Architectures REST

- REST (REpresentational State Transfer)
- pour faire des webservices
- HTTP pour le transport (essence d’un web service)
- HTTP pour la sémantique des messages !

Interface uniforme

- ressource : tout élément offert à la manipulation
- URI : Identifiant de Ressource uniforme

  http_URL = “http:” “//” host [”:” port]

Deux types de schéma d’URI sont distingués :

- URI member qui désigne une seule ressource.
- URI collection qui désigne une liste de ressources de même type.

Sémantique des messages du client au serveur : HTTP

- GET URI : récupérer la représentation d'une ressource (URI member), ou d'une liste de ressources (URI collection)
- POST URI (collection) : rajouter une ressource à une liste de ressources existantes, donc création de ressource.
- PUT URI (member) : modifier une ressource existante ou créer une nouvelle ressource. PATCH peut être utilisé pour remplacer PUT pour modifier une ressource existante.
- DELETE URI : destruction d'une (URI member) ou plusieurs (URI collection) ressources.

Sémantique des messages du serveur au client : HTTP encore

En particulier dans l'utilisation des codes de retour (status code):

- 200 Ok
- 201 Ressource créée
- 303 Redirection vers une ressource dont l'URI est donné par l'entête `Location`
- 404 Ressource non trouvée sur le serveur

Représentation des ressources

- négociation sur le format de représentation entre le client et le serveur
- client : entête `Accept` indique le ou les formats souhaités (type MIME)
- serveur et client : `Content-Type` décrit le format de représentation du corps du message
- formats le plus souvent utilisés : JSON, XML

API REST : exemple détaillé

Gestion de l’authentification d’utilisateurs

Les utilisateurs sont identifiés en tant que ressources par un schéma d’URI du type :

```
http://server/users/id
```

`id` : clef permettant d’identifier de manière unique la ressource, ici le login d’un utilisateur par exemple.

L’ensemble des utilisateurs est représenté par le schéma d’URI

```
http://server/users
```

API gestion d’utilisateurs

- récupère la liste de tous les utilisateurs (Réponse : 200 OK)

```
GET http://server/users
```

- création d’un nouvel utilisateur (Réponse : 201 Created)

```
POST http://server/users/id_user
```

- récupère la représentation de l’utilisateur identifié par `id_user` (Réponse : 200 OK ou 404 resource not found)

```
GET http://server/users/id_user
```

- modifie un utilisateur (Réponse : 200 OK ou 404 resource not found)

```
PUT http://server/users/id_user
```

- efface un utilisateur (Réponse : 200 OK ou 404 resource not found)

```
DELETE http://server/users/id_user
```

Représentation d’un ressources

JSON est utilisé pour représenter les deux types de ressources, utilisateur et liste d’utilisateurs.

```
Content-Type: application/json;charset=utf-8
```

Utilisateur

```
{user:{login: id_user, password: secret }}
```

Liste d’utilisateurs :

```
[{user:{login: id1, password: secret1}}, {user:{login:id2, password:secret2}}]
```

Un utilisateur est donc représenté ici par un login et un mot de passe.