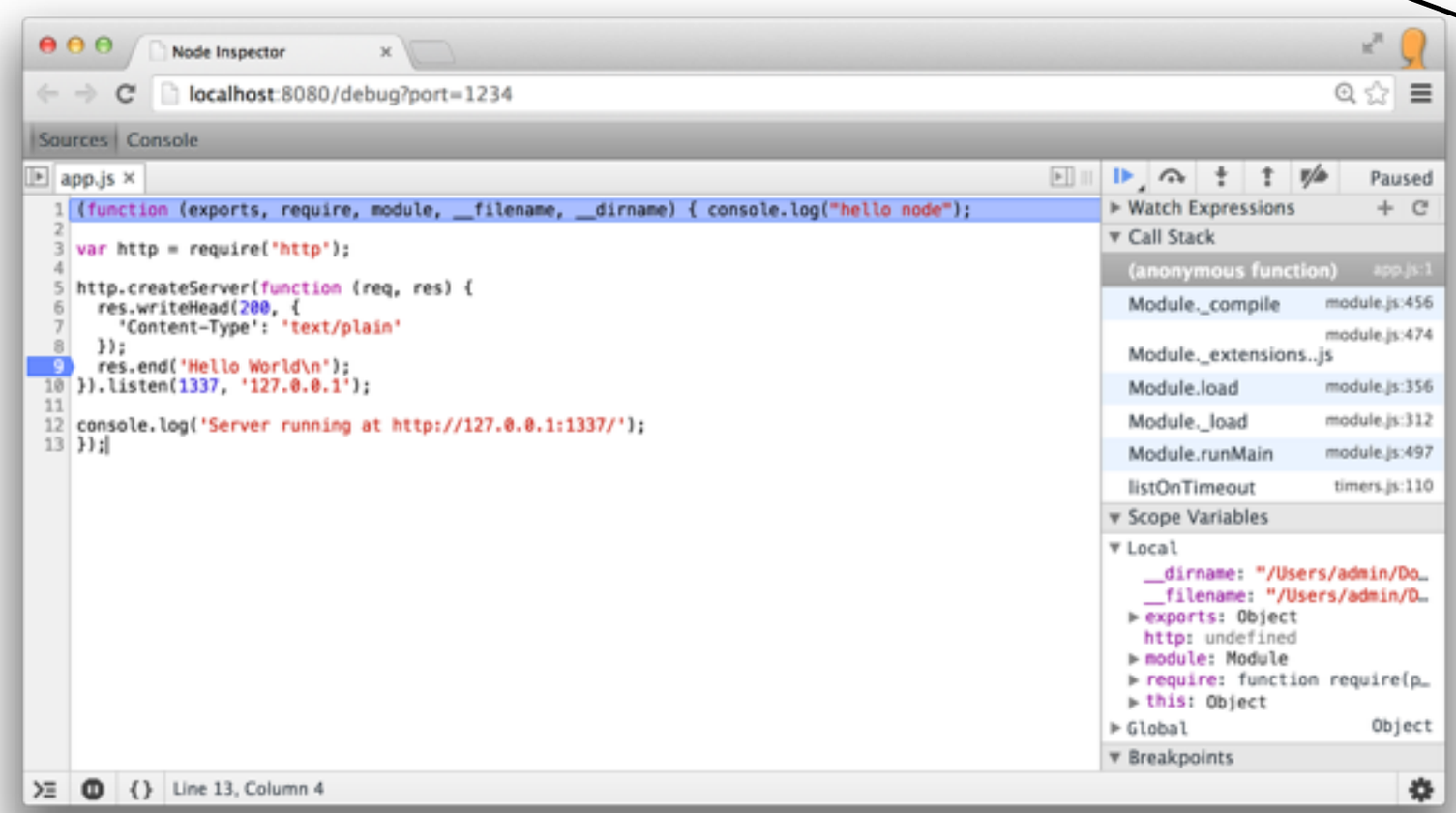
Environment Setup

How do I setup my **development environment** for Node.js applications?

- **You should use the same IDE / editor as the one you use for your client-side code**
  - Brackets works well for server-side applications
  - It's up to you to select your favorite tool, but take the time to learn how to use it efficiently (and to customize it)
- You **need** a debugger
  - Node ships with a **debugger** (http://nodejs.org/api/debugger.html). Use it with "node debug server.js"
  - A third-party module, **node-inspector**, is available and provides an integration with chrome dev tools (https://github.com/node-inspector/node-inspector).

How do I setup my **development environment** for Node.js applications?

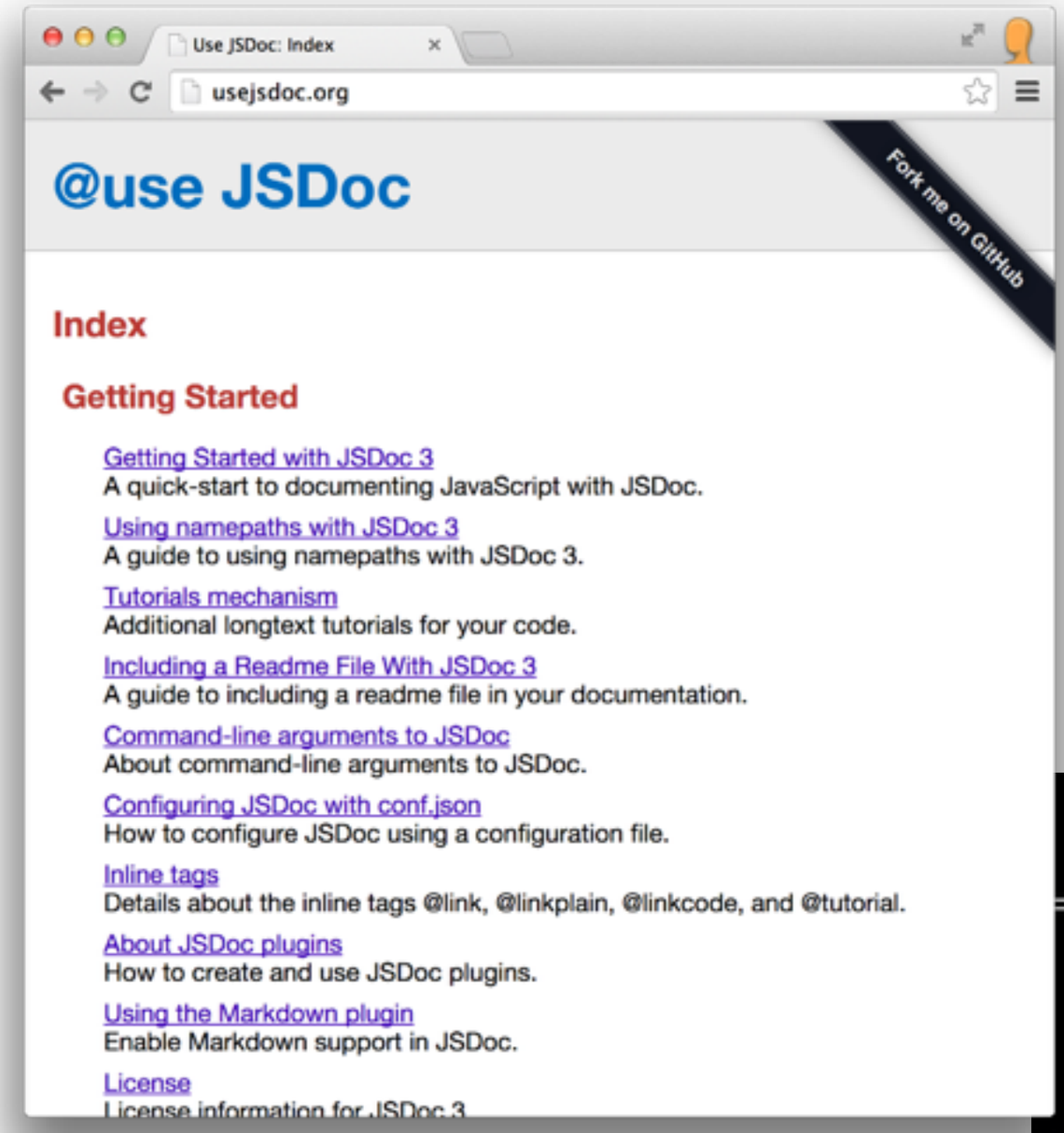
```
$ npm install -g node-inspector

$ node-debug --debug-port 5858 app.js

$ node-inspector &
```



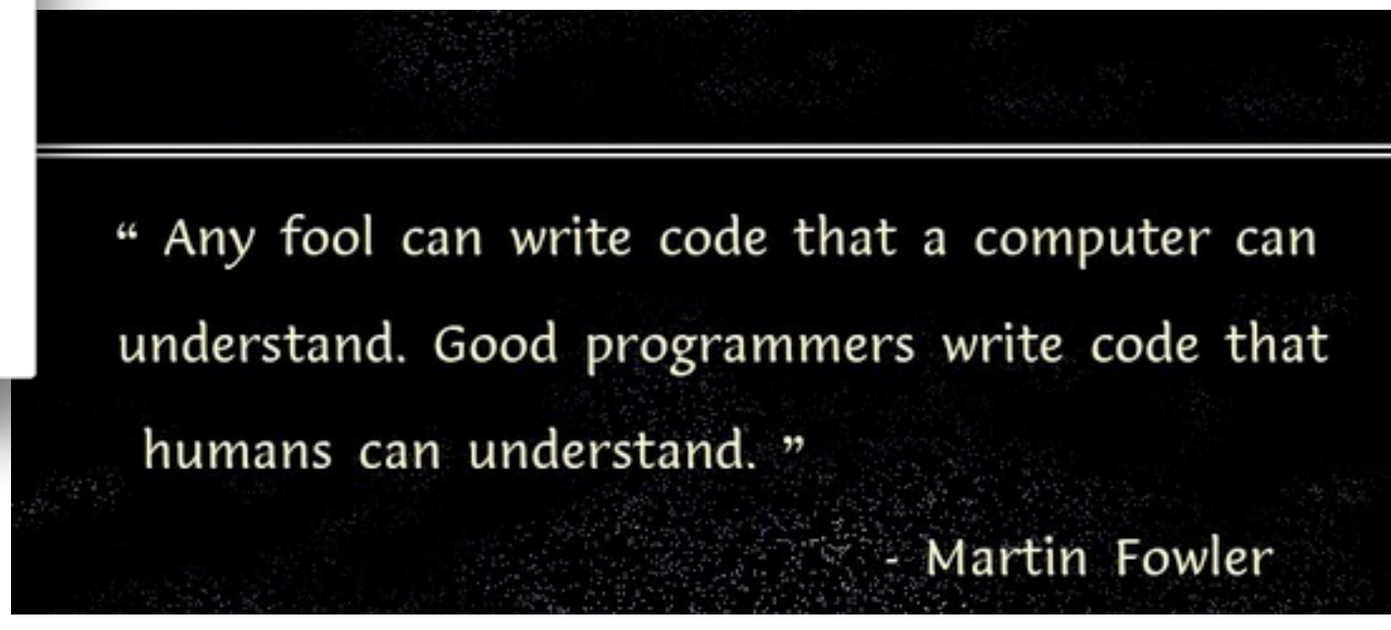
By default, Node.js uses port 5858 for the debugger. If you start several node processes on your host, you will have conflicts. In this case, make sure to use the –`debug-port` option

Document your code

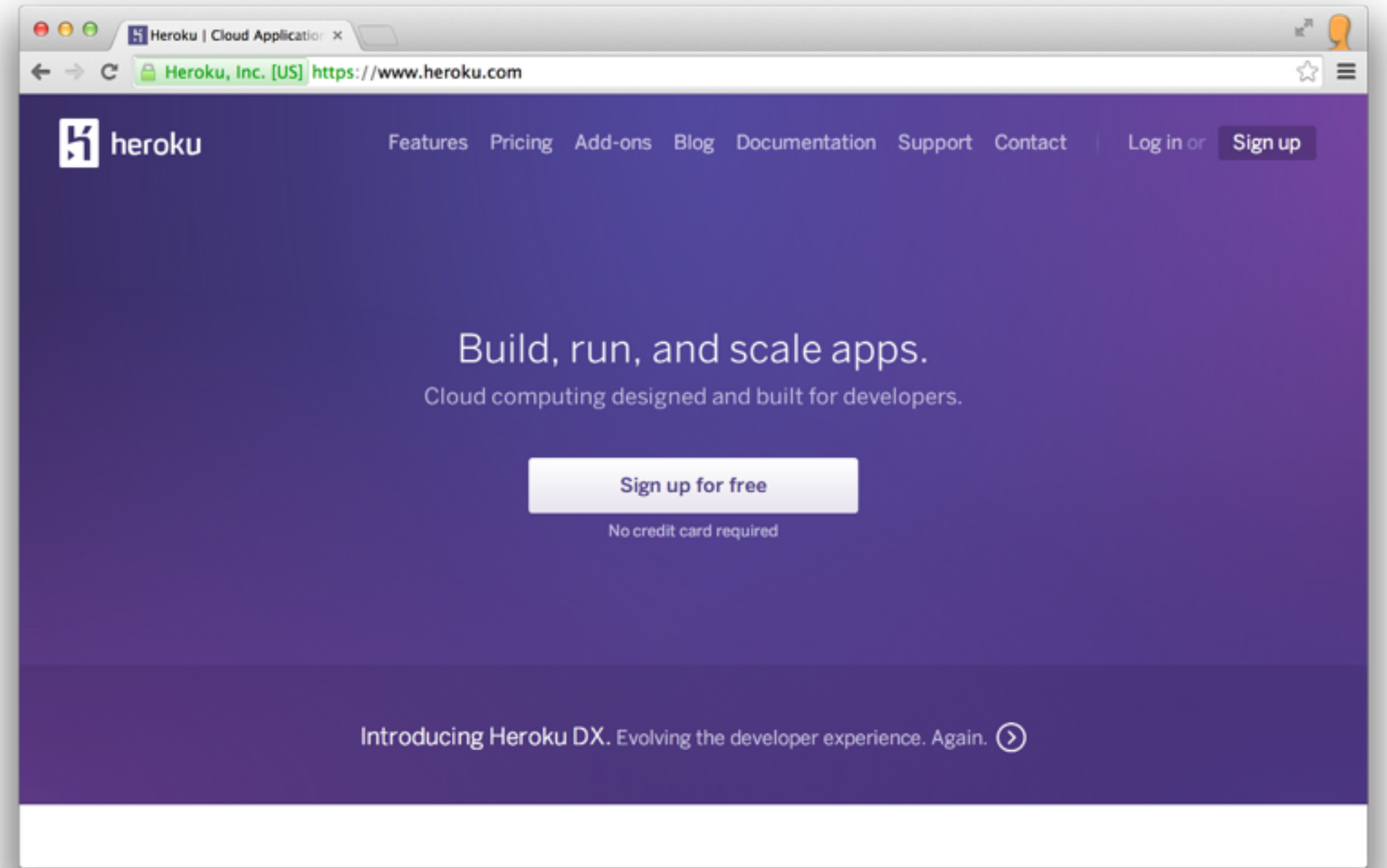
http://usejsdoc.org/
https://github.com/jsdoc3/jsdoc




```
npm install -g jsdoc
jsdoc --destination doc --recurse src
open ./doc/index.html
```

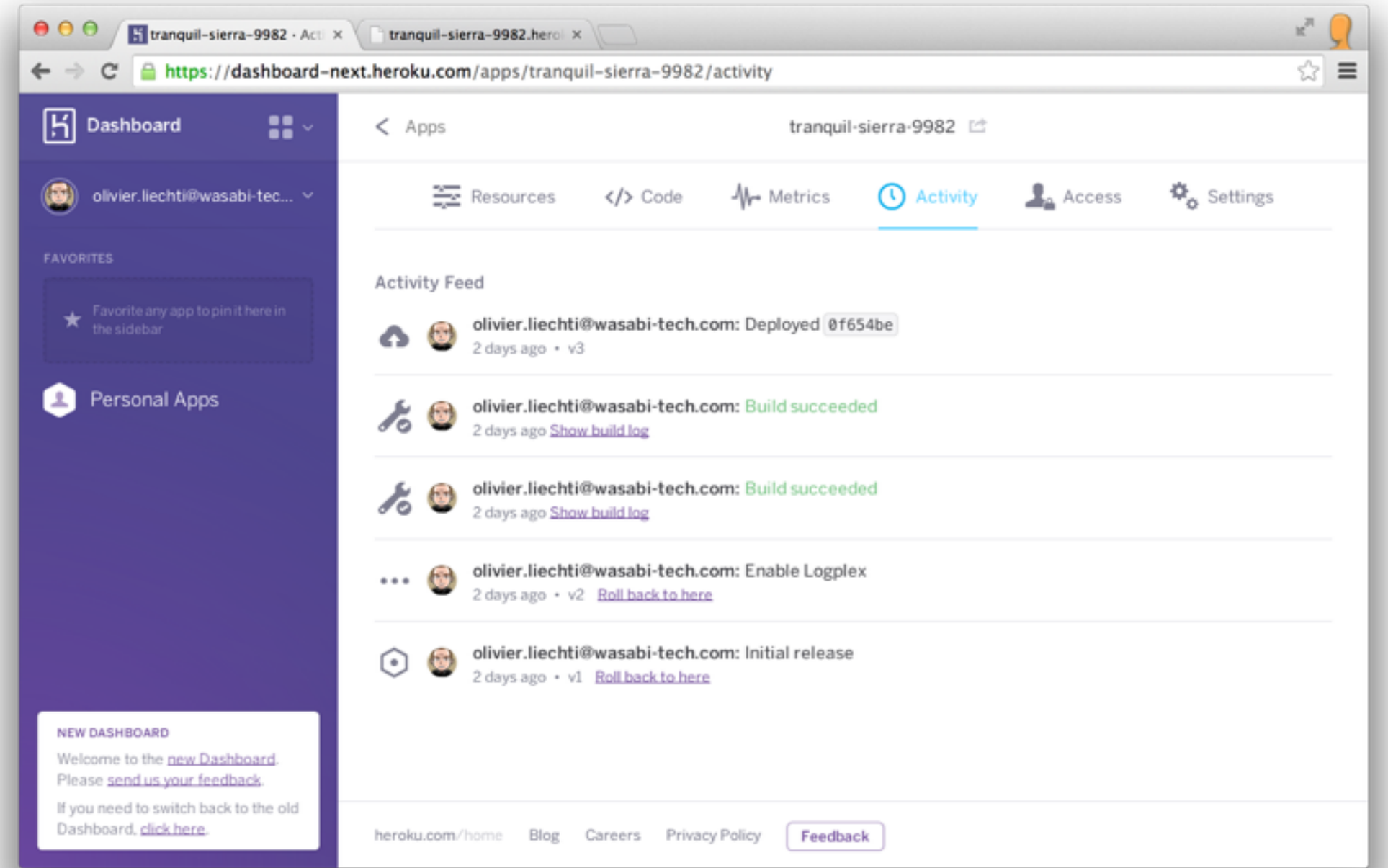


Deploy on the Cloud



https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction

Deploy on the Cloud



https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction

Setup your **Node.js environment**

- **Install Node.js**
- Create and run a very simple program
- Setup the debugger and make sure that you are able to set breakpoints and to analyze the execution of your program
- Install **jsdoc**
- Write a simple HTTP server (based on the previous example in the slides).
- **Generate the doc** with jsdoc.

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Lab

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

The goal of the lab is to develop a web app that queries GitHub repos and displays results in a nice web UI.

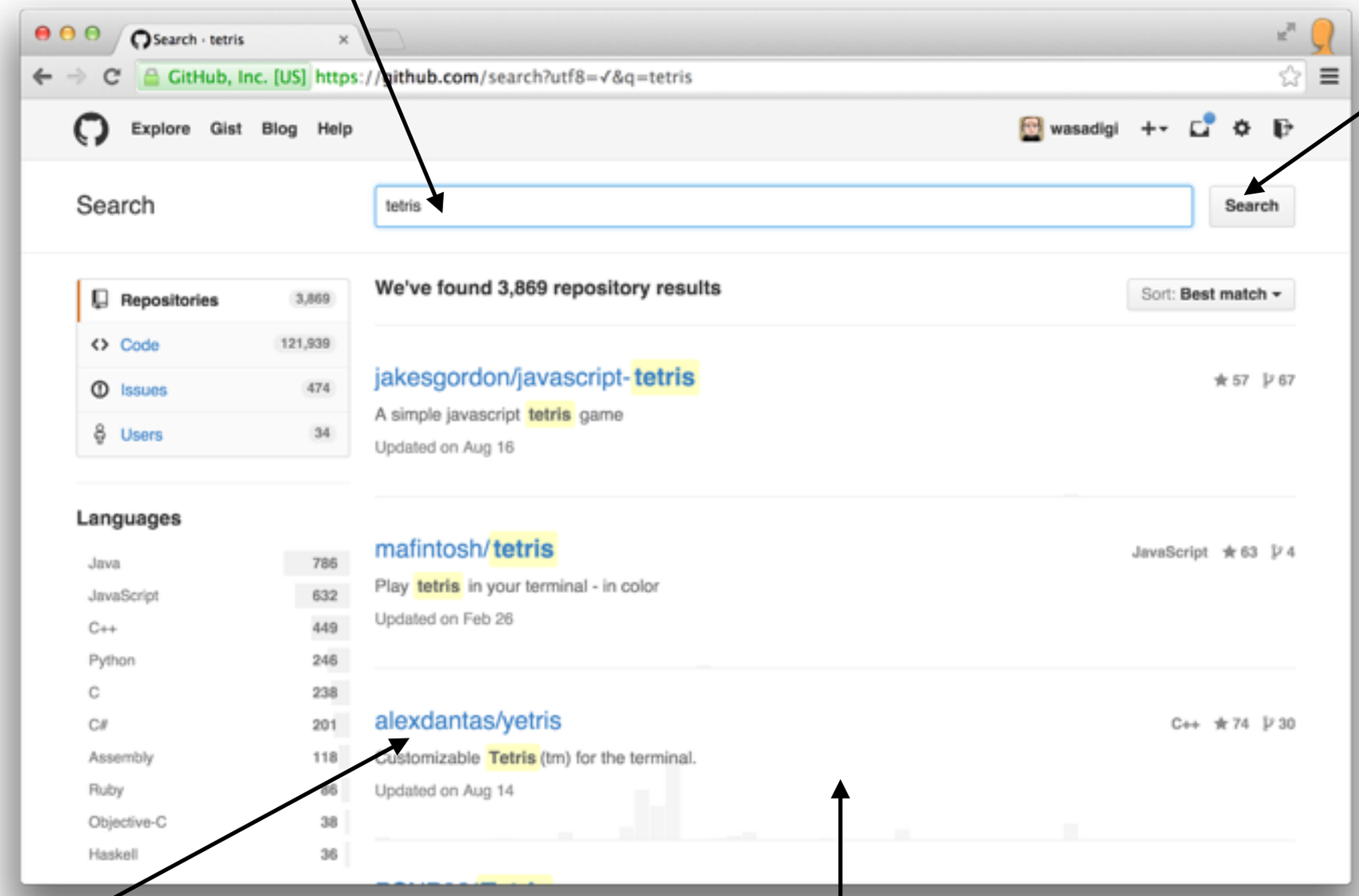
The web app must be deployed in the cloud.

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

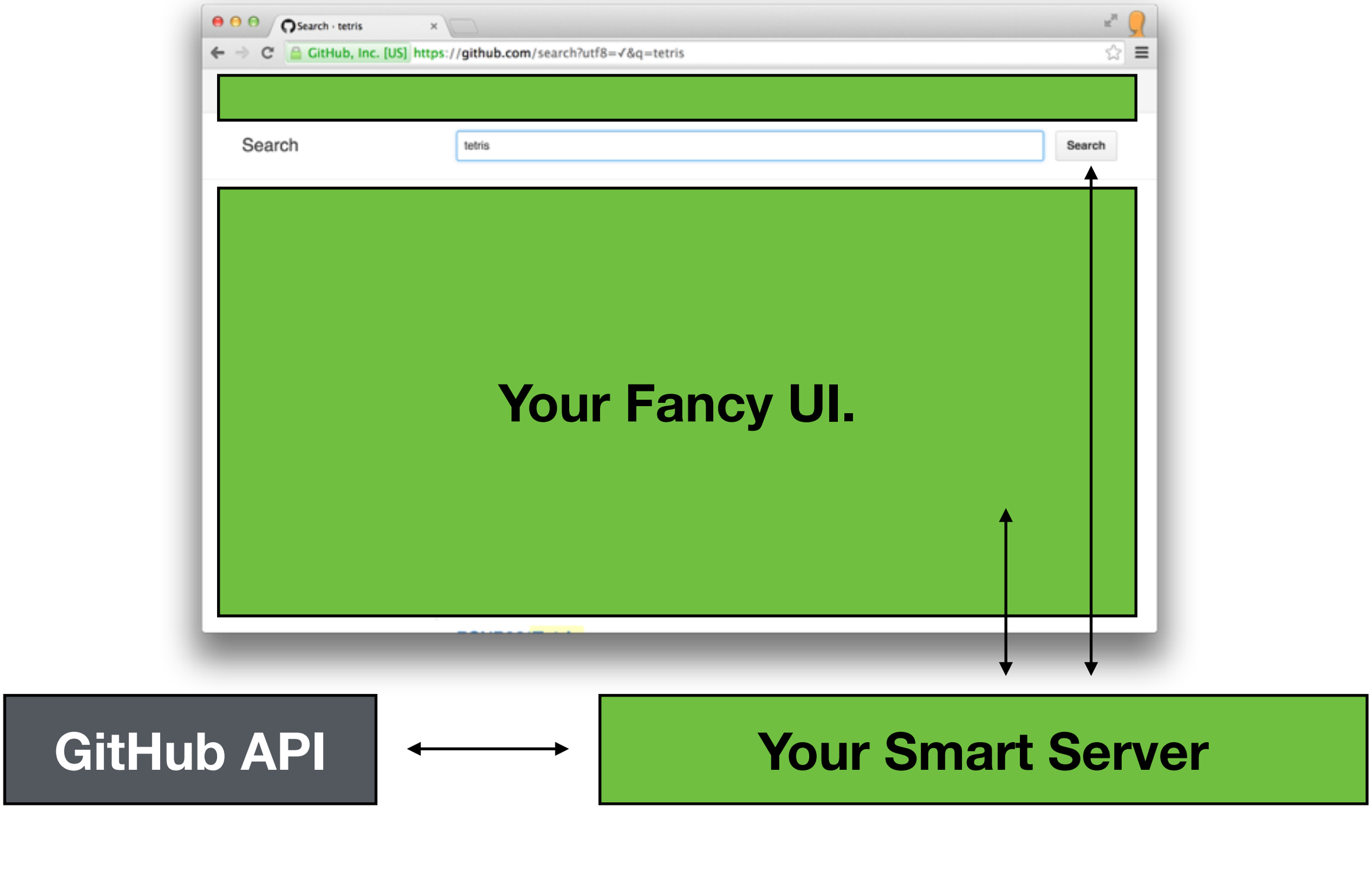
1. User types query

2. User clicks on search



Hyperlinks are provided for cross navigation

3. **List** of repositories is refreshed (no page reload!)


Search · tetris
GitHub, Inc. [US] https://github.com/search?utf8=✓&q=tetris
Search
tetris
Search
Your Fancy UI.
GitHub API
Your Smart Server

Minimum requirements

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- [ ] You have to develop a **dynamic web application**, which consists of:
  - [ ] a back-end server, implemented with **Node.js** and the **Express** framework
  - [ ] a front-end user interface, implemented with **JQuery** and **handlebars**.
- [ ] The user should be able to type a keyword in a "**query**" **text field**.
- [ ] The user should be able to hit a "**search**" **button**.
- [ ] When the user hits the search button, your front-end code should send an **AJAX request** (without blocking the UI) to your back-end.
- [ ] You back-end code should **retrieve a list of Github repositories** containing based on the content of the query text field. It should **transform and forward** the JSON data to your front-end.

Minimum requirements

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- [ ] Your front-end code should **process and render the data** sent back by your back-end service. Make sure that you cleanly separate business logic from presentation logic (think about handlebars).
- [ ] Your back-end service must be deployed in the cloud and be publicly available. You have to use the heroku cloud environment for this purpose.
- [ ] Your back-end service peforms two functions. Firstly, it serves your front-end assets (html, CSS, javascript). Secondly, it processes queries and returns data.

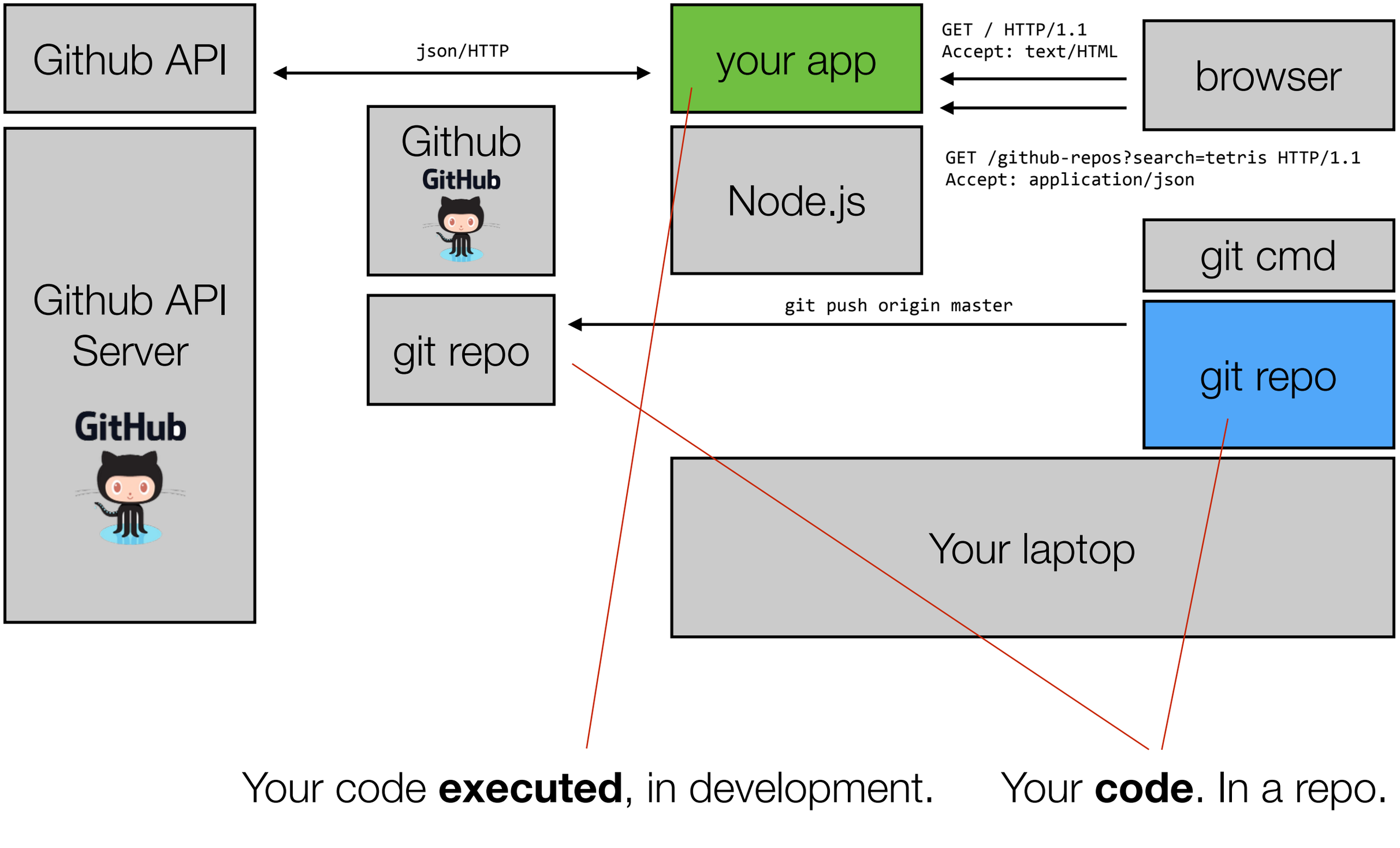
The extra mile

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- [ ] When you are done with the minimum requirements, be creative and propose extensions to the given guidelines.
- [ ] For instance, you may want to integrate other data elements from GitHub (users, activities, ratings, etc.).

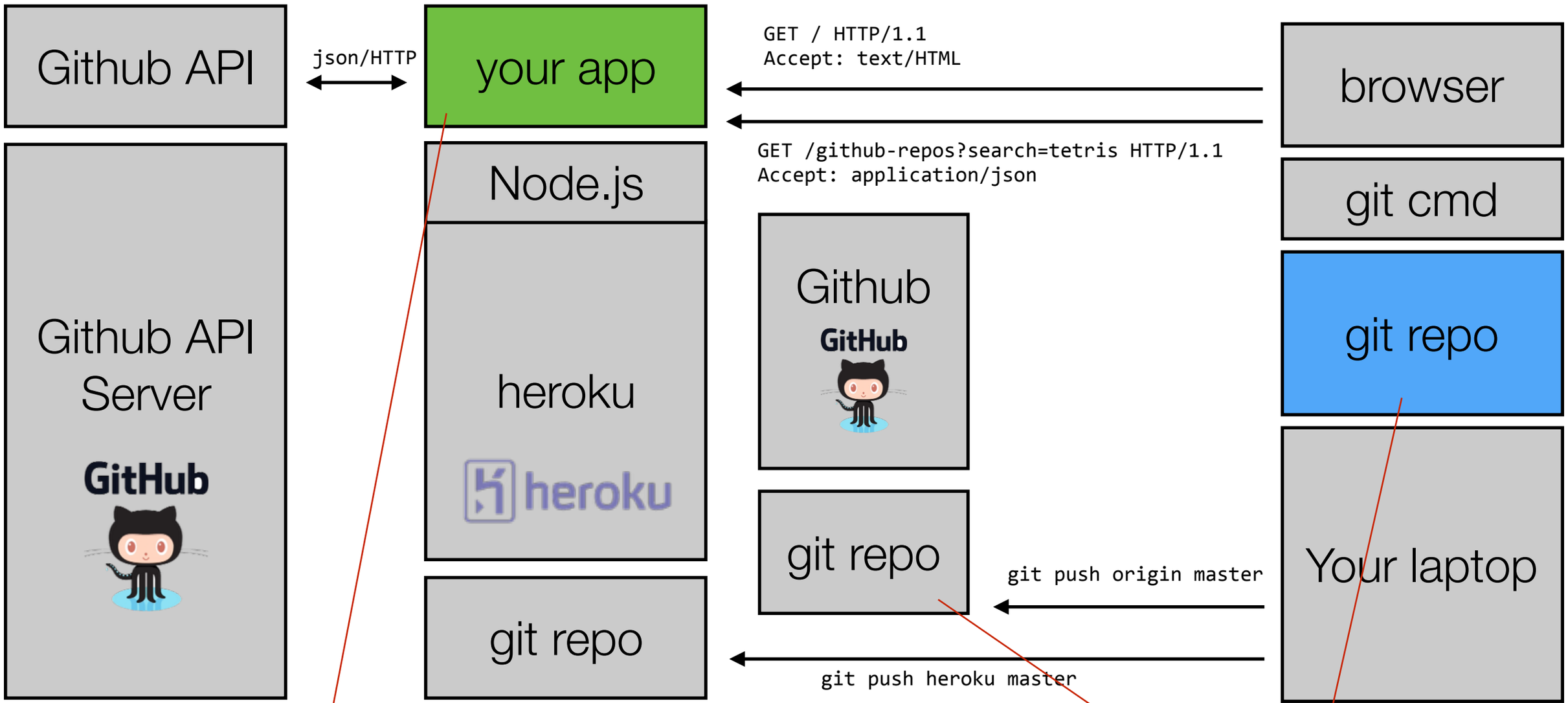
Development Environment

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



Production Environment

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



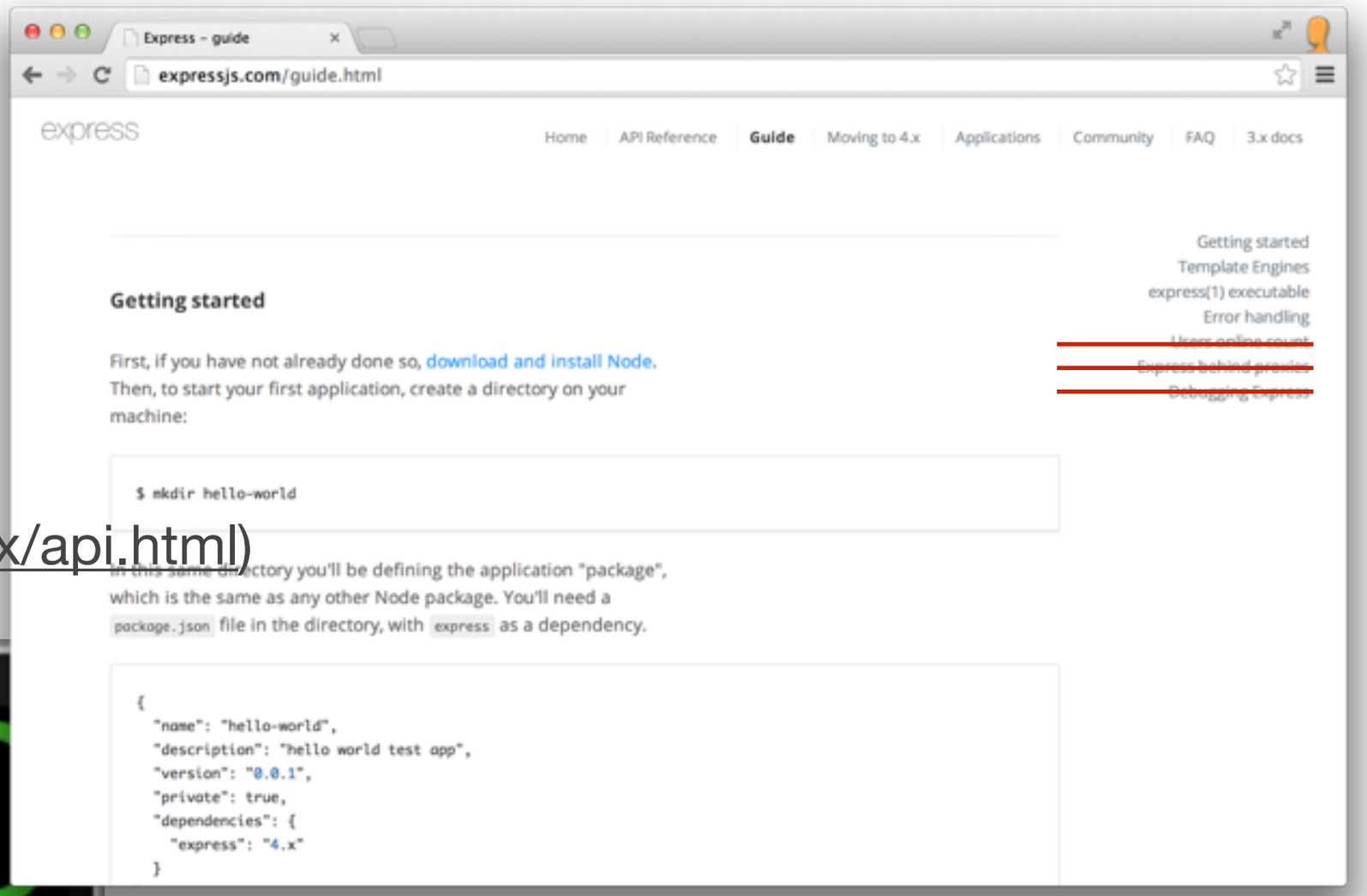
Your code **executed**, in production. **Deployment.** Your **code**. In a repo.

Setup your **environment**

- By now, you have a working Node.js environment.
- **Create a GitHub repo** for the project, right from the start.
- Commit early, commit often.

Get familiar with **Express.js** framework

- **Read and follow the instructions** in the Express.js guide (http://expressjs.com/guide.html)
  - Getting started
  - Template Engines
  - express(1) executable
  - Error handling
- Quickly go through the API (http://expressjs.com/4x/api.html)



Create your own application with **Express.js**

- In this step, the application should simply make it possible for users to access the front-end (i.e. the HTML page containing the text field and the submit button, as well as the linked assets).
- Note that you don't have to use a template engine, it's up to you.
- In other words, opening the browser and typing http://XXX:XX in the navigation bar should load your web UI from the Express.js backend.
- As usual, look & feel **does** matter.
- Test **locally**.
- **Commit and push changes** to your repository.

Read the documentation for the **express.static(root, [options])** function.

Deploy your app on **Heroku**

- Go through the **heroku tutorial** for Node.js developers (https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction)
- Make sure that you ***really*** understand how the different git repositories interact with each other (the one on your machine, the one on Github, the one on heroku).
- In the second step (**Prepare the app**), **do not clone** the “node-js-getting-started.git” app. **You have your own app** and local clone, so use this one!
- If you have issues (you will), here is the doc: https://devcenter.heroku.com/articles/getting-started-with-nodejs-o

If you want heroku to recognize your app, you will need a file named **Procfile** at the root of your project.

If you want heroku to recognize your app, you will need a file named **package.json** at the root of your project.

Heroku will select a TCP port. It will populate the PORT environment variable with this value. You have to pick it up in your Node.js app!

Implement the back-end service

- How are you going to retrieve data from GitHub?
- Don’t reinvent the wheel. People have probably already done similar things and shared their work on the **npm registry**.
- You will have to transform the JSON data obtained from GitHub into the following format.


```
GET /github-repos?search=tetris
Accept: application/json
```

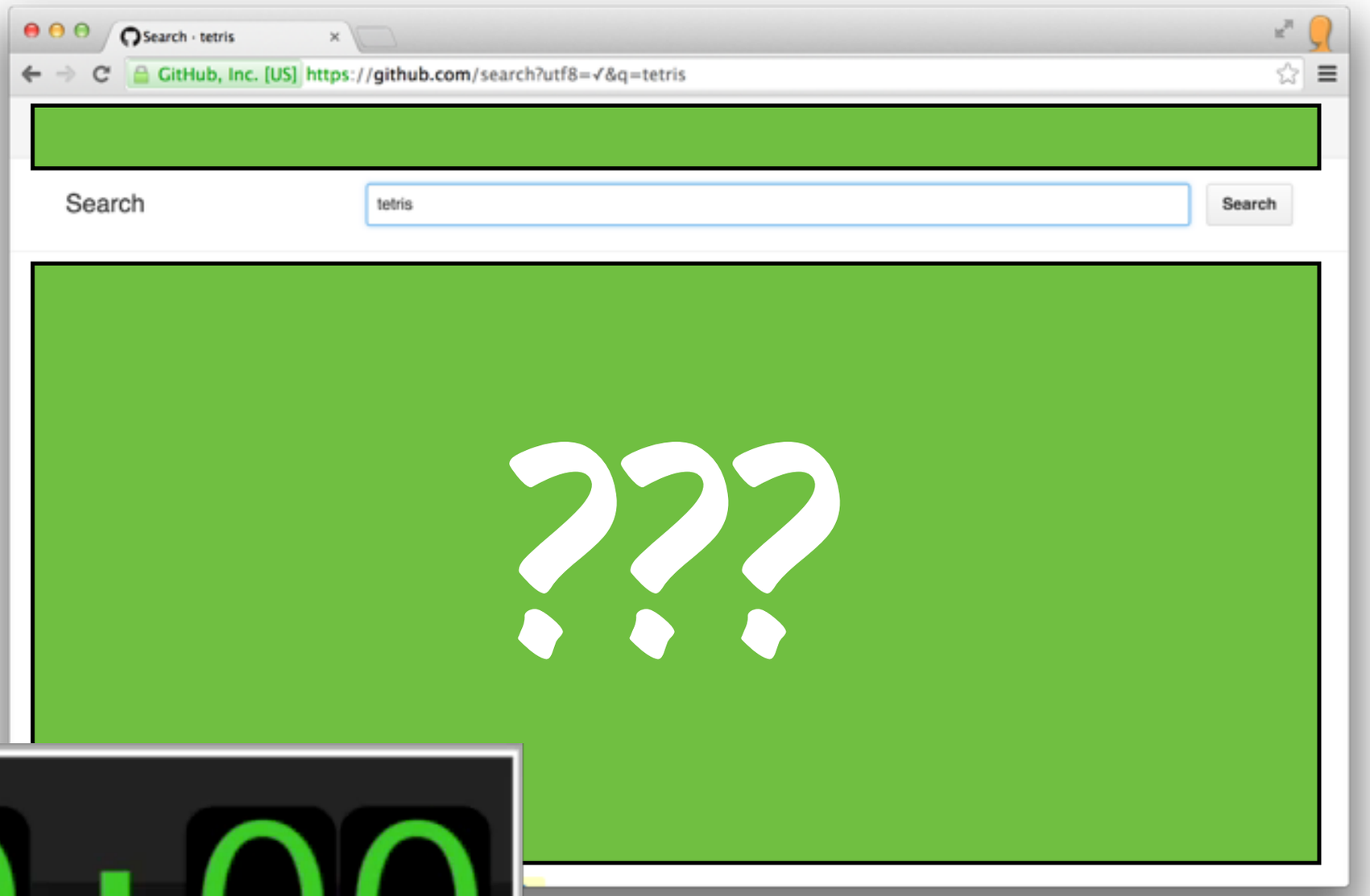
```
{
  "count": 1,
  "repositories": [
    {
      "id": 21095601,
      "name": "Tetris-Duel",
      "full_name": "Tetris-Duel-Team/Tetris-Duel",
      "owner": {
        "login": "Tetris-Duel-Team",
        "avatar_url": "https://avatars.githubusercontent.com/u/7956696?v=2",
        "url": "https://api.github.com/users/Tetris-Duel-Team",
        "html_url": "https://github.com/Tetris-Duel-Team",
      },
      "html_url": "https://github.com/Tetris-Duel-Team/Tetris-Duel",
      "description": "Multiplayer Tetris for Raspberry Pi (in bare metal assembly)",
      "url": "https://api.github.com/repos/Tetris-Duel-Team/Tetris-Duel",
      "created_at": "2014-06-22T14:23:25Z",
      "updated_at": "2014-08-25T01:23:37Z",
      "pushed_at": "2014-07-12T22:13:16Z",
      "stars": 29
    }
  }
}
```

Reploy on Heroku, test and validate with POSTMAN

Postman - REST ...

GitHub API **Your Smart Server**

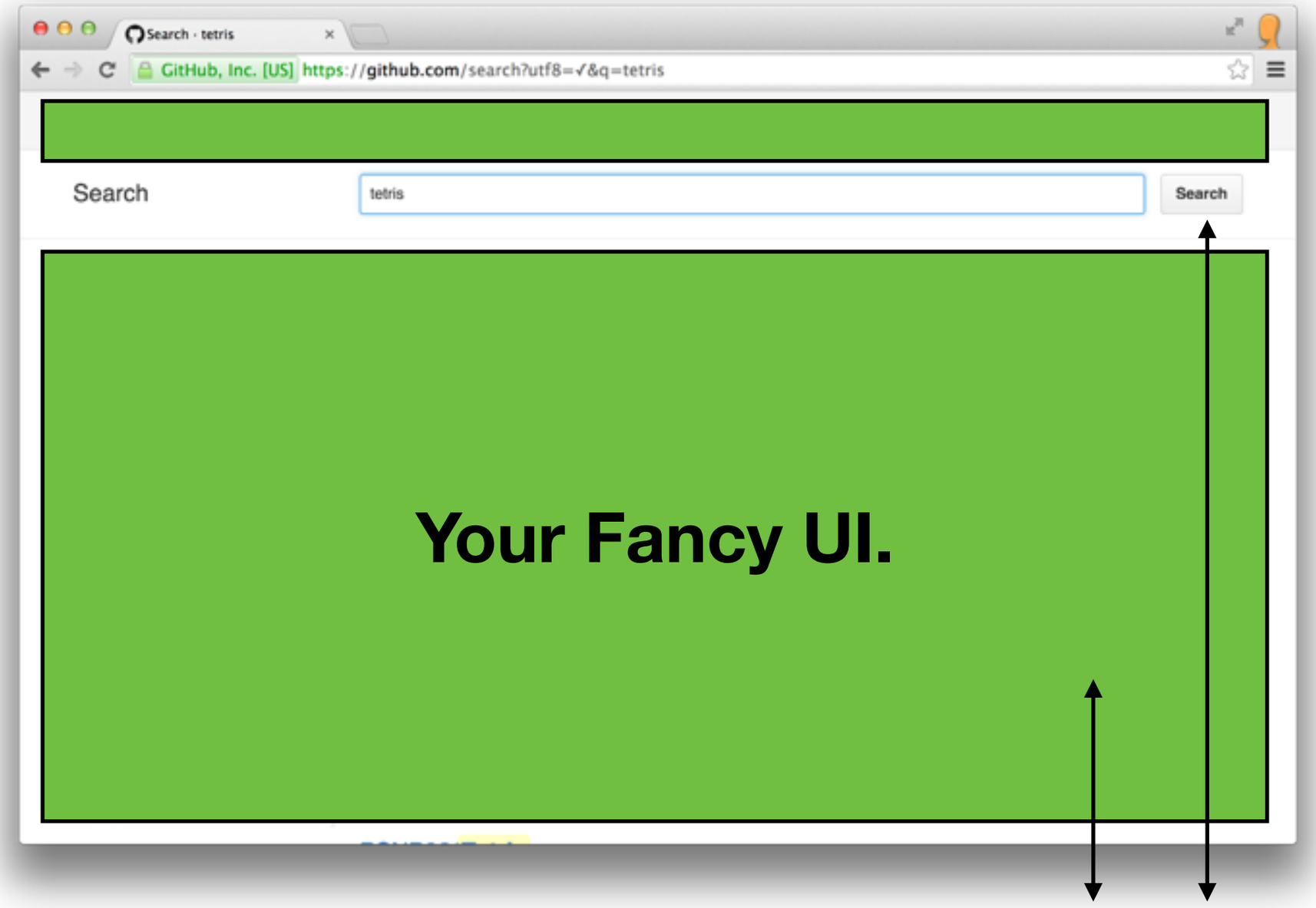
Design and **specify** your results page **on paper**



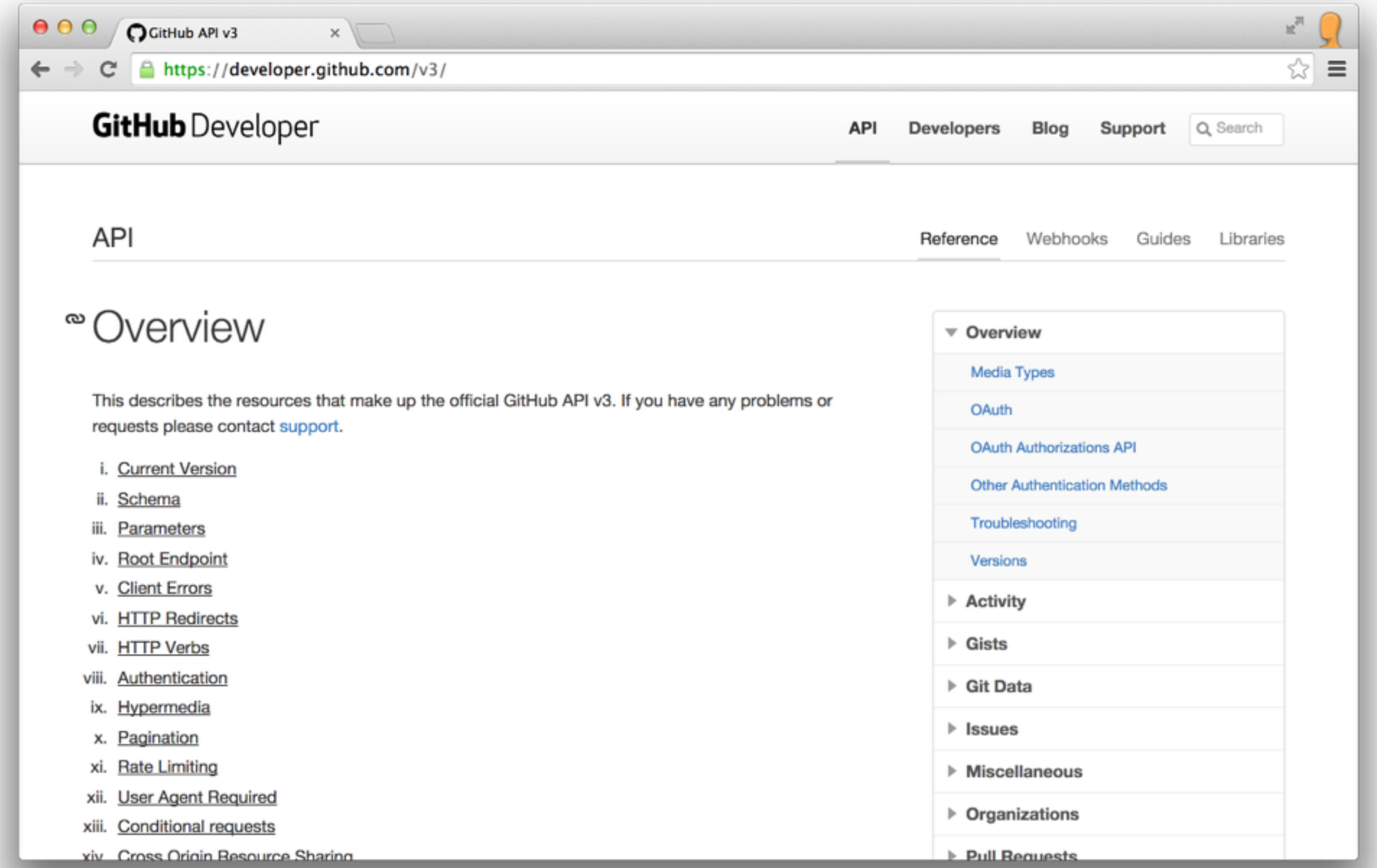
Implement the interactive UI that you have specified in the previous step.

- Use **JQuery** and **handlebars**.
- Keep your code clean and well structured.
- Document your code.

Reploy on Heroku, test and validate with a web browser



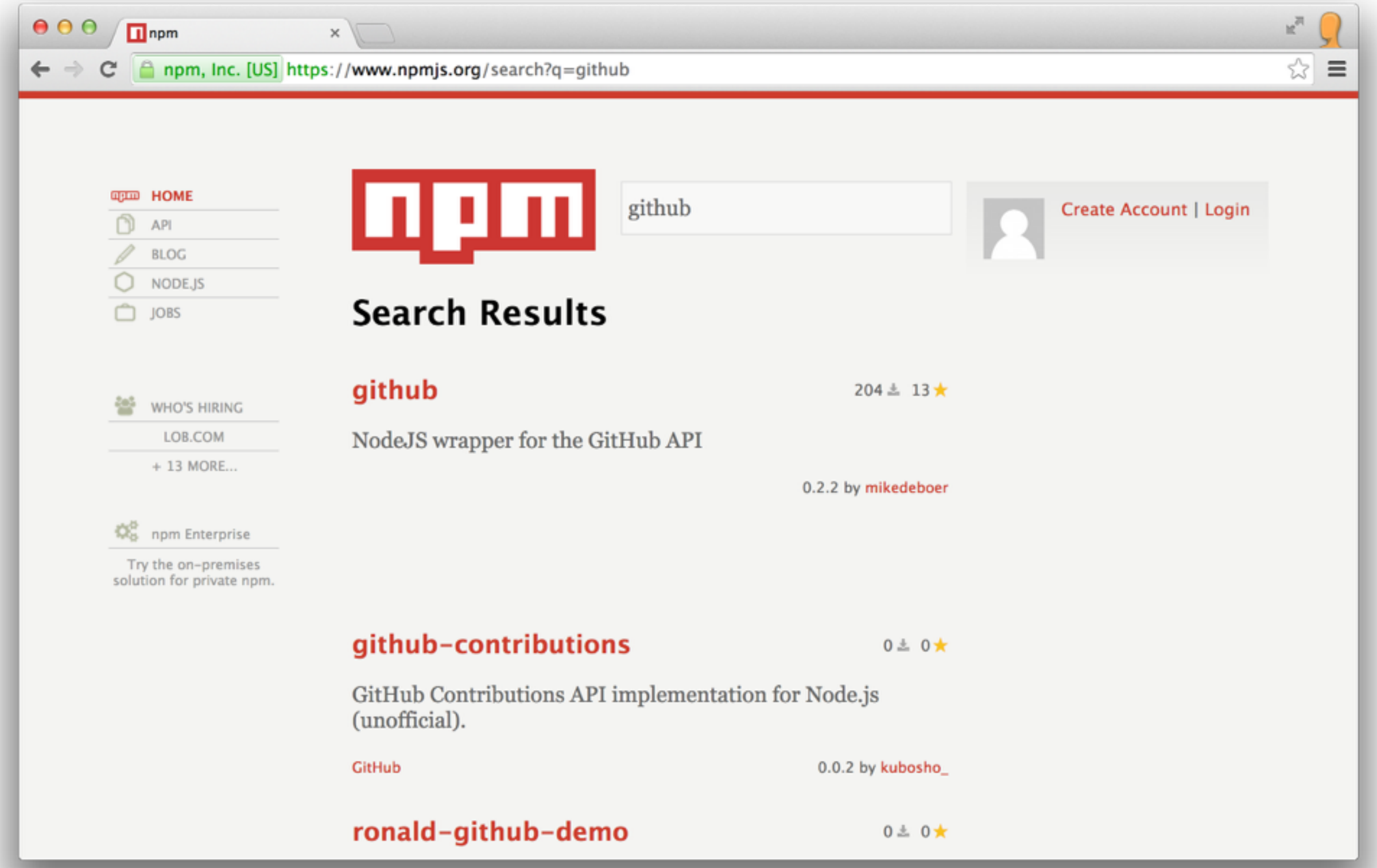
GitHub API ⟷ Your Smart Server


GitHub API v3
https://developer.github.com/v3/
GitHub Developer
API
Developers
Blog
Support
Search
API
Reference
Webhooks
Guides
Libraries
Overview
This describes the resources that make up the official GitHub API v3. If you have any problems or requests please contact support.
i. Current Version
ii. Schema
iii. Parameters
iv. Root Endpoint
v. Client Errors
vi. HTTP Redirects
vii. HTTP Verbs
viii. Authentication
ix. Hypermedia
x. Pagination
xi. Rate Limiting
xii. User Agent Required
xiii. Conditional requests
xiv. Cross Origin Resource Sharing
Overview
Media Types
OAuth
OAuth Authorizations API
Other Authentication Methods
Troubleshooting
Versions
Activity
Gists
Git Data
Issues
Miscellaneous
Organizations
Pull Requests

https://api.github.com/se ×

https://api.github.com/search/repositories?q=heig


```
{
  "total_count": 631,
  "incomplete_results": false,
  "items": [
    {
      "id": 1015650,
      "name": "jquery-iframe-auto-height",
      "full_name": "house9/jquery-iframe-auto-height",
      "owner": {
        "login": "house9",
        "id": 52154,
        "avatar_url": "https://avatars.githubusercontent.com/u/52154?v=2",
        "gravatar_id": "",
        "url": "https://api.github.com/users/house9",
        "html_url": "https://github.com/house9",
        "followers_url": "https://api.github.com/users/house9/followers",
        "following_url": "https://api.github.com/users/house9/following{/other_user}",
        "gists_url": "https://api.github.com/users/house9/gists{/gist_id}",
        "starred_url": "https://api.github.com/users/house9/starred{/owner}{/repo}",
        "subscriptions_url": "https://api.github.com/users/house9/subscriptions",
        "organizations_url": "https://api.github.com/users/house9/orgs",
        "repos_url": "https://api.github.com/users/house9/repos",
        "events_url": "https://api.github.com/users/house9/events{/privacy}",
        "received_events_url": "https://api.github.com/users/house9/received_events",
        "type": "User",
        "site_admin": false
      },
      "private": false,
      "html_url": "https://github.com/house9/jquery-iframe-auto-height",
      "description": "jQuery plugin to set the height of an iframe to its contents height",
      "fork": false,
      "url": "https://api.github.com/repos/house9/jquery-iframe-auto-height",
      "forks_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/forks",
      "keys_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/keys{/key_id}",
      "collaborators_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/collaborators{/collaborator}",
      "teams_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/teams",
      "hooks_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/hooks",
      "issue_events_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/issues/events{/number}",
      "events_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/events",
      "assignees_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/assignees{/user}",
      "branches_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/branches{/branch}",
      "tags_url": "https://api.github.com/repos/house9/jquery-iframe-auto-height/tags"
```


npm
npm, Inc. [US] https://www.npmjs.org/search?q=github
HOME
API
BLOG
NODE.JS
JOBS
WHO'S HIRING
LOB.COM
+ 13 MORE...
npm Enterprise
Try the on-premises solution for private npm.
github
Create Account | Login
Search Results
github
204 13
NodeJS wrapper for the GitHub API
0.2.2 by mikedeboer
github-contributions
0 0
GitHub Contributions API implementation for Node.js (unofficial).
GitHub
0.0.2 by kubosho_
ronald-github-demo
0 0

tranquil-sierra-9982 · Cod ×

https://dashboard-next.heroku.com/apps/tranquil-sierra-9982/code

Dashboard

olivier.liechti@wasabi-tec...

FAVORITES

Favorite any app to pin it here in the sidebar

Personal Apps

NEW DASHBOARD

Welcome to the new Dashboard.
Please send us your feedback.

If you need to switch back to the old Dashboard, click here.

Apps

tranquil-sierra-9982

Resources | Code | Metrics | Activity | Access | Settings

Install the Toolbelt

If you haven't already, you'll need to download and install the Heroku Toolbelt, which provides the Heroku command-line client and Git for version control.

Once you've installed the Toolbelt, start by logging in on the command line:

```
$ heroku login
  Enter your Heroku credentials.
  Email: olivier.liechti@wasabi-tech.com
  Password:
  Could not find an existing public key.
  Would you like to generate one? [Yn]
  Generating new SSH public key.
  Uploading ssh public key ~/.ssh/id_rsa.pub
```

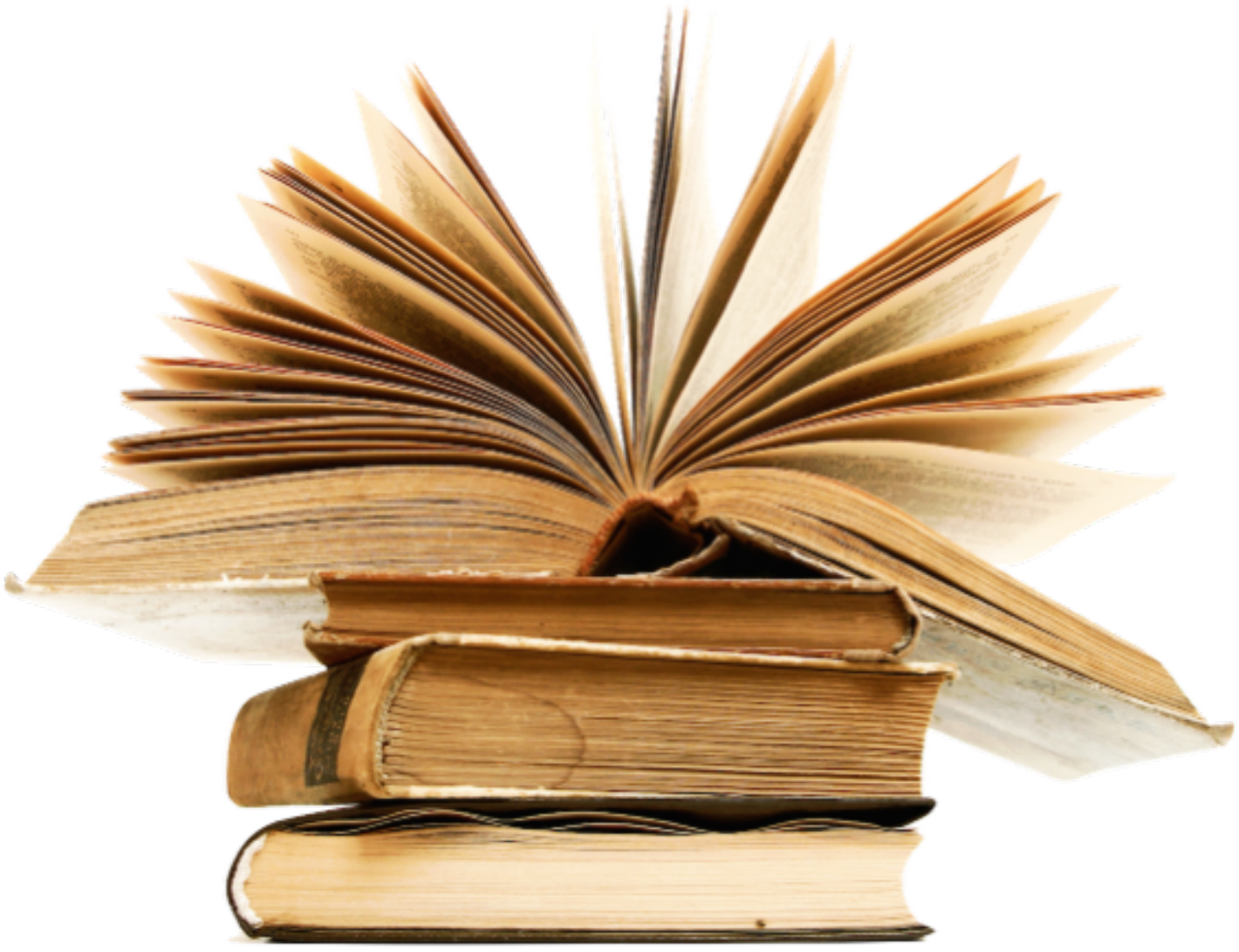
If you have previously uploaded a key to Heroku, we assume you will keep using it and do not prompt you about creating a new one during login. If you would prefer to create and upload a new key after login, simply run `heroku keys:add`.

Clone the Repository

Use Git to clone tranquil-sierra-9982's source code to your local machine.

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



References

MUST READ for the Tests

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- **Understanding the Node.js Event Loop**
  - http://strongloop.com/strongblog/node-js-event-loop/
- **Mixu's Node book: What is Node.js? (chapter 2)**
  - http://book.mixu.net/node/ch2.html
- **Node.js Explained, video**
  - http://kunkle.org/talks/