

# DOCUMENTATION API REST

## DESCRIPTION FONCTIONNELLE

### CONTEXTE

API REST pour une gestion simpliste d'une bibliothèque. L'API couvre un modèle de données simpliste qui est composée de 2 entités/ressources : Bibliothèque (Library) et Livre (Book).

Ces ressources ont pour propriétés :

#### **BOOK**

Les livres ont pour propriétés un nom (String), une date de parution (Date), un ISBN (String) et un auteur (String)

#### **LIBRARY**

Les bibliothèques ont pour propriétés un nom (String), une adresse (String), une année de construction (Int), une collection de livres (Livre)

L'API prend en charge les requêtes HTTP GET, POST, PUT, DELETE et OPTIONS sur ces 2 ressources. Avec cette API il est donc possible de :

- Récupérer une collection de tous les livres existants
- Récupérer une collection de toutes les bibliothèques existantes, leurs collections livres inclus
- Récupérer un livre ou une bibliothèque à partir d'un ID
- Créer un nouveau livre ou une nouvelle bibliothèque
- Modifier un livre ou une bibliothèque déjà existant à partir d'un ID
- Supprimer un livre ou une bibliothèque grâce à leur ID
- Obtenir une description des méthodes autorisées sur une ressource

L'API supporte le format de réponse XML et JSON pour la méthode GET

---

## DESCRIPTION TECHNIQUE

### **MÉTHODE GET :**

## LIBRARY

<http://localhost:8081/WebServiceREST/library>

**Description :** Renvoie la liste des bibliothèques existantes avec la collection de livres associée.

**Donnée d'entrée :** Fournir un header Accept (application/xml ou application/json), sinon l'API retourne une réponse au format XML par défaut.

**Réponse :** Renvoie une collection de librairie au format XML, si aucun header Accept n'a été défini, sinon renvoie le résultat au format demandé.

### Format XML :

```
<?xml version="1.0" encoding="UTF-8"?>
<list>
  <library id="1">
    <address>15 Av. Gonzales</address>
    <books>
      <book id="2" />
      <book id="1" />
      <book id="3" />
    </books>
    <name>Librairie Saint-Joseph</name>
    <yearCreated>1985</yearCreated>
  </library>
  <library id="2">
    <address>23 Av. Apotheon</address>
    <books>
      <book id="4" />
      <book id="5" />
    </books>
    <name>Librairie Hermes</name>
    <yearCreated>1900</yearCreated>
  </library>
</list>
```

Bibliothèque : <library id =""></library>

Adresse : <address></address>

Collection de livre : <books></books>

Nom de la bibliothèque : <name></name>

Année de construction : <yearCreated></yearCreated>

### Format JSON :

```
[
  {
    "id": 1,
    "address": "15 Av. Gonzales",
    "books": [
      {
        "id": 2
      },
      {
        "id": 3
      },
      {
        "id": 1
      }
    ],
    "name": "Librairie Saint-Joseph",
    "yearCreated": 1985
  },
  {
    "id": 2,
    "address": "23 Av. Apotheon",
    "books": [
      {
        "id": 5
      },
      {
        "id": 4
      }
    ],
    "name": "Librairie Hermes",
    "yearCreated": 1900
  }
]
```

**Erreurs gérées :** Pas d'erreurs.

<http://localhost:8081/WebServiceREST/library/id>

**Description :** Renvoie une représentation de la bibliothèque dont l'id a été passé en paramètre.

**Donnée d'entrée :** L'ID de la bibliothèque, dans l'url. Fournir un header Accept (application/xml ou application/json), sinon l'API retourne une réponse au format XML par défaut.

**Réponse :** Renvoie les informations de la bibliothèque demandée au format XML, si aucun header Accept n'a été défini, sinon renvoie le résultat au format demandé.

**Format JSON :**

```
{
  "id": 1,
  "address": "15 Av. Gonzales",
  "books": [
    {
      "id": 3
    },
    {
      "id": 1
    },
    {
      "id": 2
    }
  ],
  "name": "Librairie Saint-Joseph",
  "yearCreated": 1985
}
```

**Erreurs gérées** : Demande de bibliothèque inexistante, renvoie d'une erreur 404 NOT FOUND.

<http://localhost:8081/WebServiceREST/libraries>

**Description** : Renvoie la liste des bibliothèques existante avec la collection de livre associée. Il s'agit d'une redirection.

**Donnée d'entrée** : Fournir un header Accept (application/xml ou application/json), sinon l'API retourne une réponse au format XML par défaut.

**Réponse** : Renvoie une collection des bibliothèques au format XML, si aucun header Accept n'a été défini, sinon renvoie le résultat au format demandé.

**Erreurs gérées** : Pas d'erreurs.

BOOK

<http://localhost:8081/WebServiceREST/book>

**Description** : Renvoie la liste des livres existants.

**Donnée d'entrée :** Fournir un header Accept (application/xml ou application/json), sinon l'API retourne une réponse au format XML par défaut.

**Réponse :** Renvoie une collection de livres au format XML, si aucun header Accept n'a été défini, sinon renvoie le résultat au format demandé.

**Format XML :**

```
<?xml version="1.0" encoding="UTF-8"?>
<list>
  <book id="1">
    <library id="1" />
    <name>Livre des révélations</name>
    <releaseDate>2017-11-05 22:34:40.694 CET</releaseDate>
    <isbn>REV2017</isbn>
    <author>Flamarion</author>
  </book>
  <book id="2">
    <library id="1" />
    <name>Grails pour les nuls</name>
    <releaseDate>2017-11-05 22:34:40.695 CET</releaseDate>
    <isbn>GRNUL17</isbn>
    <author>Hachette</author>
  </book>
</list>
```

**Format JSON :**

```
[
  {
    "id": 1,
    "library": {
      "id": 1
    },
    "name": "Livre des révélations",
    "releaseDate": "2017-11-05T21:34:40Z",
    "isbn": "REV2017",
    "author": "Flamarion"
  },
  {
    "id": 2,
    "library": {
      "id": 1
    },
    "name": "Grails pour les nuls",
    "releaseDate": "2017-11-05T21:34:40Z",
    "isbn": "GRNUL17",
    "author": "Hachette"
  }
]
```

**Erreurs gérées :** Pas d'erreurs.

<http://localhost:8081/WebServiceREST/book/id>

**Description :** Renvoie une représentation du livre dont l'id a été passé en paramètre.

**Donnée d'entrée :** L'ID du livre, dans l'url. Fournir un header Accept (application/xml ou application/json), sinon l'API retourne une réponse au format XML par défaut.

**Réponse :** Renvoie les informations du livre demandé au format XML, si aucun header Accept n'a été défini, sinon renvoie le résultat au format demandé.

**Format JSON :**

```
{
  "id": 1,
  "address": "15 Av. Gonzales",
  "books": [
    {
      "id": 3
    },
    {
      "id": 1
    },
    {
      "id": 2
    }
  ],
  "name": "Librairie Saint-Joseph",
  "yearCreated": 1985
}
```

**Erreurs gérées :** Demande d'un livre inexistant, renvoie d'une erreur 404 NOT FOUND.

<http://localhost:8081/WebServiceREST/books>

**Description :** Renvoie la liste des livres existants. Il s'agit d'une redirection.

**Donnée d'entrée :** Fournir un header Accept (application/xml ou application/json), sinon l'API retourne une réponse au format XML par défaut.

**Réponse :** Renvoie une collection de livres au format XML, si aucun header Accept n'a été défini, sinon renvoie le résultat au format demandé.

**Erreurs gérées** : Pas d'erreurs.

## MÉTHODE POST :

LIBRARY

<http://localhost:8081/WebServiceREST/library>

**Description** : Créer une nouvelle bibliothèque.

**Donnée d'entrée** : form-data avec les paramètres correspondant aux informations adresse, nom et année de construction (address, name, yearCreated).

<input checked="" type="radio"/>	form-data	<input type="radio"/>	x-www-form-urlencoded	<input type="radio"/>	raw	<input type="radio"/>	binary
	Key						Value
<input checked="" type="checkbox"/>	address						24 av Pensilvanie
<input checked="" type="checkbox"/>	name						The UberLibrary
<input checked="" type="checkbox"/>	yearCreated						1985

**Réponse** : 200 OK, création de la nouvelle bibliothèque

**Erreurs gérées** : 400 Bad request, si un des paramètres est manquant.

BOOK

<http://localhost:8081/WebServiceREST/book>

**Description** : Créer un nouveau livre.

**Donnée d'entrée** : form-data avec les paramètres correspondant aux informations nom, isbn, date de parution, auteur et ID de la bibliothèque auquel ce livre appartient (name, isbn, releaseDate, author, library).

<input checked="" type="radio"/> form-data <input type="radio"/> x-www-form-urlencoded <input type="radio"/> raw <input type="radio"/> binary			
	Key		Value
<input checked="" type="checkbox"/>	name	Text ▾	HELLSING
<input checked="" type="checkbox"/>	isbn		ISBN
<input checked="" type="checkbox"/>	releaseDate		18/05/1985
<input checked="" type="checkbox"/>	author		Hakuma
<input checked="" type="checkbox"/>	library		1

**Réponse** : 200 OK, création du nouveau livre

**Erreurs gérées** : 400 Bad request, si un des paramètres est manquant.

## MÉTHODE PUT :

LIBRARY

<http://localhost:8081/WebServiceREST/library/id>

**Description** : Modifier une bibliothèque à partir d'un ID fournit dans l'URL.

**Donnée d'entrée** : JSON avec les paramètres correspondant aux informations adresse, nom et année de construction (address, name, yearCreated).

```
{
  "address": "15 Av. St Germain",
  "name": "Librairie 20",
  "yearCreated": 1970
}
```

**Réponse** : 200 OK, modification de la bibliothèque

**Erreurs gérées** :

- 400 Bad request, si un des paramètres est manquant.
- 400 Bad request, si l'ID n'a pas été renseigné dans l'URL

BOOK

<http://localhost:8081/WebServiceREST/book/id>



**Description :** Modifier un livre à partir d'un ID fournit dans l'URL.

**Donnée d'entrée :** JSON avec les paramètres correspondant aux informations nom, isbn, date de parution, auteur et ID de la bibliothèque auquel ce livre appartient (name, isbn, releaseDate, author, library).

```
{  
  "library": {  
    "id": 2  
  },  
  "name": "Bouquin",  
  "releaseDate": "1975-04-09T23:00:00Z",  
  "isbn": "ISBN",  
  "author": "George"  
}
```

**Réponse :** 200 OK, modification du livre

**Erreurs gérées :**

- 400 Bad request, si un des paramètres est manquant.
- 400 Bad request, si l'ID n'a pas été renseigné dans l'URL
- 400 Bad request, si la bibliothèque passée en paramètre n'existe pas

## MÉTHODE DELETE :

LIBRARY

<http://localhost:8081/WebServiceREST/library/id>

**Description :** Supprimer une bibliothèque à partir d'un ID fournit dans l'URL.

**Donnée d'entrée :** L'ID de la bibliothèque à supprimer.

**Réponse :** 200 OK, suppression de la bibliothèque

**Erreurs gérées :**

- 400 Bad request, si l'ID n'a pas été renseigné dans l'URL

BOOK

<http://localhost:8081/WebServiceREST/book/id>

**Description :** Supprimer un livre à partir d'un ID fournit dans l'URL.

**Donnée d'entrée :** L'ID du livre à supprimer.

**Réponse :** 200 OK, suppression du livre

**Erreurs gérées :**

- 400 Bad request, si l'ID n'a pas été renseigné dans l'URL
- 404 Not Found, si le livre passée en paramètre n'existe pas

## MÉTHODE OPTIONS :

LIBRARY

<http://localhost:8081/WebServiceREST/library>

BOOK

<http://localhost:8081/WebServiceREST/book>

**Description :** Obtenir les méthodes supportées par chaque ressources

**Donnée d'entrée :** aucune

**Réponse :** 200 OK, GET, PUT, POST, DELETE, OPTION

**Erreurs gérées :** Aucune