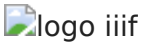
Données multimédia : Python pour le traitement d’images et de données audio

Master Humanités Numériques du CESR

Clément Plancq (MSH VDL / CITERES)



iiif.io

IIIF vise à créer un écosystème ouvert et interopérable autour des images numériques, favorisant l'accès, la recherche et la valorisation des contenus culturels

IIIF désigne :

- un cadre d’intéropérabilité pour la diffusion d’images et d’objets numériques
- une communauté d’acteurs et d’institutions

IIIF définit un cadre technique commun à destination des fournisseurs d’image numériques afin de :

- délivrer les images de manière standardisée sur le web
- les décrire à l’aide de métadonnées structurées
- les rendre consultables, manipulables, annotables

Concrètement, ce cadre techinque ce sont des APIs et des outils.

Les APIs :

- *Image API*
- *Presentation API*
- Authorization Flow
- Change Discovery
- Content Search
- Content State

En plus des standards, IIIF soutient et fédère le développement d’applications open source et partagées :

- serveur d’images

- visualiseurs
- outils d'annotation
- modules pour CMS

Voir https://github.com/IIIF/awesome-iiif

IIIF est porté par un consortium de musées, d'universités, de bibliothèques nationales, … Voir la liste des membres

Voir la carte

IIIF est né en réponse aux problèmes rencontrés après les années 2000 et la prolifération des projets de bibliothéques numériques :

- incapacité des systèmes à échanger des données : manque d'interopérabilité
- développement de serveur de diffusion d'image et/ou de visualiseur propre à chaque projet : long, coûteux et difficile à maintenir
- outils avec UIs et fonctionnalités différentes : difficile à appréhender par les chercheurs

- Approche en silo : les images et les métadonnées ne sont disponibles que dans l'entrepôt où elles sont stockées
- Approche distribuée et intéropérable : les entrepôts sont des points d'accès distants auquels des applications tierces peuvent se brancher
  On peut voir ça comme du « *streaming* » d'image

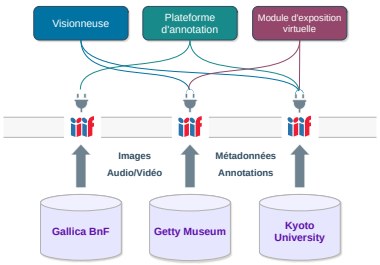


Schéma : principe général d'interopérabilité de IIIF (trois applications différentes sont branchées à trois entrepôts IIIF) source : Équipe Portail Biblissima+

Exemples d'images accessibles via les protocoles IIIF

- un manuscrit sur Gallica
- une image sur Cultural Japan
- une photo sur National Galery of Art

Zoom profond

Voir la carte sur Stanford Digital Repository

Comparaison d’images

https://www.theleidencollection.com/viewer/david-and-uriah/

Images de plusieurs sites

Le manuscrit 5 de la Bibliothèque municipale de Châteauroux, c. 1460.
Folio est à la BVMM (Bibliotheque Virtuelle de Manuscrits medievaux, IRHT)
Miniature est à la BNF

http://demos.biblissima-condorcet.fr/chateauroux/demo/

Recherche

Exemple de recherche sur manuscrit avec OCR

https://mirador-textoverlay.netlify.app/

Annotations

- Exemple avec Mirador : https://projectmirador.org/ cliquez sur "Try a live demo"
- Exemple ave Adno : https://adno.app/en/example/
- Exemple avec Annona : https://ericayhayes.github.io/audubon/exhibits/

Visites guidées

- Exemple avec Exhibit : Jane Austen

Comment ça marche ?

Deux APIs principales :

- Image API
- Presentation API

Image API

Elle spécifie un service web qui renvoie une image en réponse à une requête HTTP ou HTTPS
La requête peut porter sur :

- l'image elle-même
- de l'information sur l'image

La requête sur l'image se fait via une URI avec la syntaxe suivante :
`{scheme}://{server}{/prefix}/{identifier}/{region}/{size}/{rotation}/{quality}.{format}`

Exemple : https://ids.lib.harvard.edu/ids/iiif/25286607/full/200,/0/default.jpg

scheme: `https`
server: `ids.lib.harvard.edu`
prefix: `/ids/iiif/`
identifier: `25286607`
region: `full`
size: `200,`
rotation: `0`
quality: `default`

format: `jpg`

`region`

Valeur	Description
`full`	Image entière
`square`	Image au format carré renvoyée par le serveur
`x,y,w,h`	`x` : nombre de pixels depuis la position 0 sur l'axe horizontal, `y` position 0 sur l'axe vertical (0,0 est le coin gauche par exemple). `w` : largeur, `h` : hauteur en pixels
`pct:x,y,w,h`	Même chose mais en pourcentage de l'image originale

`size`

Valeur	Description
`max`	Taille maximale mais pas d'upscale
`w,`	L'image renvoyée doit faire cette largeur (attention pas plus grande que la largeur de l'image)

| `,h` | Idem pour la hauteur | `pct:n` | Largeur et hauteur au `n` pourcent de l'image originale | | `w,h` | Largeur, hauteur |

Et autres subtilités, voir les spécifications

`rotation`

Une valeur entre 0 et 360

`quality`

Valeur	Description
`color`	En couleur
`gray`	En niveau de gris
`bitonal`	Noir et blanc
`default`	Valeur par défaut indiquée par le serveur

`format`

Les formats supportés par le serveur parmi `jpg`, `tif`, `png`, `pdf`, `jp2`

Vous pouvez modifier les paramètres d'une requête sur cette page : https://www.learniiif.org/image-api/playground

Afficher l'image `https://ids.lib.harvard.edu/ids/iiif/25286607/full/200,/0/default.jpg` à la 10% de sa taille, inversée et en niveaux de gris

Zoom et tuilage

Les formats JPEG2000 et TIFF Pyramidal permettent le zoom profond à l'aide d'un système de tuilage

Vous pouvez vous faire une idée du principe de tuiles à l'aide de l'outil : IIIF Tile Exploder

Pour en savoir plus consultez https://doc.biblissima.fr/formation-iiif/api-image/implementation/ de la formation de la formation IIIF de Biblissima + de Régis Robineau ([Licence ouverte / Open Licence](https://www.etalab.gouv.fr/licence-ouverte-open-licence/))

Exercice

Trouver l'url de l'image représentée sur https://cultural.jp/en/item/europeana-2021659_S1985_0104

Construire les url IIIF pour :

- image pleine taille pivotée sur l'axe vertical (mirroring)
- image pleine taille retournée à l'envers (180°)
- image redimensionnée avec une hauteur de 250 pixels (en conservant les proportions)
- image au format PNG et en niveau de gris

Information

`{scheme}://{server}{/prefix}/{identifier}/info.json`

Donne des informations sur l'image (taille notamment) et sur le serveur (formats, version de l'API supportée, _)

Voir https://iiif.io/api/image/3.0/#5-image-information

Serveurs et clients pour *Image API*

Serveurs :

- Cantaloupe - Image server written in Java fully conformant to all IIIF Image API versions through 3.0.
- IIPImage Server - High performance image server.
- Loris - Written in Python.

Clients, visualiseurs :

- Mirador - Multi-up workspace. See also Awesome Mirador list.
- Tify - Slim and fast IIIF document viewer built with Vue.js.
- Universal Viewer - Rich embeddable interface.
- OpenSeadragon - IIIF tile support.

Liste complète : https://github.com/IIIF/awesome-iiif

JSON

JavaScript Object Notation
Format de données structurées de type texte
Léger, peu verbeux, beaucoup utilisé par les APIs

Il peut contenir :

- des objets Javascript (dict)

- des tableaux Javascript (list)
- booléens
- nombres
- chaînes de caractères
- valeur `null`

In [1]:
```python
import requests

# Exemple pour récupérer largeur et hauteur de l'image

r = requests.get("https://ids.lib.harvard.edu/ids/iiif/25286607/info.json")
if r.status_code == requests.codes.ok:
    info = r.json()
    width = info['width']
    height = info['height']
    print(f"Largeur : {width}, hauteur : {height}")
```

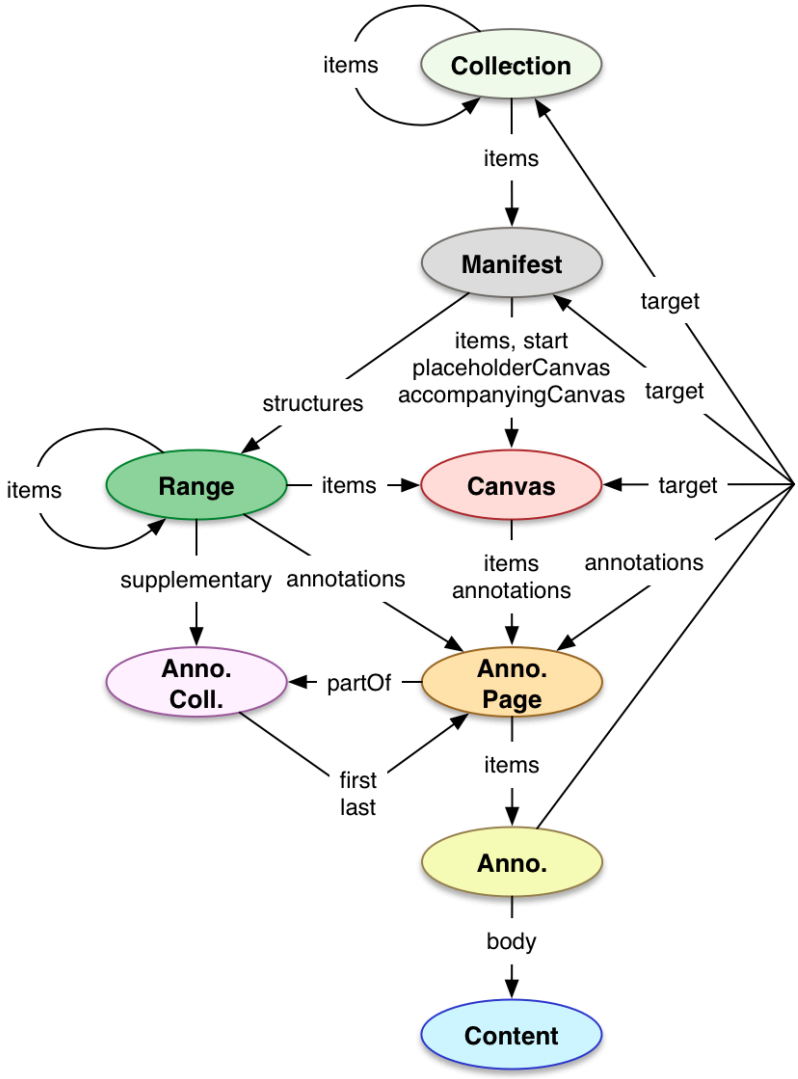
```
Largeur : 4132, hauteur : 8176
```

Presentation API

https://iiif.io/api/presentation/3.0/

Permet de décrire un objet numérique complexe comme une collection d'images par exemple

Cette description est écrite dans un *manifest* le plus souvent distribué au formant JSON-LD (JSON Linked Data)



Exemple pour une image : https://iiif.io/api/cookbook/recipe/0001-mvm-image/manifest.json
Voir dans Mirador

Un livre : https://iiif.io/api/cookbook/recipe/0009-book-1/manifest.json Voir dans Universal Viewer

Avec des métadonnées : https://iiif.io/api/cookbook/recipe/0029-metadata-anywhere/manifest.json Mirador

Voir iiif cookbook

Utiliser des manifests

Trouvez 3 manifests IIIF et importez les dans Mirador

Affichez et parcourez les informations

Importez ces manifests dans l'instance Mirador de Biblissima : https://portail.biblissima.fr/m3/?theme=dark

Écrire un manifest

- Via un programme, en Python par exemple :-)
- Avec un éditeur spécialisé : http://digital.bodleian.ox.ac.uk/manifest-editor/

Pour votre projet vous devrez enrichir les métadonnées d'un manifest

Vous pouvez tester la validité de votre fichier manifest avec https://presentation-validator.iiif.io/ mais il faut que le fichier soit accessible en ligne.

In [1]:
```
import requests
import json

r = requests.get("https://iiif.io/api/cookbook/recipe/0029-metadata-anywhere
manifest = r.json()
metadata = manifest['metadata']

## Création d'un nouvel item de métadonnées
description = {
    "label": {
        "en": ["Detected objects"],
        "fr": ["Objets détectés"]
    },
    "value": {
        "en": ["rabbit, zebra, pot"],
        "fr": ["lapin, zèbre, casserole"]
    }
}

# Insertion dans la liste de metadata
# (attention si je modifie metadata, manifest aussi est modifié)
metadata.append(description)

# Écriture du fichiers JSON résultat
with open('manifest_rabbit.json', 'w') as json_file:
    json.dump(manifest, json_file)
```

Sources

Ce notebook s'appuie sur le contenu et les références des ressources suivantes :

- Introduction aux protocoles IIIF. Formation Enssib (Régis Robineau, 23 janvier 2019)
- Formation à IIIF. Biblissima +
- IIIF Online workshop