PROJET INFORMATIQUE 2A

Pépin Rémi, Ensai, 2019

LE PLAN

1. Les grands enjeux du projet
    - Organisation/planning
    - Echéances
    - Groupes

---

2. L'analyse fonctionnelle
    - Définition
    - Diagrammes UML
3. L'architecture logiciel
    - Définition
    - Separation of concern
4. La couche data access object
    - Comment fonctionne la mémoire de votre ordinateur
    - Représentation d'un objet python
    - La DAO

GRANDS ENJEUX DU PROJET

ORGANISATION DU PROJET

- 4 cours d’1h30
- 4 TP de 3h
- 2 créneaux de 3h libéré pour le projet
- 7 séances de suivi
- Une période d'immersion de 3 jours
- Du travail perso et en équipe

Prévoir au moins 3h de travail perso chaque semaine

ECHÉANCES

- 27/09 - Rapport d’analyse
- 15/11 - Rapport final + code
- 27/11 - Soutenance

Pour les attendus allez voir la notice élève

LES GROUPES

- 4-5 par groupe : "gros" groupes
- Imposé (un peu comme dans le monde pro)
- Homogène (tirage aléatoire stratifié)
- Pas d'élève isolé
- Un seul rôle à part **chef de projet**
- Soyez polyvalent

L'ENCADRANT

- Travail dans l'informatique en entreprise
- Double rôle
  - "Client" : vous demande de réaliser une application
  - "Mentor" : vous passer sess connaissances, son expériences (=/= prof)
- Débloque si problème technique
- Si problème dans le groupe venez me voir ou Romaric Gaudel

ANALYSE FONCTIONNELLE

C'EST QUOI L'ANALYSE FONCTIONNELLE ?

- Première étape de tout projet
- Déterminer les **fonctions**, **acteurs** du produit pour répondre aux **besoins** du client
- Faire des diagrammes pour échanger avec le client
- Prioriser les fonctions

LES DIAGRAMMES UML

- Cours de 1A
- UML 2.5, Pascal Roques, Eyrolles, Mémento

ARCHITECTURE LOGICIEL

C'EST QUOI L'ARCHITECTURE LOGICIEL ?

- Le pendant de l'analyse fonctionnel
- Maintenant que l'on a le quoi, on détermine le comment
- On dessine notre application

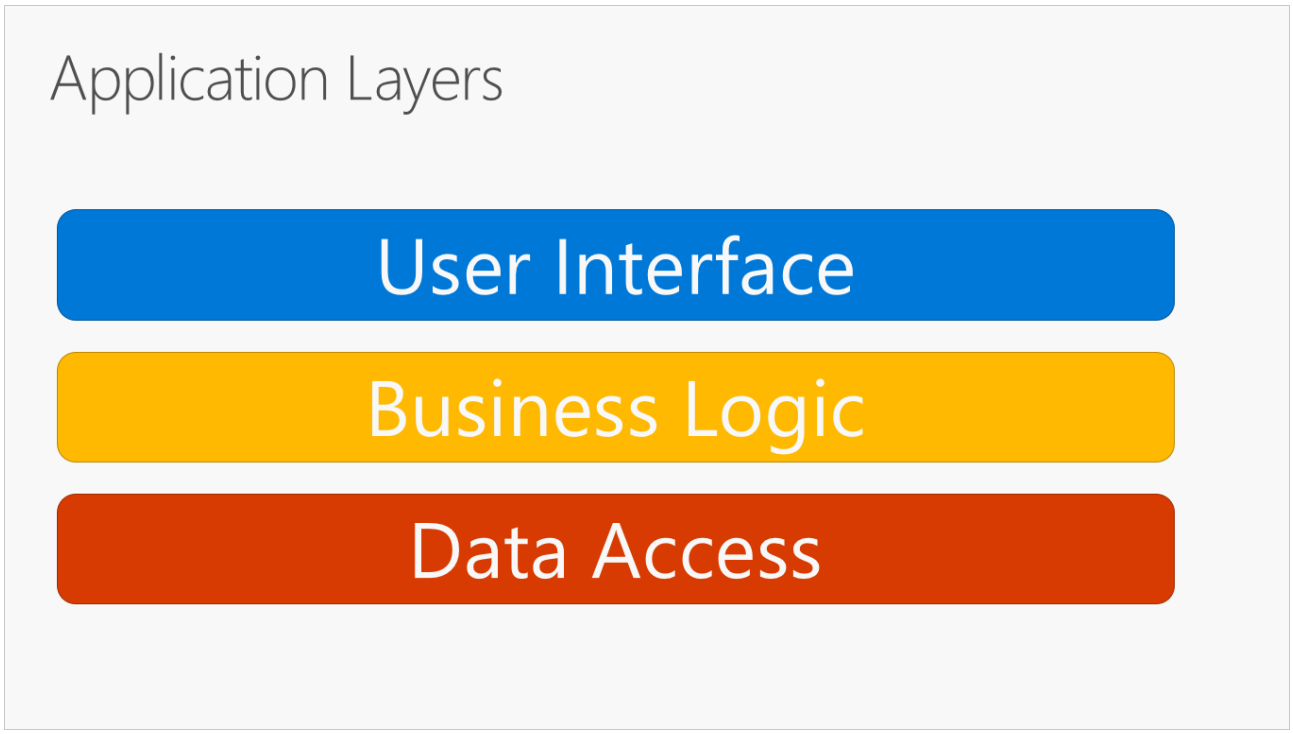
POURQUOI C'EST IMPORTANT : PARALLÈLE AVEC L'ARCHITECTURE

POURQUOI C'EST IMPORTANT : PARALLÈLE AVEC L'ARCHITECTURE

- Pièces, l'installation électrique, l'eau, le gaz, contraintes législatives, s'adapter au terrain
- Besoin de réfléchir comment il faut agencer tout ça dès le début
- **Ce n'est pas du temps perdu !**

UN GRAND PRINCIPE : SEPARATION OF CONCERN

UN GRAND PRINCIPE : SEPARATION OF CONCERN



SEPARATION OF CONCERN : EXPLICATION

- Décomposition d'un programme en module simples **cohérent**
- Les modules exposent des méthodes utilisable par d'autres modules
- Chaque module doit être une boîte noire pour les autres (ce qui importe c'est les entrées/sorties)
- Si on garde les mêmes entrées/sorties on peut changer un module sans risque

SEPARATION OF CONCERN : POURQUOI LE RESPECTER ?

- Travail en groupe
- Lisibilité du code
- Débugage

Limiter les risques d'erreur quand on modifie le code (éviter l'assiette de spaghetti) 

UN GRAND PRINCIPE : SEPARATION OF CONCERN

Un mantra :

Faible couplage, forte cohésion

EXEMPLE DE COUCHES QUE VOUS ALLER MANIPULER

- Persistance
- Data access object
- Métier
- Affichage

DATA ACCESS OBJECT (DAO)

QUESTION

Quels sont les composants "important" de votre ordinateur et leur utilité ?

UN ORDINATEUR (EN GROS)

- Un processeur (CPU) : fait UNIQUEMENT du calcul
- La mémoire RAM : mémoire volatile rapide
- Disque dur (HDD, SSD) : mémoire longue durée
- Carte graphique : unité de calcul spécialisée

QUESTION

A votre avis, où sont stockés les **variables** de votre programme python ?

Dans la RAM -> tout disparait quand on éteint le programme 😕

Vous avez un programme "one shot"

COMMENT RÉSOUDRE CE PROBLÈME ?

UNE QUESTION À CE POSER

C'est quoi une variable python ?

- une référence (le nom de la variable)
- Un objet associé (sa valeur)

UNE QUESTION QUI EN DÉCOULE

C'est quoi un objet python ?

- Des attributs (qui peuvent être eux même des objets)
- Des méthodes

POUR RÉSUMER

On veut sauvegarder des couples clefs-valeur, avec des valeurs qui peuvent être elle-même constituée de couples clef-valeur

On veut sauvegarder un arbre 

COMMENT FAIRE ÇA ?

- Écrire nos données sur le disque dur dans un fichier (serialisation, format json, xml ...) 
- Utiliser une base de données (tables + relations) 

CHECKPOINT

Rappel du probleme : persister des données

- **Quoi ?** Données, peuvent être représentées par des arbres
- **Où ?** En base, ou dans un fichier sous forme sérialisée
- **Comment ? En utilisant des Data acess objects (DAO)**

C'EST QUOI UNE DAO ?

C'EST QUOI UNE DAO ?

- Classe technique 🪛
- Une classe DAO / object métier 1:1
- Expose des méthodes pour communiquer avec la couche de persistance 💽

QUELLES MÉTHODES EXPOSER ?

CRUD

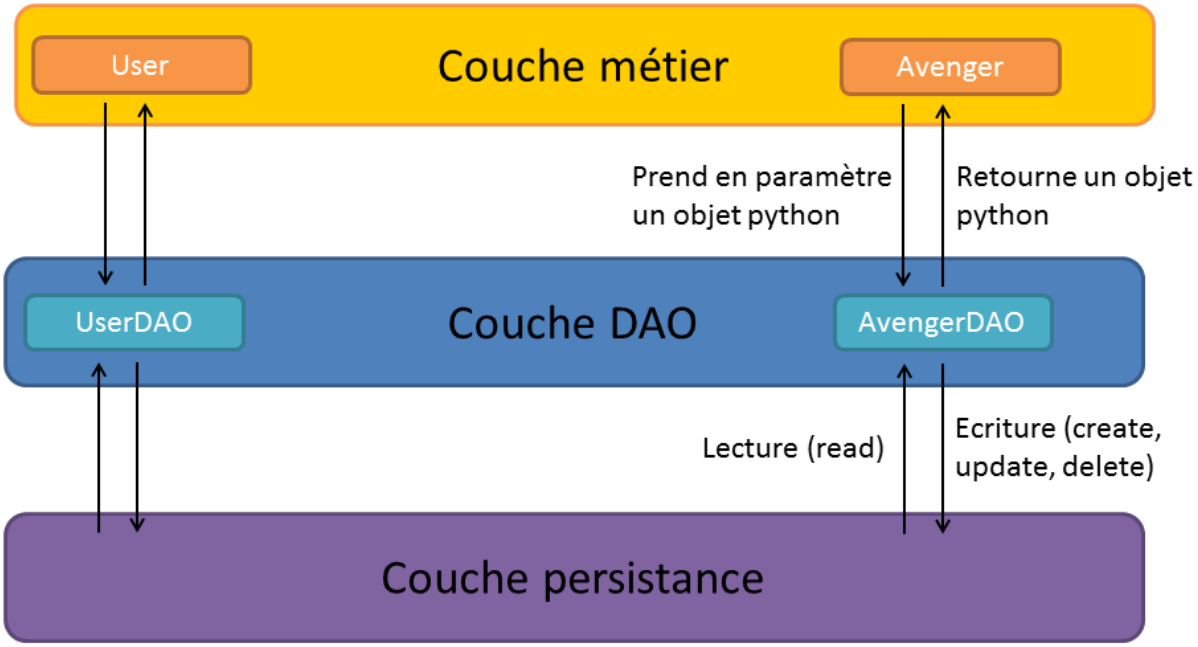
- Create
- Read
- Update
- Delete

L'INTÉRÊT D'UNE CLASSE À PART

Separation of concern

- Classe "jetable" 
- Modifiable sans risque 
- Parallélisation du travail 

PETIT RECAP



UN PETIT EXEMPLE : AVENGERDAO

```
class AvengerDAO :
    def create(self, avenger):
    """
    Persiste une instance d'avenger en base. Retourne l'avenger en
    mise à jour de son id en base
    """
        cur = connection.cursor()
        try:
            cur.execute(
                "INSERT INTO avengers (name, alias, power) VALUES
                , (avenger.name, avenger.alias, avenger.power))
            avenger.id = cur.fetchone()[0]
            # la transaction est enregistrée en base
            connection.commit()
        except psycopg2.Error as error:
```

UN PETIT EXEMPLE : AVENGERDAO

```
def read(self, id):
    with connection.cursor() as cur:
        row = cur.execute(
            "select name, alias, power from avengers where id=%s;"
        row = cur.fetchone()
        if row:
            return Avenger(name=row[0], alias=row[1],power=row[2],
        else:
            return None
```

```
def update(self, avenger):
    with connection.cursor() as cur:
        try:
            cur.execute(
                "update avengers set name=%s, alias=%s, power=%s wh
                ,(avenger.name, avenger.alias, avenger.power, aven
            connection.commit()
        except psycopg2.Error as error:
            connection.rollback()
            raise error
```


UN PETIT EXEMPLE : AVENGERDAO

```
def delete(self, avenger):
    with connection.cursor() as cur:
        try:
            cur.execute("delete from avengers where id=%s", (aveng
            connection.commit()
        except psycopg2.Error as error:
            connection.rollback()
            raise error
```