

LANGAGE DE PROGRAMMATION PYTHON

django framework web python

Formateur : IBRAHIM M. S.

ibrahim.ms@gmail.com

• formation • informatique • programmation •

IBRAHIM M.S. (IMS)

• formation • informatique • programmation •

• paris • 19.06.24 → 19.06.26 •

• paris • 19.06.24 → 19.06.26 •

1 / 20

Introduction à Django

Historique

2003 : début du dev
Adrian Holovaty & Simon Willison
journal local de Lawrence
08.05 : licence BSD
06.08 : Django Software Foundation

Promesses

développement rapide
développement simple
minimum de html/css
absence de javascript
python est suffisant
non orienté développeur web

Technologies

Python
HTML/CSS
JavaScript
XML DOM JSON
SQL POO (ORM)

Paramétrés par défaut

modèles et templates
gestion des requêtes sql
paramètres des méthodes
besoins classiques/communs
on peut tout personnaliser
mais on ne devrait pas



Atouts

modèles, vues et templates génériques

- ▶ traitement des situations les plus courantes
- ▶ pagination, date, mapping objets/requêtes sql

système d'authentification/gestion des users
création de pages statiques/dynamiques

- ▶ efficace pour des sites de petites tailles et simples
- ▶ peut générer et gérer des sites de très grandes tailles
- ▶ se connecte à tout type de base de données

communauté et documentation mais en anglais
mode débogage

- ▶ debug aisément du fait du mécanisme try/catch python

Faiblesses

extremement rigide par défaut

- ▶ personnalisation longue et fastidieuse
- ▶ évaluer les autres solutions le cas échéant

intégration AJAX côté client web – combiner avec :

- ▶ MooTools, Prototype, MochiKit, jQuery, dojo, ..

Exemples : websites

eventbrite
instagram
pinterest
youtube
dropbox
spotify
disqus
nasa

Alternatives

web2py
turbogears
cherrypy
zope
flask
twisted

Chapitre : framework web python

1 Présentation

Introduction
Le framework django
Schéma de fonctionnement
Composants du framework

2 Environnement de travail

Installation de Python
Gestionnaire de package
Environnements virtuels
TP - Installation de django

3 Exemple d'illustration

Classes
Navigation

• formation • informatique • programmation •

IBRAHIM M.S. (IMS)

• formation • informatique • programmation •

• paris • 19.06.24 → 19.06.26 •

2 / 20

• formation • informatique • programmation •

IBRAHIM M.S. (IMS)

• formation • informatique • programmation •

• paris • 19.06.24 → 19.06.26 •

4 / 20

Le framework Django



Django

framework python de haut niveau
éprouvé gratuit et open source
développement rapide et simple
de sites web statiques ou dynamiques
basé sur protocole de communication
serveur web / application web WSGI

Framework

ensemble cohérent d'outils
permettant un développement
plus rapide/simple de produits
particuliers à une/des technologie(s)
ou à un domaine d'application ciblé
utilité : gain de temps et d'efforts

IBRAHIM M.S. (IMS)

• formation • informatique • programmation • ● paris ● 19.06.24 → 19.06.26 ● 5 / 20

WSGI : Web Server Gateway Interface

spécification/protocole
simple et universelle
d'interface de communication
► server et application web
► server et framework web

Composants

models views templates
routeur url - serveur web
système d'authentification
génération de formulaires
requête sql et interface db
gestion des utilisateurs
outil de déploiement sur serveur
interface d'administration

Composants du framework



server web

django intègre un serveur web
qui reçoit les requêtes html
mais ne les traite pas lui-même
les redirige vers le routeur d'url

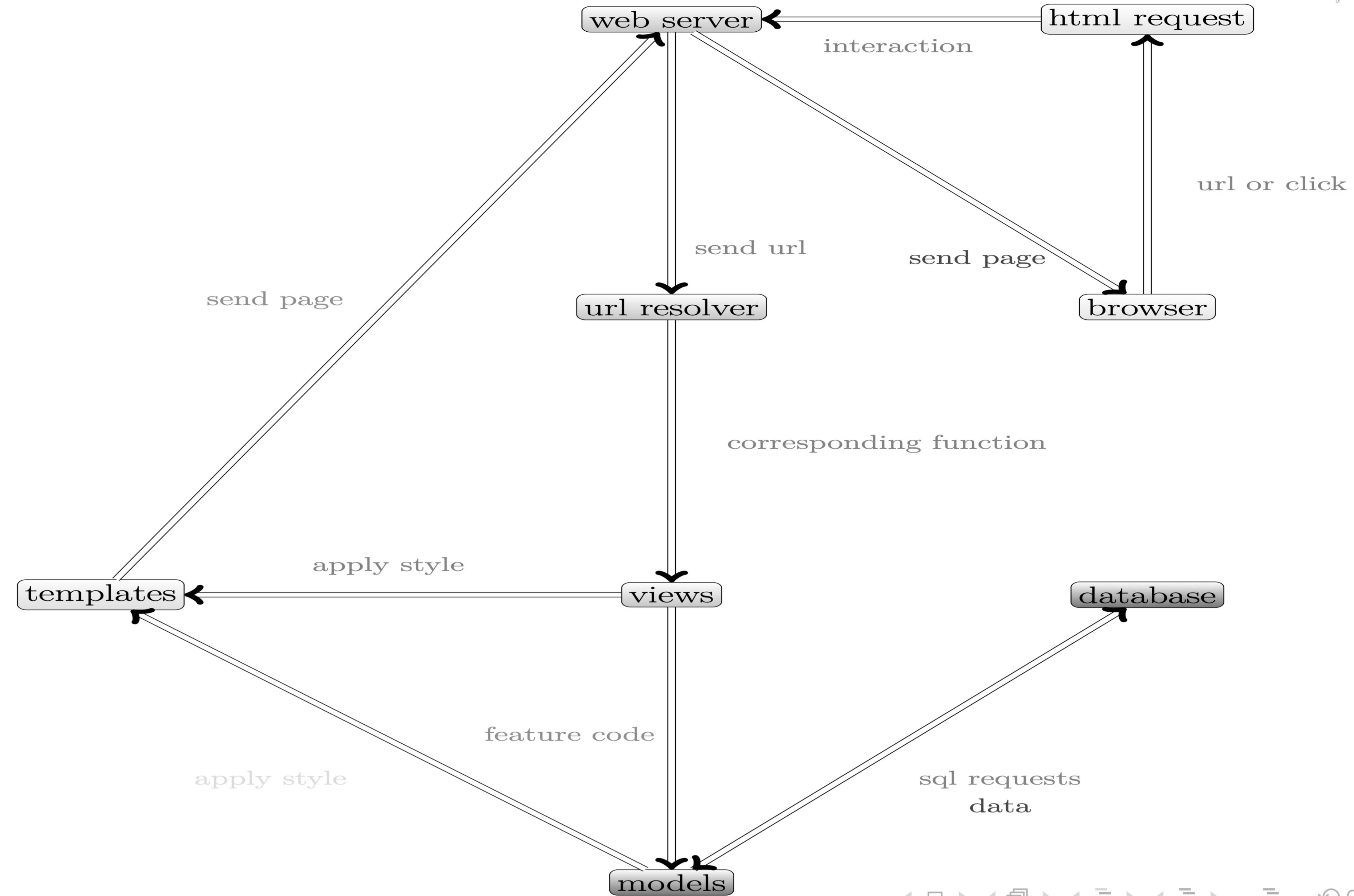
routeur d'url

celui-ci a une liste de patterns
pattern détermine la fonctionnalité
demandée par l'utilisateur
dès qu'un matching est trouvé
envoie la requête à la fonction
ceux-ci doivent être cohérents
pas de correspondance url/file

views

cette fonction est appelée vue
c'est elle qui fait le travail
► recherche de data en base
► mise-à-jour d'informations
► ajout/suppression de données
► affichage des produits/clients
se charge de valider la faisabilité
► existence d'utilisateurs
► vérification des droits d'accès
► vérification droits d'écriture
► cohérence données envoyées
► conflits à l'écriture en base
génère réponse à transmettre
à django qui redirige au serveur
puis au browser pour affichage

Fonctionnement du framework



Composants du framework



models

représentation abstraite des données
manipulées par l'application web
représentation par des objets python
des données présentes en database
mapping tables sql / classes python
mapping data sql / objets python
► une classe par table
► un champs par attribut
► une entrée par objet
application de méthodes sur objets
pas de manipulation requêtes sql
possibilité d'injecter des requêtes

templates

mise en forme des informations
par application de styles
html css javascript
structures de contrôle

database

pas de manipulation directe
mais via l'orm, moins de travail
en dev db.sqlite3 possible
type quelconque de database
multiples configurations

IBRAHIM M.S. (IMS)

• formation • informatique • programmation • ● paris ● 19.06.24 → 19.06.26 ● 7 / 20

IBRAHIM M.S. (IMS)

• formation • informatique • programmation • ● paris ● 19.06.24 → 19.06.26 ● 6 / 20

8 / 20

Installation de Python

Windows/Mac – binaire officiel : www.python.org/downloads

Linux : debian et dérivés

```
sudo apt-get install python3 ipython3  
idle3 python3-pip python3-pyqt4  
ipython3-notebook
```



```
sudo apt-get install python3-numpy  
python3-scipy python3-matplotlib
```



```
idle3
```

Distribution complète^a: anaconda – www.continuum.io/downloads

a. epd, python(x,y), winpython, pythonanywhere, activepython, intel's distribution



pip : le gestionnaire de packages

Présentation

ajout de modules Python dont on peut choisir les versions
installation préalable des modules/librairies nécessaires
pas d'installation partielle : dépendances téléchargées avant

Une solution

un système de gestion de paquets est disponible pour cela
un dépôt central répertoriant les modules disponibles
sauvegarde/restauration d'une configuration Python

Librairies avec des extensions en C

extensions plus performantes que celle en Python
lancement de la compilation de ces dernières
ici des dépendances peuvent faire défaut

Problématiques

plusieurs projets en parallèle
conflits entre différents projets
différentes versions des modules

conflits avec modules systèmes
résolution des dépendances entre projets
outils non python potentiellement utiles

Intégrité du système

basculement projets contraignant
perte de temps configuration

perte de temps réinstallation
environnement instable ou inopérant

Gestionnaire de packages

l'outil pip est fourni avec Python

un dépôt central est mis à jour

Environnements virtuels

l'outil virtualenv est fourni avec Python

chaque environnement est indépendant

Outils C, C++, Fortran,..

conda inclu dans la distribution anaconda permet bien plus de chose

python gère bien les outils non-Python
programmes plus performants
programmes existants déjà

pas de réécriture des outils
faire cohabiter différentes technologies
l'outil conda permet de faciliter cela

IBRAHIM M.S. (IMS)

formation informatique programmation

paris 19.06.24 → 19.06.26 10 / 20

Commandes fréquentes

Principales commandes pip

pip help
pip help command
pip search packageNameOrSubject
pip install packageName
pip install packageName==versionNumber
pip uninstall packageName
pip list
pip list -O
pip install -U packageName

(outdated)
(upgrade)

Upgrader tous les packages

```
pip freeze --local | grep -v '^-\e' |  
    cut -d= -f1 | xargs -n1 pip install -U
```

IBRAHIM M.S. (IMS)

formation informatique programmation

paris 19.06.24 → 19.06.26 12 / 20

Finalité

- disposer de différents instances de Python
- disposer de différentes versions de Python
- disposer d'un environnement de tests de modules
- disposer d'une copie de l'environnement de production
- pas d'interférence sur la version du système
- travailler en parallèle sur différents projets

Caractéristiques

ces instances n'ont pas d'incidence les unes sur les autres

- modules installés potentiellement différents
- dans des versions potentiellement différentes
- disposant de modules précis en versions spécifiques
- isolées de l'instance système donc sans problème de stabilité

conda as a virtual environments manager

conda as virtual environment manager

conda environment manager is cross-platform with packages platform specific

no compilation

no root/admin need

c, c++, ... libs

conda as virtual environment manager : simple commands

`conda env list`

`conda create --name newEnvName`

`conda env remove --name toBeDeletedEnvName`

`conda install --name existingEnvName [packageName]`

conda as virtual environment manager : select current environment

`source activate existingEnvName`

`source deactivate`

conda as virtual environment manager : with specific python version

```
conda create --name newEnvName python=2.7.13
```

conda as virtual environment manager : using file for configuration

```
conda create --name newEnvName --file specificConfig.txt
conda install --name existingEnvName packagesList.txt
```

conda as virtual environment manager : convert package to platforms

```
conda convert myPythonPackage-1.0-py33.tar.bz2 -p win-64
conda convert --platform=all pathToPackageToConvert
```

conda as virtual environment manager : cloning is not crossplatform

```
conda create --name myclone --clone myenv
conda list --explicit > specificConfig.txt
conda install --name existingEnv --file specificConfig.txt
conda create --name newClonedEnv --file specificConfig.txt
```

TP — Installation de django et vérification

1/2

0 - tools installation

```
○ims@sh: apt install python3-pip
○ims@sh: apt install python3-venv
○ims@sh: pip install virtualenvwrapper
```

1 - new virtual python environnement

```
○ims@sh: python3 -m venv new_env_name
○ims@sh: virtualenv -p python3 new_env_name
○ims@sh: mkvirtualenv new_env_name
○ims@sh: conda create --name new_env_name
```

2 - activate virtual python environnement

```
○ims@sh: source env_name/bin/activate
○ims@sh: workon env_name
○ims@sh: conda activate env_name
```

(virtualenvwrapper) (conda)

3 - work on django project

```

ims@sh: conda list
ims@sh: conda install django ; conda list
ims@sh: py -c "import django ; print(django.get_version())"
ims@sh: mkdir djg_projs
ims@sh: cd djg_projs
ims@sh: django-admin startproject dj_verif
ims@sh: cd dj_verif
ims@sh: ./manage.py runserver 12345
ims@sh: kill `ps aux | grep '12345' | awk '{print $2}'` 
ims@sh: browser localhost:12345

```

2 - deactivate python environnement

```

ims@sh: deactivate
ims@sh: conda deactivate
(ims@sh: conda activate)

```

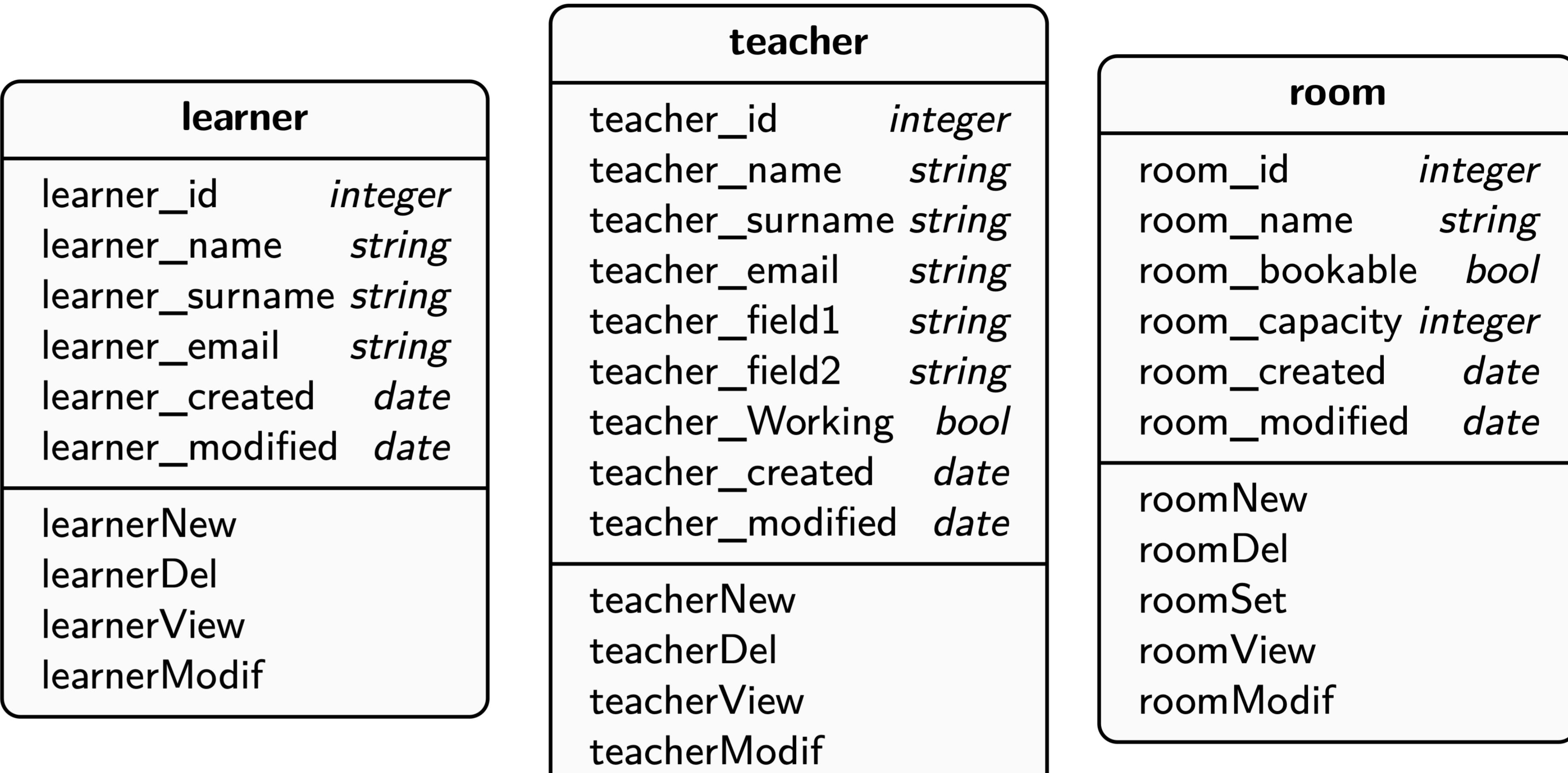
IBRAHIM M.S. (IMS)

• formation • informatique • programmation • paris • 19.06.24 → 19.06.26 • 17 / 20



python

Programming Language

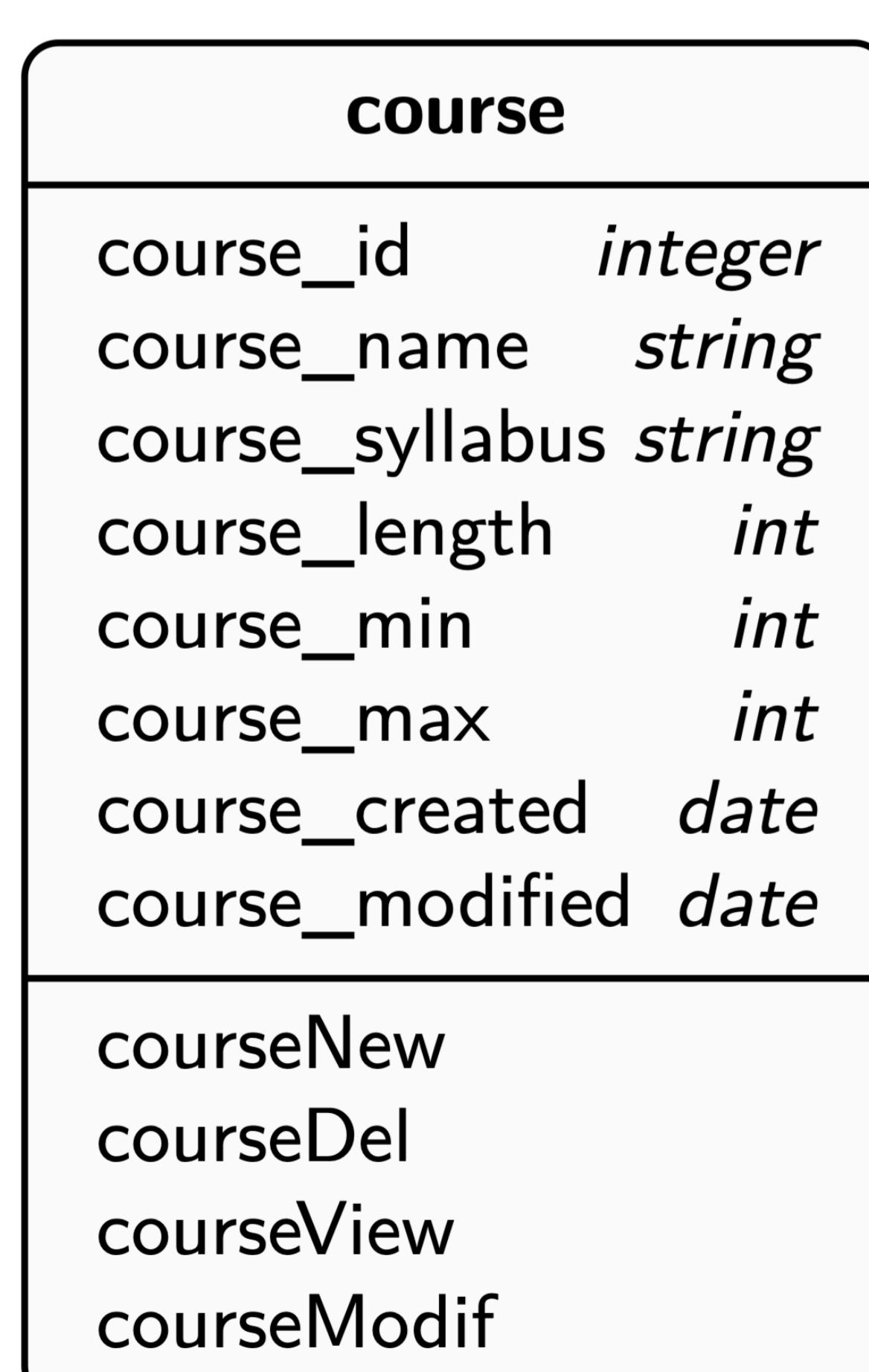


IBRAHIM M.S. (IMS) • formation • informatique • programmation • paris • 19.06.24 → 19.06.26 • 18 / 20

training center session management – classes



Programming Language



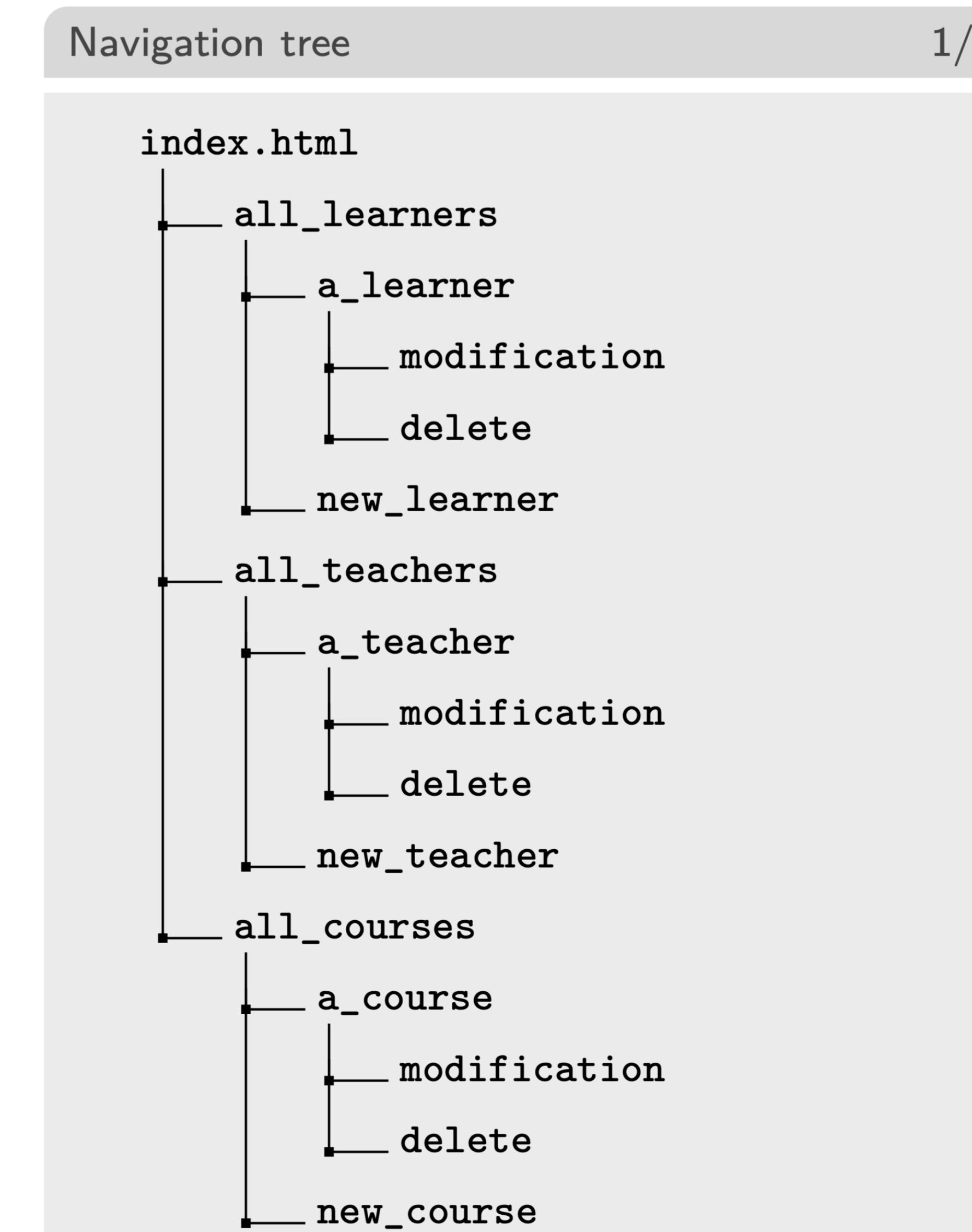
IBRAHIM M.S. (IMS)

• formation • informatique • programmation • paris • 19.06.24 → 19.06.26 • 19 / 20

training center session management – pages navigation



Programming Language

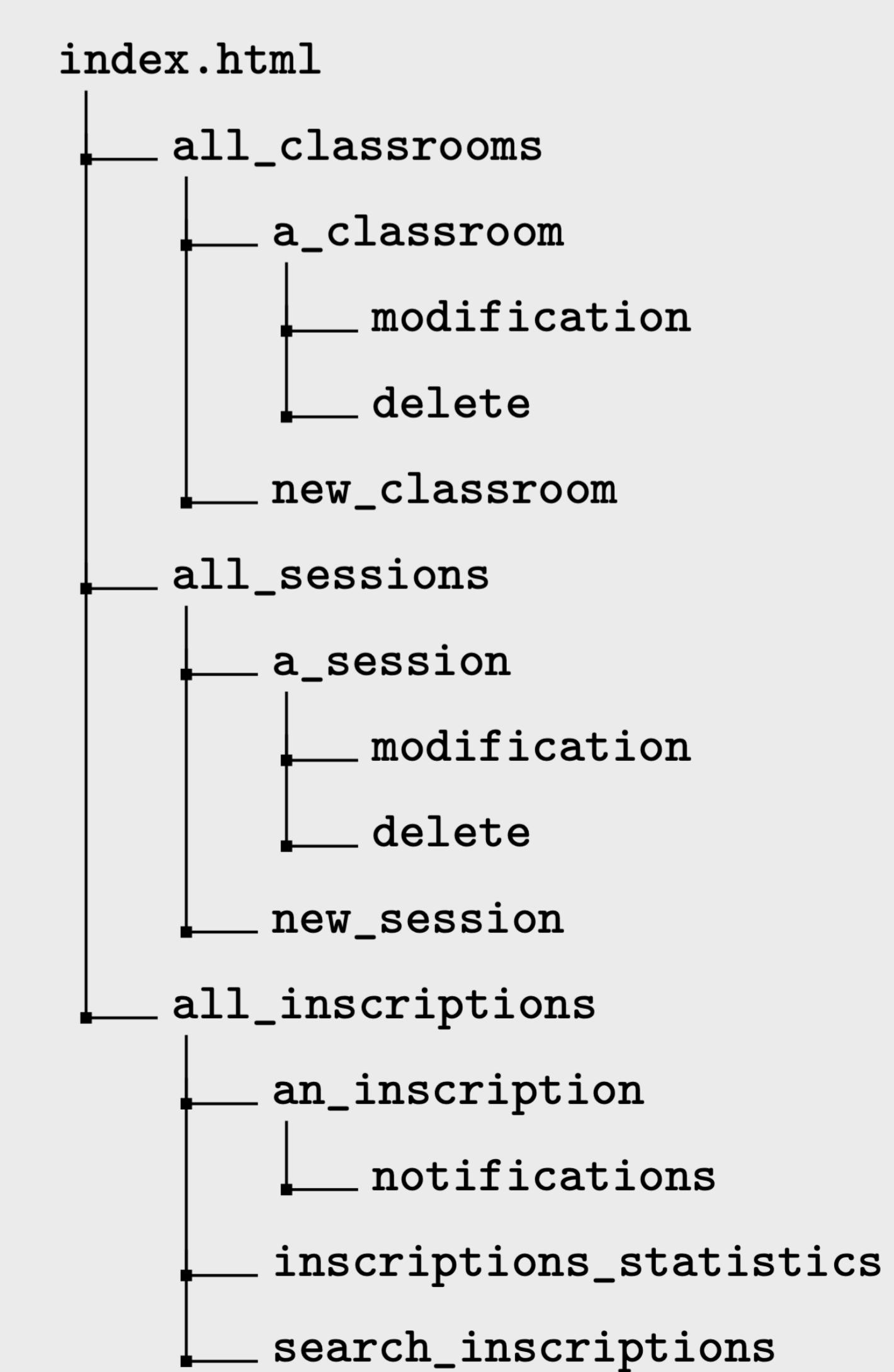


IBRAHIM M.S. (IMS)

• formation • informatique • programmation • paris • 19.06.24 → 19.06.26 • 20 / 20



Programming Language



IBRAHIM M.S. (IMS) • formation • informatique • programmation • paris • 19.06.24 → 19.06.26 • 20 / 20