Distributed Objects

Request-Response

- A message exchange pattern
    - Knock-Knock Protocol.
    - High-Low Guess (HW #1)
    - HTTP
- Synchronous communication
- Advantage: Simple. Easy to detect problems.
- Disadvantage: Less throughput because client waits

Remote Procedure Call

- Causing a subroutine to execute in another process (maybe on another computer)
- Implemented as a request-response protocol
- To the programmer an RPC looks like a local procedure call.
  - There is some additional overhead
    - Server has to publish the remote object
    - Client has to obtain a reference to the remote object

Java RMI
Remote Method Invocation

- Source: https://docs.oracle.com/javase/tutorial/rmi/

RMI Overview

- Client-server model using distributed objects
- Objects can be in another JVM
  - Why? Share resources. Load balance.
- Any Java object can be passed as a parameter or return type
- Server
  - Creates remote objects
  - Makes references to remote objects available
- Client
  - Gets references to remote objects
  - Calls methods on remote objects

java.rmi.Naming

- Provides methods for getting/setting references to remote objects in a remote object registry
  - A name server for remote Java objects
- Methods
  - void bind(String name, Remote obj)
  - String[] list()
  - Remote lookup(String name)
  - void rebind(String name, Remote obj)
  - void unbind(String name)

RMI Example

Directory Structure

```
[project home]/
    app.policy        <- Security Policy
    build.xml
    build/
        myrmi/
            *.class
    src/
        myrmi/
            *.java
```

Note: the directory structure matches the package naming.

Hello Interface

```java
package myrmi;

import java.rmi.Remote;
import java.rmi.RemoteException;

public interface Hello extends Remote {
   String sayHello(String name) throws RemoteException;
}
```

Server 1of 2

```java
package myrmi;

import java.rmi.Naming;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class Server extends UnicastRemoteObject implements Hello {

   public Server() throws RemoteException {
   }

   @Override
   public String sayHello(String name) throws RemoteException {
      if (name == null) {
         name = "world";
      }

      return "Hello, " + name;
   }
```

Server 2 of 2

```
    public static void main(String[] args) {
        if (System.getSecurityManager() == null) {
            System.setSecurityManager(new SecurityManager());
        }

        try {
            Hello obj = new Server();
            Naming.rebind(“Hello", obj);

            System.out.println("Server ready");
        } catch (Exception re) {
            re.printStackTrace();
        }
    }
}
```

Client

```
package myrmi;

import java.rmi.Naming;

public class Client {
   public static void main(String[] args) {
      // check for 1 command line argument
      String host = args[0];
      if (System.getSecurityManager() == null) {
         System.setSecurityManager(new SecurityManager());
      }

      try {
         String url = "rmi://" + host + "/Hello";
         Hello stub = (Hello)Naming.lookup(url);

         System.out.println("response: " + stub.sayHello(null));
         System.out.println("response: " + stub.sayHello(“UND"));
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Match binding name

Security Policy

- The server and client have set a SecurityManager
- Requires a security policy file to grant security permissions
- Type following into app.policy

```
grant codeBase “file:<path to build directory>” {
     permission java.security.AllPermission;
};
```

Compile

```
$ cd [project home]
$ ant
```

or

```
$ java -d build src/myrmi/*.java
```

Start the RMI Registry

```
$ cd build

$ rmiregistry
```

Start the Server

```
$ cd [project home]

$ java -cp build -Djava.security.policy=app.policy myrmi.Server
```

Did you create this file in the project directory?

Start the Client

```
$ cd [project home]

$ java -cp build -Djava.security.policy=app.policy myrmi.Client localhost
```

Did you create this file in the project directory?