

# INF5190 - Révision pour l'examen intra

Jean-Philippe Caissy

16 octobre 2019

# Examen intra

- ▶ La semaine prochaine, 23 octobre 18h
- ▶ 3 heures : 18h à 21h
- ▶ 2 salles :
  - ▶ SH-2560
  - ▶ SH-2580
- ▶ Feuille de note 8.5" x 11" recto-verso, aucun autre matériel permis
- ▶ Matière vue aux cours 1 à 7 (notes de cours 01 à 13), incluant les laboratoires
- ▶ Questions à réponse courte, choix de réponse et réponse longues (type compréhension)

# Révision intra

## 01 - Introduction

### HTTP

Différences entre HTTP 1.0, 1.1 et 2+

- ▶ Évolution des fonctionnalités
  - ▶ Réutilisation des connexions
  - ▶ Compression
    - ▶ Corps de la requête et réponse
    - ▶ Compression des entêtes
  - ▶ Support multi-hôtes
  - ▶ Multi-plexage des requêtes
  - ▶ Server Push

# Révision intra

## 01 - Introduction

### HTTP

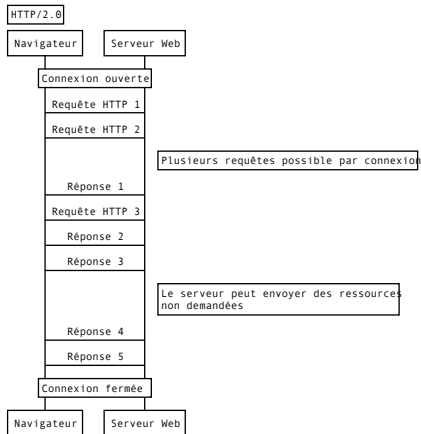


Figure 1: HTTP?

# Révision intra

## 01 - Introduction

### HTTP

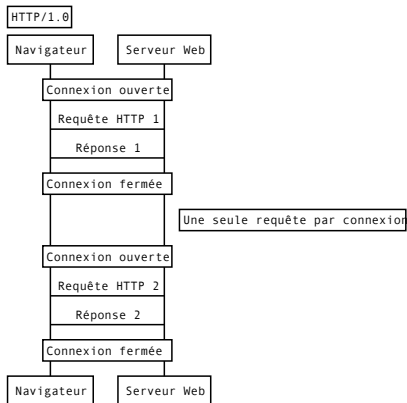


Figure 2: HTTP?

# Révision intra

## 01 - Introduction

### HTTP

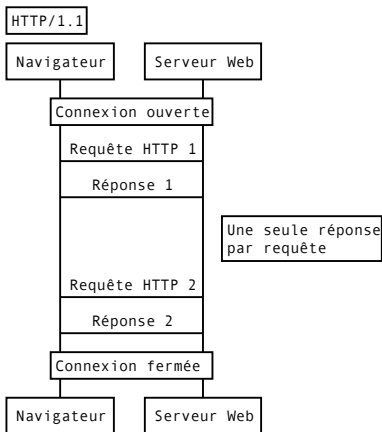


Figure 3: HTTP?

# Révision intra

## 01 - Introduction

### Ressource d'une requête HTTP

`http://user:pass@example.com:81/blog/?tag=net&sort=old#top`

- ▶ Quelles sont les 6 composantes de cette requête HTTP?

# Révision intra

## 01 - Introduction

### Ressource d'une requête HTTP

`http://user:pass@example.com:81/blog/?tag=net&sort=old#top`

Composante	Description
<code>http:</code>	Protocole
<code>user:pass</code>	Nom d'utilisateur et mot de passe
<code>example.com</code>	Domaine de l'hôte
<code>:81</code>	Port
<code>/blog/</code>	Route, emplacement (path)
<code>?tag=net&amp;sort=old</code>	Requête GET (query string)
<code>#top</code>	Ancrage



# Révision intra

02 - Python

@ TODO

# Révision intra

## 03 - Python (suite)

- ▶ Exceptions
- ▶ Modules
  - ▶ Comment est-ce qu'un module est défini?
  - ▶ À quoi sert le fichier `__init__.py` à la racine d'un dossier?
- ▶ Quel est l'utilité d'un environnement virtuel, tel que `virtualenv`?

# Révision intra

## 03 - Python (suite)

Que représente ce code, et pourquoi est-ce qu'on utilise `with`?

```
with open("file.data", "r") as f:  
    print(f.readlines())
```

# Révision intra

## 04 - Introduction aux applications Web

### Requête HTTP

- ▶ Que contient une requête HTTP?

# Révision intra

## 04 - Introduction aux applications Web

### Requête HTTP

- ▶ Que contient une requête HTTP?
  1. Ligne de début
  2. Entêtes
  3. Corps de la requête (body)
- ▶ Que représentent ces 3 composantes d'une requête HTTP?  
Donnez un exemple valide.
- ▶ Quand est-ce que le corps de la requête n'est pas envoyé?
- ▶ Que contient la ligne de début?
- ▶ Quel entête est obligatoire depuis HTTP/1.1?
  - ▶ Cet entête permet quelle fonctionnalité de HTTP/1.1?

## 04 - Introduction aux applications Web

### Entêtes

- ▶ Quel est le format d'un entête?
- ▶ Est-ce que le nom de l'entête est sensible à la case?
- ▶ Est-ce que tous les caractères sont permis dans la valeur d'un entête?
- ▶ Comment est-ce qu'on encode les caractères qui ne sont pas permis?

# Révision intra

## 04 - Introduction aux applications Web

### Entêtes

- ▶ Que veulent dire les entêtes suivants :

- ▶ Referer
- ▶ User-Agent
- ▶ Content-Type
- ▶ Accept-Language
- ▶ Accept-Encoding
- ▶ Accept

- ▶ Que signifie cet entête :

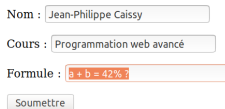
`Accept: text/html, text/plain; q=0.6, */*; q=0.1`

# Révision intra

## 04 - Introduction aux applications Web

### Corps de requête

- Un corps de la requête peut être envoyé lors des requêtes de type POST, PATCH/PUT



Nom :

Cours :

Formule :

Figure 4: Exemple de formulaire

- Comment est représenté ce formulaire dans une requête POST?

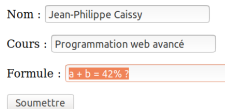


# Révision intra

## 04 - Introduction aux applications Web

### Corps de requête

- Un corps de la requête peut être envoyé lors des requêtes de type POST, PATCH/PUT



Nom :

Cours :

Formule :

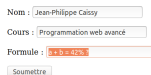
Figure 5: Exemple de formulaire

- Comment est représenté ce formulaire dans une requête POST?

# Révision intra

## 04 - Introduction aux applications Web

### Corps de requête



A screenshot of a web form. It contains three input fields stacked vertically. The first field is labeled 'Nom :' and contains the text 'Jean-Philippe Caissy'. The second field is labeled 'Cours :' and contains the text 'Programmation web avancé'. The third field is labeled 'Formule :' and contains the text 'a + b \* c / 2'. Below these fields is a button labeled 'Soumettre'.

Figure 6: Exemple de formulaire

Grâce à l'annotation hexadécimale avec des pourcentages, cela donne :

```
nom=Jean-Philippe+Caissy&cours=Programmation+web+
avanc%E9&formule=a+%2B+b+%3D+42%25+%3F
```

# Révision intra

## 04 - Introduction aux applications Web

### Réponse

Le format de réponse d'une requête par une application Web est similaire à la requête :

1. Ligne de début
  2. Liste d'entêtes
  3. Contenu de la réponse
- ▶ Que contient la ligne de début?
  - ▶ Est-ce que le contenu de la réponse peut être vide? Si oui, dans quel cas cela est-il approprié?

# Révision intra

## 04 - Introduction aux applications Web

### Réponse

- ▶ Que représentent les statut HTTP :
  - ▶ 1XX
  - ▶ 2XX
  - ▶ 3XX
  - ▶ 4XX
  - ▶ 5XX

# Révision intra

## 04 - Introduction aux applications Web

### Cookie

- ▶ Qu'est-ce qu'un cookie? À quoi servent les cookies?
- ▶ Comment est-ce que les cookie sont transférés de l'application Web au client?
- ▶ Quelle est la différence entre un cookie de session et un cookie persisté?
  - ▶ Comment est-ce qu'on déclare un cookie persisté?
- ▶ Quelles sont les deux méthodes de protection des cookies?
  - ▶ À quoi servent-ils, que protègent-ils?

## 04 - Introduction aux applications Web

### Balanceur de charge

- ▶ Qu'est-ce qu'un balanceur de charge?
- ▶ Quelle est l'utilité d'un balanceur de charge?
- ▶ Il existe 4 algorithmes de routage (Round-Robin, Nombre de connexion, Temps de réponse, Table de hashage)
  - ▶ Quelle est la différence entre un algorithme basé sur le nombre de connexion et celui sur le temps de réponse?

# Révision intra

## 04 - Introduction aux applications Web

Interface

@TODO

## 05 - Architecture d'applications Web

### Patrons de communications

- ▶ Qu'est-ce que les patrons de communications suivant :
  - ▶ Requête-réponse (*request-response*)
  - ▶ Unidirectionel (*one way*)
  - ▶ Publier s'abonner (*publish-subscribe*)
- ▶ Quel est le patron de communication approprié pour une application de type chat? Pourquoi?
- ▶ Quel est l'un des avantages d'un patron de communication asynchrone par rapport à un patron synchrone?



# Révision intra

## 05 - Architecture d'applications Web

### Routage de méthode

#	Expression régulière	Méthode HTTP
1	<code>^\\/\$</code>	GET
2	<code>^\\/ressource\$</code>	GET
3	<code>^\\/ressource\$</code>	POST
4	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	GET
5	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	DELETE
6	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)/images\$</code>	GET

Quelle sera la route pour cette requête :

GET /

# Révision intra

## 05 - Architecture d'applications Web

### Routage de méthode

#	Expression régulière	Méthode HTTP
1	<code>^\\/\$</code>	GET
2	<code>^\\/ressource\$</code>	GET
3	<code>^\\/ressource\$</code>	POST
4	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	GET
5	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	DELETE
6	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)/images\$</code>	GET

Quelle sera la route pour cette requête :

GET /ressource/toto

# Révision intra

## 05 - Architecture d'applications Web

### Routage de méthode

#	Expression régulière	Méthode HTTP
1	<code>^\\/\$</code>	GET
2	<code>^\\/ressource\$</code>	GET
3	<code>^\\/ressource\$</code>	POST
4	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	GET
5	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	DELETE
6	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)/images\$</code>	GET

Quelle sera la route pour cette requête :

POST /ressource

# Révision intra

## 05 - Architecture d'applications Web

### Routage de méthode

#	Expression régulière	Méthode HTTP
1	<code>^\\/\$</code>	GET
2	<code>^\\/ressource\$</code>	GET
3	<code>^\\/ressource\$</code>	POST
4	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	GET
5	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	DELETE
6	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)/images\$</code>	GET

Quelle sera la route pour cette requête :

DELETE /ressource

# Révision intra

## 05 - Architecture d'applications Web

### Routage de méthode

#	Expression régulière	Méthode HTTP
1	<code>^\\/\$</code>	GET
2	<code>^\\/ressource\$</code>	GET
3	<code>^\\/ressource\$</code>	POST
4	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	GET
5	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	DELETE
6	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)/images\$</code>	GET

Quelle sera la route pour cette requête :

DELETE /ressource/5

# Révision intra

## 05 - Architecture d'applications Web

### Routage de méthode

#	Expression régulière	Méthode HTTP
1	<code>^\\/\$</code>	GET
2	<code>^\\/ressource\$</code>	GET
3	<code>^\\/ressource\$</code>	POST
4	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	GET
5	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)\$</code>	DELETE
6	<code>^\\/ressource\\/(?P&lt;id&gt;\\d+)/images\$</code>	GET

Quelle sera la route pour cette requête :

GET /ressource/5/images

# Révision intra

## 05 - Architecture d'applications Web

### MVC

- ▶ Le patron de conception MVC divise une application en trois composantes :
  - ▶ Modèle (*model*)
  - ▶ Vue (*view*)
  - ▶ Contrôleur (*controller*)
- ▶ Quelle est la responsabilité de chacun de ces trois composantes?
- ▶ Quelles sont les avantages et désavantages principales de ce patron de conception?

# Révision intra

## 05 - Architecture d'applications Web

### Traitement d'une requête POST

- ▶ Qu'est-ce le paradigme **Post-Redirect-Get**?
- ▶ Quel problème est-ce que ce paradigme tente de résoudre?



## 06 - Tests d'applications Web

Il existe trois niveaux de tests :

1. Tests unitaires
2. Tests fonctionnelles
3. Tests d'intégration

- ▶ Qu'est-ce qui différencie chacun d'entre eux?
- ▶ Qu'est-ce qu'un effet de bord dans le contexte d'un test?
  - ▶ Quels est le ou les types de tests qui peuvent avoir un effet de bord?
  - ▶ Comment est-ce qu'on gère les effets de bords dans un test?

# Révision intra

## 06 - Tests d'applications Web

Une suite de test efficace possède les 4 caractéristiques suivantes :

1. Rapide
  2. Complet
  3. Fiable
  4. Hermétique
- ▶ Dans quel cas est-ce qu'un test n'est pas fiable?
  - ▶ Quel est le lien entre des tests hermétiques et les effets de bords?
  - ▶ Rouler les tests de manière aléatoire permet de valider quelle(s) caractéristiques?

## 06 - Tests d'applications Web

### Tests de performance

- ▶ Qu'est-ce qu'un test de performance dans le contexte d'une application Web?
- ▶ Qu'est-ce que la capacité d'une application Web?
- ▶ Quelle est l'utilité de faire des tests de performances sur une application Web?

# Révision intra

## 07 - MVC - Modèle

### Sérialisation des données

Modéliser en JSON et XML les informations suivants :

- ▶ Prénom : Jean-Philippe
- ▶ Nom : Caissy
- ▶ Chargé de cours : oui
- ▶ Étudiant : non
- ▶ Adresse :
  - ▶ Adresse : 201, Président Kennedy
  - ▶ Ville : Montréal
  - ▶ Code postal: H2X 3Y7
- ▶ Téléphones :
  1. 514-555-1234
  2. 514-123-3214

# Révision intra

## 07 - MVC - Modèle

### Sérialisation des données

Est-ce que ces deux éléments JSON sont les mêmes? Pourquoi?

```
{ "name" : "Coffee Extra Coarse",  
  "price" : 299,  
  "id" : 1248,  
  "in_stock" : true }
```

et

```
{ "id" : 1248,  
  "in_stock" : true,  
  "name" : "Coffee Extra Coarse",  
  "price" : 299 }
```

# Révision intra

## 07 - MVC - Modèle

### Sérialisation des données

Est-ce que ces deux éléments JSON sont les mêmes? Pourquoi?

```
{  
  "name" : "Coffee Extra Coarse", "images":  
    ["image-1", "image-2", "image-3"]  
}
```

et

```
{  
  "name" : "Coffee Extra Coarse",  
  "images": ["image-3", "image-2", "image-1"]  
}
```

# Révision intra

## 07 - MVC - Modèle

### Base de données

- ▶ Qu'est-ce qu'une base de donnée relationnelle?
- ▶ Quelle est la différence entre une base de donnée relationnelle et une base de donnée de type stockage de documents?
- ▶ Est-ce que les bases de données de type clé-valeur supportent les jointures de données? Pourquoi?

# Révision intra

## 08 - MVC - Modèle (suite)

### Active Record

Le patron de conception Active Record expose une approche pour accéder aux données d'une BD. Les principales fonctionnalités sont les suivantes :

1. Récupération d'un objet ou de plusieurs objet
2. Insertion d'une nouvel objet
3. Modification d'un objet existant
4. Suppression d'un objet

Dans le cadre de ce patron de conception, que représente :

- ▶ Une table
- ▶ La colonne d'une table
- ▶ Un enregistrement d'une table



# Révision intra

## 08 - MVC - Modèle (suite)

### Object Relation Manager (ORM)

Un ORM est composé de ces 3 couches :

1. L'abstraction d'accès à la base de donnée
2. La gestion bi-directionnel des données entre la persistance (BD) et la représentation mémoire (couche du domaine d'affaire)
3. Le patron Active Record qui expose en objet les données

Quelle(s) couche(s) est responsable de :

- ▶ `poll = Poll()`  
`poll.name = "Mon premier sondage"`
- ▶ Lancer la requête SQL à la base de donnée
- ▶ La méthode `Poll.get_by_id`

# Révision intra

## 08 - MVC - Modèle (suite)

### Object Relation Manager (ORM)

Que représente ce modèle Peewee?

```
from peewee import *
```

```
class Product(Model):  
    id = AutoField(primary_key=True)  
    name = CharField(unique=True)  
    price = DecimalField(constraints=[Check('price <  
↪ 10000')])  
    created = DateTimeField(  
        constraints=[  
            SQL("DEFAULT (datetime('now'))")  
        ]  
    )
```

# Révision intra

## 08 - MVC - Modèle (suite)

### Object Relation Manager (ORM)

Après avoir roulé ce code, que va contenir les tables poll et choice?

```
from datetime import datetime
```

```
Poll.create(name="Mon premier sondage",
```

```
    ↪ date=datetime.now())
```

```
Choice.create(choice="Premier choix", poll=poll)
```

```
Choice.create(choice="Deuxième choix", poll=poll)
```

```
choice = Choice.fist()
```

```
choice.name = "Mon vrai premier choxi"
```

```
choice.save()
```

# Révision intra

## 09 - MVC - Contrôleur

### Engin de templating

- ▶ Qu'est-ce qu'un engin de templating HTML?
- ▶ Quels sont 3 avantages d'utiliser un engin de templating pour générer des pages HTML?

# Révision intra

## 09 - MVC - Contrôleur

### i18n et l10n

- ▶ Quel est l'objectif principal de l'internalisation et la localisation d'une application?
- ▶ Quelles données peuvent être localisées?
- ▶ Que représentent les paramètres régionaux suivants :
  - ▶ fr\_CA
  - ▶ fr\_FR
  - ▶ en\_US
- ▶ Comment peut-on fournir la localisation à une application Web?

# Révision intra

## 10 - API Web

API Web : Une interface de programmation exposé à travers une application web

L'interface peut être exposé publiquement et accessible à tous, ou être derrière un système d'authentification

- ▶ Qu'est-ce qu'un appel de type RPC (Remote Procedural Call)?
- ▶ Expliquez comment fonctionne un appel de type RPC localement, et sur le serveur distant
- ▶ Quelles sont les différences majeurs entre SOAP et REST?

# Révision intra

## 10 - API Web

### REST

Vrai ou faux (et pourquoi) :

- ▶ L'état d'une connexion est partagée entre les requêtes REST
- ▶ On peut utiliser du XML dans un API REST
- ▶ Le client sait sur quel serveur il s'est connecté et va toujours communiquer avec celui-ci
- ▶ L'exposition d'une API REST en différente couche permet d'y implémenter un système d'authentification
- ▶ `POST /article/?operation=new` est une interface considérée comme RESTful

# Révision intra

## 10 - API Web

### REST

À quoi servent les 4 méthodes HTTP suivantes, et quand sont-elles utilisées?

1. GET
2. DELETE
3. POST
4. PATCH
5. PUT



## 10 - API Web

### REST

- ▶ Pour une API REST, quelle est la différence entre un code HTTP 400 et 422?
- ▶ Dans quelle situation est-ce qu'on retourne un code HTTP 404?
- ▶ Est-ce qu'on peut retourner une erreur de traitement avec un code HTTP 200? Si non, quelle serait un code approprié?

# Révision intra

## 10 - API Web

### REST

@TODO : Exercice bon et mauvais API REST

# Révision intra

## 10 - API Web

### REST

Vous devez modéliser une API REST pour une application de type Twitter.

Il existe deux ressources principales :

- ▶ Tweets
- ▶ Utilisateurs

Modélisez une API REST qui correspond aux requêtes suivantes :

- ▶ Afficher tous les tweets
- ▶ Publier un nouveau tweet
- ▶ Modifier un tweet
- ▶ Afficher les réponses d'un tweet
- ▶ Afficher les gens qui suivent un utilisateur
- ▶ Suivre un utilisateur