2019

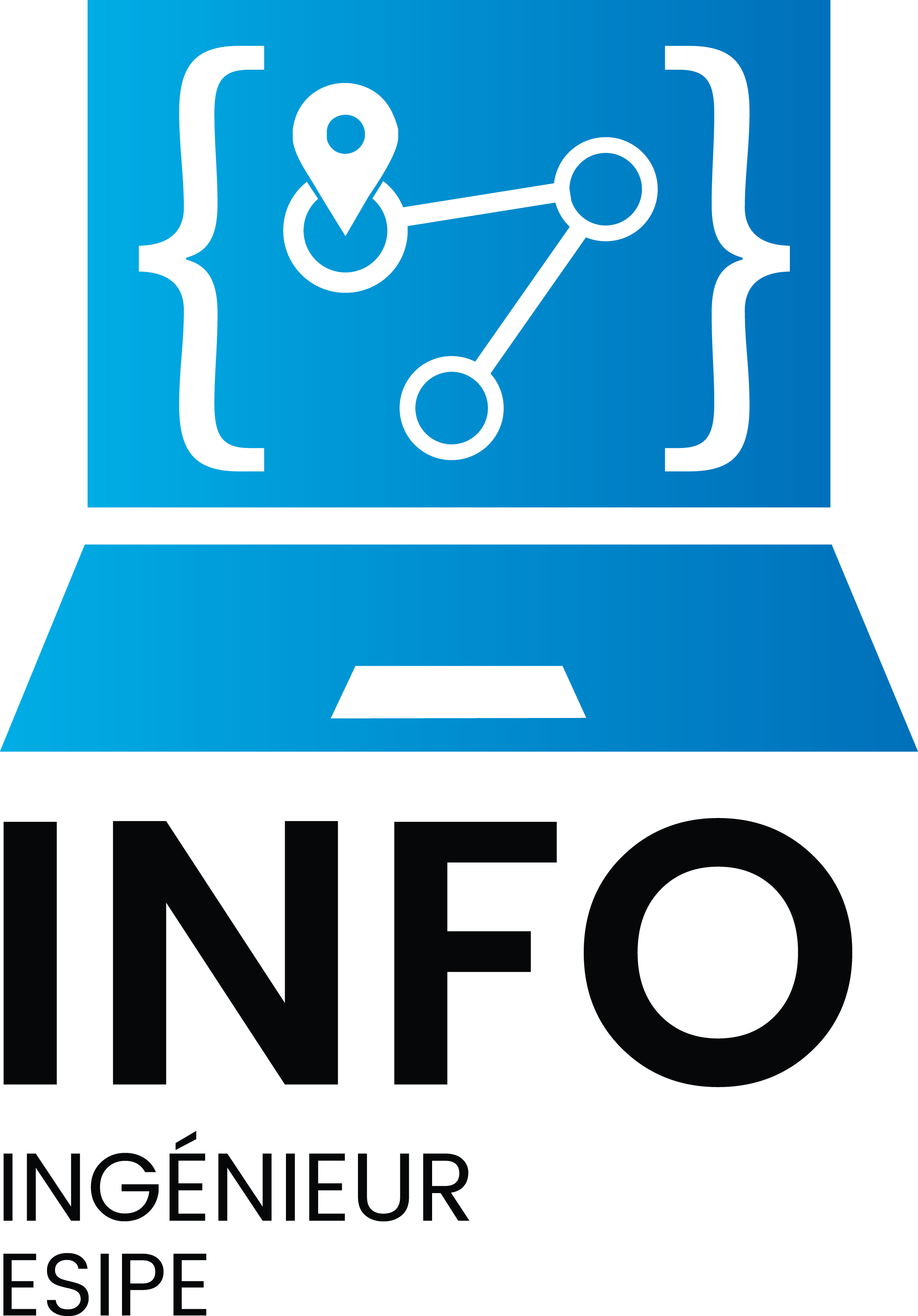
Jonathan CRÉTÉ – Filière Informatique

Compte rendu de TP : Interface fonctionnelle, la puissance du déclaratif

Java avancé – TP n°3









Compte rendu de TP : Interface fonctionnelle, la puissance du déclaratif

**Exercice 1 - Docteur, je me sens mal ?**

**1)** Écrire la méthode healthCheck sachant que dans notre cas, le contenu de la réponse HTTP ne nous intéresse pas, seul un code de réponse de 200 est suffisant.

Note : la doc de la classe HttpClient vous donne un exemple d'utilisation.

|  |
| --- |
|  |

**2)** Définir l'interface fonctionnelle URIFinder qui possède une méthode find pour que le code d'utilisation d'un URIFinder ci dessous fonctionne

|  |
| --- |
|  |

**3)** On cherche maintenant à écrire la méthode fromArguments qui prend en paramètre un tableau de String (celui fourni par le main) et renvoie un URIFinder qui considère le premier argument, et si il existe, permet de le transformer en URI (en utilisant URI.create).

Où doit-on placer la méthode fromArguments et quels sont les modificateurs (public etc) de celle-ci ?

Écrire la méthode fromArguments.

|  |
| --- |
|  |

**4)** On cherche maintenant à écrire une méthode fromURI qui prend une chaîne de caractères et la transforme en URI.

Que se passe-t-il si la chaîne de caractères n'est pas une URI valide ?

Comment faire si l'on souhaite ignorer les chaînes de caractères qui ne sont pas des URI valides ?

Écrire le code de la méthode fromURI et réviser le code de fromArguments pour que lui aussi ignore les chaînes de caractères qui ne sont pas des URI valides.On appel la méthode close() sur notre stream :

|  |
| --- |
|  |

**5)** On souhaite maintenant écrire une méthode or qui permet de combiner deux URIFinder de telle façon que si le premier URIFinder ne trouve pas d'URI, alors le second essaie de trouver son URI.

Où doit-on placer la méthode or et quels sont les modificateurs de celle-ci ?

Écrire la méthode or.On développe le code avec le bloc finally

|  |
| --- |
|  |

**6)** On souhaite écrire la méthode fromMapGetLike qui prend comme premier paramètre un nom de clé/propriété et comme second paramètre une fonction ayant la même signature et la même sémantique que map::get (avec des String comme type de clé et de valeur dans la Map). Cette méthode devra renvoyer un URIFinder permettant de chercher la valeur de la clé dans la Map.

Voici un exemple d'utilisation.

Écrire le code de la méthode fromMapGetLike et vérifier aussi que le code fromMapGetLike("HEALTH\_CHECK\_URI", System::getenv) est valide.

Rappel: map.get renvoie null si il n'y a pas de valeur associée à une clé donnée.

|  |
| --- |
|  |

**7)** En fait, la méthode fromMapGetLike pourrait avoir des types de paramètre (une signature) acceptant plus de cas valides. Par exemple, le code suivant dans lequel les clés sont des Integer devrait aussi fonctionnerOn utilise un try-catch pour reprendre sur l'erreur, gérer l'exception.

**8)** Enfin, on souhaite ajouter une méthode fromPropertyFile qui prend en paramètre le chemin d'un fichier et le nom d'une clé et renvoie un URIFinder qui renvoie l'URI associé à la clé dans le fichier de properties si l'association existe et que la valeur associée est bien une URI valide.

Écrire la méthode fromPropertyFile.

Note : il existe une méthode Properties.load().

Note 2 : vous vous rappelez surement qu'une bonne façon de lire un fichier de caractères est d'utiliser un BufferedReader...

|  |
| --- |
|  |

**Conclusion**

Durant ce troisième TP de JAVA avancé,

