INITIATION À NODE.JS

LE CONCURRENT De PHP/APACHE

Lycée Louise Michel

OBJECTIFS

CONNAISSANCES ET COMPÉTENCES

- **CONNAISSANCES**
  - Connaître les notions spécifiques à node.JS
  - Connaître le fonctionnement des API
  - Découvrir un environnement de travail Low Code (Node-Red)

- **COMPÉTENCES**
  - B1.5.1 Déployer un service
  - B2.1.3 Utiliser des composants d'accès aux données
  - B2.1.7 Exploiter les technologies Web et mobile pour mettre en œuvre les échanges entre applications, y compris de mobilité

INTRODUCTION

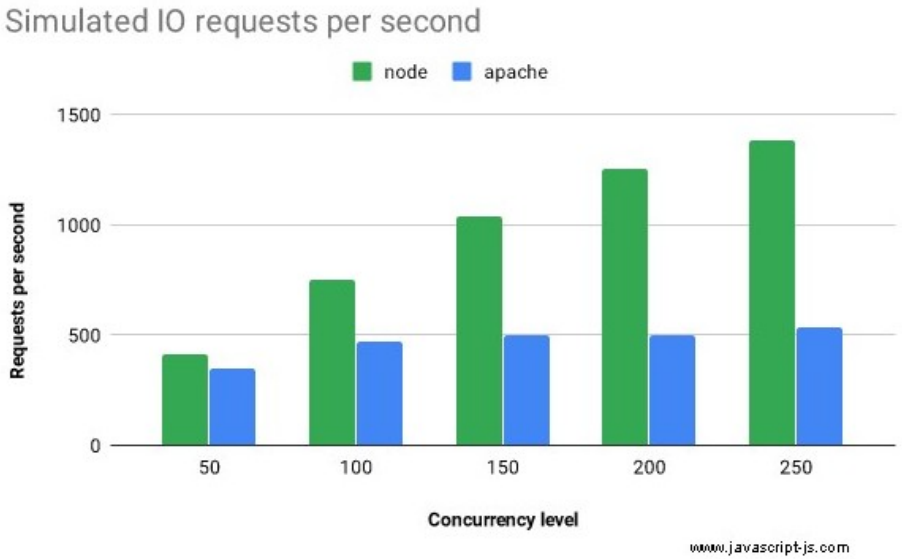
- **QUE SAVEZ-VOUS DE NODE.JS ?**

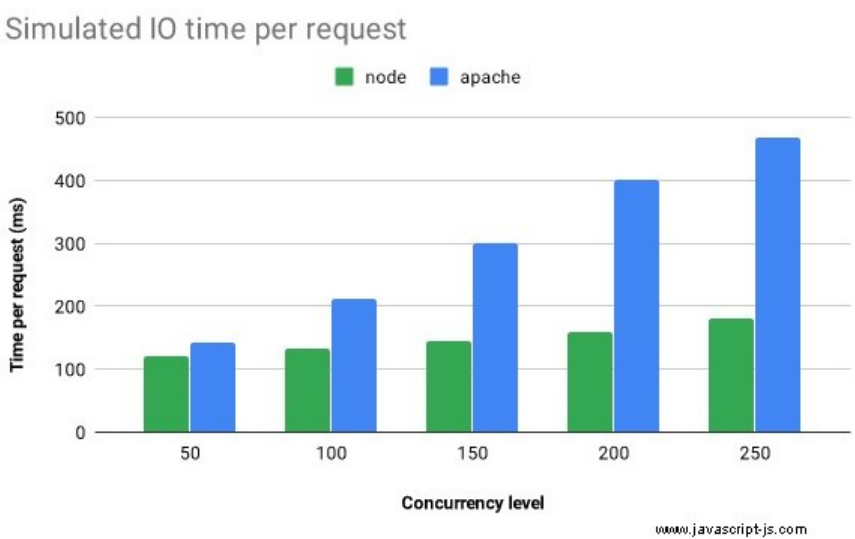
- **QUE SAVEZ-VOUS DE NODE.JS ?**
  - NODE.JS EST JAVASCRIPT
    - JavaScript est principalement un langage côté navigateur depuis 1995
    - JavaScript ressemble à la syntaxe de Java, mais est plus simple
    - JavaScript est inventé en 1995 par Brendan Eich, qu'il conçoit en 10 jours
  - NODE.JS EST CÔTÉ SERVEUR
    - Netscape Enterprise tente un système s'exécutant sous Java : Rhino (1997)
    - Node.JS contient le Runtime, la machine virtuelle JavaScript et des API
    - Il est l'œuvre de Ryan DAHL en 2009, lorsqu'il découvre JavaScript

NODE.JS : POURQUOI JAVASCRIPT

- LENTEUR DE RUBY ON RAIL
- COMPLEXITÉ DE C
- BLOCAGES AVEC PYTHON, LUA, HASKELL
- JAVASCRIPT ?
  - Contient déjà des fonctions **non bloquantes**
  - Gère l'**asynchronisme**
  - Gère l'**enclosure*** et dispose d'une **boucle d'évènement**
  - Première démo de NodeJS avec une **barre de progression** de téléchargement !

- **NODE.JS : POURQUOI JAVASCRIPT**
  - PERFORMANCES

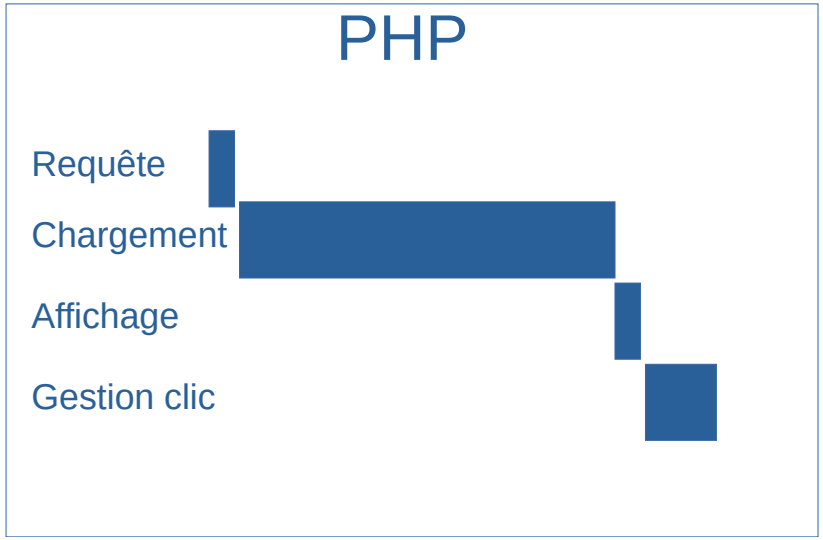


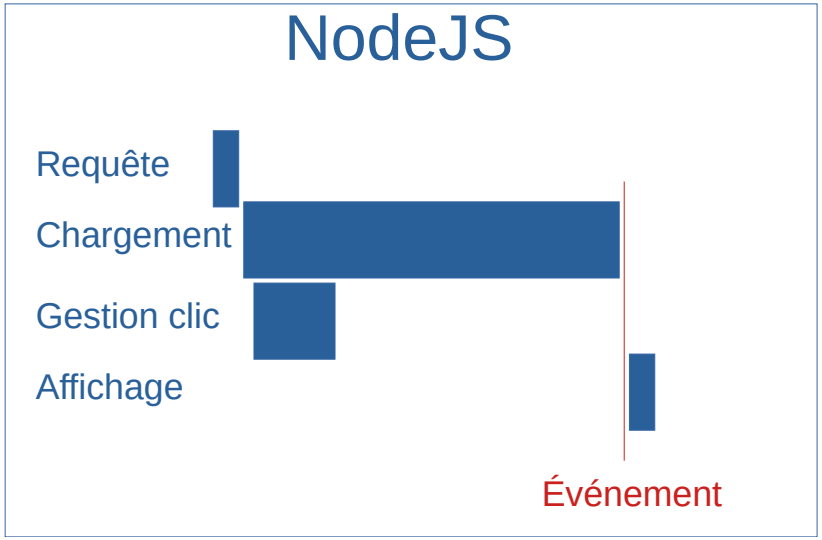


https://javascript.19633.com/fr/Node-10/1010054605.html

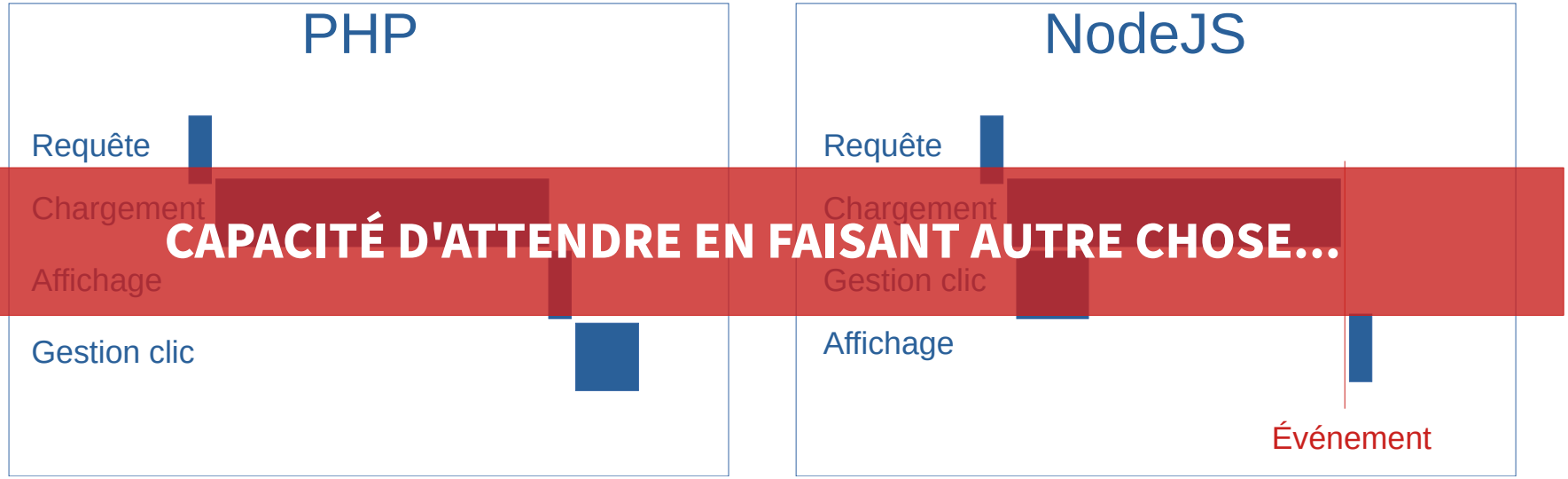
FONCTIONNEMENT

- **FONCTIONNEMENT**
  - ASYNCHRONISME

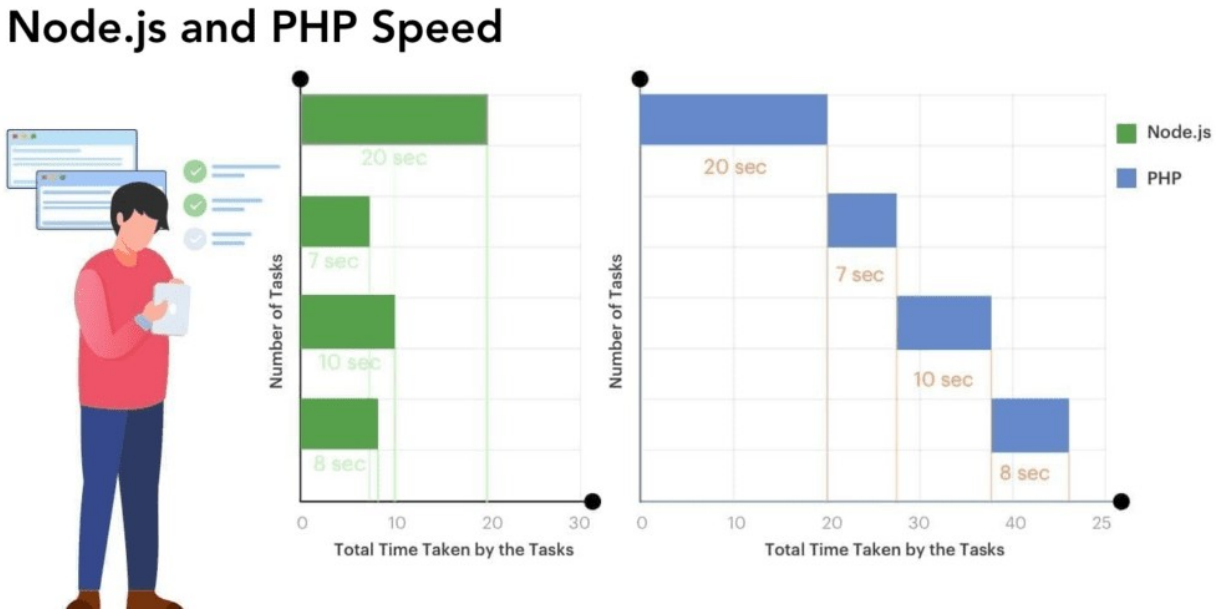




- **FONCTIONNEMENT**
  - ASYNCHRONISME



- **FONCTIONNEMENT**
  - ASYNCHRONISME



https://externlabs.com/blogs/node-js-vs-php/

- **FONCTIONNEMENT**
  - CLOSURE

```
function createCounter() {
    let count = 0;
    return function increment() {
        count++;
        console.log(count);
    };
}

const counter = createCounter();
counter(); // Affiche 1
counter(); // Affiche 2
```

CLOSURE

- **FONCTIONNEMENT**
  - CLOSURE

```
function createCounter() {
    let count = 0;
    return function increment() {
        count++;
        console.log(count);
    };
}

const counter = createCounter();
counter(); // Affiche 1
counter(); // Affiche 2
```

CLOSURE

CAPACITÉ DE CONSERVATION D'UN ENVIRONNEMENT

- **FONCTIONNEMENT**
  - CLOSURE

    https://codepen.io/prRoumanet/pen/JodjLYP

ÉCOSYSTÈME

ÉCOSYSTÈME

Le gestionnaire de paquets NPM

Node Package Manager

- npm **install**, npm **version**, npm **update**, npm **run**, npm **init**

Fichier package.json

- Contient la liste des paquets utiles à un projet (spécifications)
- La commande **npm install** installe automatiquement les dépendances (paquets) nécessaires au projet
- Le fichier **package-lock.json** contient les paquets réellement installés

- **LE GESTIONNAIRE DE PAQUETS NPM**

Lycée Louise Michel

  - FICHIER **PACKAGE.JSON** VERSUS **PACKAGE-LOCK.JSON**

```json
{
  "dependencies": {
    "express": "^4.17.1",
    "body-parser": "^1.19.0"
  }
}
```

Je veux express en version supérieure ou égale à 4.17.1

```json
{
  "dependencies": {
    "express": {
      "version": "4.17.1",
      "resolved": "https://registry.npmjs.org/express/-/express-4.17.1.tgz",
      "integrity": "...",
      "requires": {
        "body-parser": "1.19.0"
      }
    },
    "body-parser": {
      "version": "1.19.0",
```

Voici les versions que j'ai trouvé qui correspondent à tes attentes

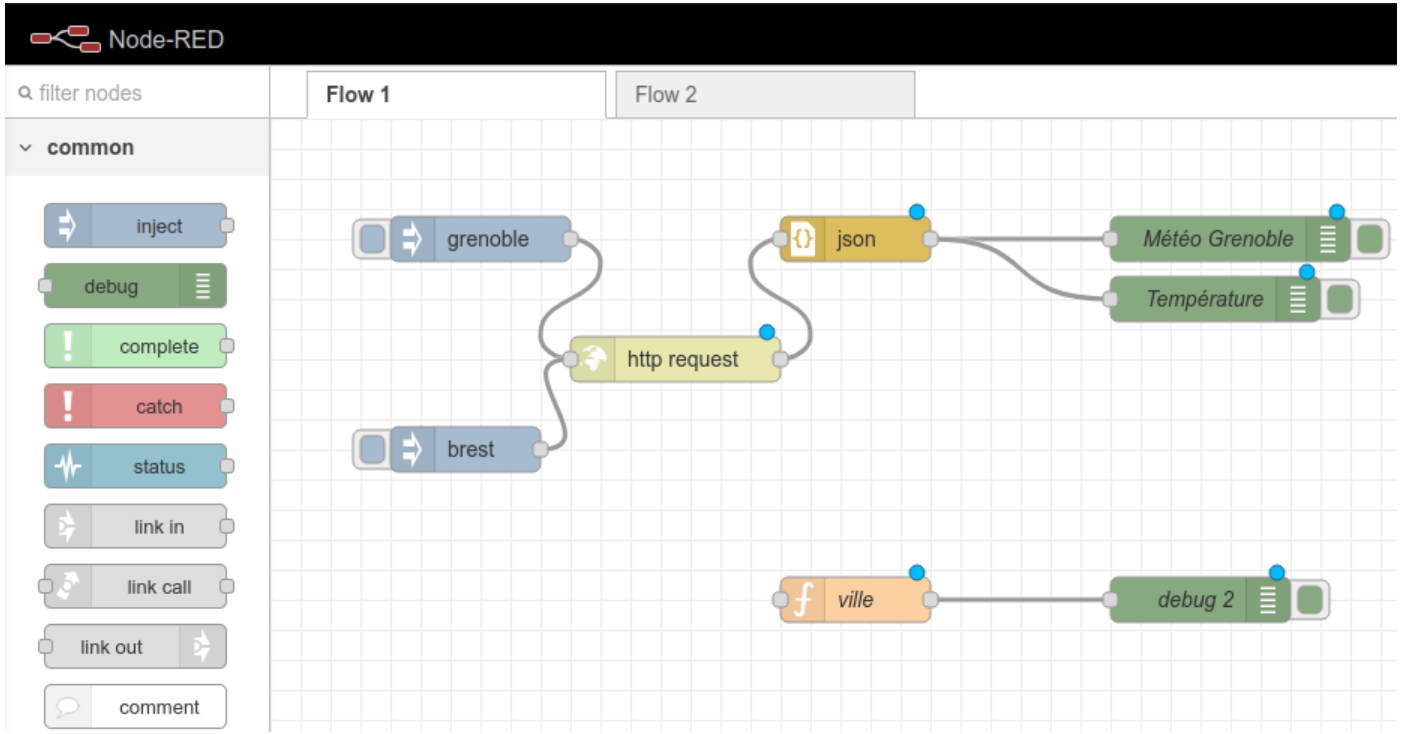
ÉCOSYSTÈME

Les autres gestionnaires de paquets

- Yarn
  - créé par Facebook en 2017 pour compenser les lenteurs de npm de l'époque
  - Yet Another
  - npm install -g yarn
- PNPM
  - Performant NPM (à l'époque, npm était vraiment lent)
- BUN
  - Projet entièrement nouveau, remplaçant NPM, Node.JS et Webpack (regroupeur de modules)

- **LE FRAMEWORK NODE-RED**
  - ORIGINE DU PROJET
    - Inventé par IBM en 2013
    - Possibilité de coder "graphiquement"
    - Objectif : faciliter la gestion des systèmes IoT et le prototypage d'applications
    - Gestion de plusieurs protocoles nativement (MQTT, HTTP, websocket...)

- **LE FRAMEWORK NODE-RED**
  - ASPECT



LES API AVEC NODEJS

USAGE DES SERVICES WEB 3.0

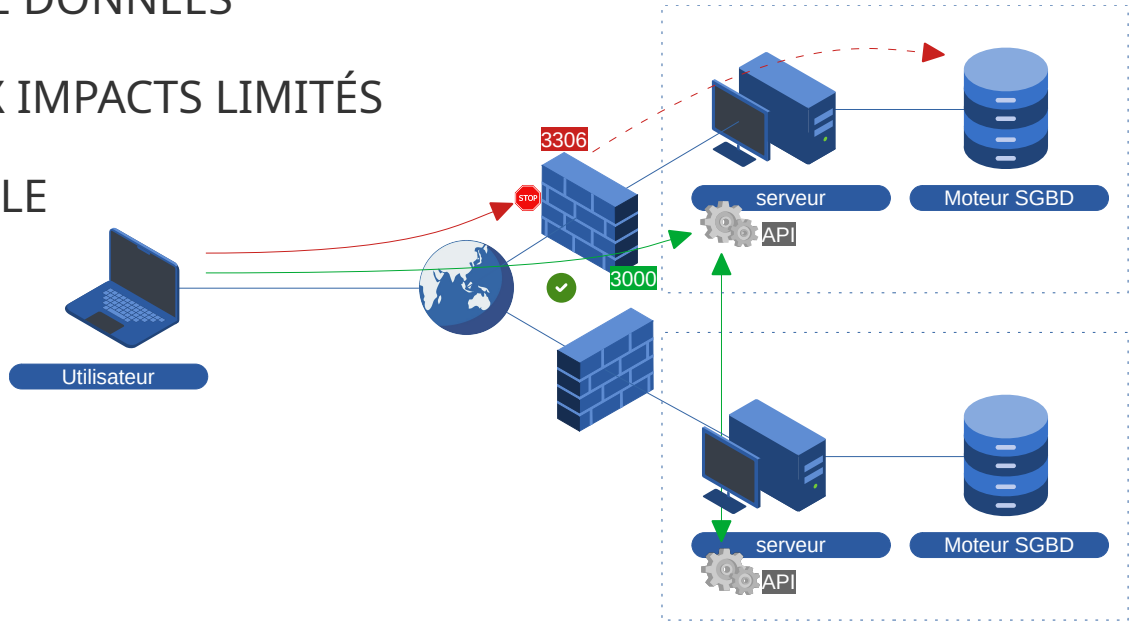
Lycée Louise Michel

HISTORIQUE DES API

- ## Application Programming Interface

  - Une API, pour **Application programming interface**, est un programme permettant à deux applications distinctes de communiquer entre elles et d'échanger des données. Cela évite notamment de recréer et redévelopper entièrement une application pour y ajouter ses informations.

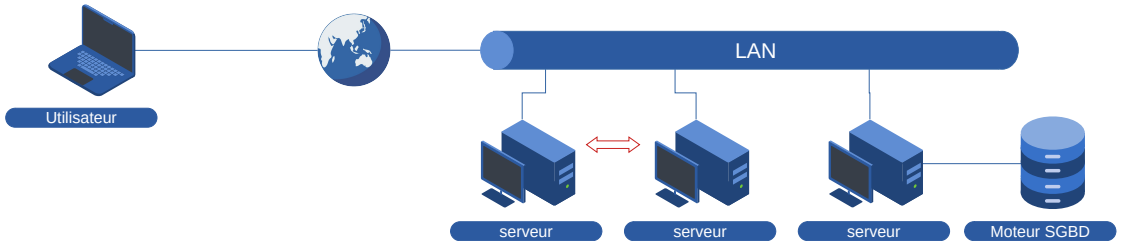
- **POURQUOI UNE API**
  - SÉCURITÉ DES BASES DE DONNÉES
  - SERVICE PUBLIQUE AUX IMPACTS LIMITÉS
  - ADMINISTRATION SIMPLE



CORBA, DCOM… LES PREMIÈRES API

DE L'INFORMATIQUE PARTAGÉE À L'INFORMATIQUE DISTRIBUÉE

- Trouver un moyen de communication entre application
- S'affranchir du média (au sein du même ordinateur ou via le réseau)
- Standardiser les échanges de données



- **CORBA, DCOM… LES PREMIÈRES API**
  - DE L'INFORMATIQUE PARTAGÉE À L'INFORMATIQUE DISTRIBUÉE
    - Complexe à mettre en œuvre
    - Coûts de licences (IBM)
    - DCOM architecture propriétaire de Microsoft

- **SOAP… LA STANDARDISATION**

  - DE L'INFORMATIQUE DISTRIBUÉE INTERNATIONALE

    - Utilisation de XML pour le format des données (format ouvert)
    - Intégration d'éléments de sécurité (WS-Security, ACID compliance...)
    - Très apprécié du milieu bancaire
    - Serverless (requêtes indépendantes)
    - Cependant…
    - Courbe d'apprentissage plus difficile (très normalisé)
    - Messages très verbeux
    - Évolutivité plus complexe impliquant de modifier le WSDL (Web Service Description Language)

- **SOAP… LA STANDARDISATION**

  - EXEMPLE SOAP

```
client = Client('https://www.example.com/exampleapi')
result = client.service.GetUser(123) # request user with ID 123
```

  - FORMAT DE LA REQUÊTE

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="https://www.w3.org/2003/05/soap-envelope">
  <soap:Header>
  </soap:Header>
  <soap:Body>
    <m:GetUser>
      <m:UserId>123</m:UserId>
    </m:GetUser>
  </soap:Body>
</soap:Envelope>
```

- **REST… SIMPLIFICATION**

  - LA MATURITÉ DES ÉCHANGES

    - Utilisation de JSON (mais accepte XML ou texte)
    - Utilise les sécurités HTTP
    - Protocole **stateless** [sans état] : la réponse se suffit à elle-même
    - Quatre méthodes (citez-les ?)
    - Basé sur les ressources plutôt qu'un catalogue (comme WSDL)
    - REST est plus flexible et évolutif

- **REST… SIMPLIFICATION**

  - EXEMPLE REST

    ```
    client = Client('https://www.example.com/exampleapi/users/23')
    ```

  - FORMAT DE LA REQUÊTE

    ```
    GET https://www.example.com/exampleapi/users/23
    ```

FONCTIONNEMENT REST

API REST

- **REST**
  - Association du CRUD des bases de données aux requêtes HTTP
    - **GET** = READ
    - **POST** = CREATE ou UPDATE
    - **DELETE** = DELETE
    - **PUT** ou **PATCH** = UPDATE
  - État des requêtes
    - 20x ✅
    - 30x
    - 40x ❌

API REST

- **REST**
  - REQUÊTES
    - On travaille sur des ressources
    - GET http://localhost/chambres renverra la liste des chambres (voir resaHotelCalifornia)
      - ⚠️ Les formulaires web (HTML) ne peuvent gérer que GET et POST
      - 💡 Pour utiliser PUT/PATCH et DELETE, on utilise AJAX ou la commande fetch de JavaScript
    - POST http://localhost/chambres créera une chambre et renverra le code 201 (Created)
    - Bonnes pratiques :
      - Les URL auront une forme /api/v1/chambres pour désigner une API REST en version 1
      - Un changement de version s'écrit donc /api/v2/chambres
      - Usage de Token JWT pour les requêtes nécessitant une authentification

- **REST**
  - EXEMPLE DE REQUÊTE
    - POST http://localhost/chambres

```
// POST /api/chambres
fetch('https://api.hotel.com/api/chambres', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify({
    numero: 101,
    capacite: 2,
    prix: 120.00,
    categorie: "Standard"
  })
});
```

API REST

- **REST**
  - Codes HTTP

    https://fr.wikipedia.org/wiki/Liste_des_codes_HTTP

C'EST À VOUS !

Lycée Louise Michel

ACTIVITÉ

- **INSTALLATION DE NODE.JS**
  - ACTIVITÉ
    - http://david.roumanet.free.fr/BTS-SIO/Bloc2-Dev/300%20-%20Frameworks%20JS/B2-DEV%20300%20installation%20NodeJS%20et%20NPM.pdf
- **INSTALLATION ET UTILISATION DE NODE-RED**
  - ACTIVITÉ
    - http://david.roumanet.free.fr/BTS-SIO/Bloc2-Dev/350%20Low%20Code%20(Node%20Red)/B2-Dev%20350%20d%C3%A9couverte%20Low%20Code%20avec%20Node-Red.pdf