

JAVA RMI

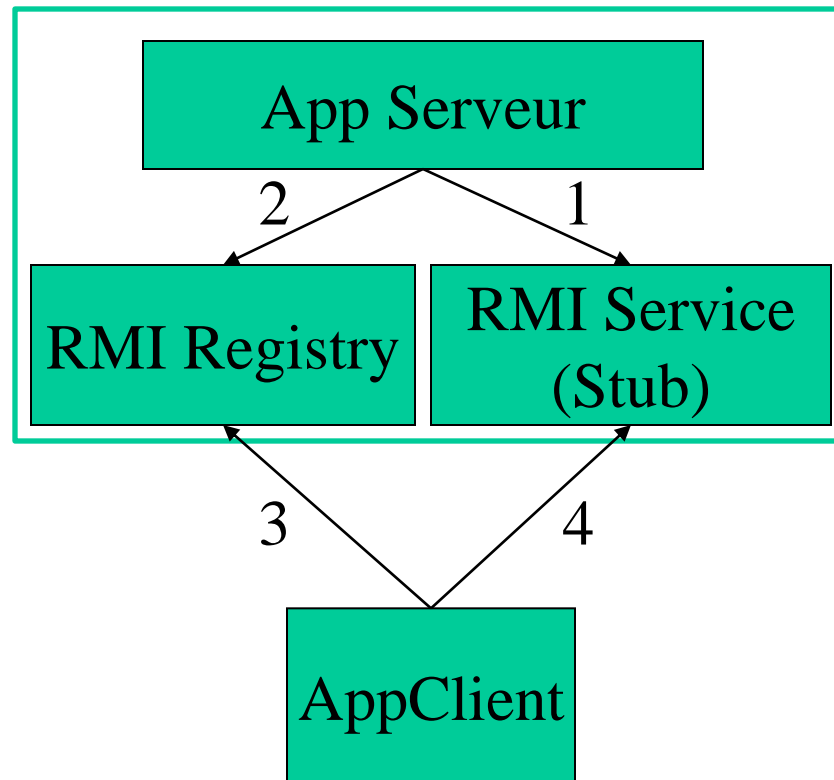
Remote Method Invocation

RMI

- Permet de faire de la programmation orientée objet distribuée (facilement) en Java.
- Une fois le petit framework en place, la distance devient transparente pour le programmeur.

RMI

- L'architecture d'un programme RMI simple:



RMI

- Nous utiliserons une librairie en remplacement du RMI natif de Java
- Une version modifiée de la librairie RipeRMI (qui est un fork de LipeRMI) vous sera distribuée sur LÉA.
- Ça va nous simplifier la vie:
 - Architecture
 - **Android** (ne supporte pas RMI natif)
 - Connexion bidirectionnelle simple (MVC)

Construire une application avec RipeRMI:

1. Déterminer l'interface du serveur.
2. Implémenter le serveur.
 1. Enregistrer notre stub auprès du service RMI
 2. Exposer notre serveur
3. Implémenter le client.
 1. Se connecter au serveur
 2. Récupérer un proxy sur notre interface serveur
 3. Appeler une méthode distante
4. Partir le serveur.
5. Partir le client.

DEMO

- HelloWorld

RMI – Exemple (Interface)

- HelloWorld à distance:
 - Notre interface serveur propose une fonction

```
public interface IServer
```

```
{
```

```
public void sayHello(String helloFromWho);
```

```
}
```

RMI – Exemple (Serveur)

```
public class MyServer extends Server implements IServer
{
    public void sayHello(String helloFromWho)
    {
        System.out.println("HelloWorld from : " + helloFromWho);
    }
    ...
}
```


Toujours nécessaire pour du remote

RMI – Exemple (Serveur)

```
public MyServer(){
```

```
    CallHandler callHandler = new CallHandler();
```

On enregistre une instance de notre serveur
par l'entremise de son interface

```
    callHandler.registerGlobal(IServer.class, this);
```

On expose notre serveur sur le port 12345

```
    this.bind(12345, callHandler);
```

On garde notre serveur actif

```
    while(true){Thread.sleep(500);}
}
```

Toujours nécessaire pour du remote

RMI – Exemple (Client)

```
public static void main(String... args){  
    CallHandler callHandler = new CallHandler();
```

Connexion au serveur 192.168.1.200:12345

```
    Client client = new Client("192.168.1.200", 12345, callHandler);
```

Obtention du Proxy

```
    IServer myServiceCaller = client.getGlobal(IServer.class);
```

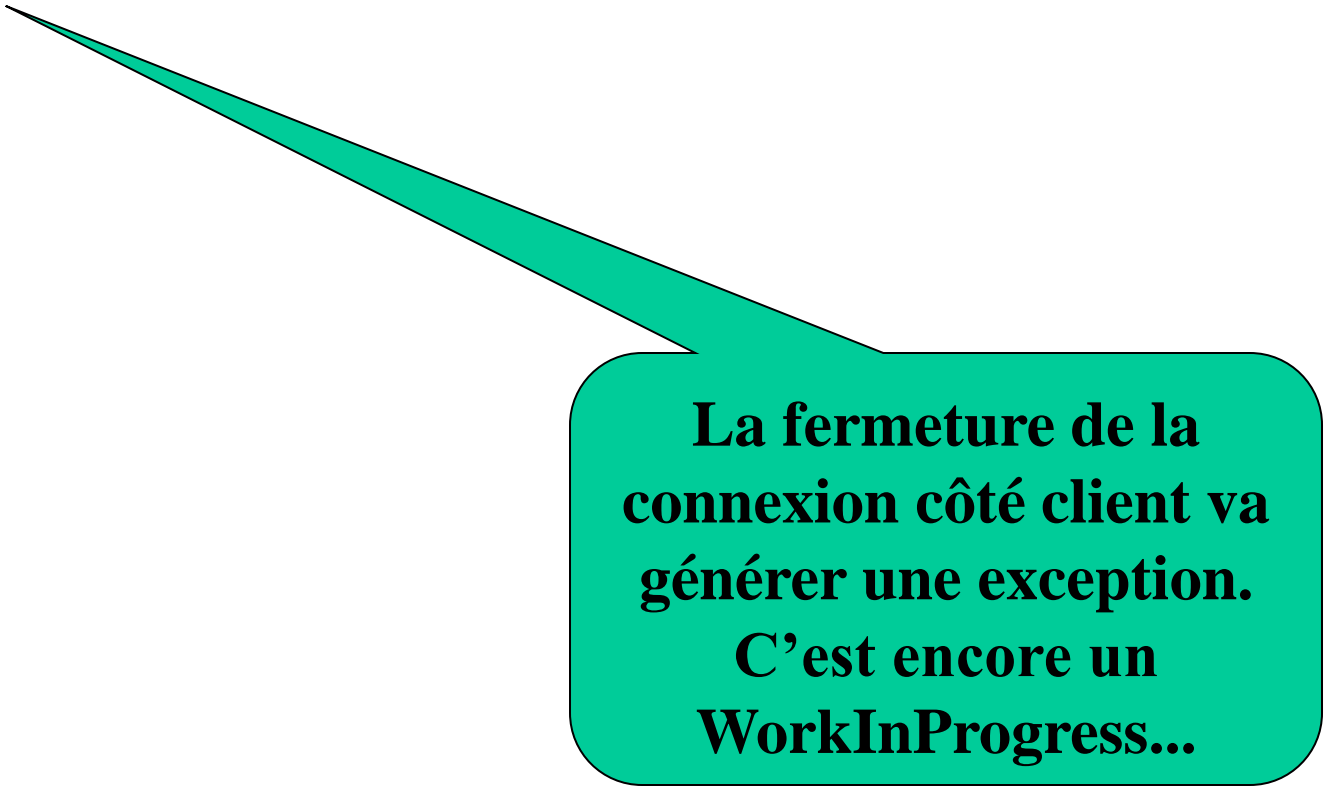
Appel distant

```
    myServiceCaller.sayHello("Frank");  
    client.close();}
```

Deconnexion du serveur

RMI – Exemple (Client)

```
public static void main(String... args){  
...  
client.close();  
...  
}
```



**La fermeture de la
connexion côté client va
générer une exception.
C'est encore un
WorkInProgress...**

RMI – Exemple

Essayons notre exemple HelloWorld ensemble

- Vous avez besoin du .jar de RMI sur LÉA, à inclure dans votre projet.

*Il est possible que le démarrage du serveur soit bloqué par un Firewall. Soit modifier les règles ou le désactiver.

RMI - Exemple

```
IServer myServiceCaller = (IServer)  
client.getGlobal(IServer.class);
```

```
myServiceCaller.sayHello("FrankTiger");
```



Que ce passe-t-il ici ?

RMI – Exemple

- myServiceCaller est un proxy vers un objet distant (sur le serveur) et non une copie de l'objet.
- L'accès aux ressources de l'objet passeront par le réseau.
- La String en paramètre sera « sérialisée » et sa COPIE sera transférée sur le réseau !!!

RMI – Exemple

- Modifiez votre code client pour vous connecter sur mon serveur:
 - IP = ???
 - Port = 12345
- Exécutez votre client.

Serialization

- Java propose l'interface Serializable.
- Permet de transformer automatiquement une instance en une séquence binaire (serialize) et reconstruire une instance à partir d'une séquence binaire (deserialize).
- L'interface n'a aucune fonction.

Serialization

- Une classe qui implémente Serializable doit définir un champ

`static final long serialVersionUID`

- Ce champs permet de savoir si une séquence binaire est bien compatible avec la version actuelle de la classe.
- Ce champ devrait être modifié à chaque changement majeur de la classe.

RMI - Trucs

- Les paramètres (et la valeur de retour) des fonctions appelées sur le stub seront transmis (copiés pour vrai) sur le réseau.
- Ils doivent donc être Serializable.

RMI - Trucs

- Attention, lorsque les objets se déplacent sur le réseau (e.g., paramètres), la fonction equals() d'Object n'est plus adéquate car on a vraiment deux copies en mémoire d'un objet équivalent.
- Il faut donc la redéfinir.

RMI - Trucs

- RMI est multi-thread par défaut. Lorsque le client appelle une fonction sur le serveur, ça va automatiquement créer un nouveau thread sur le serveur.
- Plusieurs clients = plusieurs threads.

Suite

- La semaine prochaine, nous verrons des fonctionnalités plus avancées de RipeRMI.