Bastien
LEMARIÉ

FISA2025

Rapport de COO Avancée
-
Introduction (semaine 1)

IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Lien du repository GitHub : https://github.com/bastibraz/COO_Pharo/settings/access

1 / Apprendre à connaître les collections dans Pharo et leurs itérateurs

1. **Qu'est-ce qu'une collection et à quoi sert-elle ?**

Une collection est une structure de données qui permet de regrouper plusieurs objets. Elle sert à stocker, organiser, et manipuler des ensembles d'éléments de manière efficace.

2. **Quels types de collections la bibliothèque standard du Pharo propose-t-elle ?**

La bibliothèque standard de Pharo propose plusieurs types de collections :

- Séquences : Listes, tableaux (Array), chaînes de caractères (String).
- Ensembles : Sets (sans doublons).
- Dictionnaires : Mappings clé-valeur (Dictionary).
- Bag : Collections qui comptent le nombre d'occurrences d'éléments.

3. **Comment itérer les collections et quelles sont les différences entre elles ?**

On peut itérer sur les collections en utilisant des blocs de code ou des méthodes d'itération (comme *do:, collect:, select:, reject:*, etc.).

Les différences résident dans leur comportement : les séquences maintiennent l'ordre, les ensembles éliminent les doublons, et les dictionnaires permettent l'accès via une clé.

4. **Comment avez-vous trouvé ces informations ?**

Ces informations proviennent de la documentation officielle de Pharo et de l'expérience avec le langage Smalltalk/Pharo, qui est bien documenté dans ses ressources en ligne et guides d'apprentissage.

2 / Apprendre les conditionnels en Pharo

1. **Comment écrire des conditionnelles dans Pharo ?**

En Pharo, les conditionnelles s'écrivent en utilisant des messages envoyés aux objets booléens. Par exemple, on utilise *ifTrue: [...] ifFalse: [...]* pour exécuter du code en fonction de la valeur de l'objet booléen.

2. **Qu'est-ce qui les différencie des autres langages de programmation ?**

En Pharo, les conditionnelles sont des envois de messages aux objets plutôt que des structures de contrôle classiques. Cela s'aligne avec la philosophie orientée objet de Smalltalk, où tout est un objet, y compris les booléens et les blocs de code.

3. **Pouvez-vous réfléchir aux avantages et aux inconvénients de cette approche ?**

Avantages : La cohérence dans la philosophie orientée objet et la lisibilité du code.

Inconvénients : Peut paraître inhabituel pour les programmeurs venant d'autres langages, et les performances peuvent être légèrement impactées par rapport aux structures de contrôle natives d'autres langages.

4. **Comment avez-vous trouvé ces informations ?**

Ces informations proviennent des connaissances sur le langage Pharo, de sa documentation officielle et des principes de programmation Smalltalk.

3 / Apprendre à créer des classes et des méthodes

1. **Comment écrire un petit programme avec des classes et des méthodes dans Pharo ?**

Dans Pharo, on crée des classes et des méthodes directement via l'IDE. On utilise le navigateur de classes pour définir une nouvelle classe en spécifiant son nom et sa superclasse. Ensuite, on ajoute des méthodes en les éditant dans le navigateur de méthodes.

2. **Pharo est en effet très orienté IDE et il faut s'habituer à l'outillage. Comment avez-vous trouvé ces informations ?**

Ces informations viennent de l'utilisation pratique de Pharo et de ses tutoriels officiels, qui fournissent des exemples détaillés de création de classes et de méthodes.

3. **Quel programme avez-vous écrit ?**

```
Object << #Counter
	slots: { #count . #new };
	package: 'MyCounter'

count
	^ count

count: anInteger
	count := anInteger

increment
	count := count + 1

testCountIsSetAndRead
	| c |
	c := Counter new.
	c count: 7.
	self assert: c count equals: 7

testIncrement
	| c |
	c := Counter new.
	c count: 2.
	c
	increment;
	increment.
	self assert: c count equals: 4
```

4. **Quels problèmes avez-vous rencontrés ?**

Les principales difficultés étaient liées à l'usage de l'IDE : s'habituer aux outils pour naviguer dans le code, comprendre comment sauvegarder les modifications, et manipuler les fenêtres de l'IDE, qui est différent des éditeurs de texte classiques.