.NET Remoting

1 Partie TD

1.1 Étude détaillée d'exemples

Question 1. Exemple Singleton (listings 1, 2 et 3)

a- On n'utilise pas ici d'interface. Expliquez pourquoi c'est une mauvaise idée. Que faudrait-il faire pour utiliser une interface ?

b- La ligne 14 de HelloServer enregistre notre service distant. Comment sait-on quelle instance de HelloServer sera récupérée par le client aux lignes 35 et 43 ?

c- Quel type d'affichage aurait-on chez le client et le serveur si on passait de Singleton à SingleCall ? Et à l'activation côté client ?

d- Que veut dire le `true` de la ligne 13 du listing 2 ?

e- Que se passe t-il si on remplace la ligne 35 du listing 3 par la ligne qui suit ?

```
1 RemotingConfiguration.RegisterWellKnownClientType(typeof(HelloServer),"tcp://
    localhost:8089/SayHello");
2 HelloServer obj = new HelloServer();
```

Question 2. Exemple Passage de références (listings 1, 4 et 5)

a- À la ligne 15 du listing 5, l'objet param est-il créé chez le client ou chez le serveur ?

b- Si on met la ligne 25 au lieu des lignes 18 à 24 du listing 4, quelle est la ligne du listing 5 qui va déclencher une erreur ? Pourquoi ?

Question 3. Exemple asynchronisme

a- Que fait la ligne 7 du listing 7 ?

b- À la ligne 57 du listing 8, pourquoi obj reçoit-il un proxy au lieu d'une instance nouvellement créée en local ?

c- Que se passe t-il quand la ligne 65 du listing 8 s'exécute ?

d- Que se passe t-il à la ligne 73 du listing 8 ?

e- Que se passe t-il à la ligne 77 du listing 8 ?

f- Pourquoi récupère t-on immédiatement la main suite à l'exécution de la ligne 82 du listing 8 ? Pourquoi passe t-on un delegate en paramètre à cette ligne ?

g- À quoi sert la ligne 104 du listing 8 ?

h- Quand sera appelée la méthode définie à la ligne 18 du listing 8 ?

i- Quel évenement permet de sortir de la boucle des lignes 101 à 105 du listing 8 ?

2 Partie TP

2.1 Exemples de TD

Récupérez les exemples vus en TD et disponibles via :
http ://www.lirmm.fr/~nebut/index.php/Enseignement/GMIN204.
Faites fonctionner ces exemples en faisant varier les projets à démarrer, dans les propriétés de la solution. Prenez le temps pour bien les comprendre.

2.2 Bibliothèque

Une bibliothèque souhaite informatiser son système d'informations pour permettre aux personnes le souhaitant de consulter à distance son catalogue, et aux abonnés de pouvoir entrer des commentaires sur des ouvrages, qui seront alors consultables également. Dans un premier temps, les réservations et les emprunts d'ouvrage resteront non-informatisés. On ne distinguera donc pas la notion d'ouvrage de la notion d'exemplaire. Pour la première version, on considèrera un seul type d'ouvrage : le livre, dont les champs à stocker sont : son titre, son auteur, son ISBN, son nombre d'exemplaires, son éditeur.

On ne réalisera que 2 méthodes de consultation : la recherche d'ouvrage par ISBN et par nom d'auteur. Chaque abonné se voit attribuer un numéro d'abonné (unique), et choisit un mot de passe. Ce numéro d'abonné et ce mot de passe lui sera demandé lors de l'ajout d'un commentaire. On ne s'occupera pas du cryptage de ce mot de passe, qui sera stocké et envoyé en clair.

On gardera bien à l'esprit que le but n'est pas de créer une application de consultation réaliste, mais de comprendre la mise en œuvre d'applications réparties via le .net remoting. En ce sens, vous ne devez pas vous attarder trop sur les mécanismes de recherche avancés par auteur, par exemple ne prenez pas en compte le fait qu'on peut mal connaitre l'orthographe du nom de l'auteur. De même, on se contentera d'un objet Livre minimum. On ne réalisera une interface graphique cliente que si l'on a fini le TP.

Question 4. Réfléchissez à la structure de votre application, que vous développerez ensuite en TP. Vous réaliserez au minimum un diagramme de classes. Essayez de faire une conception "propre", en évitant notamment le piège de la grosse classe serveur omnipotente. Vous réfléchirez également à quelles classes seront accessibles à distance, et comment (utilisation de références ou de copie, publication en mode singleton, singlecall, activation par le client, utilisation de fabriques ?).

Question 5. Développez une première version de votre application sans client ni objet distribué : vous la testerez avec un Main simulant le client. Vous veillerez à utiliser à bon escient des propriétés. Vous pouvez utiliser des tables de hachage (classe Dictionary<TKey,TValue> de System.Collections.Generic). Cela vous permettra d'utiliser les indexeurs pour accéder aux éléments de votre table de hachage, documenté dans Property→Item. On utilisera des canaux TCP.

Attention : pour des raisons de sécurité, les lecteurs réseaux ne sont pas par défaut considérés comme des endroits de confiance, et vous ne pourrez (peut être) pas utiliser le remoting depuis votre compte (dans la mesure où vous n'avez pas les droits d'administrateur pour ajouter le lecteur correspondant dans la zone de confiance).

Question 6. Mettez en œuvre la distribution en n'utilisant pas d'interfaces pour les objets distribués : il n'y a presque rien à changer dans votre application. Quand ça fonctionne (testez !), utilisez des interfaces (comme avec java RMI). Faites un projet dédié aux interfaces, et partagé par le serveur et le client (ajouter une référence sur le projet contenant les interfaces). Au moment du déploiement, il faudra partager la dll correspondante.

Question 7. Modifiez le mode de publication de vos objets (Singleton, Singlecall, activation par le client) et testez de manière à vous assurer que le mode que vous avez choisi était bien le bon. Attention, le moment où les objets sont créés sur le serveur est souvent déroutant, faites des affichages vous permettant de déterminer quand les objets initiaux sont effectivement créés (par objets initiaux, on entend ici : ceux qui sont récupérés par le client soit par un (faux) new, soit par un GetObject(), et pas transmis par un objet du serveur qu'on connait déjà).

Question 8. Faites en sorte que la gestion des abonnés et du catalogue soit une application cliente du serveur de bibliothèque, de manière à ce que plusieurs bibliothécaires puissent administrer la bibliothèque en même temps.