Java Network II
(RMI)

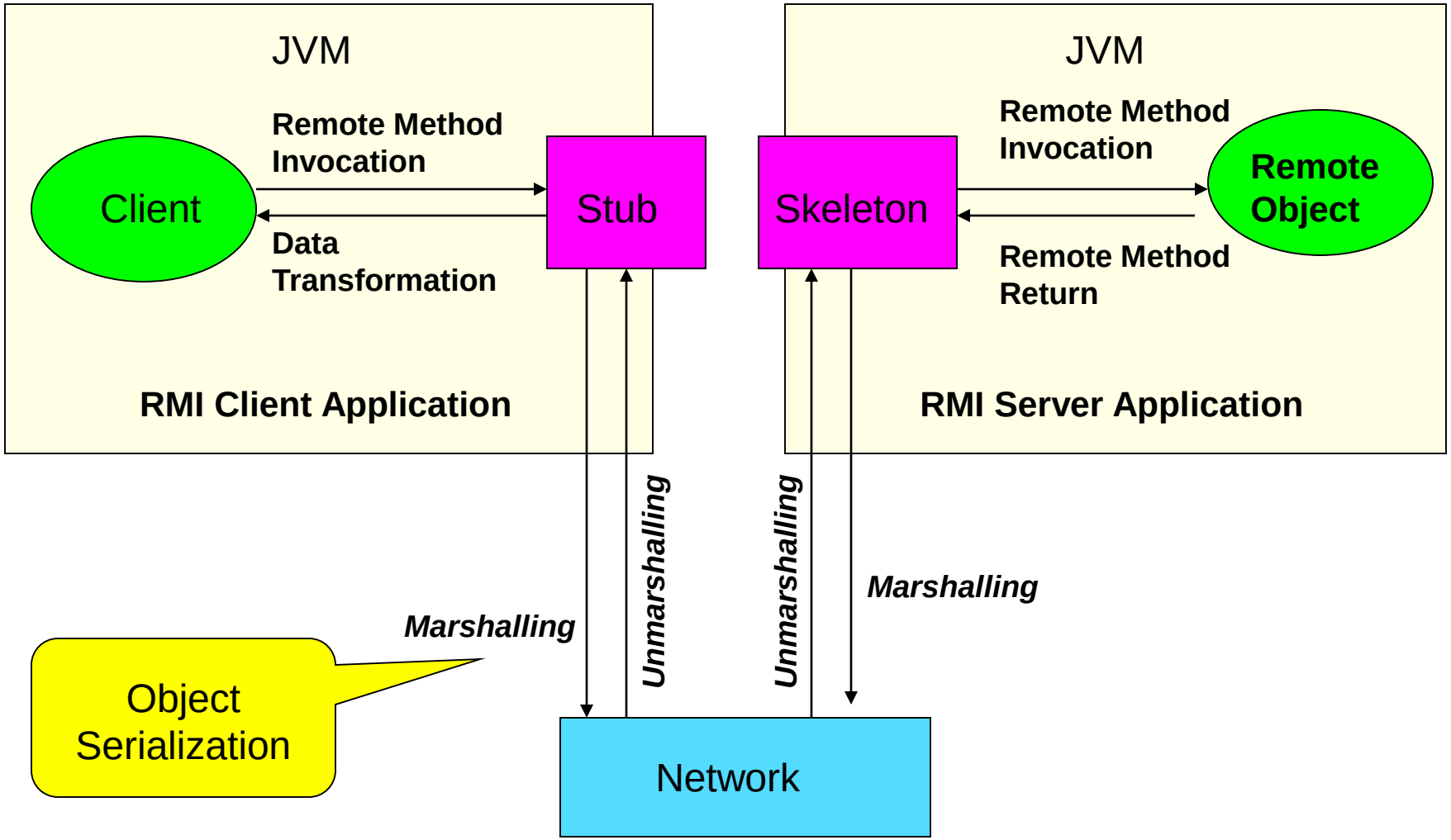
Contents

- Java RMI
- Communication of Remote Object and Client
- Writing Java RMI Application
- Hello Example

Java Remote Method Invocation (RMI)

- Can use objects on remote different run-time environments as like objects on a local run-time environment
- Abstraction of low-level network code on distributed network to provide developers an environment where they focus on their application development.

Communication of Remote Object and Client



Stub and Skeleton

- Stubs and Skeleton
  - When RMI client invokes a remote method of a remote object, it uses stub reference of the remote object instead of remote object reference.
  - For marshalling and unmarshalling of stub and skeleton, object serialization and deserialization are used.
  - Condition of Object for Object Serialization
    - The class must implement the java.io.Serializable interface.
    - The members of the class should be serializable. If one or more of the members are not to be serialized, they should be marked as transient.

```
public class Employee implements java.io.Serializable {
   public String name;
   public String address;
   public transient int SSN;
   public void mailCheck() {
      System.out.println("Mailing a check to " + name + " " + address);
   }
}
```

Writing Java RMI Application

- Writing RMI Application
  - Definition of Remote Interface
  - Definition of Remote Implementation Class
  - Write RMI Server Application
  - Write Client Application
- Compile and Run the Application
  - Compilation of the Implementation Class
  - Creation of Stub and Skeleton using "rmic" command
  - Compilation of the Server Application
  - Run the RMI Registry and Start the Server Program
  - Compilation of the Client Program
  - Run the Client

Hello Example : RMI

Interface

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

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

Hello Example : RMI

Implementation (Server)

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

public class HelloImpl extends UnicastRemoteObject
    implements Hello {

    public HelloImpl() throws RemoteException {
        super();
    }

    public String sayHello() {
        return  "Hello World!";
    }

    public static void main(String args[]) {

        // Create and install a security manager
        if (System.getSecurityManager() == null) {
                    System.setSecurityManager(new
RMISecurityManager());
        }
```

"HelloImpl" object

For using host server and port →
"//hostserver:4321/HelloServer"
LocateRegistry.createRegistry(4321);

```
try {
                    HelloImpl obj = new HelloImpl();
                    // Bind this object instance to the
name "HelloServer"
                    Naming.rebind("HelloServer", obj);
                    System.out.println("HelloServer bound
in registry");
        } catch (Exception e) {
                    System.out.println("HelloImpl err: " +
e.getMessage());
                    e.printStackTrace();
        }
    }
}
```

Compile & Skeleton Creation :

% javac Hello.java

% javac HelloImpl.java

(% rmic HelloImpl)

Deprecated! At 1.7

Hello Example : RMI

A Client Application

```
import java.rmi.Naming;
import java.rmi.RemoteException;

public class HelloClient {

  public static void main(String args[]) {
    String message = "Hello: This is my test message";

    // "obj" is the identifier that we'll use to refer
    // to the remote object that implements the "Hello"
    // interface
    Hello obj = null;

    try {
                    obj = (Hello)Naming.lookup("//" + "/HelloServer");
                    message = obj.sayHello();
                } catch (Exception e) {
                    System.out.println("HelloClient exception: " + e.getMessage());
                    e.printStackTrace();
                }
    System.out.println("Message = " + message);
  } // end of main
} // end of HelloClient
```

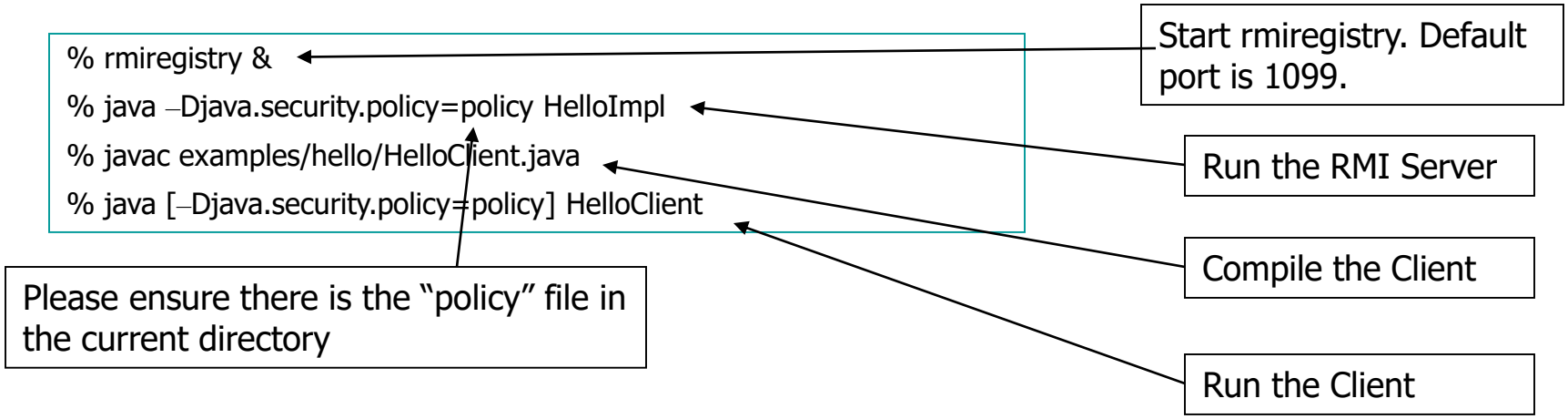
For using host server and port → **"//hostserver:4321/HelloServer"**

Hello Example : RMI

File "policy"

```
grant {
        // Allow everything for now
        permission java.security.AllPermission;
};
```

Start Registry Server & Run Server and Client



RMI in Eclipse

http://www.ejbtutorial.com/java-rmi/a-step-by-step-implementation-tutorial-for-java-rmi