

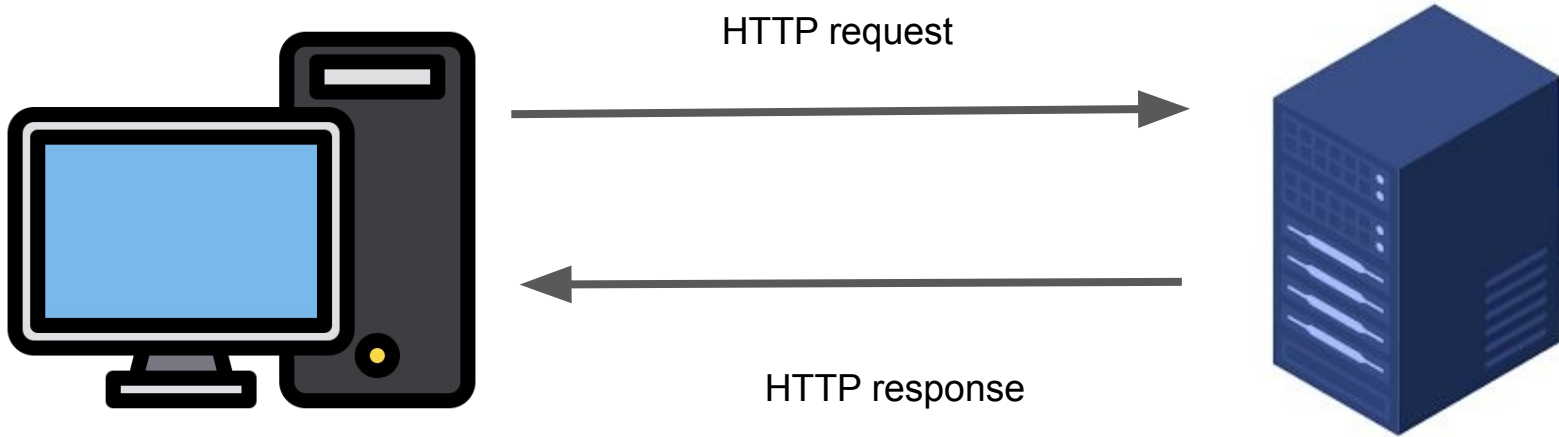
micro-services

Web Servers

A **web server** is [computer software](#) and underlying [hardware](#) that accepts requests via [HTTP](#) (the [network protocol](#) created to distribute [web content](#)) or its secure variant [HTTPS](#). A user agent, commonly a [web browser](#) or [web crawler](#), initiates communication by making a request for a [web page](#) or other [resource](#) using HTTP, and the [server](#) responds with the content of that resource or an [error message](#). A web server can also accept and store resources sent from the user agent if configured to do so.^{[1] [2]}

Un **serveur web** est, soit un [logiciel](#) de service de ressources web (serveur HTTP), soit un [serveur informatique](#) ([ordinateur](#)) qui répond à des requêtes du [World Wide Web](#) sur un réseau public ([Internet](#)) ou privé ([intranet](#))^{1,2,3}, en utilisant principalement le [protocole HTTP](#).

Webserver





NGINX



LITESPEED



Microsoft
IIS



THE
APACHE[™]
SOFTWARE FOUNDATION



CLOUDFLARE[®]



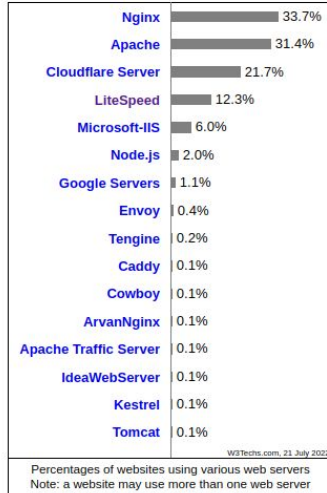
gunicorn



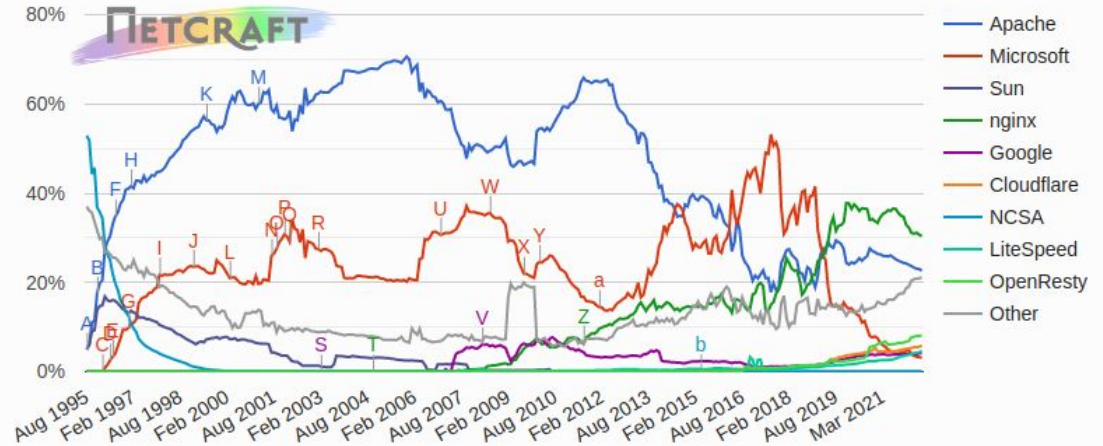
Apache Tomcat

Server by popularity

Percentages of websites using various web servers



Web server developers: Market share of all sites



Web page composition

HTML



JavaScript



CSS



HTML, it's XML...

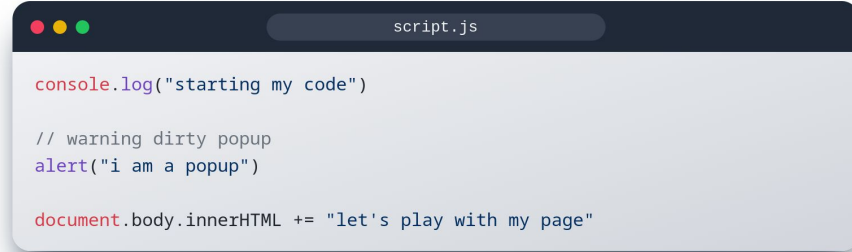


```
<!DOCTYPE html>
<html>
<body>

<h1>Title</h1>
<p>first content</p>
<div>First Section</div>

</body>
</html>
```

JS : javascript, it's code



```
console.log("starting my code")

// warning dirty popup
alert("i am a popup")

document.body.innerHTML += "let's play with my page"
```


CSS, how to make things pretty ?



Combining all together

```
index.html

<!DOCTYPE html>
<html>
<head>
<style>
  body {
    background-color: lightblue;
  }
  h1 {
    color: white;
    text-align: center;
  }

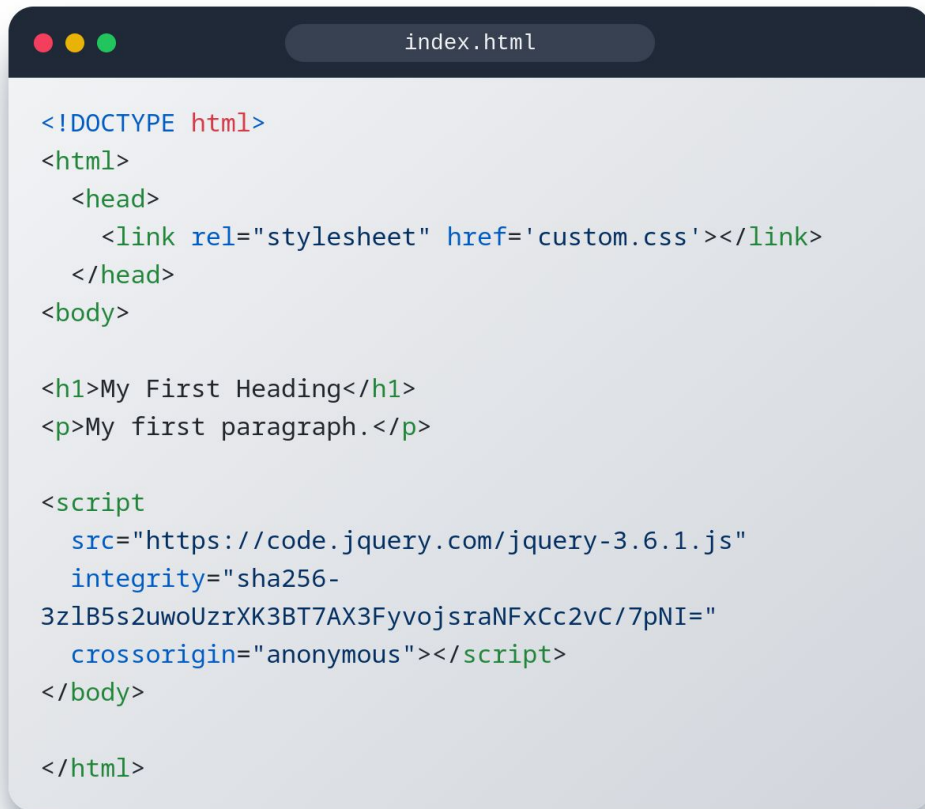
  p {
    font-family: verdana;
    font-size: 20px;
  }
</style>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

<script>
console.log("start")
alert("ok")
document.body.innerHTML += "coucou"
</script>

</body>
</html>
```

better seperate things



```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href='custom.css'></link>
  </head>
  <body>

    <h1>My First Heading</h1>
    <p>My first paragraph.</p>

    <script
      src="https://code.jquery.com/jquery-3.6.1.js"
      integrity="sha256-
3zlB5s2uwoUzrXK3BT7AX3FyvojsraNFxCc2vC/7pNI="
      crossorigin="anonymous"></script>
  </body>

</html>
```

Static vs API

Static file : cachable, same for everyone

API : Application Programming interface = code = specific functionality

a first server

```
while true; do (echo -e 'HTTP/1.1 200 OK\r\n'; echo -e "\t$(date)\n") | nc -Nlp 8080; done
```



NodeJS



our first real server

1. install nodejs

```
sudo apt-get install nodejs
```

2. vérification

```
node -v
```

3. création d'un projet

```
mkdir myproject
```

```
npm init
```

Before a server, a script

```
// script.js  
console.log("Hello World")  
  
node script.js
```


Nodejs : getting started

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

const port = 3000;

const server = http.createServer((req, res) => {

  res.statusCode = 200;

  res.setHeader('Content-Type', 'text/plain');

  res.end('Hello World');

});

server.listen(port, () => {

  console.log(`Server running at http://localhost:${port}/`);

});
```

Nodejs Express

Express

Infrastructure Web
minimaliste, souple et
rapide pour [Node.js](#)

- static file server
- templating
- session management
-

nodejs & Express

```
const express = require('express')

const app = express()

const port = 3000

app.get('/', (req, res) => {

  res.send('Hello World!')

})

app.listen(port, () => {

  console.log(`Example app listening on port ${port}`)

})
```

static files are important ! simple and cachable

favicon.ico

CSS

JS

HTML simple

image

video

....

handeling static file

```
app.get('/file', (req, res) => {  
  fs.readFile('./www/index.html', function read(err, data) {  
    res.writeHead(200, { "Content-Type": "text/html" });  
    res.write(data);  
    res.end();  
  });  
})
```



Express : static server

```
const express = require('express')

const app = express()

const port = 3000

app.use(express.static('www'));

app.get('/', (req, res) => {
  res.send('Hello World!')
})

app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

callback js



```
function salutation(name) {  
    alert('Bonjour ' + name);  
}  
  
function processUserInput(callback) {  
    var name = prompt('Entrez votre nom.');
```

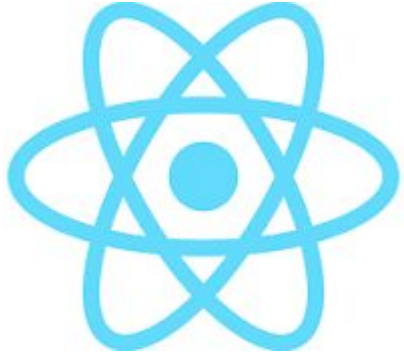
The code editor window has a dark blue title bar with three colored window control buttons (red, yellow, green) on the left and the filename 'index.js' in the center. The code is displayed in a light gray area with syntax highlighting: keywords are red, function names and variables are purple, and strings are blue.

```
    callback(name);  
}  
  
processUserInput(salutation);
```

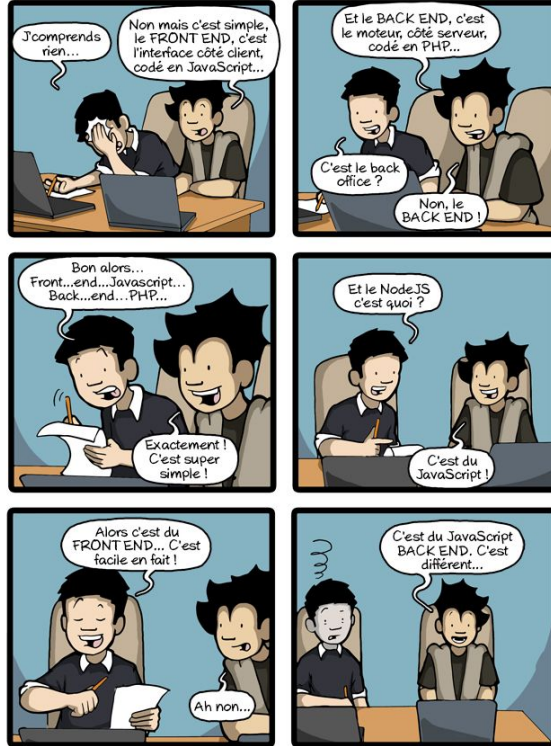
some front end ?



Front end Frameworks

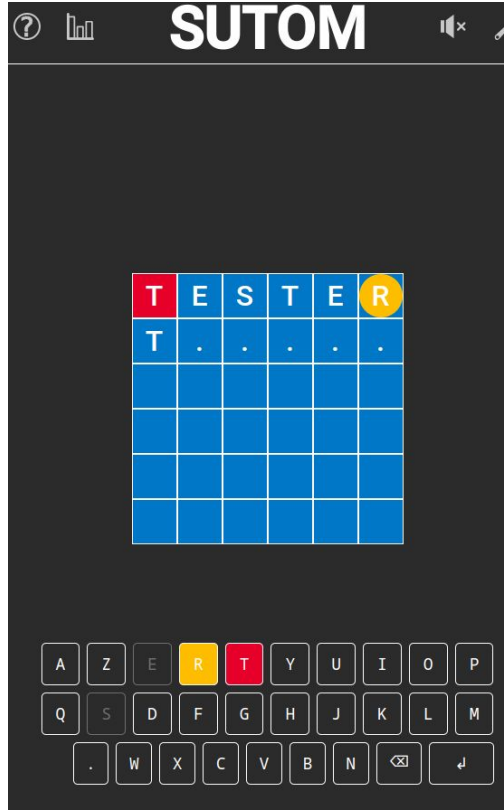


JS is everywhere



TP : MOTUS

<https://sutom.nocle.fr/>



Jquery & miligram



Let's code

https://simongomezuniv.github.io/td2_NODE