# FORMATION FULL STACK

הרווחה, משרד העבודה
והשירותים החברתיים

# Node.js

By Charles Cohen

Charles Cohen
Senior Full Stack Developer
charlesc@edandweb.co.il
050-4449764

# Programme du cours Node.js

1. Introduction to Node.js
   - What is Node and what is it not ?
   - Node.js Features ?
   - Our first Node.js script: Hello World
   - Building a web server in Node.js
   - Debugging node applications
2. Building your Stack
   - Pulling in other libraries
   - Building custom libraries
   - A-synchronicity and callbacks
   - Blocking vs. non-blocking I/O
   - Working within the event loop
3. Modular JavaScript with Node.js
   - Writing Modular JavaScript with Node.js
   - Core Modules
   - Installing Packages
   - Publishing packages
4. Avoiding common pitfalls with Async.js
   - Introducing the Asynchronous problem
   - Async.js Library to the rescue
   - Collections
   - Flow Controllers
5. Working with the file system
   - Files manipulations
   - Folder manipulations
   - Putting the file-system module together Async.js
6. Building Web applications with the Express Framework
   - Introduction to Express, installation and basic setup
   - Application configuration
   - Routing
   - Views and Templating options
   - Persistence with Cookies, In-Memory Sessions and session-stores
   - Social Authentication with Passport.js
7. Connecting MySQL Server
   - Database connection
   - A-synchronicity Queries from node.js

# Cours 3

## Asynchrone et requête

# Plan du module Node.js

| Cours | Date | Cours |
|---|---|---|
| 1 | Lun. 20/05 | • Introduction to Node.js (1) |
| 2 | Mer. 29/05 | Building your Stack (2)<br>• Command Line<br>• File System<br>• Arrow Functions<br>• Modular JavaScript with Node.js |
| 3 | Mer. 05/06 | • Asynchronous JS<br>• Consuming API – HTTP requests |
| 4 | Lun. 10/06 | • Consuming API – HTTP requests<br>• Building Web applications with the Express Framework |
| 5 | Lun. 17/06 | • Express framework – templates with Handlebars<br>• Building a get API entry point based on previous exercices (get coordinates and weather from mapbox.com and darksky.net) and returning data as JSON<br>• Implement a search address box that consuming the GET API |
| 6 | Lun. 24/06 | • Promise, await, async, Async module …<br>• Avoiding common pitfalls with Async.js |
| 7 | Lun. 01/07 | • Working with MongoDB 1/2 |
| 8 | Lun 08/07 | • Working with MongoDB 1/2 |

# Environnement de travail

Editeur de code:

- Visual Studio Code
    - Extensions:

Liens Utiles:

- Node.js: https://nodejs.org/
- Moteur V8 Javascript: https://v8.dev/

# Questions/Sujets à enrichir

- Files: If Node.js reading all the file or do we have a buffer ?
- How can we run our code outside the command line ?
  Write our own server and set an entry point (API: POST, GET, PUT, PATCH)

# Résumé du cours 2

- Création d'un serveur basique avec node.js (http.createServer …)
- Utilisation de la librairie chalk pour coloriser les messages dans la console
- Découverte du packet nodemon (lancement en continu d'un script)
- Installation d'un module en mode globalle npm install -g nodemon  s ( pas d'impact sur package.json de notre projet)
- Ligne de commande (process.argv ), affichage argument (process.argv [2])-
- Yargs: ligne de commande:  yargs.command, command, describe, handler, builder, describe, demandOption, type
- Gestion des erreurs: try { } catch (e) {}
- JSON.parse , JSON.stringify
- writeFileSync, readFileSync
- Debugger avec Node.js node inspect app.js et aller dans chrome et tapper chrome://inspect/ et verifier dans configure que les adresses localhost:9229 et 127.0.0.1:9229 sont bien renseignées.

# Résumé du cours 1

- Node.js is a Javascript **runtime** built on **Chrome's V8** Javascript engine.
- Node.js uses an **event-driven**, **non-blocking**
- Node.js' package ecosystem, **npm**, is the largest ecosystem of open source libraries
- Node.js => JS Code => V8 (C++) => Result
- Le nom de l'objet global est **global** et l'objet équivalent à Document dans le browser est **process**
- Pour utiliser la librairie des fichiers (FileSystem) on utilisera la commande const fs = require('fs');
- Pour écrire on utilisera la méthode WriteFileSync
- Système de modules de Node.js (Filesystem …), API: https://nodejs.org/dist/latest-v10.x/docs/api/
- Nos scripts require('./utils.js'); ➔ module.exports
- npm init
- Packets npm (npm install validator)
- Projet existant: npm install

# Implémentation de la fonction addNote

```
8   const addNote = function(title, body){
9       let notes = loadNotes(NOTES_FILE);
10
11      notes.push({
12          title: title,
13          body: body
14      });
15
16      saveNotes(notes);
17  };
18
19  const saveNotes = function(notes){
20      const dataJSON = JSON.stringify(notes);
21      fs.writeFileSync(NOTES_FILE, dataJSON);
22  };
23
```

Maintenant que nous avons réussit à charger les notes nous allons ajouter à chaque appel de la commande la nouvelle note.

La variable notes contient un array et pour y ajouter une nouvelle note nous pouvons utiliser la méthode **push** .

Créons une fonction saveNote qui sera responsable de la sauvegarde des notes:

1. JSON.stringify
2. wrtiteFileSync

Relancer la commande add et vérifier que le fichier notes.json se met à jour.

# RAPPEL - Array - Méthodes Itératives

5 méthodes itératives existent pour les Array.
Elle reçoive en arguments 2 paramètres le premier obligatoire et le second optionnel.
Le premier est **une fonction qui sera exécuté sur chacun des éléments de l'Array**.
Le second est le contexte dit le scope (influant sur la valeur de ***this***) on verra plus tard.

| Méthodes itératives | |
|---|---|
| .every() | Exécute la fonction sur tous les éléments de l'Array et retourne true si la fonction à retourné true pour tous les éléments de l'Array . |
| .filter() | Exécute la fonction sur tous les éléments de l'Array et retourne un Array de tous les éléments pour lesquels la fonction à retourné true. |
| .forEach() | Exécute la fonction sur tous les éléments de l'Array et ne retourne aucune valeur. |
| .map() | Exécute la fonction sur tous les éléments de l'Array et retourne un Array avec les résultats. |
| .some() | Exécute la fonction sur tous les éléments de l'Array et retourne true si la fonction à retourné true pour au moins un élément de l'Array . |

# Vérification de note dupliquée

Pour vérifier que le titre de la note n'existe pas, nous utiliserons filter sur notre array.

Le but est de créer un array avec les notes dupliquées et de vérifier que sa propriété length est bien à 0.

```
const addNote = function(title, body){
    const notes = loadNotes(FILE_NOTES);
    const duplicatNotes = notes.filter(function(note){
        return note.title === title;
    });

    if (duplicatNotes.length === 0) {
        notes.push({
            title: title,
            body: body
        });

        saveNotes(notes);
        console.log(chalk.green.inverse('New note added!'));
    } else {
        console.log(chalk.blue.inverse('Note title taken!'));
    }


};
```

# Les fonctions Arrow => en JS

# Arrow function (x) => { }

```
const square = function(x){
    return x * x;
};

console.log(square(3));
```

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app\playground>

[nodemon] 1.18.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
```

```
// const square = function(x){
//     return x * x;
// };

const square = (x) => {
    return x * x;
};

console.log(square(3));
```

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app\playground>

[nodemon] 1.18.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
```

# Shorthand arrow function

File Edit Selection View Go Debug ··· 2-arrow-function.js - n

app.js | notes.js | 2-arrow-function.js × | {..} notes

```
// const square = function(x){
//     return x * x;
// };

// const square = (x) => {
//     return x * x;
// };

const square = (x) => x * x;

console.log(square(3));
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: node

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app\playground>

[nodemon] 1.18.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
```

# Arrow function for methods

Le context d'une fonction arrow depend de l'endroit ou elle est crée

```
// const square = (x) => x * x;

// console.log(square(3));

const event = {
    name: 'Birthday Party',
    printGuestList: function(){
        console.log('Guest List for ' + this.name);
    }
};

event.printGuestList();
```

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app\playground> nodemon

[nodemon] 1.18.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for Birthday Party
[nodemon] clean exit - waiting for changes before restart
```

```
// const square = (x) => x * x;

// console.log(square(3));

const event = {
    name: 'Birthday Party',
    printGuestList: () => {
        console.log('Guest List for ' + this.name);
    }
};

event.printGuestList();
```

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app\playground> nodemon .\

[nodemon] 1.18.10
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for Birthday Party
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for undefined
[nodemon] clean exit - waiting for changes before restart
```

# Solution - arrow functions for methods

File Edit Selection View Go Debug ··· 2-arrow-function.js - notes-app - Vis

app.js | notes.js | 2-arrow-function.js × | notes.json

```
// const square = (x) => x * x;

// console.log(square(3));

const event = {
    name: 'Birthday Party',
    printGuestList() {
        console.log('Guest List for ' + this.name);
    }
};

event.printGuestList();
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: node

```
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
9
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for Birthday Party
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for undefined
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for Birthday Party
[nodemon] clean exit - waiting for changes before restart
[nodemon] restarting due to changes...
[nodemon] starting `node .\2-arrow-function.js`
Guest List for Birthday Party
[nodemon] clean exit - waiting for changes before restart
```

0 0 Live Share Go Live Ln 18, Col 3 Spaces: 4 UTF-8 CRLF JavaScript

app.js | notes.js | notes.json | package.json | notes_oldstyle.js | 2-a

```
// console.log(square(3));

const event = {
    name: 'Birthday Party',
    guestList: ['Charles', 'Sarah', 'Michelle'],
    printGuestList() {
        console.log('Guest List for ' + this.name);
        this.guestList.forEach( (guest) => {
            console.log(guest + ' is attending ' + this.name);
        });
    }
};

event.printGuestList();
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app\playground> node .\2-arrow-function.js
Guest List for Birthday Party
Charles is attending Birthday Party
Sarah is attending Birthday Party
Michelle is attending Birthday Party
```

# Transformation de notre application avec des fonctions arrow

File Edit Selection View Go Debug Terminal Help notes.js - notes-app - Visual Stu

app.js | notes.js | notes.json | notes_oldstyle.js

```
41  const saveNotes = (notes) => {
42      const dataJSON = JSON.stringify(notes);
43      fs.writeFileSync(FILE_NOTES, dataJSON);
44  };
45
46  const list = () => {
47      const notes = loadNotes(FILE_NOTES);
48
49      console.log(FILE_NOTES);
50      console.log(notes);
51
52      notes.forEach(note => {
53          console.log(chalk.magenta.inverse(note.title));
54          console.log(chalk.yellow.inverse(note.body));
55      });
56  };
57
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: powershell

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app> node .\app.js list
Listing all the notes
notes.json
[ { title: 't', body: 'b' },
  { title: 'Course ideas', body: 'New method List' } ]
t
b
Course ideas
New method List
PS C:\Users\charl\projects\learn_NODEJS\notes-app>
```

File Edit Selection View Go Debug Terminal Help notes.js - notes-app - Visual Studio Code

app.js | notes.js | notes.json | notes_oldstyle.js

```
50
51  const readNote = (noteTitle) => {
52
53      const notes = loadNotes(FILE_NOTES);
54
55      const note = notes.find((note) => note.title === noteTitle);
56
57      if (!note) {
58          console.log(chalk.green.inverse('Note: ' + noteTitle + ' not f
59      } else {
60          console.log(chalk.green.inverse('This is the requested note:')
61          console.log(note.title);
62          console.log(note.body);
63      }
64  };
65
66  const saveNotes = (notes) => {
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: powershell

```
PS C:\Users\charl\projects\learn_NODEJS\notes-app> node .\app.js read --title="t"
Reading a specific note
This is the requested note:
t
b
PS C:\Users\charl\projects\learn_NODEJS\notes-app>
```

EXERCICES & TP

# Install a npm package

Ecrire une fonction getToDo fonction qui devra récupérer les taches non complétées (false) à l'aide de filter.

Utiliser une arrow fonction pour les objets avec ES6.

```
const todo = {
        todos: [
                {
                text: 'Achats pour la maison',
                completed: true
                },
                {
                text: 'Rendre le livre au voisin',
                completed: false
                },
                {
                text: 'Aller chercher mon costume',
                completed: false
                }
        ]
};
```

# Asynchrone avec setTimeout et setInterval

Edit Selection View Go Debug Terminal Help app.js - weather-app - Visu

app.js

```javascript
console.log('Starting');

setTimeout(() => {
    console.log('2 seconds Timer');
}, 2000);

setTimeout(() => {
    console.log('0 seconds Timer');
}, 0);


console.log('Stop');
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: powershell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
Starting
Stop
2 seconds Timer
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
Starting
Stop
0 seconds Timer
2 seconds Timer
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
```

Live Share Ln 9, Col 5 Spaces: 4 UTF-8 CRLF JavaScript

L'asynchrone permet l'exécution de code, pendant qu'un traitement en fond est exécuté par Node.js .

En soit l'exécution de notre code utilise le système d'event loop de Node.js qui est singlethread mais Node.js utilise les possibilités multithread du C++ .

A chaque exécution de notre script, Node.js va englober notre script dans une fonction appelé main.

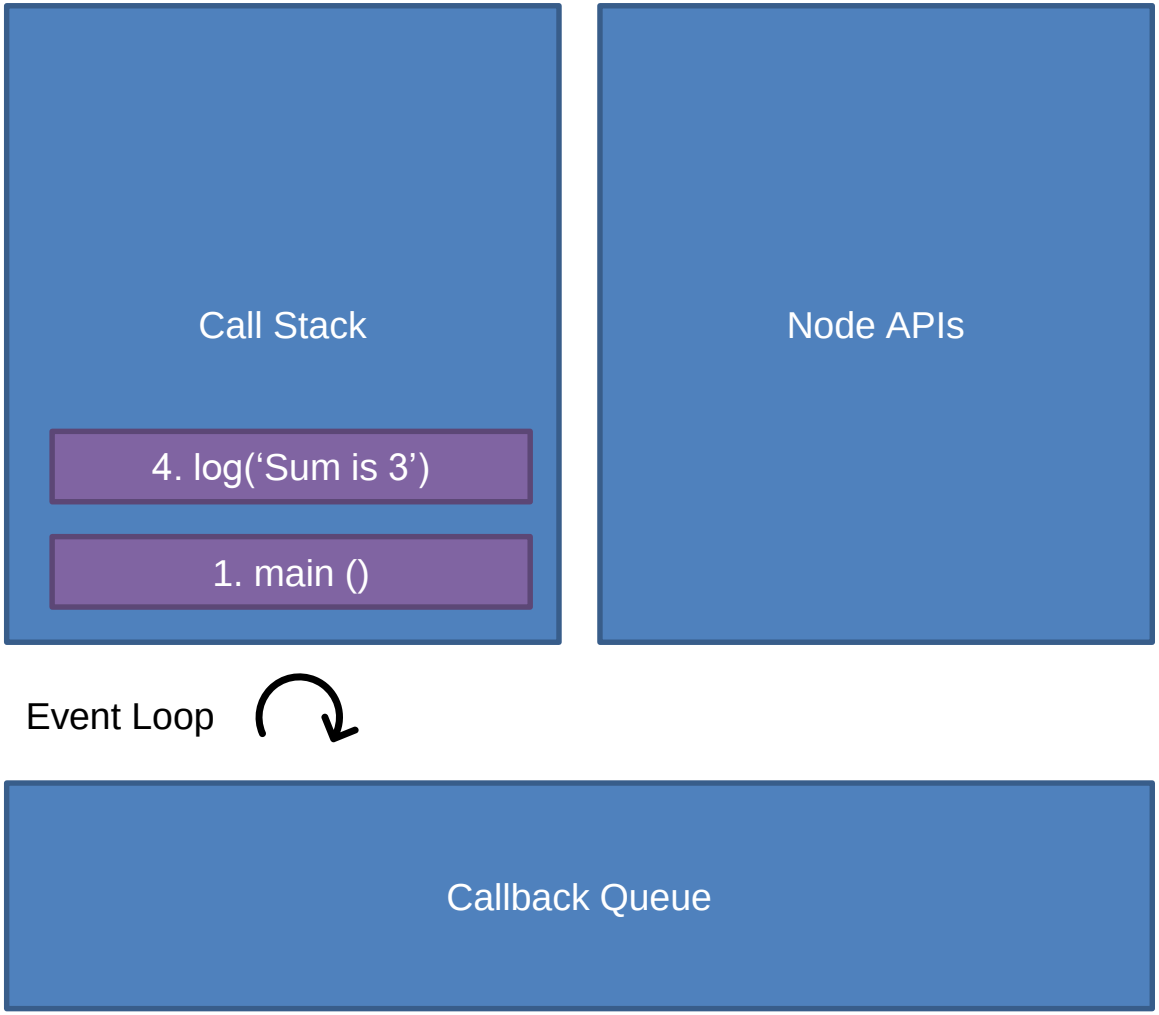
# Exemple code synchrone

```
2. const x = 1;

3. const y = x + 2;

4. console.log('Sun is ' + y);
```

Sum is 3



# Exemple code synchrone 2

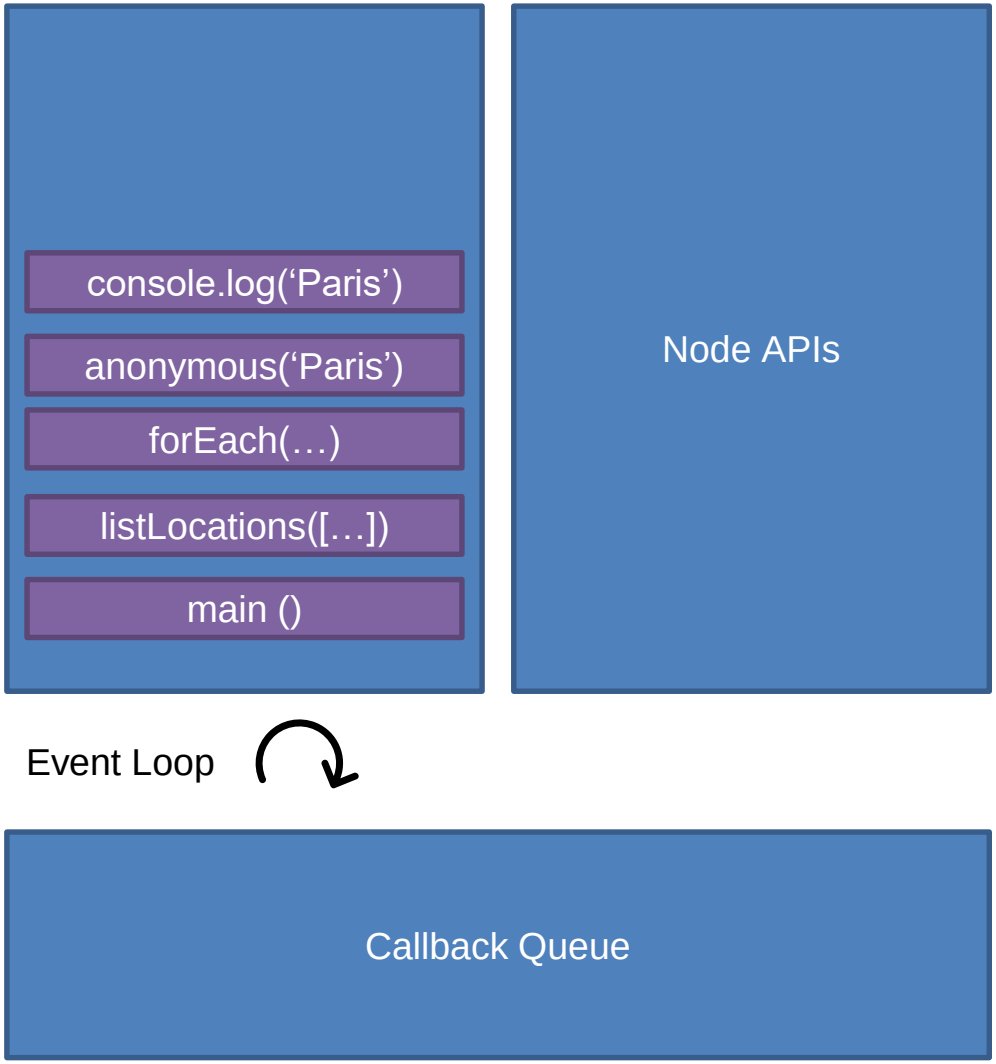
```
const listLocations = (locations) => {
    locations.forEach((location) => {
        console.log(location);
    });
};

const myLocations = ['Paris', 'Crete'];

listLocations(myLocations);
```

Paris

Crete



# Exemple code asynchrone

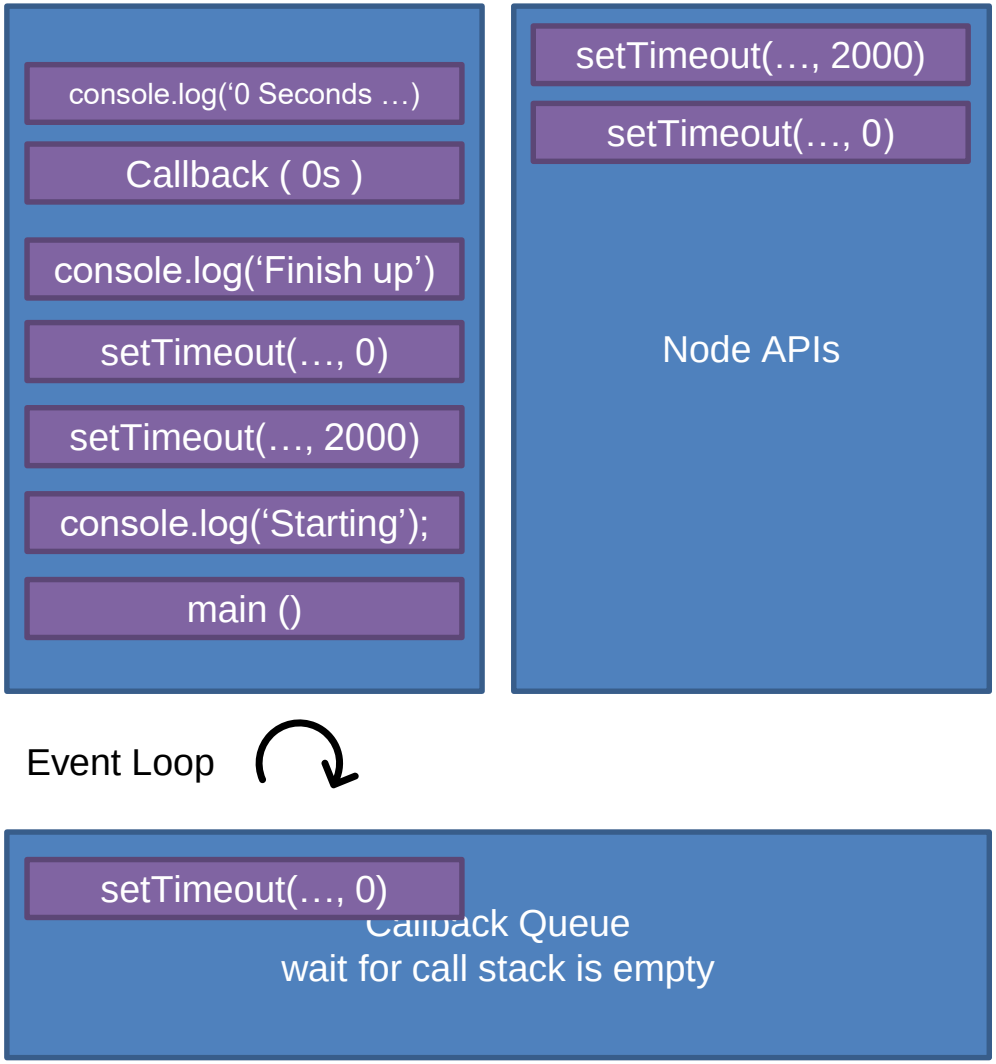
```
console.log('Starting');

setTimeout(() => {
    console.log('2 seconds Timer');
}, 2000);

setTimeout(() => {
    console.log('0 seconds Timer');
}, 0);


console.log('Stop');
```

```
PS C:\Users\charl\projects\learn_NODEJS\
Starting
Stop
0 seconds Timer
2 seconds Timer
```



# Requête asynchrone avec Node.js

```
const request = require('request');

const url = 'https://api.darksky.net/forecast/8bdb6ef6cf884b1b7d10

request({ url: url}, (error, response) => {
    const data = JSON.parse(response.body);
    console.log(data);
} );
```

```
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
{ latitude: 37.8267,
  longitude: -122.4233,
  timezone: 'America/Los_Angeles',
  currently:
   { time: 1554705999,
     summary: 'Mostly Cloudy',
     icon: 'partly-cloudy-night',
     nearestStormDistance: 0,
     precipIntensity: 0.002,
     precipIntensityError: 0,
     precipProbability: 0.45,
     precipType: 'rain',
     temperature: 58.1,
```

- Nous allons utiliser le packet request pour exécuter nos requêtes.
- Nous allons dans un premier temps utiliser une api de meteo appeler darksky.net (créer un compte).
- Nous allons dans un second temps utiliser une api de Geocode appeler mapbox.com (créer un compte).
- API:
- temperature, precipProbability
- 1. Ecrire une phrase la temperature est de … et le risque de pluie est de …

# Use of json parameter

## Super simple to use

Request is designed to be the simplest way possible to make http calls. It supports HTTPS and follows redirects by default.

```
var request = require('request');
request('http://www.google.com', function (error, response, body) {
  console.log('error:', error); // Print the error if one occurred
  console.log('statusCode:', response && response.statusCode); // Print the response status cod
  console.log('body:', body); // Print the HTML for the Google homepage.
});
```

## Table of contents



Request also offers convenience methods like `request.defaults` and `request.post`, and there are lots of usage examples and several debugging techniques.

- `qsStringifyOptions` - object containing options to pass to the qs.stringify method. Alternatively pass options to the querystring.stringify method using this format `{sep:';', eq:':', options:{}}`. For example, to change the way arrays are converted to query strings using the `qs` module pass the `arrayFormat` option with one of `indices|brackets|repeat`
- `useQuerystring` - if true, use `querystring` to stringify and parse querystrings, otherwise use `qs` (default: `false`). Set this option to `true` if you need arrays to be serialized as `foo=bar&foo=baz` instead of the default `foo[0]=bar&foo[1]=baz`.

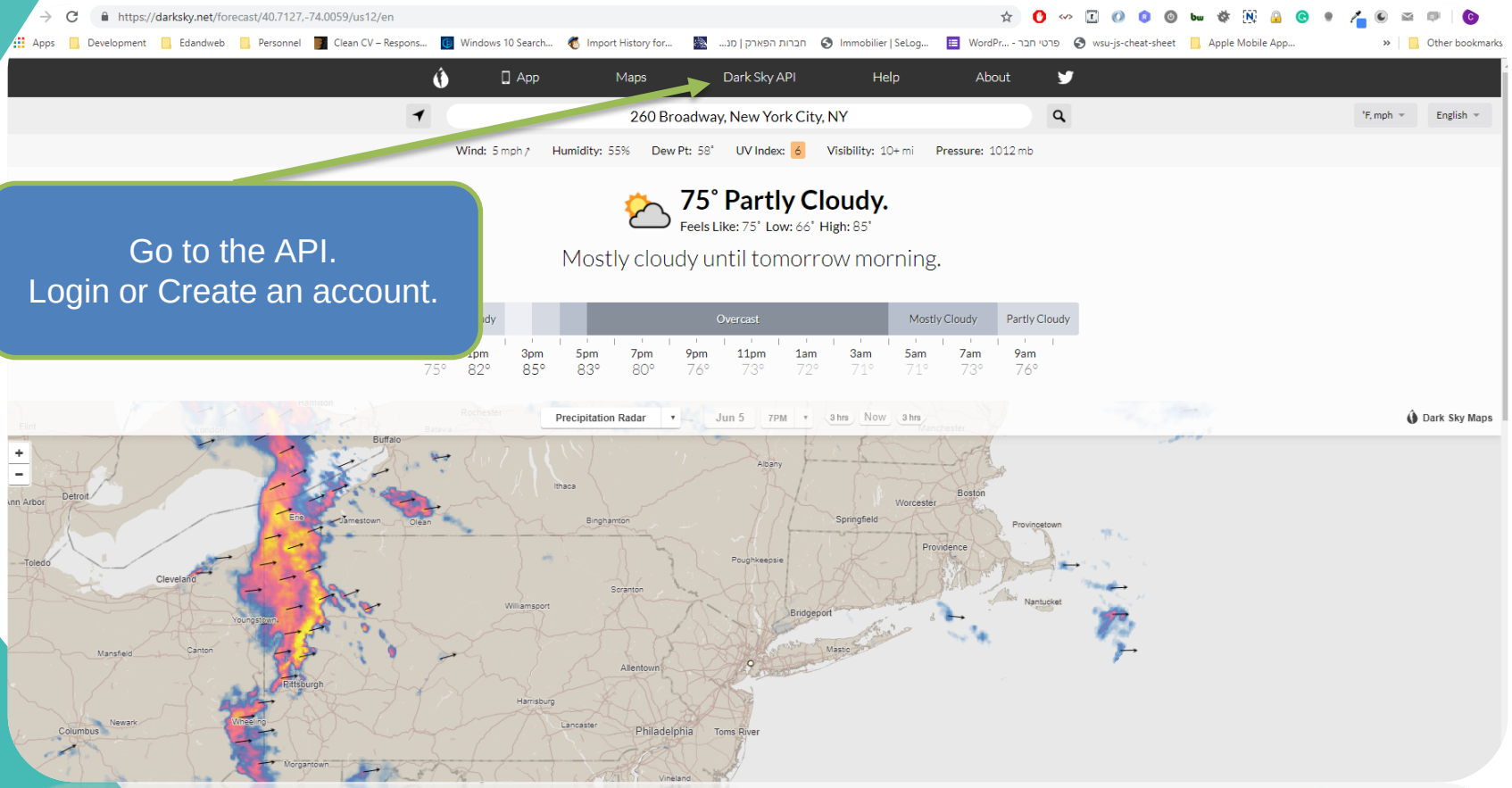
---

- `body` - entity body for PATCH, POST and PUT requests. Must be a `Buffer`, `String` or `ReadStream`. If `json` is `true`, then `body` must be a JSON-serializable object.
- `form` - when passed an object or a querystring, this sets `body` to a querystring representation of value, and adds `Content-type: application/x-www-form-urlencoded` header. When passed no options, a `FormData` instance is returned (and is piped to request). See "Forms" section above.
- `formData` - data to pass for a `multipart/form-data` request. See Forms section above.
- `multipart` - array of objects which contain their own headers and `body` attributes. Sends a `multipart/related` request. See Forms section above.
  - Alternatively you can pass in an object `{chunked: false, data: []}` where `chunked` is used to specify whether the request is sent in chunked transfer encoding In non-chunked requests, data items with body streams are not allowed.
- `preambleCRLF` - append a newline/CRLF before the boundary of your `multipart/form-data` request.
- `postambleCRLF` - append a newline/CRLF at the end of the boundary of your `multipart/form-data` request.
- `json` - sets `body` to JSON representation of value and adds `Content-type: application/json` header. Additionally, parses the response body as JSON.
- `jsonReviver` - a reviver function that will be passed to `JSON.parse()` when parsing a JSON response body.
- `jsonReplacer` - a replacer function that will be passed to `JSON.stringify()` when stringifying a JSON request body.

---

- `auth` - a hash containing values `user` || `username`, `pass` || `password`, and `sendImmediately` (optional). See documentation above.
- `oauth` - options for OAuth HMAC-SHA1 signing. See documentation above.
- `hawk` - options for Hawk signing. The `credentials` key must contain the necessary signing info, see hawk docs for details.
- `aws` - `object` containing AWS signing information. Should have the properties `key`, `secret`, and optionally `session` (note that this only works for services that require session as part of the canonical string). Also requires the property `bucket`, unless you're specifying your `bucket` as part of the path, or the request doesn't use a bucket (i.e. GET Services). If you want to use AWS sign version 4 use the parameter `sign_version` with value `4` otherwise the default is

# darksky.net




Go to the API.
Login or Create an account.

# Darksky.net

## *requête GET sur l'url d'exemple*

```
const request = require('request');

const url = 'https://api.darksky.net/forecast/8bdb6ef6cf884b1b7d10

request({ url: url, json: true}, (error, response) => {
    console.log(response.body.currently);
} );
```

```
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
{ time: 1554706206,
  summary: 'Mostly Cloudy',
  icon: 'partly-cloudy-night',
  nearestStormDistance: 2,
  nearestStormBearing: 207,
  precipIntensity: 0,
  precipProbability: 0,
  temperature: 58.09,
  apparentTemperature: 58.09,
  dewPoint: 54.65,
  humidity: 0.88,
  pressure: 1024.58,
  windSpeed: 2.6,
  windGust: 3.98,
  windBearing: 231,
  cloudCover: 0.65,
  uvIndex: 0,
  visibility: 7.48,
  ozone: 287.81 }
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
```

Une fois connecté, vous aurez accès à votre compte.

Darksky.net vous donnera une clé secrète que vous utiliserez dans vos appels d'API.

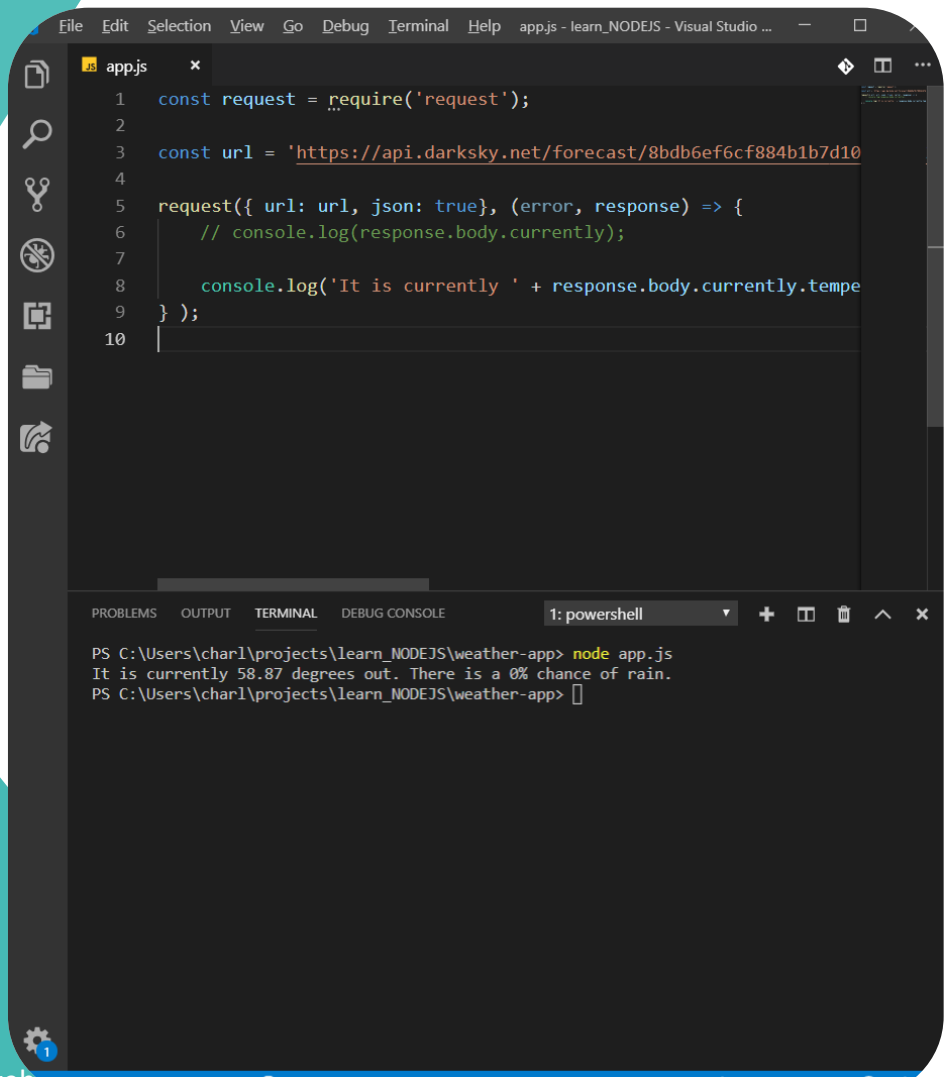
Utilisons le packet request pour faire notre première requête.

Nous utiliserons le paramètre json à true.

Nous travaillerons avec le nœud currently de l'API de darksky.net.

# Darksky.net

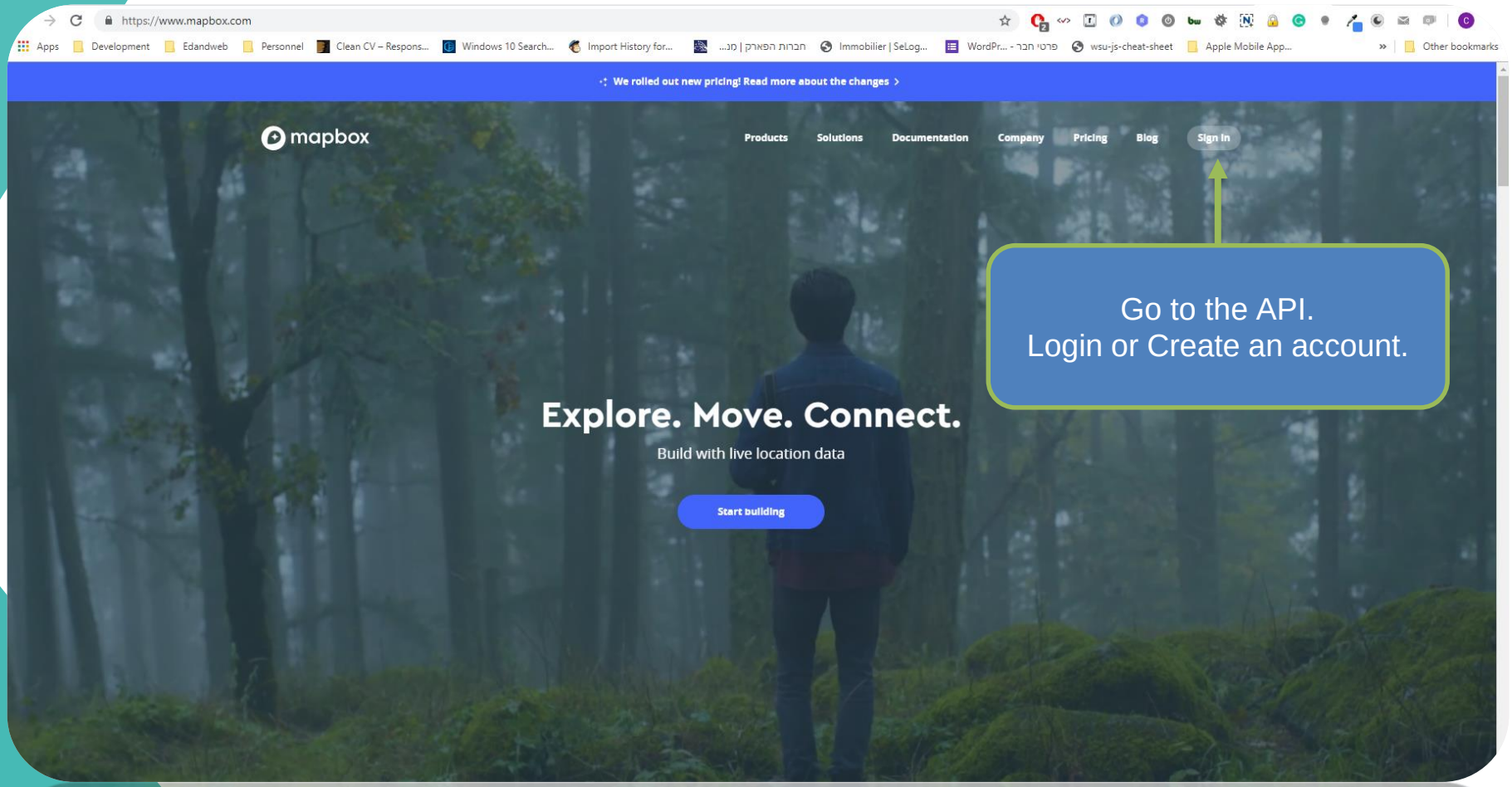
## *Afficher une phrase en exploitant les données de l'API*




Ecrivez la phrase suivante en utilisant les clés temperature et precipProbability:

Ecrire une phrase la temperature est de … et le risque de pluie est de …

# Mapbox.com


https://www.mapbox.com
We rolled out new pricing! Read more about the changes >
mapbox
Products
Solutions
Documentation
Company
Pricing
Blog
Sign In
Explore. Move. Connect.
Build with live location data
Start building
Go to the API.
Login or Create an account.

# Mapbox.com

## *Dashboard*

https://account.mapbox.com

We rolled out new pricing so your invoice may look a little different. Read more about the changes >

mapbox | Account

Dashboard Tokens Statistics Invoices Settings

Go to the API documentation

Welcome, edandweb!

Start by designing a map >

Or install the Maps SDK:

- Web
- iOS
- Android
- Unity

Plan View billing

Pay-as-you-go

Upgrade to new API pricing

Current billing cycle usage

*No usage information to display.*

Learn more about our new product offerings and how you can get started for free >

Tools & resources

- Integrate Mapbox
- Design in Mapbox Studio
- Documentation

Access tokens + Create a token

You need an API access token to configure Mapbox GL JS, Mobile, and Mapbox web services like routing and geocoding. Read more about API access tokens in our documentation.

# Mapbox.com

## API Documentation

mapbox | Docs

Maps Navigation Search All docs Help

API Documentation

Introduction
- Reading this documentation
- Access tokens and token scopes
- API versioning
- Rate limit headers
- Rate limits
- HTTPS and CORS
- Coordinate format
- Date and time format
- Pagination
- High DPI images
- SDK and library support

Maps service

Navigation service

Search service

Accounts service

Mapbox API changelog

# Introduction

The Mapbox web services APIs allow you to programmatically access Mapbox tools and services. You can use these APIs to retrieve information about your account, upload and change resources, use core Mapbox tools, and more.

Mapbox APIs are divided into four distinct *services*: **Maps**, **Directions**, **Geocoding**, and **Accounts**. Each of these services has its own overview page in this documentation. These overview pages are divided into the individual *APIs* that make up the service. The documentation for each API is structured by *endpoints*. An endpoint is a specific method within an API that performs one action and is located at a specific URL.

| | |
|---|---|
| **Maps service** | Overview of the Mapbox Maps service APIs. → |
| **Navigation service** | Overview of the Mapbox Navigation service APIs. → |
| **Search service** | Overview of the Mapbox Search service APIs. → |
| **Accounts service** | Overview of the Mapbox Accounts service APIs. → |

The Mapbox APIs described in this documentation are subject to Mapbox's Terms of Service.

We'll use the Search service

## Reading this documentation

Each API endpoint in this documentation is described using several parts:

- **The HTTP method.** Includes `GET`, `POST`, `PUT`, `PATCH`, `DELETE`.
- **The base path.** All URLs referenced in the documentation have the base path `https://api.mapbox.com`. This base path goes before the endpoint path.

API Documentation



# Search service

The Mapbox Search service is composed of the following APIs:

- Geocoding API

## Geocoding

The Mapbox Geocoding API does two things: *forward geocoding* and *reverse geocoding*.

Forward geocoding converts location text into geographic coordinates, turning `2 Lincoln Memorial Circle NW` into `-77.050,38.889`.

Reverse geocoding turns geographic coordinates into place names, turning `-77.050, 38.889` into `2 Lincoln Memorial Circle NW`. These location names can vary in specificity, from individual addresses to states and countries that contain the given coordinates.

For more background information on the Mapbox Geocoding API and how it works, see the How geocoding works guide.

### Endpoints

The geocoding API includes two different endpoints: `mapbox.places` and `mapbox.places-permanent`.

#### mapbox.places

The `mapbox-places` endpoint is accessible to all geocoding customers. Requests to this endpoint must be triggered by user activity. Any results must be displayed on a Mapbox map and cannot be stored permanently, as described in Mapbox's terms of service.

API Documentation



# Forward geocoding

GET /geocoding/v5/{endpoint}/{search_text}.json

The **forward geocoding** query type allows you to look up a single location by name and returns its geographic coordinates.

Try forward geocoding in the Search Playground.

| Required parameters | Description |
|---|---|
| `endpoint` | One of `mapbox.places` or `mapbox.places-permanent`, as described in the Endpoints section. |
| `search_text` | The feature you're trying to look up. This could be an address, a point of interest name, a city name, etc. When searching for points if interest, it can also be a category name (for example, "coffee shop"). For information on categories, see the Point of interest category coverage section. The search text should be expressed as a URL-encoded UTF-8 string, and must not contain the semicolon character (either raw or URL-encoded). Your search text, once decoded, must consist of at most 20 words and numbers separated by spacing and punctuation, and at most 256 characters. |

You can further refine the results of a forward geocoding query with the following optional parameters:

| Optional parameters | Description |
|---|---|
| `autocomplete` | Specify whether to return autocomplete results (`true`, default) or not (`false`). When autocomplete is enabled, results will be included that start with the requested string, rather than just responses that match it exactly. For example, a query for `India` might return both `India` and `Indiana` with autocomplete enabled, but only `India` if it's disabled. |
| `bbox` | Limit results to only those contained within the supplied bounding box. Bounding boxes should be supplied as four numbers separated by commas, in `minLon,minLat,maxLon,maxLat` order. The bounding box cannot cross the 180th meridian. |

API Documentation



## Example request: Forward geocoding

```
# A basic forward geocoding request
# Find Los Angeles

$ curl "https://api.mapbox.com/geocoding/v5/mapbox.places/Los%20Angeles.json?
access_token=pk.eyJ1IjoiZWRhbmR3ZWIiLCJhIjoiY2p1ODB3bm1uMHo4djN5cGZ6ejJzbnh4NCJ9.2La2amfGm8YVPVwb5YkviA"

# Find a town called 'Chester' in a specific region
# Add the proximity parameter with local coordinates
# This ensures the town of Chester, New Jersey is in the results

$ curl "https://api.mapbox.com/geocoding/v5/mapbox.places/chester.json?
proximity=-74.70850,40.78375&access_token=pk.eyJ1IjoiZWRhbmR3ZWIiLCJhIjoiY2p1ODB3bm1uMHo4djN5cGZ6ejJzbnh4NCJ9.2L
```

### Endpoint support

Mapbox wrapper libraries help you integrate Mapbox APIs into your existing application. The following SDKs support this endpoint:

- Mapbox CLI SDK
- MapboxDirections.swift (Objective-C and Swift)
- Mapbox Java SDK
- Mapbox JavaScript SDK
- Mapbox Python SDK
- Mapbox React Geocoding Plugin
- Mapbox Ruby SDK

See the SDK documentation for details and examples of how to use the relevant methods to query this endpoint.

## Response: Forward geocoding

The API response for a forward geocoding query returns a GeoJSON feature collection in Mapbox Geocoding Response format. For more details on how a response from the Geocoding API is formatted, see the Geocoding API response object section.

## Reverse geocoding

File Edit Selection View Go Debug Terminal Help • app.js - learn_NODEJS - Visual Studio...

app.js

```js
const request = require('request');

const urlCoordinates = 'https://api.mapbox.com/geocoding/v5/mapbox
const url = 'https://api.darksky.net/forecast/8bdb6ef6cf884b1b7d10

request({url: urlCoordinates, json: true}, (error, response) => {
    const latitude = response.body.features[0].center[1];
    const longitude = response.body.features[0].center[0];

    console.log(latitude, longitude);

} );

// request({ url: url, json: true}, (error, response) => {
//     // console.log(response.body.currently);

//     console.log(response.body.daily.data[0].summary + ' It is c
// } );
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: powershell

```
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
31.771667 35.186166
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
```

0 Live Share Go Live Ln 10, Col 13 Spaces: 4 UTF-8 CRLF JavaScript ESLint Prettier

File Edit Selection View Go Debug Terminal Help app.js - learn_NODEJS - Visual Studio ...

app.js

```
1  const request = require('request');
2
3  const urlCoordinates = 'https://api.mapbox.com/geocoding/v5/mapbox
4  const url = 'httpss://api.darksky.net/forecast/8bdb6ef6cf884b1b7d1
5
6  // request({url: urlCoordinates, json: true}, (error, response) =>
7  //     const latitude = response.body.features[0].center[1];
8  //     const longitude = response.body.features[0].center[0];
9
10 //     console.log(latitude, longitude);
11
12 // } );
13
14 request({ url: url, json: true}, (error, response) => {
15     // console.log(response.body.currently);
16
17     if (error) {
18         console.log(error);
19         console.log('Unable to process the request');
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: powershell

```
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
31.771667 35.186166
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
Unable to process the request
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
Ciel couvert toute la journée. It is currently 14.93 degrees out. There is a 0.05% chance o
f rain.
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
Ciel couvert toute la journée. It is currently 14.93 degrees out. There is a 0.05% chance o
f rain.
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
```

0 Live Share Go Live Ln 4, Col 20 Spaces: 4 UTF-8 CRLF JavaScript ESLint Prettier

# API Geocode – longitude , latitude

Mapbox Inc. [US] | https://api.mapbox.com/geocoding/v5/mapbox.places/Los%20Angeles.json?access_token=pk.eyJ1IjoiZWRhbmR3ZWliLCJhIjoiY2p1ODB3bm1uMHo4d...

Apps | Development | Edandweb | Personnel | Clean CV – Respons... | Windows 10 Search... | Import History for... | חברות הפארק | מנ... | Immobilier | SeLog... | WordPr... - פרטי חבר | wsu-js-cheat-sheet | Apple Mobile App... | Other bookmarks


```
{
    type: "FeatureCollection",
  - query: [
        "los",
        "angeles"
    ],
  - features: [
    - {
          id: "place.739750309342764 0",
          type: "Feature",
        - place_type: [
              "place"
          ],
          relevance: 1,
        - properties: {
              wikidata: "Q65"
          },
          text: "Los Angeles",
          place_name: "Los Angeles, California, United States",
        - bbox: [
              -118.521456965901,
              33.9018913203336,
              -118.121305008073,
              34.161440999758
          ],
        - center: [
              -118.2439,
              34.0544
          ],
        - geometry: {
              type: "Point",
            - coordinates: [
                  -118.2439,
                  34.0544
              ]
          },
        - context: [
            - {
                  id: "region.11319063928738010",
                  short_code: "US-CA",
                  wikidata: "Q99",
                  text: "California"
              },
            - {
                  id: "country.9053006287256050",
                  short_code: "us",
                  wikidata: "Q30",
                  text: "United States"
              }
          ]
      },
```

EXERCICES & TP

```
add(1, 4, (sum) => {
    console.log(sum);
});
```

Ecrire une fonction add en utilisant setTimeout avec un timer de 2s pour que cette fonction puisse afficher la somme.

EXERCICES & TP

```
const add = (a, b, callback) => {
setTimeout(() => {
    callback(a + b)
}, 2000);
};

add(1, 4, (sum) => {
console.log(sum);
});
```

EXERCICES & TP

```
forecast(-75.7088, 44.1545, (error, data) =>
{
    console.log('Error', error)
    console.log('Data', data)
});
```

Ecrire une fonction forecast dans utils/forecast.js .
Gérer 2 types d'erreurs:

- Erreur low level (network …)
- Problème de coordonnées

File Edit Selection View Go Debug Terminal Help • app.js - learn_NODEJS - Visual Studio Code

app.js forecast.js geocode.js

```javascript
const request = require('request');
const geocode = require('./utils/geocode.js');
const forecast = require('./utils/forecast.js');



geocode.getCoordinates('Jerusalem', (error, data) => {
    if (error) {
        console.log('Error occured: ' + error);
    } else {
        console.log(data);
    }
});


forecast.getWeather(-75.7088, 44.1545, (error, data) => {

    if (error) {
        console.log('Error', error)
    } else {
        // console.log('Data', data)
        console.log(data.daily.data[0].summary + ' It is currently ' + data.currently.temperature + ' degrees out. There is a ' + data.currently.prec
    }
});

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 1: powershell

```
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
{ latitude: 31.771667, longitude: 35.186166 }
Brumeux toute la journée. It is currently -43.61 degrees out. There is a 0.07% chance of rain.
5
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
```

0 Live Share learn_NODEJS Go Live Ln 30, Col 1 Spaces: 4 UTF-8 CRLF JavaScript ESLint Prettier

File Edit Selection View Go Debug Terminal Help

app.js - learn_NODEJS - Visual Studio Code

app.js | forecast.js | geocode.js

```javascript
const request = require('request');
const geocode = require('./utils/geocode.js');
const forecast = require('./utils/forecast.js');



geocode.getCoordinates('Jerusalem', (error, data) => {
    if (error) {
        console.log('Error occured: ' + error);
    } else {
        forecast.getWeather(data.latitude, data.longitude, (error, data) => {

            if (error) {
                console.log('Error', error)
            } else {
                // console.log('Data', data)
                console.log(data.daily.data[0].summary + ' It is currently ' + data.currently.temperature + ' degrees out. There is a ' + data.curren
            }
        });
    }
});
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

1: powershell

```
PS C:\Users\charl\projects\learn_NODEJS\weather-app> node app.js
Ciel dégagé toute la journée. It is currently 23.12 degrees out. There is a 0% chance of rain.
PS C:\Users\charl\projects\learn_NODEJS\weather-app>
```

0 Live Share learn_NODEJS Go Live Ln 11, Col 58 Spaces: 4 UTF-8 CRLF JavaScript ESLint Prettier

# Résumé