

SCALE FOR PROJECT DJANGO - 0 - OOB (/PROJECTS/DJANGO-0-OOB)

You should evaluate 1 student in this team



Git repository

git@vogsphere.42paris.fr:vogsphere/intra-uuid-481b0758-34d0-45da-bd0



Introduction

Nous vous demandons pour le bon déroulement de cette évaluation de respecter les règles suivantes :


- Restez courtois, polis, respectueux et constructifs en toutes situations lors de cet échange. Le lien de confiance entre la communauté 42 et vous en dépend.
- Mettez en évidence auprès de la personne (ou du groupe) notée les dysfonctionnements éventuels du travail rendu, et prenez le temps d'en discuter et d'en débattre.
- Acceptez qu'il puisse y avoir parfois des différences d'interprétation sur les demandes du sujet ou l'étendue des fonctionnalités. Restez ouvert d'esprit face à la vision de l'autre (a-t-il ou elle raison ou tort ?), et notez le plus honnêtement possible. La pédagogie de 42 n'a de sens que si la peer-évaluation est faite sérieusement.

Guidelines

- Vous ne devez évaluer que ce qui se trouve sur le dépôt GiT de rendu de l'étudiant(e) ou du groupe.
- Prenez soin de vérifier que le dépôt GiT est bien celui correspondant à l'étudiant(e) ou au groupe, et au projet.
- Vérifiez méticuleusement qu'aucun alias malicieux n'a été utilisé pour vous induire en erreur et vous faire évaluer autre chose que le contenu du dépôt officiel.
- Tout script censé faciliter l'évaluation fourni par l'un des deux partis doit être rigoureusement vérifié par l'autre parti pour éviter des mauvaises surprises.
- Si l'étudiant(e) correcteur/correctrice n'a pas encore fait ce projet, il est obligatoire pour cet(te) étudiant(e) de lire le sujet en entier avant de commencer cette soutenance.
- Utilisez les flags disponibles sur ce barème pour signaler un rendu vide, un cas de triche, etc. Dans ce cas, l'évaluation est terminée et la note finale est 0 (ou -42 dans le cas spécial de la triche). Toutefois, hors cas de triche, vous êtes encouragés à continuer d'échanger autour du travail effectué (ou non effectué justement) pour identifier les problèmes ayant entraîné cette situation et les éviter pour le prochain rendu.
- Vous devez arrêter de compter les points à partir du premier exercice faux, même si les exercices suivants sont bons.

Attachments

 subject.pdf (<https://cdn.intra.42.fr/pdf/pdf/64990/fr.subject.pdf>)

 d02.tar.gz (<https://cdn.intra.42.fr/document/document/12028/d02.tar.gz>)

Preliminaires

Cette section est dédiée au démarrage de l'évaluation et à la vérification des prérequis. Elle ne rapporte pas de points, mais si quelque-chose est faux à ce stade ou à n'importe quel point de l'évaluation, la note est 0, et un flag peut être coché si nécessaire.

Respect des consignes

- Le dépôt contient le travail de l'étudiant(e) (ou du groupe) noté(e).
- L'étudiant(e) (ou le groupe) noté(e) est capable d'expliquer son travail à n'importe quel moment de l'évaluation.
- Les règles générales et les éventuelles règles propres à la journée sont respectées tout au long de l'évaluation.
- Pour ce projet, vous devez cloner le dépôt Git sur l'ordinateur de la personne évaluée.

 Yes

 No

Formation Python-Django - 0 - OOB

- Si vous trouvez une erreur dans le travail d'un exercice, l'intégralité de cet exercice est faux. Pas de demi-mesure ou de "toute petite erreur de rien du tout", soit le travail est parfait, soit il ne l'est pas. - Le corrigé doit avoir fourni des tests pour chaque fichier python (bloc "if __name__ == '__main__':") sauf en cas d'indication contraire dans le sujet. - L'affichage des pages doit être maîtrisé : pas de comportement bizarres, de caractères spéciaux (dont accentués) non gérés dans l'affichage, etc. Du propre et rien que du propre, sinon voir au dessus.

ex00: À la conquête de la Silicon Valley!

- Le programme render.py gère correctement le nombre d'arguments (un message d'erreur si le programme reçoit aucun argument ou plus d'un).
- Le programme gère les cas où le nom de fichier passé en paramètre ne possède pas l'extension .template (afficher un message d'erreur).
- Le programme écrit bien le résultat dans un fichier dont le nom est identique au nom du .template passé en paramètre, avec pour seule différence une extension .html à la place d'une extension .template.
- Le programme convertit bien le template rendu en un fichier HTML, les motifs sont bien remplacés par les valeurs du fichier settings.py et les informations demandées dans le sujet sont bien présentes dans le fichier HTML obtenu.
- Modifiez les valeurs contenues dans le fichier settings.py, vérifiez que ces changements sont bien pris en compte lorsqu'on lance le programme à nouveau.
- Les informations demandées (Nom, prénom, titre de la page, âge, profession) doivent être présentes dans le CV produit.
- L'exemple du sujet est reproductible.

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

 Yes

 No

ex01: Startup innovante cherche stagiaire. 10 ans d'expérience requis.

- Votre classe Intern est bien présente et peut être instanciée avec ou sans nom passé en paramètre.
- L'utilisation de print() pour afficher une instance permet bien l'affichage de l'attribut "Name" de l'instance (le "name" passé en paramètre à l'instanciation ou la valeur par défaut).

- Une classe `Coffee` est bien présente dans le scope de la classe `Intern`. La méthode `str()` est bien définie dans cette classe `Coffee` de manière à ce qu'elle retourne "This is the worst coffee you ever tasted."
- La méthode `work()` de la classe `Intern` est bien présente et lève bien une exception avec pour texte "I'm an intern, I can't do that...". Le type utilisé est bien l'exception de base (`Exception`).
- La méthode `make_coffee()` est bien implémentée dans la classe `Intern` et elle retourne bien une instance de la classe `Coffee` implémentée dans le scope de la classe `Intern` (`return self.Coffee()`).
- Utiliser `print()` sur le retour de la méthode `make_coffee()` (une instance de la classe `Intern.Coffee` donc) affiche bien le retour de la méthode `str()` de l'instance de la classe `Intern.Coffee`, c'est-à-dire "This is the worst coffee you ever tasted."
- L'exception levée par l'appel de la méthode `work()` est bien catchée par un bloc `try/except`.

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

✓ Yes

✗ No

ex02: 5 classes 1 cup.

- Chaque instance de la classe `HotBeverage` ou d'une de ses classes dérivées est affichée de la bonne manière :

```
name : <name>
price : <price limited to a two decimal point>
description : <description>
```
- Les informations affichées pour chaque instance de chaque classe correspondent bien aux informations demandées dans le sujet.
- Le contenu de chaque classe fille ne doit être explicitement codé que s'il est différent de celui de la classe mère.
Ainsi, si le prix, le nom, ou la description d'une classe est explicitement réécrit dans la classe fille alors que l'information qu'il contient est identique à celle héritée de la classe mère, l'exercice est considéré comme non valide.

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

✓ Yes

✗ No

ex03: Glorious coffee machine!

- Une classe `EmptyCup` héritant de `HotBeverage` est bien présente dans le scope de la classe `CoffeeMachine` et les informations (description, prix, et nom) correspondent bien aux informations fournies dans le sujet.
- La classe `CoffeeMachine` peut être instanciée et l'instance possède bien une méthode `serve()` qui prend en paramètre une classe `HotBeverage` ou dérivée et a 50% de chance de retourner une instance de la classe passée en paramètre, ou 50% de retourner une instance d'`EmptyCup`. Vérifiez que le retour est bien aléatoire et que les probabilités demandées sont respectées.
- La classe `CoffeeMachine` possède bien un constructeur.
- Une classe `BrokenMachineException` héritant de la classe `Exception` est également implémentée dans le scope de la classe `CoffeeMachine`. Le message "This coffee machine has to be repaired." est bien défini dans le constructeur.
- Après 10 appels à la méthode `serve()`, un nouvel appel à cette méthode provoque la levée de l'exception `CoffeeMachine.BrokenMachineException`. Utiliser `print()` sur cette expression permet l'affichage du message "This coffee machine has to be repaired."
- Après l'appel de la méthode `repair()`, la méthode `serve()` peut de nouveau être appelée 10 fois avant que la machine ne tombe à nouveau "en panne" et ne lève à nouveau l'exception pour chaque nouvel appel.

- Chaque levée d'exception est correctement gérée (=catchée avec un bloc try/catch) dans vos tests.

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

 Yes

 No

ex04: Une classe de base ft. RMS.

- Le fichier de tests en annexe s'exécute avec succès.
- La structure demandée dans le sujet est correctement reproduite et affichée dans les tests.

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

 Yes

 No

ex05: Faites vos propres éléments.

- Les classes demandées sont toutes bien présentes et héritent toutes de Elem de l'exercice précédent. Elles sont toutes bien fonctionnelles
- print(Html([Head(), Body()]))) affiche bien :

```
<html>
<head></head>
<body></body>
</html>
```
- La structure demandée dans le sujet est correctement reproduite et affichée SANS USAGE DE LA CLASSE Elem.

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

 Yes

 No

ex06: Validation.

- La classe Page est présente.
- print(Page(Html([Head(), Body()]))) affichera bien :

```
<!doctype html>
<html>
<head></head>
<body></body>
</html>
```
- Page(Html([Head(), Body()])).is_valid() retournera bien False.
- Page(Html([Head(Title(Text('title'))), Body()])).is_valid() retournera bien True.
- Page(Html([Head(Title(Text('title'))), Body(Li())]).is_valid() retournera bien False.
- Page(Html([Head(Title(Text('title'))), Body(OL(Li(Text('foo')))]))).is_valid() retournera bien True
- Testez tous les cas qui vous sembleront nécessaires. Vous DEVEZ éprouver le travail rendu par l'étudiant et le tester de manière exhaustive.
- Page(Html([Head(Title(Text('title'))), Body()])).write_to_file("test.html") crée un fichier test.html contenant :

```
<!doctype html>
<html>
<head>
<title>
title
</title>
</head>
<body></body>
</html>
```

L'exercice est considéré comme faux si un des points demandés n'est pas parfaitement réalisé.

 Yes

 No

Ratings

Don't forget to check the flag corresponding to the defense

 Ok


 Outstanding project


 Empty work

 Incomplete work

 Cheat

 Crash

 Concerning situation

 Forbidden function

Conclusion

Leave a comment on this evaluation

Finish evaluation