Parul® University

Advanced Java Technology

Mr. Prashant Sahatiya, Assistant Professor
Information Technology Engineering

Parul®
University

DIGITAL LEARNING CONTENT

CHAPTER-4

JAVA RMI Programming

Outline

- RMI architecture
- RMI registry
- Writing distributed application with RMI
- Naming services
- Naming And Directory Services
- Overview of JNDI
- Object serialization and Internationalization

What is RMI?

- The **RMI (Remote Method Invocation)** is an API that provides a mechanism to create **distributed** application in java. The RMI allows an object to invoke methods on an object running in another JVM.
- The RMI provides remote communication between the applications using two objects **stub** and **skeleton**.

Understanding stub and skeleton

- RMI uses **stub** and **skeleton** object for communication with the remote object.

- A remote object is an object whose method can be invoked from another JVM. Let's understand the stub and skeleton objects:

stub

- The stub is an object, acts as a gateway for the client side. All the outgoing requests are routed through it. It resides at the client side and represents the remote object. When the caller invokes method on the stub object, it does the following tasks:

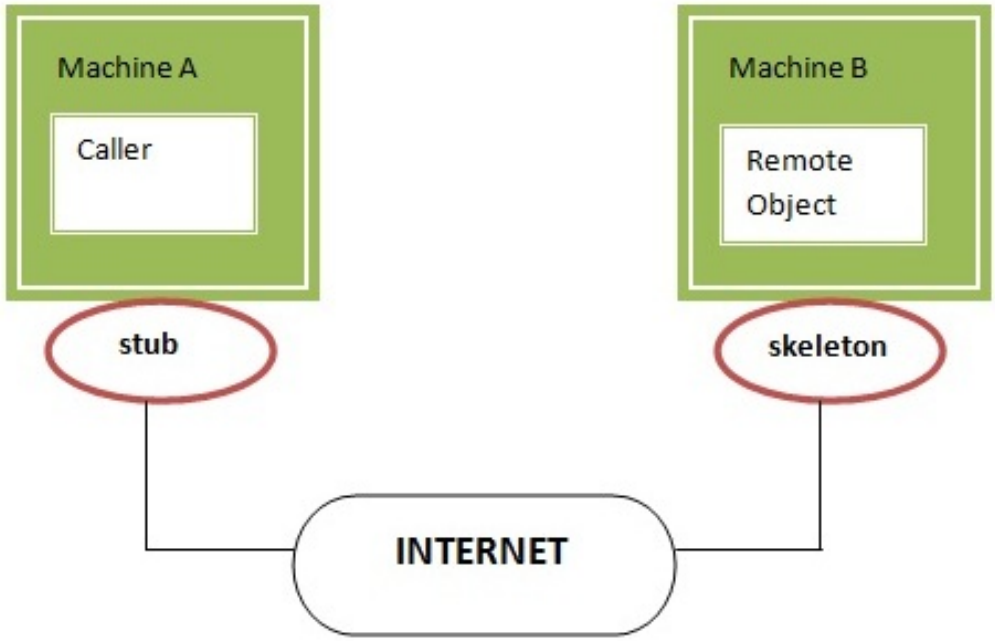
- It initiates a connection with remote Virtual Machine (JVM),
- It writes and transmits (marshals) the parameters to the remote Virtual Machine (JVM),
- It waits for the result
- It reads (unmarshals) the return value or exception, and
- It finally, returns the value to the caller.

skeleton

- The skeleton is an object, acts as a gateway for the server side object. All the incoming requests are routed through it. When the skeleton receives the incoming request, it does the following tasks:

- It reads the parameter for the remote method
- It invokes the method on the actual remote object, and
- It writes and transmits (marshals) the result to the caller.

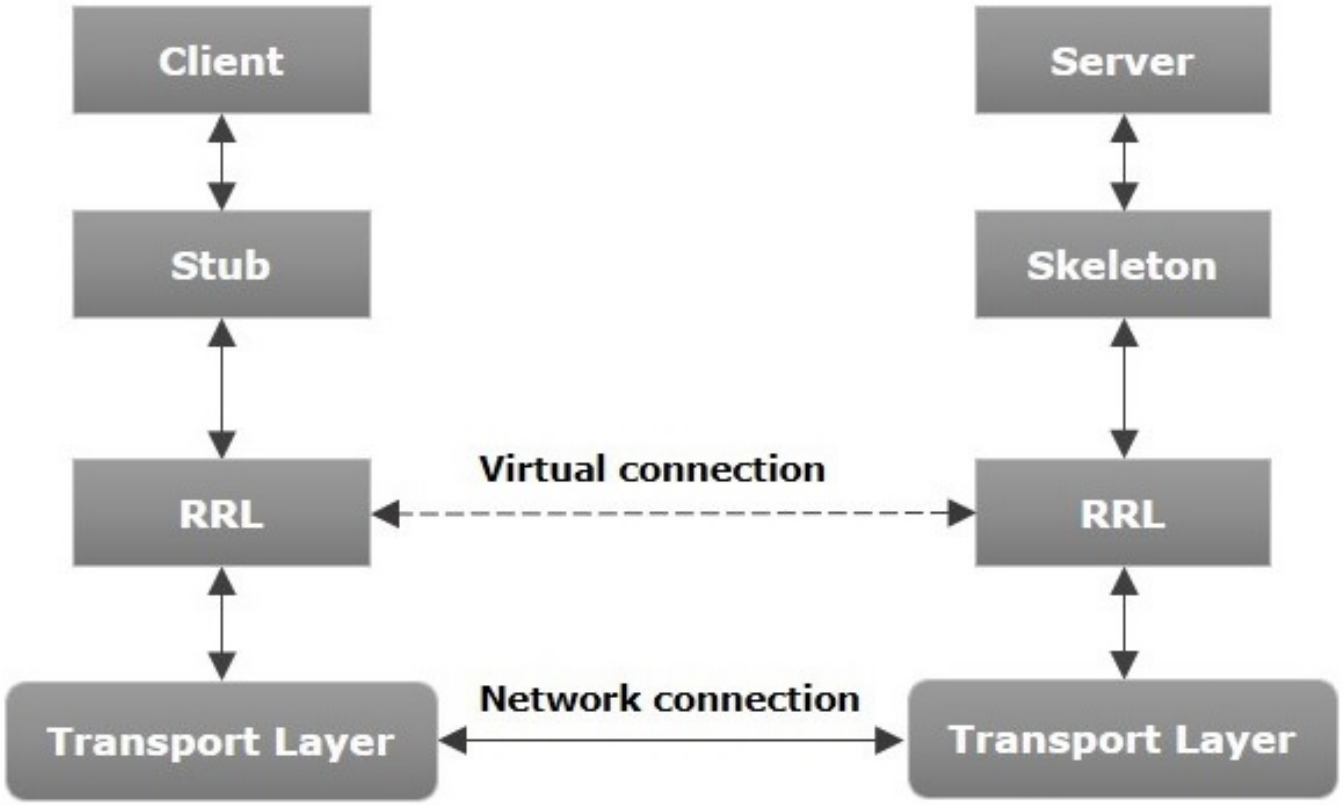
RMI


Machine A
Caller
stub
Machine B
Remote Object
skeleton
INTERNET

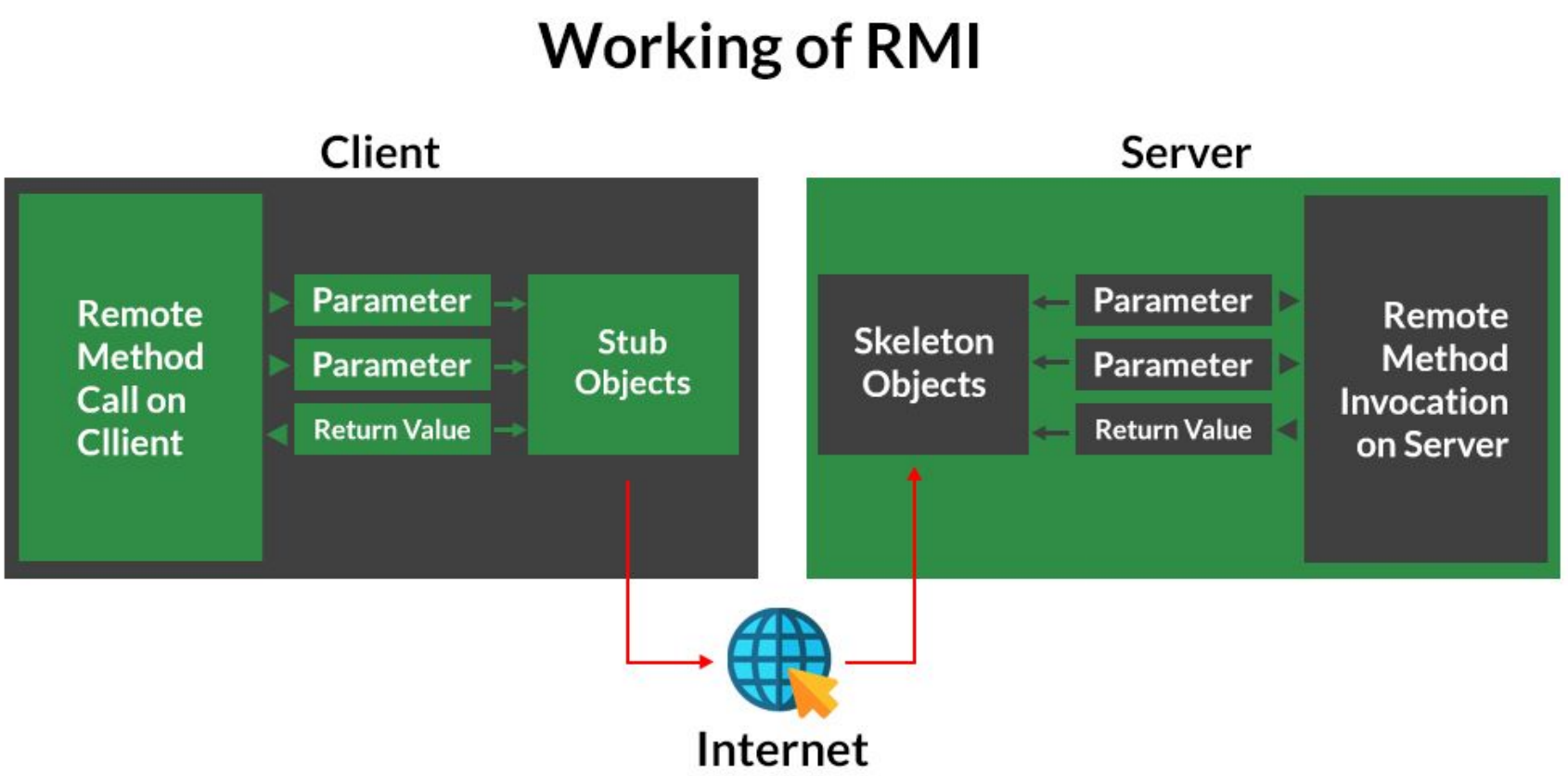
Architecture of an RMI Application

- In an RMI application, we write two programs, a server program (resides on the server) and a client program (resides on the client).
- Inside the server program, a remote object is created and reference of that object is made available for the client (using the registry).
- The client program requests the remote objects on the server and tries to invoke its methods.

Architecture of an RMI Application



Working of an RMI Application



Let us now discuss the components of this architecture.

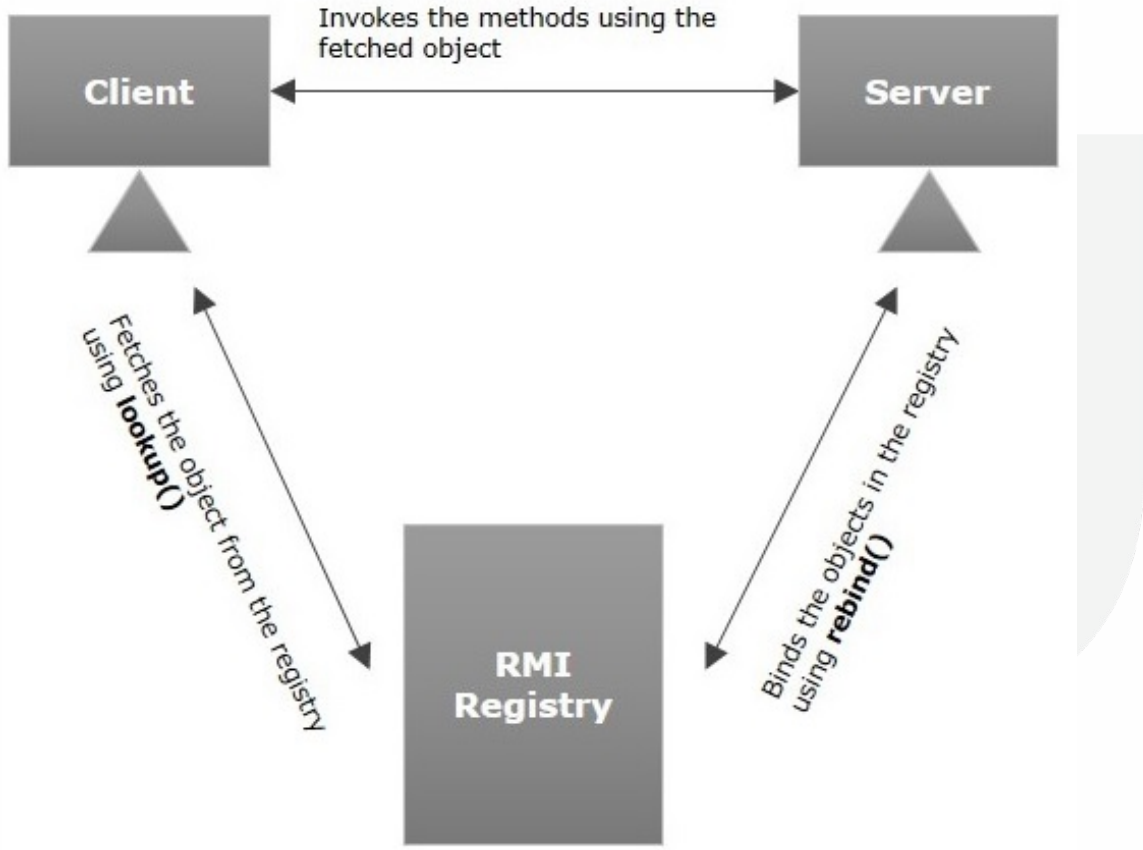
- **Transport Layer** – This layer connects the client and the server. It manages the existing connection and also sets up new connections.
- **Stub** – A stub is a representation (proxy) of the remote object at client. It resides in the client system; it acts as a gateway for the client program.
- **Skeleton** – This is the object which resides on the server side. stub communicates with this skeleton to pass request to the remote object.
- **RRL(Remote Reference Layer)** – It is the layer which manages the references made by the client to the remote object.

RMI Registry

- RMI registry is a namespace on which all server objects are placed. Each time the server creates an object, it registers this object with the RMIregistry (using bind() or reBind() methods). These are registered using a unique name known as bind name.

- To invoke a remote object, the client needs a reference of that object. At that time, the client fetches the object from the registry using its bind name (using lookup() method).

RMI Registry



Goals of RMI

- To minimize the complexity of the application.
- To preserve type safety.
- Distributed garbage collection.
- Minimize the difference between working with local and remote objects.

java.rmi.Naming Class in Java

- Java.rmi.Naming class contains a method to bind, unbind or rebind names with a remote object present at the remote registry. This class is also used to get the reference of the object present at remote registries or the list of name associated with this registry.
- **Syntax:** Class declaration
  - public final class Naming extends Object

java.rmi.Naming Class in Java

- Let us do discuss some major methods of this class which are as follows:
- The most commonly used methods are described below as follows:

1. bind()
2. list()
3. lookup()
4. rebind()
5. unbind()

Method 1: bind()

- **Syntax:** static void bind(String name, remote object)
- **Parameters:** Name of the remote registry
- **Exceptions:**

1. AlreadyBoundException occurs when the name was already bounded
2. MalformedURLException occurs when the format of the name is not an URL
3. RemoteException occurs when the contact to remote registry fails
4. AccessException occurs when access to this operation is not permitted

Method 3: lookup() looks for the reference of the remote object to which this name is associated

- **Syntax:** static remote lookup(String name)
- **Parameters:** The name of the remote registry
- **Return Type:** The reference for a remote object
- **Exceptions:**

1. NotBoundException occurs when the name does not bound with the current operation
2. RemoteException occurs when the contact to remote registry fails
3. AccessException occurs when access to this operation is not permitted
4. MalformedURLException occurs when the format of the name is not an URL

Method 4: rebind() method rebinds this name with the associated remote object

- **Syntax:** static void rebind(String name, remote object)
- **Parameters:** It takes two parameters namely name and object.

1. The name of the remote registry
2. The remote object to associate with the name

- **Exceptions:**

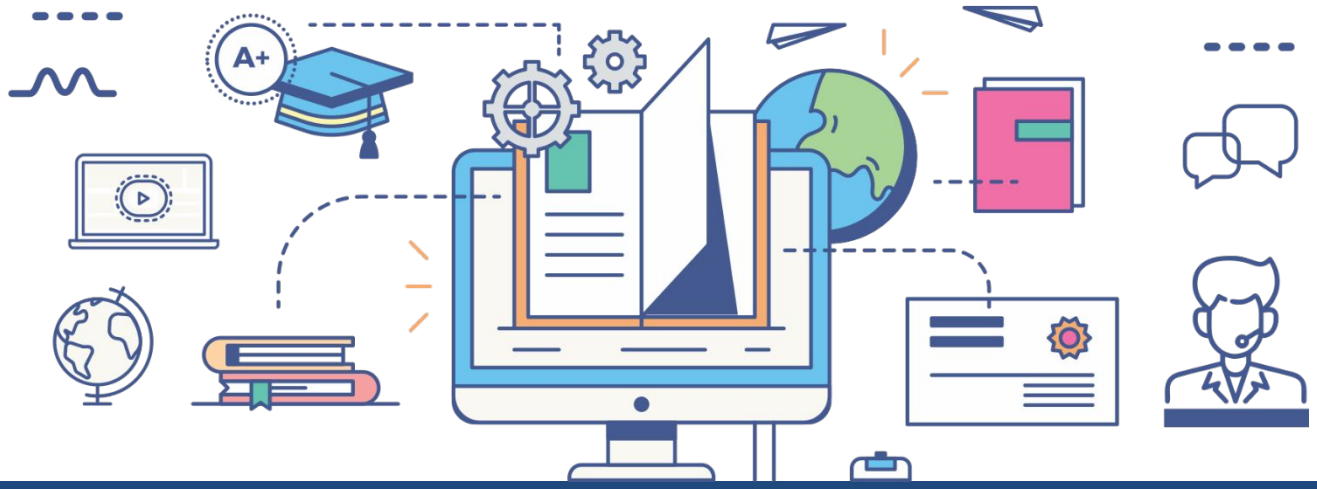
1. MalformedURLException occurs when the format of the name is not an URL
2. RemoteException occurs when the contact to remote registry fails
3. AccessException occurs when access to this operation is not permitted

Method 5: unbind() unbinds this name with the associated remote object

- **Syntax:** static void unbind(String name)
- **Parameters:** The name of the remote registry.
- **Exceptions:**

1. NotBoundException occurs when the name does not bound with the current operation
2. MalformedURLException occurs when the format of the name is not an URL
3. RemoteException occurs when the contact to remote registry fails
4. AccessException occurs when access to this operation is not permitted

DIGITAL LEARNING CONTENT



Parul® University

www.paruluniversity.ac.in