Programmation Orientée Objet

TD 1 : Classes et objets

1 Compteur

1.1 Compteur simple

On souhaite créer un objet `Compteur` qui retient le compte de quelque chose et est capable d'ajouter 1 à son compte quand on le lui demande. Proposez une modélisation UML de cet objet simple. Cette modélisation doit être capable de répondre au code suivant :

```
from counter import Counter

c1 = Counter()
c2 = Counter()
c1.count()
c2.count()
c1.count()

print(c2.get_value())
```

Que valent `c1.get_value()`, `c2.get_value()` à la fin de l'exécution ?

1.2 Code de classes

Le fichier `counter.py` contient le code la classe `Counter` :

```
class Counter:
    def __init__(self):
        self.value = 0

    def count(self):
        self.value = self.value + 1

    def get_value(self):
        return self.value
```

Exécutez le code de la partie 1.1 un utilisant le code python de la classe ci-avant. Montrez tous les *namespaces* utilisés.

1.3 Compteur à pas choisi

On souhaite maintenant pouvoir choisir le pas de notre compteur (c'est-à-dire ajouter 2 à chaque fois plutôt que 1 par exemple). Que faut-il ajouter à notre classe ?

2 Un Dé

2.1 Modèle d'un dé classique

On veut créer une classe `Dice`. Elle doit être capable de :

- créer un objet sans paramètre,
- créer un objet avec sa valeur initiale,
- connaître et donner la valeur du dé (avec les méthodes `get_position` et `set_position`),
- lancer un dé.

Proposez une modélisation UML de la classe `Dice`. Donnez des exemples de manipulation d'objets de cette classe : créer un objet, modifier la valeur de sa position, obtenir sa position et le lancer.

2.2 Modèle d'un dé pipé

On souhaite améliorer notre classe `Dice` pour pouvoir tricher. Complétez la modélisation UML précédente afin d'associer à chaque dé une liste correspondant aux probabilités de tomber sur chaque face.

Que faut-il modifier ?