Introduction aux API Web

Contenu

- **Concept d'API**
- **Les API de type REST**
  - REST, c'est quoi?
  - JSON, c'est quoi?
  - Exemple d'API REST
- **Tester son API REST**

Concept d'API

Concept d’API

- **API signifie *Application Programming Interface***
  - En français : Interface de programmation d’application
- **L'ensemble des classes, fonctions, etc. permettant à un programme informatique de communiquer avec une librairie ou un autre programme**
  - Ex:
    - PDO
    - API de Windows

API Web

- **Dans le contexte du Web, on parle d’une interface permettant d’échanger des données avec le backend d'une application Web**
  - soit à partir de son frontend
  - soit à partir d'une autre application
- **Exemples d'APIs Web:**
  - API Météo
  - API de Facebook
  - API de Paypal

API Web

- **Une API Web est typiquement implantée sur le protocole HTTP**

Rappel: Protocole

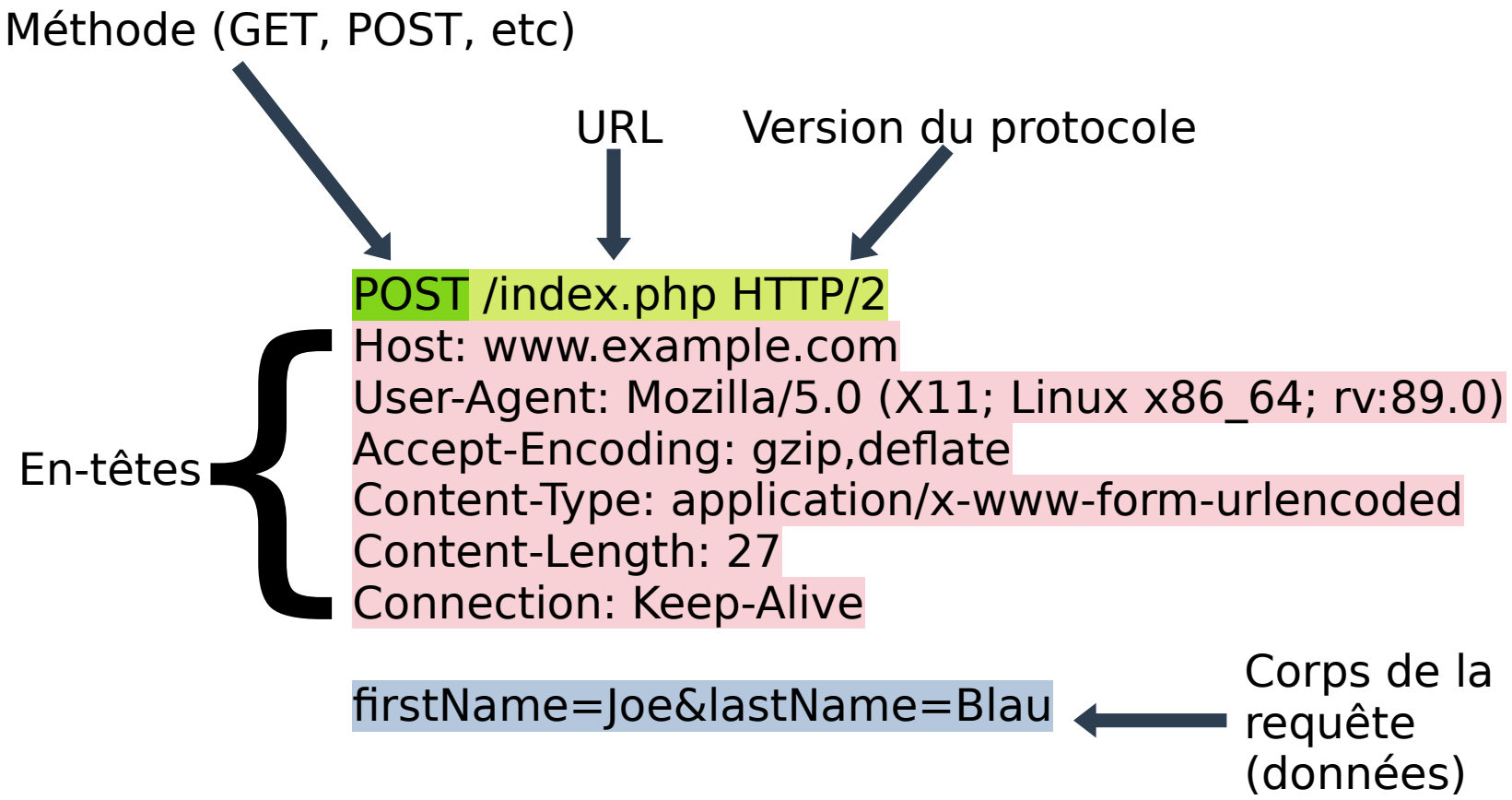
- **Un protocole est l'ensemble des règles qui régissent la communication entre deux logiciels (ex: un client et un serveur) à travers un réseau**
- **Il existe de nombreux protocoles, pour différents types d'applications, ex:**
  - HTTP
  - FTP
  - SSH
  - WebSocket

Protocole HTTP et API Web

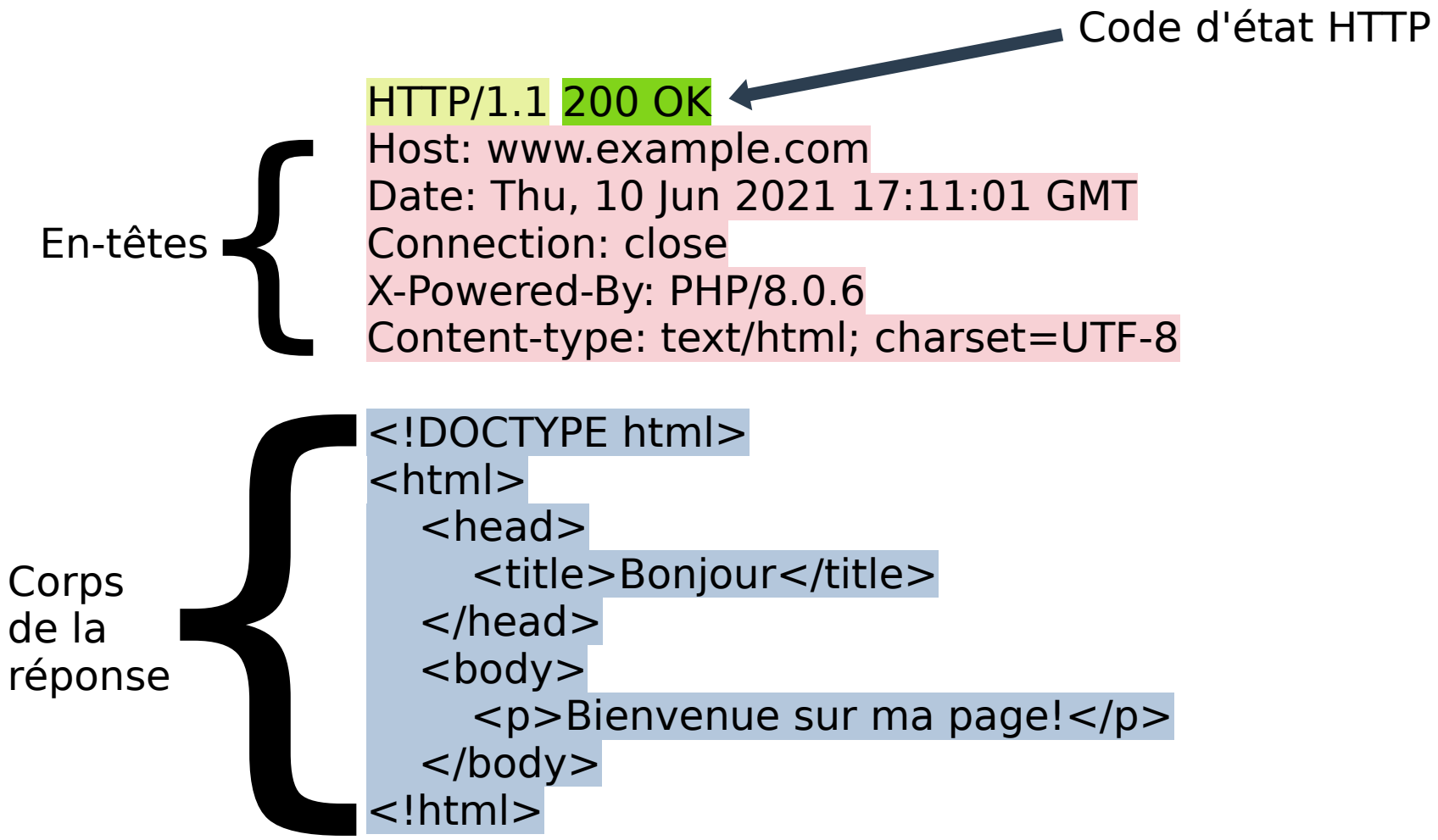
- **Le protocole HTTP est habituellement utilisé pour récupérer du HTML, du CSS, du JavaScript, des images, etc.**
  - Ou transmettre les données d'un formulaire

- **Dans le cadre d'une API Web, on l'utilise plutôt pour recevoir et transmettre des données structurées (représentées sous forme de texte dans un format particulier)**

Exemple de requête HTTP



Exemple de réponse HTTP



REST, c’est quoi?

REST, c’est quoi?

- **REST (*Representational State Transfer*) est un type d'API Web basé sur…**
  - Les méthodes du protocole HTTP
  - Les codes d'état (*status codes*) du protocole HTTP

Requêtes d'API

- **Une requête d'API REST est constituée minimalement:**
  - d'une méthode HTTP
    - Indique l'action à effectuer (lire, créer, mettre à jour, supprimer)
  - d'une route
    - La partie de l'URL qui vient après l'URL de base (***base URL***)
      - Ex: https://example.org/api**/users**
    - Représente la ressource avec laquelle on veut interagir

Requêtes d’API

- **Une requête peut aussi comprendre:**
  - des en-têtes (*headers*) HTTP
    - Contiennent des informations supplémentaires dont le serveur a besoin pour traiter la requête (ex: format des données transmises, informations d’authentification, etc)
  - un corps (*body*)
    - Contient les données sur la ressource à créer ou mettre à jour

Les méthodes HTTP

- **Quatre méthodes HTTP sont couramment utilisées dans une API REST:**
  - GET
  - POST
  - PUT
  - DELETE

La méthode GET

- **La méthode GET est utilisée pour récupérer les données concernant une ressource**
- **La route identifie la ressource à récupérer, ex:**
  - **/users** récupère la liste des utilisateurs
  - **/users/42** récupère l'utilisateur dont l'ID est 42

Les méthodes POST et PUT

- **Les méthodes POST et PUT sont utilisées pour créer ou mettre à jour une ressource**
  - Les données sur la ressource à créer ou mettre à jour sont placées dans le corps de la requête

POST vs PUT

- **Dans quel cas choisit-on PUT plutôt que POST?**

- **L'explication simple et inexacte:**
  - POST = Créer
    - Ex: **POST /users** pour un nouvel utilisateur
  - PUT = Mettre à jour
    - Ex: **PUT /users/42** pour mettre à jour l'utilisateur dont l'ID est 42

POST vs PUT

- **L'explication plus complexe et exacte:**
  - On utilise POST si on utilise une route qui ne désigne pas spécifiquement la ressource à créer ou mettre à jour, ex:
    - **POST /users** pour créer un nouvel utilisateur, qui n'a donc pas encore d'ID
  - On utilise PUT dans le cas contraire, ex:
    - **PUT /users/42** pour mettre à jour l'utilisateur dont l'ID est 42

La méthode DELETE

- **La méthode DELETE est utilisée pour supprimer une ressource**
- **Exemple:**
  - **DELETE /users/42** supprime l'utilisateur dont l'ID est 42

Réponse à une requête

- **La réponse retournée par le serveur suite à une requête est définie par**
  - Le code d'état HTTP
  - Le corps de la réponse (les données retournées)

Les codes d'état HTTP

- **Quelques codes d'état HTTP sont fréquemment utilisés dans les API de type REST:**
  - **200**: OK
  - **400**: Bad Request
    - Les données transmises avec la requêtes sont invalides
  - **401**: Unauthorized
    - L'utilisateur n'est pas authentifié et ne peut donc pas accéder à la ressource
  - **403**: Forbidden
    - L'utilisateur est authentifié, mais n'a pas les droits requis pour accéder à la ressource
  - **404**: Not Found
    - La ressource demandée n'existe pas
  - **500:** Internal Server Error

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

JSON, c’est quoi?

JSON, c’est quoi?

- **JSON = *JavaScript Object Notation***
- **Format d’échange de données couramment utilisé par les API REST**
  - Autre option: XML (moins populaire de nos jours)
- **Dérivé de la syntaxe utilisée pour déclarer des objets en JavaScript**

Exemple de JSON

```
{
    "firstName": "Harry",
    "middleName": "James",
    "lastName": "Potter",
    "birthDate": {
        "year": 1980,
        "month": 7,
        "day": 31
    },
    languagesSpoken: ["English", "Parseltongue"]
}
```

Types de données

- **JSON supporte deux structures de données**
  - Les objets (`{ key1: value1, key2: value2, ... }`)
    - Équivalent des tableaux associatifs en PHP
  - Les tableaux (`[value1, value2, ...]`)
- **Les valeurs peuvent être des types suivants:**
  - Objet
  - Tableau
  - Nombre
  - Chaîne de caractères (entre guillemets)
  - Booléen (`true` ou `false`)
  - Valeur nulle (`null`)

Exemple d’API REST

Exemple d’API REST

- **API permettant de récupérer les informations concernant**
  - Les étudiants
  - Les cours

Entité « student »

```
{
   "id": "1842421",
   "firstName": "Lisa",
   "lastName": "Simpson",
   "program": "420.B0"
   "semesterPaid": true,
   "classes": [
      "420-103-SH",
      "420-146-SH",
      "420-123-SH",
      "350-153-SH",
      "420-135-SH",
      "601-101-04",
      "340-103-04",
      "109-101-MQ"
   ]
}
```

Entité « class »

```
{
    "id": "420-123-SH",
    "title": "Création de pages Web",
    "hours": {
        "theory": 1,
        "practicalWork": 2,
        "homework": 2
    }
}
```

Récupérer la liste de tous les étudiants

- **Méthode: GET**
- **Route: /students**
- **Format des données retournées:**

```
[
   { "id": "1842421", "firstName": "Lisa", "lastName": "Simpson", ... },
   { "id": "1712345", "firstName": "William", "lastName": "Byers", ... },
   ...
]
```

Récupérer les données d’un étudiant spécifique

- **Méthode: GET**
- **Route: /students/id**
- **Format des données retournées:**

```
{
    "id": "1842421",
    "firstName": "Lisa",
    "lastName": "Simpson",
    "program": "420.B0"
    "semesterPaid": true,
    "classes": [...]
}
```

Créer un nouvel étudiant

- **Méthode: POST**
- **Route: /students**
- **Format du corps de la requête:**

```
{
    "firstName": "Lisa",
    "lastName": "Simpson",
    "program": "420.B0"
}
```

Mettre à jour un étudiant

- **Méthode: PUT**
- **Route: /students/id**
- **Format du corps de la requête:**
  - Un objet contenant les attributs à mettre à jour avec leurs nouvelles valeurs

Supprimer un étudiant

- **Méthode: DELETE**
- **Route: /students/id**

Récupérer la liste des cours

- **Méthode: GET**
- **Route: /classes**
- **Format des données retournées:**

```
[
   { "id": "420-103-SH", "title": "Création de pages Web", ... },
   ...
]
```

Récupérer les données d’un cours spécifique

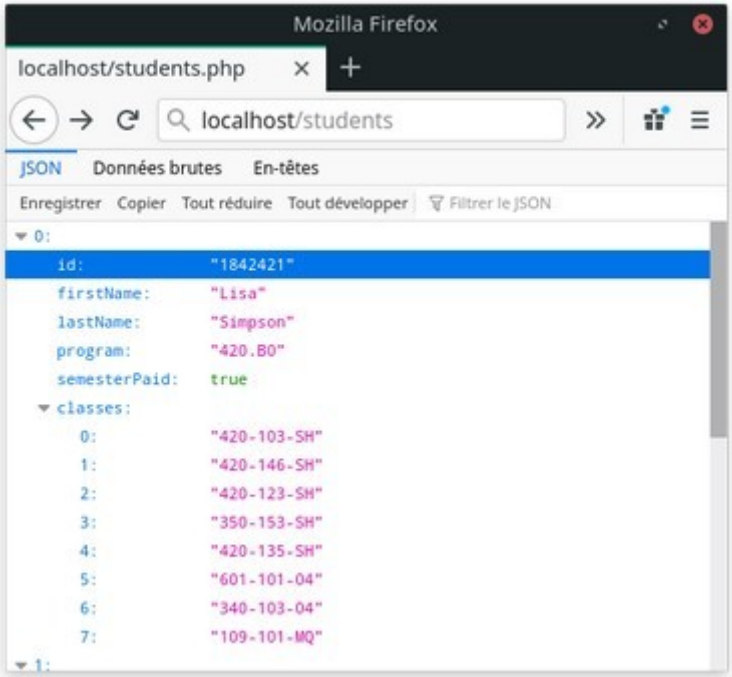
- **Méthode: GET**
- **Route: /classes/id**
- **Format des données retournées:**

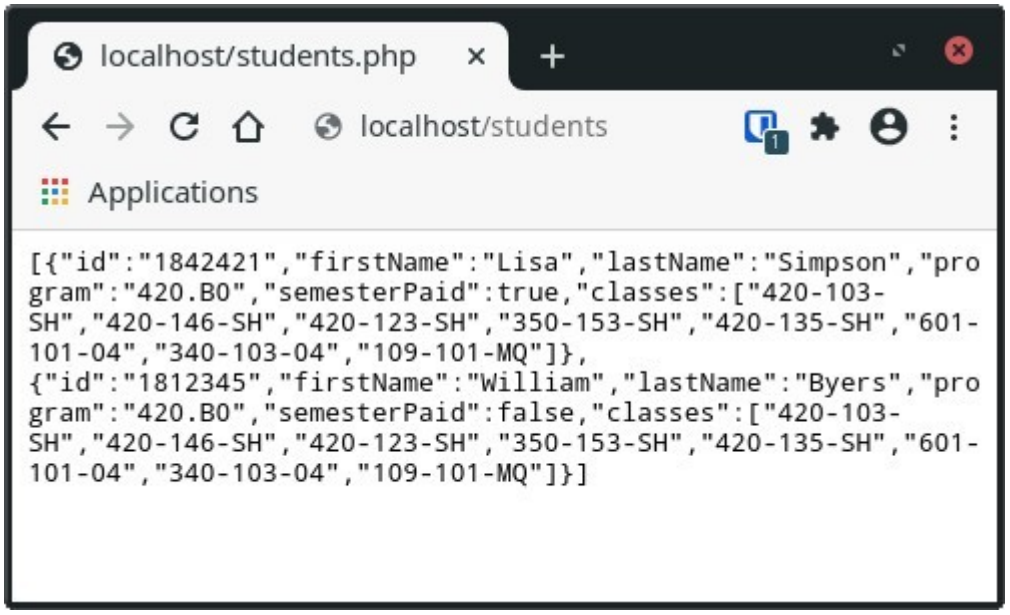
```
{
    "id": "420-123-SH",
    "title": "Création de pages Web",
    "hours": {
        "theory": 1,
        "practicalWork": 2,
        "homework": 2
    }
}
```

Tester son API

Tester son API

- **Si on accède à une route d'API directement dans le navigateur, celui-ci utilisera la méthode GET**
  - La réponse de l'API sera affichée directement dans le navigateur





Tester son API

- **Des extensions pour les navigateurs permettent d'effectuer des requêtes plus complexes ou utilisant d'autres méthodes**
  - Exemple: Rested pour Firefox et Chrome

</> RESTED

https://addons.mozilla.org/fr/firefox/addon/rested/

Fin de la présentation

Des questions?

Photo par Matt Walsh sur Unsplash