

Développez un programme logiciel en Python

OpenClassRooms - Parcours Python - Projet 4

Bérenger Ossété Gombé

22 mai 2022

Sommaire

- 1 Introduction
- 2 Méthodes et outils
 - Agilité et tests
 - Style et conventions
 - Bibliothèques Python
- 3 Modélisation objet
 - Conception et modélisation
 - Domaine métier
 - Contrôle de l'application
 - Interface utilisateur
- 4 Démonstration
- 5 Conclusion

Rappel du sommaire

- 1 Introduction
- 2 Méthodes et outils
- 3 Modélisation objet
- 4 Démonstration
- 5 Conclusion

Présentation



Bérenger Ossété Gombé

- Master 1 en informatique (2016-2017)
 - Université de Franche-Comté (UFR-ST)
 - Spécialité génie logiciel
- Formation Python chez OpenClassRooms depuis janvier 2022

Le projet : développer une application python

Objectifs

- Produire du code robuste
- Utiliser la POO
- Structurer un projet python

Contexte fictif : le club d'échecs



Figure – Logo du club d'échecs

Personnages

- Nous incarnons un développeur indépendant
- Quelques membres du club d'échecs
 - Élie, notre amie
 - Édouard, l'organisateur
 - Charlie, l'assistant informatique

Contexte fictif : le club d'échecs

Problématique

- «*Nous utilisons actuellement une **application en ligne** pour nous aider à gérer nos tournois d'échecs hebdomadaires. Malheureusement, cette application nous a déçus par le passé. Elle **tombe souvent en panne**, ce qui signifie que les matchs sont retardés.*» - Spécification technique
- «*Après que nous avons eu terminé le premier tour, **Internet a cessé de fonctionner** et le directeur du tournoi n'a pas pu entrer les scores sur le site des résultats*» - Témoignage d'Élie

Exigences fonctionnelles

- Gestion des tournois
- Gestion des classements
- Génération de rapports pouvant être exportés¹
- Fonctionnement hors-ligne
- Interaction utilisateur *via* une interface console
- Persistance *via* une base de données

1. Il faut permettre cette future fonctionnalité.

Exigences non-fonctionnelles²

■ **Maintenabilité**

- «Le code doit être aussi propre et maintenable que possible pour éviter les bugs.»

■ **Fiabilité**

- «Le code doit être aussi propre et maintenable que possible pour éviter les bugs.»

■ **Portabilité** (GNU/Linux, Windows, MacOS)

- «Le programme devrait fonctionner sous Windows, Mac ou Linux»

■ **Utilisabilité**

- «Tant que le programme affiche les résultats du tournoi proprement, nous serons heureux !»

Rappel du sommaire

1 Introduction

2 Méthodes et outils

- Agilité et tests
- Style et conventions
- Bibliothèques Python

3 Modélisation objet

4 Démonstration

5 Conclusion

Mise en place du backlog

- Utilisation d'un tableau Kanban *via* kanboard³
- Découpage des *user stories* avec critères d'acceptations

Pourquoi tester ?


- La fiabilité est un enjeu du projet
 - «*Le code doit être aussi propre et maintenable que possible pour éviter les bugs.*»⁴
- L'algorithme du tournoi Suisse doit être fiable car au cœur du domaine métier

Outils de tests

- Tests unitaires *via* PyTest⁵
- Tests d'acceptations *via* Behave (cucumber)⁶

4. D'après la spécification technique.

5. <https://docs.pytest.org/en/7.1.x/>

6. <https://behave.readthedocs.io/en/stable/> 

Style et conventions

PEP 8

- Vérification de la conformité du programme avec PEP 8 *via* Flake8⁷

Google et OpenStack

- Google Python Style Guide⁸
- OpenStack Style Guidelines⁹
- Utilisation de l'extension hacking¹⁰ de Flake8

7. <https://flake8.pycqa.org/en/latest/>

8. <https://google.github.io/styleguide/pyguide.html>

9. <https://docs.openstack.org/hacking/latest/user/hacking.html>

10. <https://pypi.org/project/hacking/>

Bibliothèques

- Style et conventions
 - Flake8
 - Flake8-html
 - Hacking
- Tests
 - PyTest
 - Behave
- Base de données
 - TinyDB

Rappel du sommaire

1 Introduction

2 Méthodes et outils

3 Modélisation objet

- Conception et modélisation
- Domaine métier
- Contrôle de l'application
- Interface utilisateur

4 Démonstration

5 Conclusion

Vocabulaire : définitions

Définitions

Vocabulaire : traductions

Traduction en anglais du vocabulaire

Tournoi *tournament*

Classement *ranking*

Rapport *report*

Joueur *player*

Pair de joueur *pair of players*

Un tour *a round*

Résultats *scores*

Match nul *draw*

Coup rapide

Découpage en modules

Architecture du projet

Tournois et classement

Génération de rapports

Persistence

Contrôle de l'application

Interface utilisateur

Rappel du sommaire

- 1 Introduction
- 2 Méthodes et outils
- 3 Modélisation objet
- 4 Démonstration**
- 5 Conclusion

Démonstration

Fonctionnalités clefs

- 1 Création et exécution d'un tournoi
- 2 Mise-à-jour du classement
- 3 Génération d'un rapport
- 4 Sauvegarde et chargement à tout moment

Rappel du sommaire

- 1 Introduction
- 2 Méthodes et outils
- 3 Modélisation objet
- 4 Démonstration
- 5 Conclusion**

Conclusion

Merci de votre attention

1 Introduction

2 Méthodes et outils

- Agilité et tests
- Style et conventions
- Bibliothèques Python

3 Modélisation objet

- Conception et modélisation
- Domaine métier
- Contrôle de l'application
- Interface utilisateur

4 Démonstration

5 Conclusion