

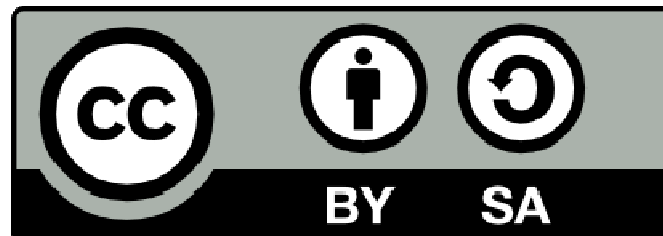


## Creative Commons

*Contrat Paternité*

*Partage des Conditions Initiales à l'Identique*

2.0 France



<http://creativecommons.org/licenses/by-sa/2.0/fr>

# Plan du cours

---

- L'utilisation du Web aujourd'hui ...
- C'est quoi REST ?
  - Ressources
  - Verbes
  - Représentations
- Exemples
- Web Services REST Versus Etendus
- Outils

# Déroulement du cours

---

## ➤ Pédagogie du cours

- Des bulles d'aide tout au long du cours
- Survol des principaux concepts en évitant une présentation exhaustive

## ➤ Logiciels utilisés

- Navigateur Web, CURL, Poster



cURL

## ➤ Pré-requis

- Ingénierie des données
- Schema XML



**Ceci est une astuce**

## ➤ Remerciements

- Djug

**Ceci est une alerte**



# Ressources : Liens sur le Web

---

## ➤ Billets issus de Blog

- [wintermuteblog.blogspot.com/2010/01/wadl-toolbox.html](http://wintermuteblog.blogspot.com/2010/01/wadl-toolbox.html)
- [bitworking.org/news/193/Do-we-need-WADL](http://bitworking.org/news/193/Do-we-need-WADL)
- [www.pompage.net/pompe/comment-j-ai-explique-rest-a-ma-femme](http://www.pompage.net/pompe/comment-j-ai-explique-rest-a-ma-femme)
- [www.biologeek.com/rest,traduction,web-semantique/pour-ne-plus-etre-en-rest-comprendre-cette-architecture](http://www.biologeek.com/rest,traduction,web-semantique/pour-ne-plus-etre-en-rest-comprendre-cette-architecture)

## ➤ Articles

- [www.ibm.com/developerworks/webservices/library/ws-restvsoap](http://www.ibm.com/developerworks/webservices/library/ws-restvsoap)
- [fr.wikipedia.org/wiki/Representational\\_State\\_Transfer](http://fr.wikipedia.org/wiki/Representational_State_Transfer)

## ➤ Cours

- ...

## ➤ Présentations

- [www.slideshare.net/gouthamrv/restful-services-2477903](http://www.slideshare.net/gouthamrv/restful-services-2477903)
- [www.parleys.com/#id=306&st=5&sl=14](http://www.parleys.com/#id=306&st=5&sl=14)

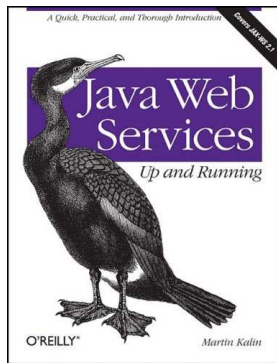
# Ressources : Bibliothèque

---



## ➤ RESTful Web Services

- Auteur : Leonard Richardson & Sam Ruby
- Éditeur : Oreilly
- Edition : Dec. 2008 - 448 pages - ISBN : 0596529260



## ➤ Java Web Services : Up and Running

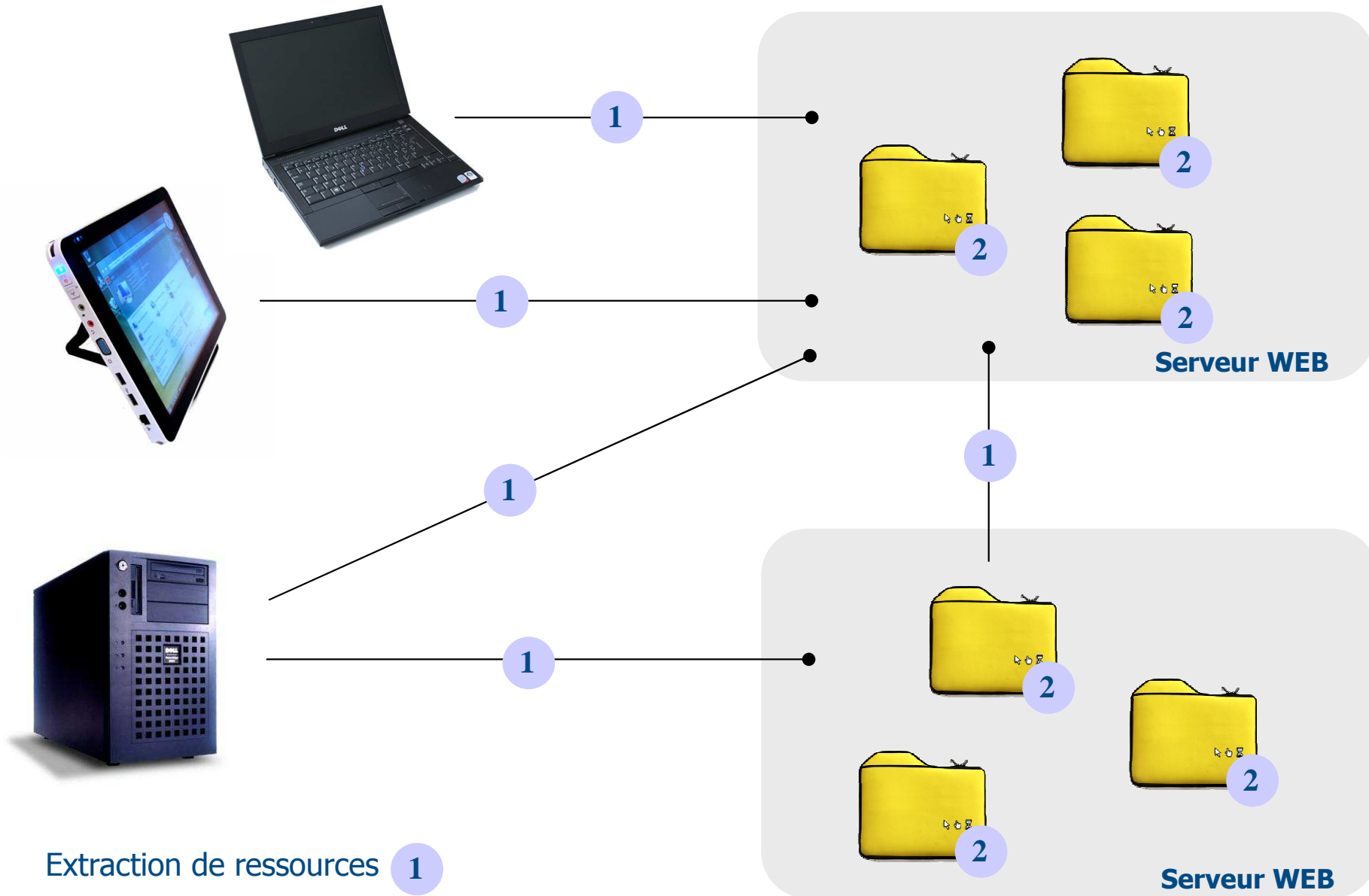
- Auteur : Martin Kalin
- Éditeur : Oreilly
- Edition : Février 2009 - 316 pages - ISBN : 059652112X



## ➤ RESTful .NET

- Auteur : Jon Flanders
- Éditeur : Oreilly
- Edition : Nov. 2008 - 320 pages - ISBN : 0596519206

# L'utilisation du Web aujourd'hui ...



Extraction de ressources 1

Ressources 2

# L'utilisation du Web aujourd'hui ...

➤ Les ressources sont récupérées au travers les URLs

The screenshot shows a Mozilla Firefox browser window with the address bar containing the URL `http://www.ldlc.com/fiche/PB00096757.html`, highlighted in red. The page content includes the LDLC logo, navigation menus, and a product listing for an HP ProLiant ML150 G6 server. The product details section shows the price as 899,95 € TTC and a 'COMMANDE' button. A red arrow points from the text below to the product name in the screenshot.

Une ressource (serveur) est identifiée par une URL

# C'est quoi REST ?

---

- REST est l'acronyme de **RE**presentational **S**tate **T**ransfert
- Principes définis dans la thèse de Roy FIELDING en 2000
  - Principaux auteurs de la spécification HTTP
  - Membre fondateur de la fondation Apache
  - Développeur du serveur Web Apache
- REST est un style d'architecture inspiré de l'architecture du Web
- **REST** est
  - un style d'architecture
  - une approche pour construire une application
- **REST** n'est pas
  - un format
  - un protocole
  - un standard

# C'est quoi REST ?

---

- Les Services Web REST sont utilisés pour développer des architectures orientées ressources
- Différentes nominations disponibles dans la littérature
  - Architectures Orientées Données (**DOA**)
  - Architectures Orientées Ressources (**ROA**)
- Les applications qui respectent les architectures orientées ressources sont respectivement nommées **RESTful**
- Dans la suite du cours nous utiliserons indifféremment la nomination REST et RESTful

# C'est quoi REST ? Les fournisseurs

---

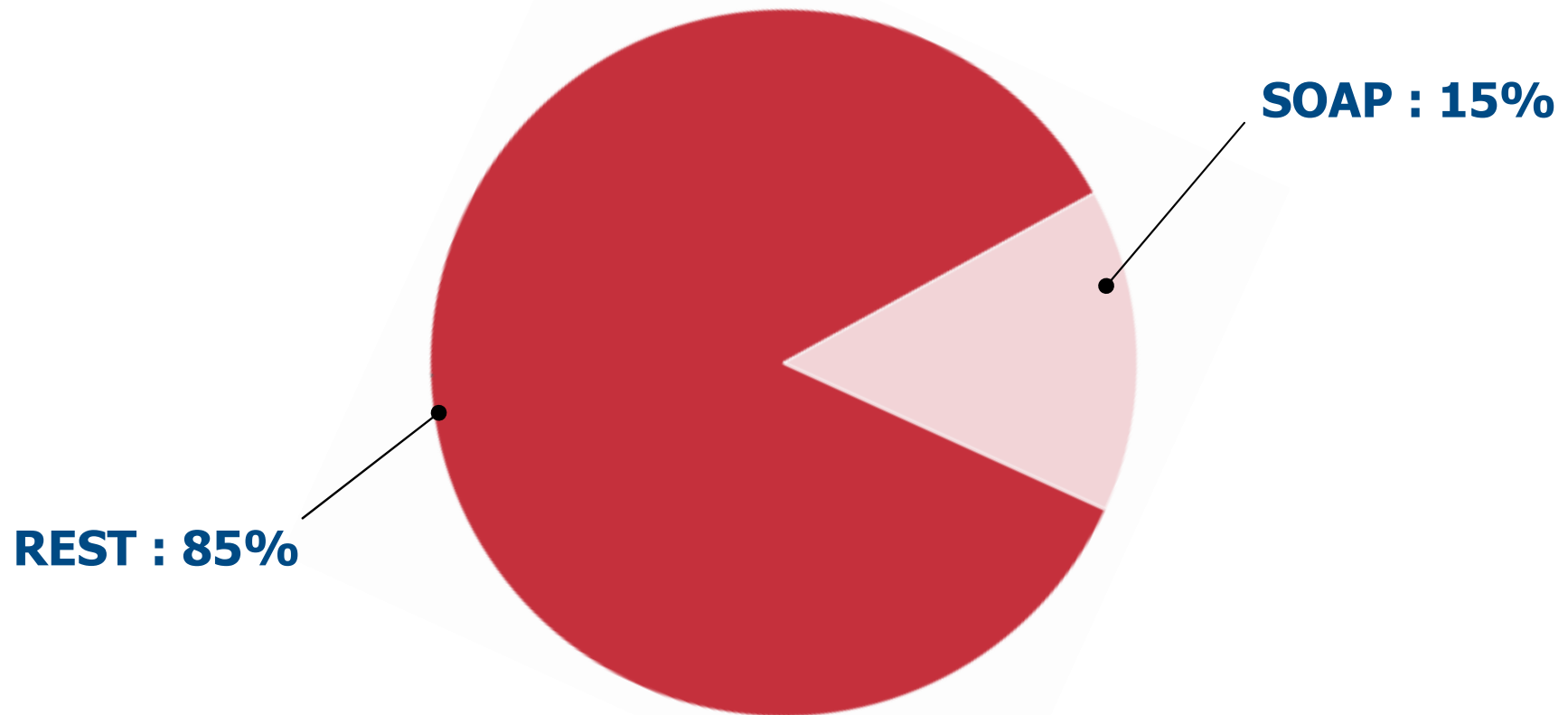
- Certains acteurs qui fournissent des Services Web REST



# C'est quoi REST ? Les fournisseurs

---

- Statistiques de l'utilisation de Services Web REST et SOAP chez AMAZON
  - [www.oreillynet.com/pub/wlg/3005](http://www.oreillynet.com/pub/wlg/3005)



# C'est quoi REST ? : caractéristiques

---

- Les services Web REST sont sans états (Stateless)
  - Chaque requête envoyée vers le serveur doit contenir toutes les informations à leur traitement
  - Minimisation des ressources systèmes, pas de session ni d'état
- Les services Web REST fournissent une interface uniforme basée sur les méthodes HTTP
  - GET, POST, PUT et DELETE
- Les architectures orientées REST sont construites à partir de ressources qui sont uniquement identifiées par des URIs

# C'est quoi REST ? : caractéristiques

---

- Dans une architecture orientée REST, les ressources sont manipulées à travers des formats de représentations
  - Une ressource liée à un *Bon de Commande* est représentée par un document XML
  - La création d'un *Bon de Commande* est réalisée par la combinaison d'une méthode HTTP Post et d'un document XML
- Dans une architecture orientée REST, la communication est obtenue par le transfert de la représentation des ressources
- L'état est maintenue par la représentation d'une ressource
- Par conséquent, le client est responsable de l'état de la ressource

# C'est quoi REST ? : Requête REST

---

## ➤ **Ressources** (Identifiant)

➤ Identifié par une URI

➤ Exemple : *http://localhost:8080/libraryrestwebservice/books*

## ➤ **Méthodes** (Verbes) pour manipuler l'identifiant

➤ Méthodes HTTP : GET, POST, PUT and DELETE

## ➤ **Représentation** donne une vue sur l'état

➤ Informations transférées entre le client et le serveur

➤ Exemples : XML, JSON, ...

# Ressource et URI

---

- Une ressource est quelque chose qui est identifiable dans un système
  - Personne, Agenda, Collection, Document, Image, Carte, ...
- Une URI (Uniform Resource Identifier) identifie une ressource de manière unique sur le système
- Une ressource peut avoir plusieurs URI et la représentation de la ressource peut évoluer avec le temps
- Exemple

*http://localhost:8080/books/aventure/harrypotter/2*

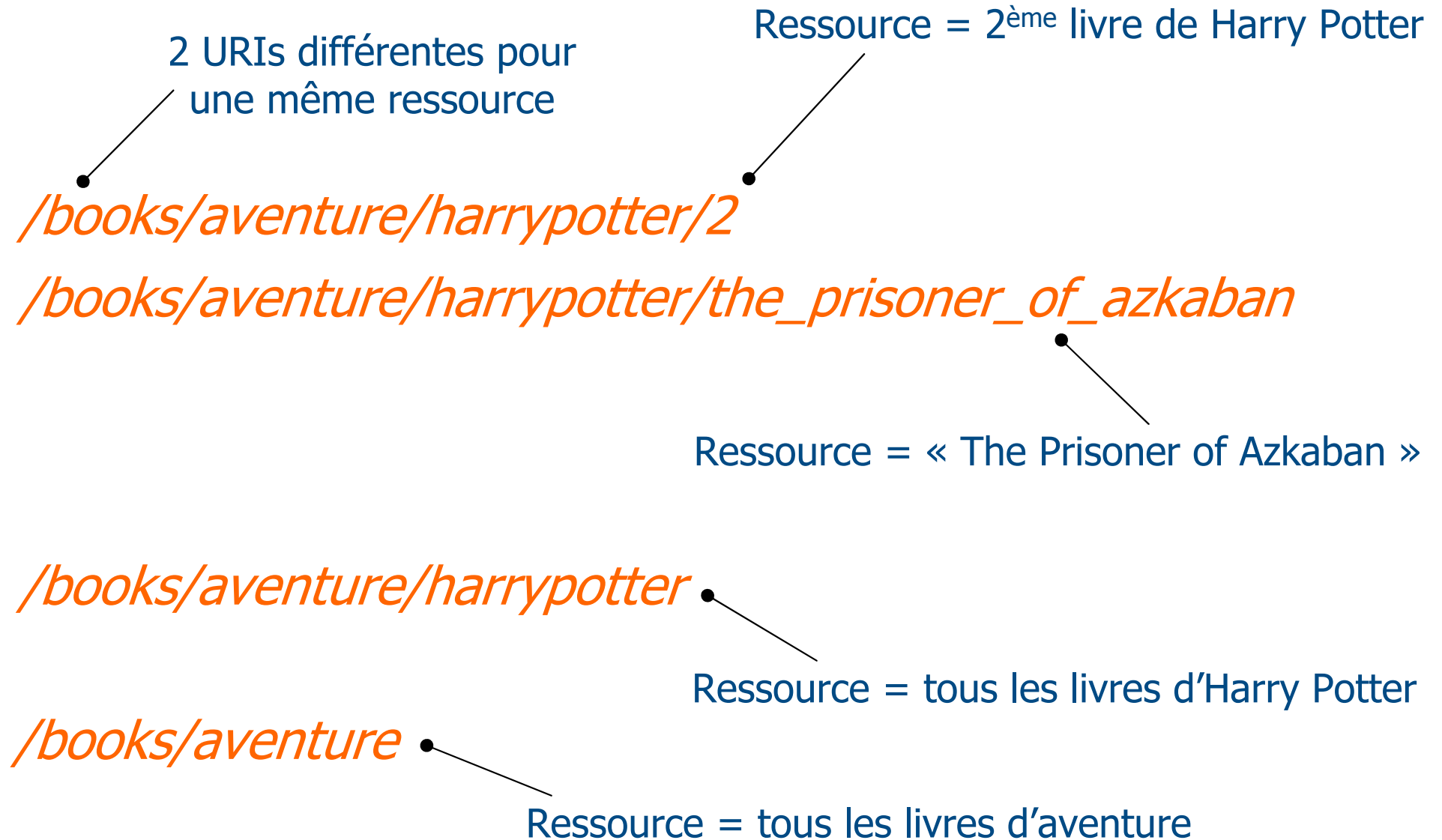
Identifiant primaire  
de la ressource

Ressource de type  
collection

# Ressource et URI

---

## ► Exemples d'URIs



# Méthode

---

- Une ressource quelconque peut subir quatre opérations de base désignées par **CRUD**
  - Create (Créer)
  - Retrieve (Lire)
  - Update (mettre à jour)
  - Delete (Supprimer)
- REST s'appuie sur le protocole HTTP pour exprimer les opérations via les méthodes HTTP
  - Create par la méthode **POST**
  - Retrieve par la méthode **GET**
  - Update par la méthode **PUT**
  - Delete par la méthode **DELETE**
- Possibilité d'exprimer des opérations supplémentaires via d'autres méthodes HTTP (HEAD, OPTIONS)

# Méthode : GET

- Méthode **GET** fournit la représentation de la ressource
  - Idempotent

Action : récupérer

**GET** /books/aventure/harrypotter/2



Client



Serveur

HTTP Status: 200 (OK)  
En-tête + Représentation

# Méthode : POST

- Méthode **POST** crée une ressource
  - Non idempotente (plusieurs créations de la même ressource)

Action : créer

**POST** /books/aventure/harrypotter/  
Représentation dans le corps



**Client**



**Serveur**

HTTP Status: 201 (Created)  
En-tête

# Méthode : DELETE

- Méthode **DELETE** supprime une ressource
  - Idempotent

Action : supprimer

**DELETE** */books/aventure/harrypotter/2*



**Client**



**Serveur**

HTTP Status: 200 (Ok)

En-tête

# Méthode : PUT

- Méthode **PUT** met à jour une ressource
  - Idempotent

Action : mise à jour

**PUT** */books/aventure/harrypotter/2*  
Représentation dans le corps



Client



Serveur

HTTP Status: 200 (Ok)  
En-tête

# Représentation

---

- Fournir les données suivant une représentation pour
  - le client (GET)
  - pour le serveur (PUT et POST)
- Données retournées sous différents formats
  - XML
  - JSON
  - (X)HTML
  - CSV
  - ...
- Le format d'entrée (POST) et le format de sortie (GET) d'un service Web d'une ressource peuvent être différents

# Représentation

## ► Exemples : format JSON et XML

**GET** <https://www.googleapis.com/urlshortener/v1/url?shortUrl=http://goo.gl/fbsS>

```
{
  "kind": "urlshortener#url",
  "id": "http://goo.gl/fbsS",
  "longUrl": "http://www.google.com/",
  "status": "OK"
}
```

Représentation des  
données en JSON

**GET** <http://localhost:8080/librarycontentrestwebservice/contentbooks/string>

```
<?xml version="1.0"?>
<details>
  Ce livre est une introduction sur la vie
</details>
```

Représentation des  
données en XML

## Exemple : Google URL Shortener

---

- Google URL Shortener est un service de réduction d'URL
  - *code.google.com/apis/urlshortener/*
- Les actions proposées sont les suivantes
  - Réduire une URL en une URL plus courte (POST)
  - Retrouver une URL longue à partir d'une URL courte (GET)
  - Afficher les statistiques sur l'utilisation d'une URL réduite (GET)
- Service gratuit et proposant une API REST pour le développement
- Pour utiliser l'API, nécessite l'utilisation d'une clé pour certaines actions lors de l'envoi d'une requête
- Utilisation du plugin **Poster** disponible sous Firefox pour l'émission de requête HTTP

# Exemple : Google URL Shortener (Démonstration)

---

- Présentation du site et des outils
  - Présenter le site URLShortener : *code.google.com/apis/urlshortener/*
    - *code.google.com/apis/urlshortener/v1/reference.html*
  - Présenter le site APIConsole : *https://code.google.com/apis/console*
  - Activation du service URLShortener
  - Présenter le plugin Firefox Poster
- Création d'une URL réduite
- Reconstruire une URL réduite
- Information sur les URLs réduites

# Exemple : Google URL Shortener (Démonstration)

## ➤ Réduction URL (requête)

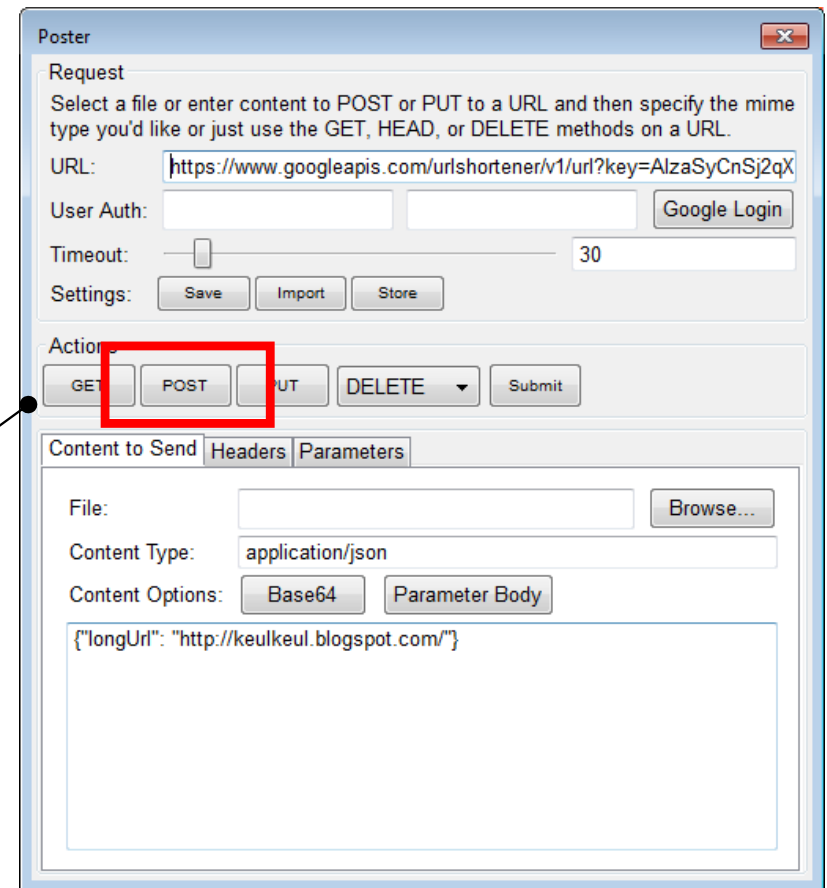
Nécessite une authentification via l'utilisation d'une clé

**POST** `https://www.googleapis.com/urlshortener/v1/url?key=...`

Content-Type: application/json

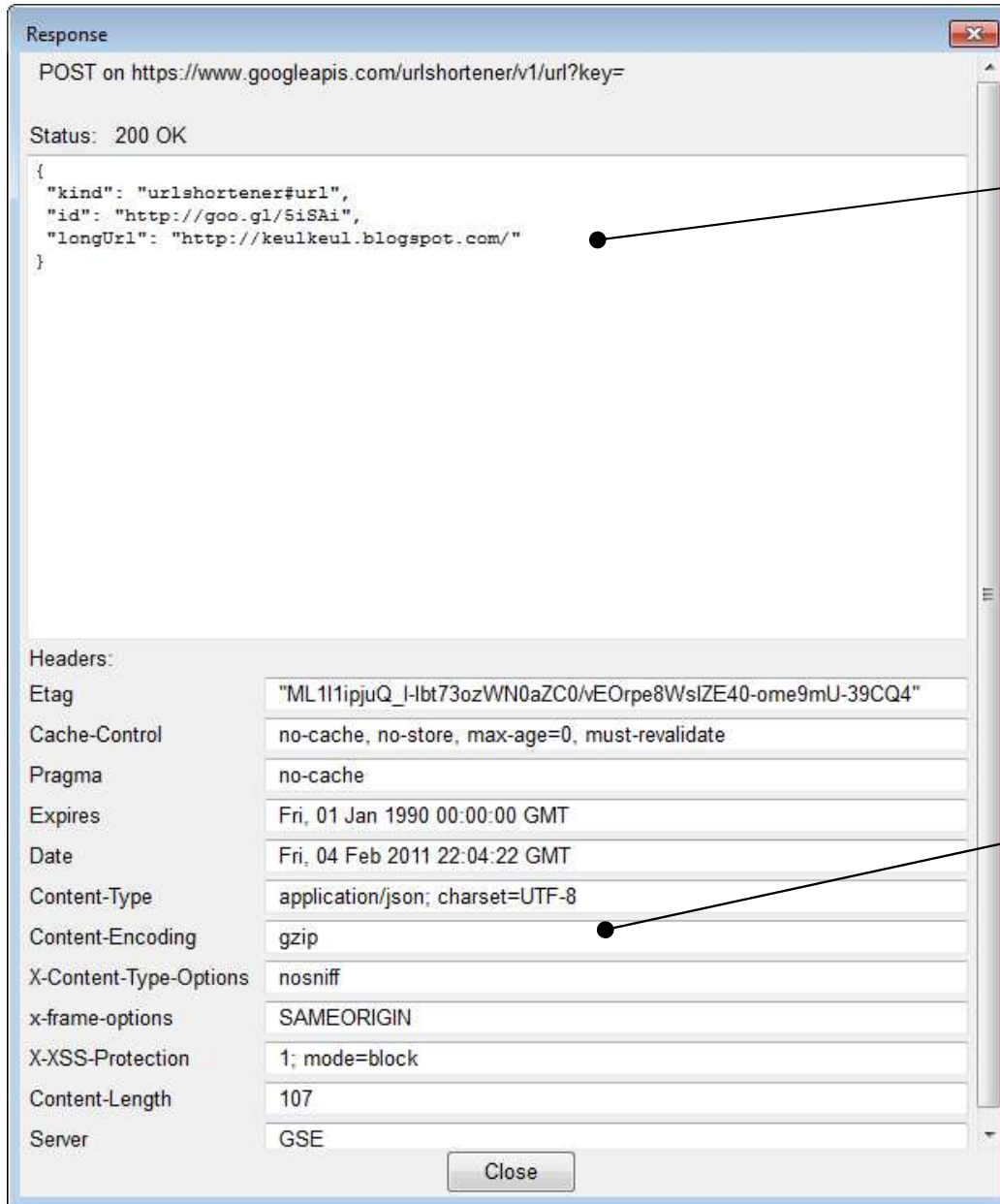
`{"longUrl": "http://www.google.com/"}`

Requête HTTP de type POST avec contenu de type JSON



# Exemple : Google URL Shortener (Démonstration)

## ► Réduction URL (réponse)



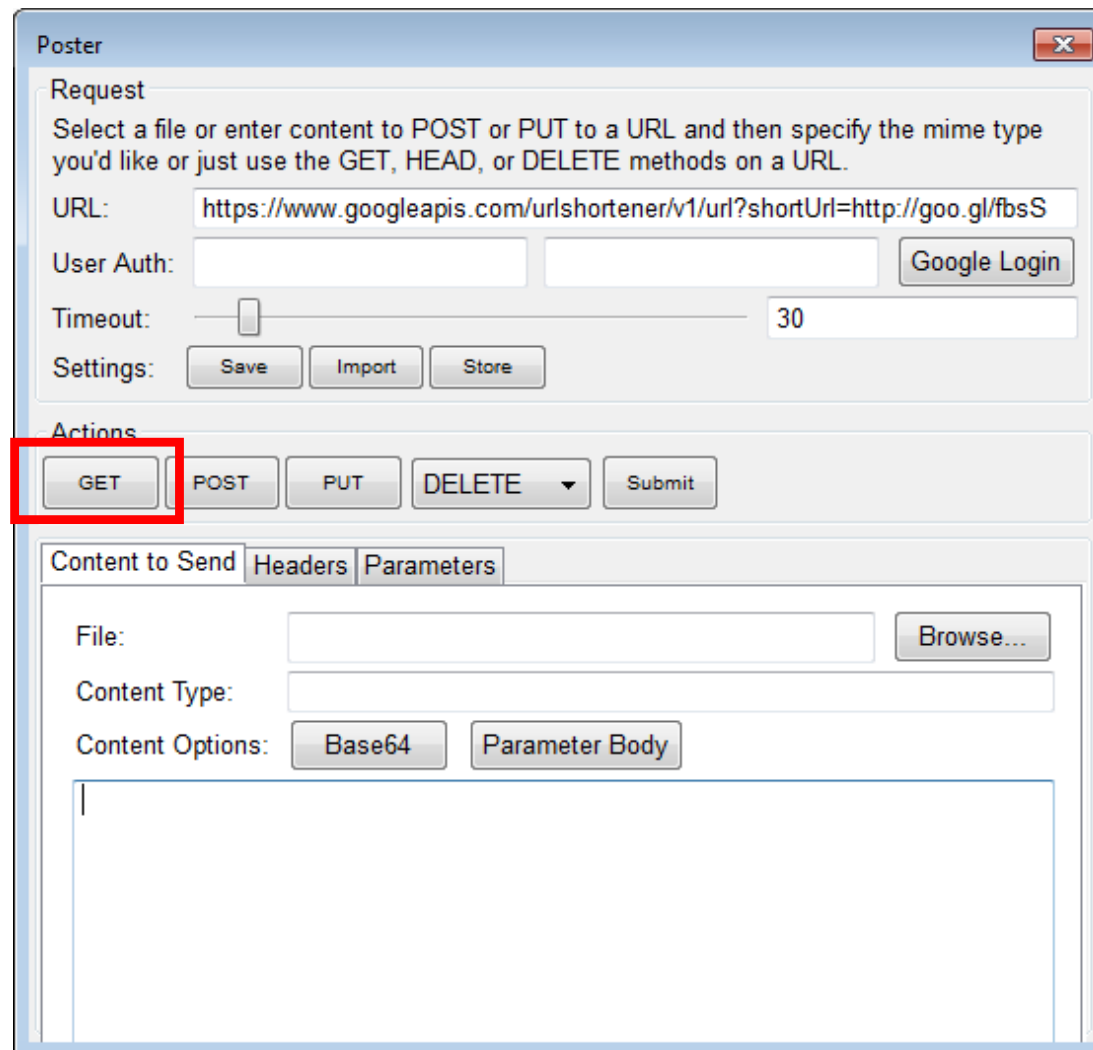
Contenu du corps de la réponse qui correspond à la ressource nouvellement créée au format JSON

Informations placées dans l'en-tête de la réponse

# Exemple : Google URL Shortener (Démonstration)

- Reconstruire une URL réduite (requête)

**GET** <https://www.googleapis.com/urlshortener/v1/url?shortUrl=http://goo.gl/fbsS>



# Exemple : Google URL Shortener (Démonstration)

## ► Reconstruire une URL réduite (réponse)

Response

POST on https://www.googleapis.com/urlshortener/v1/url?shortUrl=http://goo.gl/fbsS&key=AlzaSyCnSj2qXoHuZu-xs2q5ChyLEf\_YfxAWG2o

Status: 200 OK

```
{
  "kind": "urlshortener#url",
  "id": "http://goo.gl/fbsS",
  "longUrl": "http://www.google.com/"
}
```

Headers:

Etag	"ML111ipjuQ_I-Ibt73ozWN0aZC0/bzzW4bYcGdYK2pEqMZbheyyMn_E"
Cache-Control	no-cache, no-store, max-age=0, must-revalidate
Pragma	no-cache
Expires	Fri, 01 Jan 1990 00:00:00 GMT
Date	Mon, 07 Feb 2011 13:07:58 GMT
Content-Type	application/json; charset=UTF-8
Content-Encoding	gzip
X-Content-Type-Options	nosniff
x-frame-options	SAMEORIGIN
X-XSS-Protection	1; mode=block
Content-Length	100
Server	GSE

Close

Contenu du corps de la réponse qui correspond à la ressource nouvellement créée au format JSON

# Service Web Etendus VERSUS REST



Client

```
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:hel="http://helloworldwebservice.lisi.ensma.fr/">
  <soapenv:Header/>
  <soapenv:Body>
    <hel:GetOrderDetail>
      <value>14546-xx-45</value>
    </hel:GetOrderDetail>
  </soapenv:Body>
</soapenv:Envelope>
```



Serveur

SOAP

REST



Client

```
GET http://localhost:8080/order?ordernum=14546-xx-45
```



Serveur

# Service Web Etendus VERSUS REST

---

- Les Services Web étendus (SOAP) et les Services Web REST différent par le fait que
  - Services Web étendus reposent sur des standards
  - REST est un style d'architecture
- Services Web étendus (SOAP)
  - **Avantages**
    - Standardisé
    - Interopérabilité
    - Sécurité (WS-Security)
    - Outillé
  - **Inconvénients**
    - Performances (enveloppe SOAP supplémentaire)
    - Complexité, lourdeur
    - Cible l'appel de service

# Service Web Etendus VERSUS REST

---

## ➤ Services Web **REST**

### ➤ **Avantages**

- Simplicité de mise en œuvre
- Lisibilité par l'humain
- Evolutivité
- Repose sur les principes du Web
- Représentations multiples

### ➤ **Inconvénients**

- Sécurité restreinte par l'emploi des méthodes HTTP
- Cible uniquement l'appel de ressource

# WADL

---

- WADL (**W**eb **A**pplication **D**escription **L**anguage) est un langage de description XML de services de type REST
- WADL est une spécification W3C initiée par SUN
  - *[www.w3.org/Submission/wadl/](http://www.w3.org/Submission/wadl/)*
- Description des services par éléments de type
  - *Ressource, Méthode, Paramètre, Requête, Réponse*
- L'objectif est de pouvoir générer automatiquement les APIs clientes d'accès aux services REST
- Remarques
  - Peu d'outils exploite la description WADL (*<http://wadl.java.net/>*)
  - Apparue bien plus tard

## ► Exemple : Afficher le WADL de services REST

```
<application>
<doc jersey:generatedBy="Jersey: 1.4 09/11/2010 10:30 PM"/>
<resources base="http://localhost:8088/librarycontentrestwebservice/">
  <resource path="/contentbooks">
    <resource path="uribuilder2">
      <method name="POST" id="createURIBooks">
        <request>
          <representation mediaType="application/xml"/>
        </request>
        <response>
          <representation mediaType="*/*/>
        </response>
      </method>
    </resource>
    <resource path="uribuilder1">
      <method name="POST" id="createBooksFromURI">
        <request>
          <representation mediaType="application/xml"/>
        </request>
        <response>
          <representation mediaType="*/*/>
        </response>
      </method>
    </resource>
    ...
  </resource>
</resources>
</application>
```

# Outils et bibliothèques

---

- Des outils pour appeler des services REST
  - CURL : *curl.haxx.se*
  - Poster (plugin Firefox) : *https://addons.mozilla.org/en-US/firefox/addon/poster/*
  - SOAPUI : *www.soapui.org*
- Des plateformes pour développer (serveur) et appeler (client) des services REST
  - JAX-RS (Jersey)  pour la plateforme Java
  - .NET 
  - PHP 
  - Python, ... 