Dossier Application réflexive JAVA

Projet réalisé par Macé Loïc Gr201 et Abelhaj Youssef Gr202

Table des matières

[Rappel du projet 2](#_Toc3744227)

[Fonctionnalités 2](#_Toc3744228)

[Amateur 2](#_Toc3744229)

[Programmeur 2](#_Toc3744230)

[Services 2](#_Toc3744231)

[Spécifications techniques 2](#_Toc3744232)

[Serveur / Client 2](#_Toc3744233)

[Le gestionnaire de service 2](#_Toc3744234)

[Transfert de fichiers 3](#_Toc3744235)

[Services 3](#_Toc3744236)

[Obligations 3](#_Toc3744237)

[Multiclasses 3](#_Toc3744238)

[Inversion de texte 3](#_Toc3744239)

[Analyseur d’XML 3](#_Toc3744240)

[Messagerie 3](#_Toc3744241)

[Bilan 4](#_Toc3744242)

[Difficultés 4](#_Toc3744243)

# Rappel du projet

Dans le cadre du cours « Application réflexive », nous devions réaliser un gestionnaire de service permettant à des programmeurs de déposer des services et des amateurs d’utiliser ceux-ci.

Pour ce faire, nous avons eu recours à la réflexivité en JAVA.

# Fonctionnalités

## Amateur

Les amateurs se connectent à un port spécifique.

Un amateur peut accéder aux services qui ont été ajoutés au préalable par des programmeurs.

Il peut être amené à se connecter pour bénéficier de services spécifiques comme une messagerie par exemple.

## Programmeur

Les programmeurs se connectent à un port spécifique.

Le programmeur doit s’identifier (login et mot de passe) pour pouvoir effectuer les actions suivantes :

* Ajouter un service
* Mettre à jour un service
* Changer son adresse de FTP
* Démarrer ou arrêter un service
* Supprimer un service

## Services

Le gestionnaire de services est agrémenté des 3 services suivants :

* Inversion d’une chaine de charactère.
* Analyse d’un fichier XML et génération des résultat de l’analyse sur le serveur FTP.
* Messagerie.

# Spécifications techniques

L’amateur et le programmeur ont tout deux un port dédié.

## **Serveur / Client**

## Le gestionnaire de service

Le gestionnaire de service permet comme son nom l’indique de gérer les services.

Le gestionnaire ne contient que des méthodes statiques.

Il a une liste statique de class dans laquelle sont stockés les classes des services ainsi que des méthodes pour :

* Ajouter un service
* Mettre à jour un service
* Démarrer ou arrêter un service
* Supprimer un service
* Renvoyer la String de la liste de tous les services
* Vérifier si un service est valide

## Transfert de fichiers

## Services

### Obligations

Les services ont quelques obligations :

* Avoir une classe principale Main implémentant l’interface Service et donc être Runnable.
* La classe du service ne doit pas être abstract.
* Ils doivent contenir un constructeur avec un unique paramètre Socket.
* Ils doivent avoir un attribut private final Socket.
* Ils doivent avoir une méthode public toStringue.
* Ils doivent avoir une méthode init avec paramètre String pouvant lever une Exception.

### Multiclasses

Les services multiclasses ont un attribut Class< ?> pour chaque classe dont ils ont besoin.

La méthode init dans le Main charge les classes, autres que le Main dans les différents attributs Class< ?>.

Le service utilise ensuite ces attributs.

### Inversion de texte

Le service d’inversion demande à l’amateur de saisir une chaine qui lui ai retournée inversée.

### Analyseur d’XML

L’analyseur de XML, permet de lire un fichier XML, s’il n’y arrive pas, le fichier n’est pas valide.

Après demande à l’amateur du path de son fichier, le fichier est analysé récursivement.

Une structure arborescente « balise » est créé puis affichée à l’amateur.

### Messagerie

L’Amateur qui utilise ce service doit d’abord s’enregistrer en saisissant login puis mot de passe, puis se connecter avec ses identifiants pour accéder aux fonctionnalités offertes par le service. De ce fait, il a la possibilité de voir les messages qu’il a reçu, ou, d’envoyer des messages à d’autres utilisateurs.

Nous distinguons les utilisateurs du gestionnaire de service, des utilisateurs du service de la Messagerie. Les messages sont constitués d’un expéditeur, d’un objet, d’un destinataire et du message à transmettre. Les messages sont tous stockés dans la même liste et des méthodes sont utilisées pour renvoyer, par exemple, tous les messages d’un utilisateur donné.

# Bilan

Au cours de ce projet, nous avons renforcé nos capacités à travailler en équipe.

Grâce à git, le projet a pu être mené en parallèle, nous avons ainsi gagné du temps et cela nous a permis de mettre se temps à profit dans le code.

## Difficultés

Nous avons eu quelques difficultés pour charger un service utilisant plusieurs classes. Mais après réflexion (sans mauvais jeu de mots), la solution est venue d’elle-même en codant.