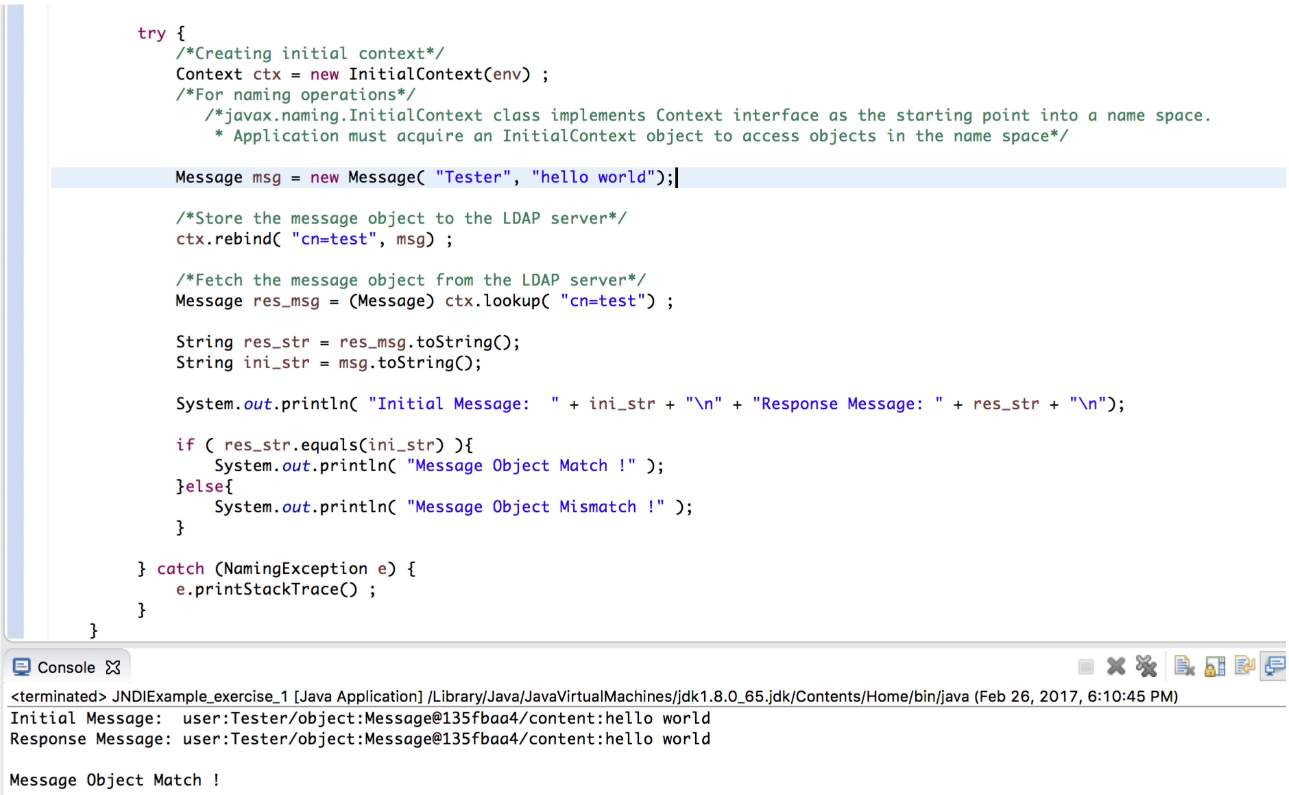
COMP3402 Tutorial 04 JNDI

1. Exercise 1:
    1. Requirements :
        1. Modify JNDI example
        2. Create a class called Message with 3 fields
            1. String user;
            2. String object;
            3. String content;
        3. Bind a Message object instead of a string and run the program
    2. **JNDIExample_exercise_1.java** and **Message.java** have been created to meet the requirements (files attached for your reference)
        1. A Message object called **msg** is created with **name="Tester"**, and **content="hello_world"**
        2. Register this Message object with the LDAP server
        3. Lookup this Message object from the LDAP server
        4. Compare the hashCode and content of 2 message objects
        5. Eventually, I conclude that they are the same Message Object because they have the same content and the same hashCode
        6. **Therefore, Message object is bound successfully**

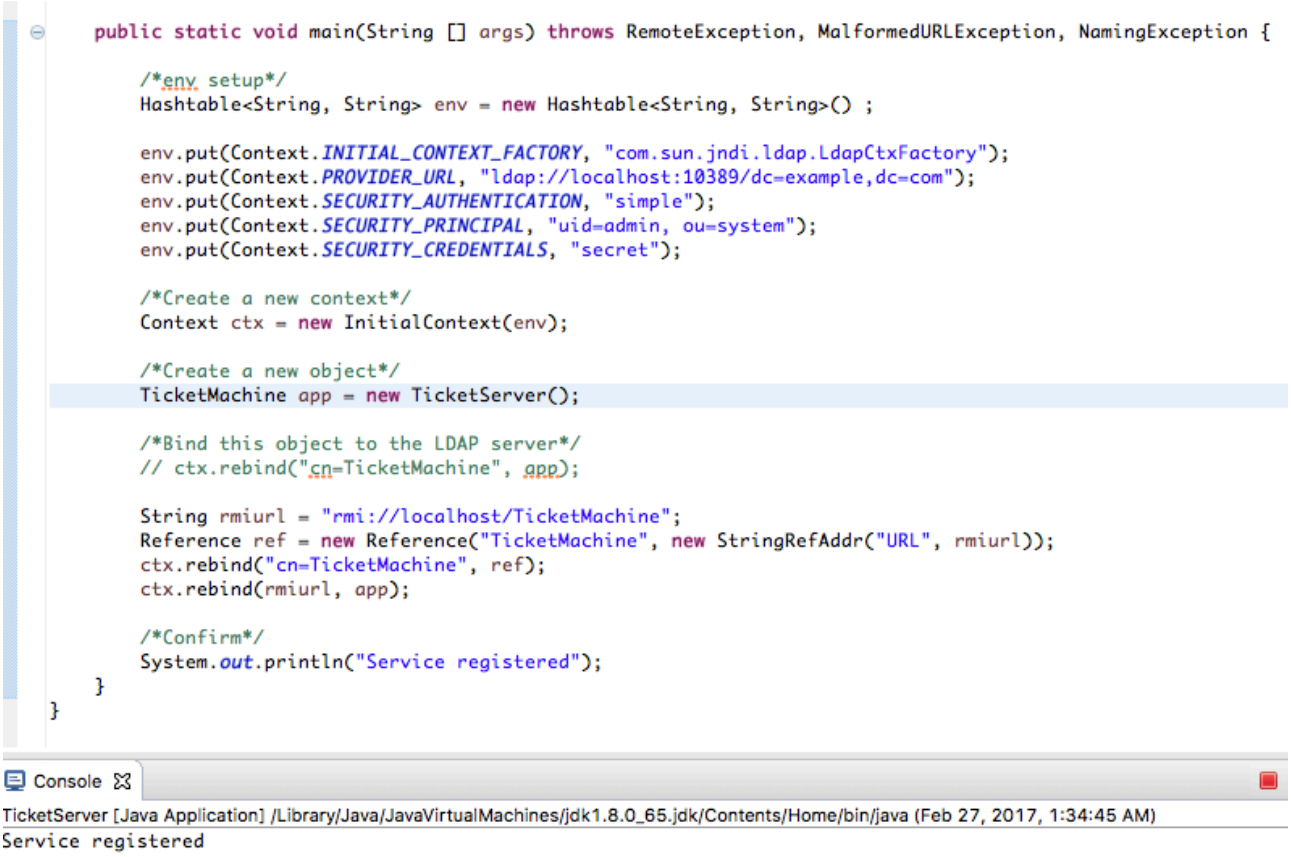


```
        try {
            /*Creating initial context*/
            Context ctx = new InitialContext(env) ;
            /*For naming operations*/
               /*javax.naming.InitialContext class implements Context interface as the starting point into a name space.
                * Application must acquire an InitialContext object to access objects in the name space*/

            Message msg = new Message( "Tester", "hello world");

            /*Store the message object to the LDAP server*/
            ctx.rebind( "cn=test", msg) ;

            /*Fetch the message object from the LDAP server*/
            Message res_msg = (Message) ctx.lookup( "cn=test") ;

            String res_str = res_msg.toString();
            String ini_str = msg.toString();

            System.out.println( "Initial Message:  " + ini_str + "\n" + "Response Message: " + res_str + "\n");

            if ( res_str.equals(ini_str) ){
                System.out.println( "Message Object Match !" );
            }else{
                System.out.println( "Message Object Mismatch !" );
            }

        } catch (NamingException e) {
            e.printStackTrace() ;
        }
    }
```

```
<terminated> JNDIExample_exercise_1 [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_65.jdk/Contents/Home/bin/java (Feb 26, 2017, 6:10:45 PM)
Initial Message:  user:Tester/object:Message@135fbaa4/content:hello world
Response Message: user:Tester/object:Message@135fbaa4/content:hello world

Message Object Match !
```

2. Exercise 2:
    1. Requirement: Use RMI and bind the remote object to LDAP server
    2. **TicketServer.java** has been modified as follow
    3. Run the RMI service in background

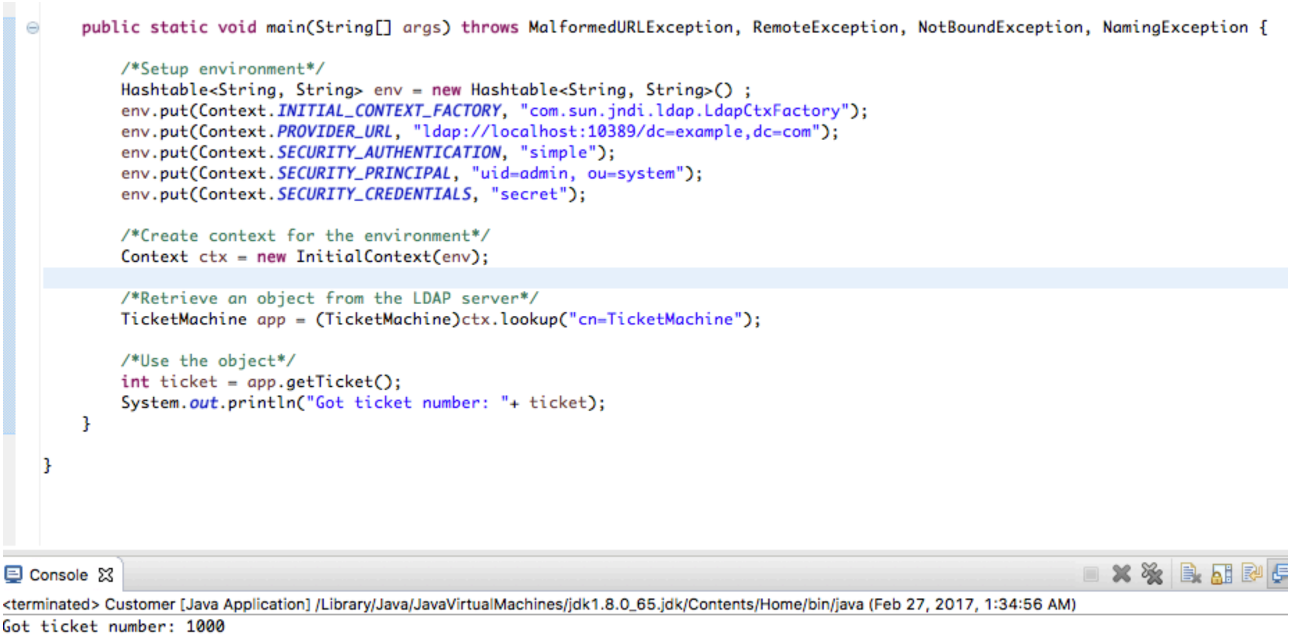
4. The image shows that the service is registered successfully



```
    public static void main(String [] args) throws RemoteException, MalformedURLException, NamingException {

        /*env setup*/
        Hashtable<String, String> env = new Hashtable<String, String>() ;

        env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
        env.put(Context.PROVIDER_URL, "ldap://localhost:10389/dc=example,dc=com");
        env.put(Context.SECURITY_AUTHENTICATION, "simple");
        env.put(Context.SECURITY_PRINCIPAL, "uid=admin, ou=system");
        env.put(Context.SECURITY_CREDENTIALS, "secret");

        /*Create a new context*/
        Context ctx = new InitialContext(env);

        /*Create a new object*/
        TicketMachine app = new TicketServer();

        /*Bind this object to the LDAP server*/
        // ctx.rebind("cn=TicketMachine", app);

        String rmiurl = "rmi://localhost/TicketMachine";
        Reference ref = new Reference("TicketMachine", new StringRefAddr("URL", rmiurl));
        ctx.rebind("cn=TicketMachine", ref);
        ctx.rebind(rmiurl, app);

        /*Confirm*/
        System.out.println("Service registered");
    }
}
```

```
TicketServer [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_65.jdk/Contents/Home/bin/java (Feb 27, 2017, 1:34:45 AM)
Service registered
```

5. The customer successfully obtains the remote object



```
    public static void main(String[] args) throws MalformedURLException, RemoteException, NotBoundException, NamingException {

        /*Setup environment*/
        Hashtable<String, String> env = new Hashtable<String, String>() ;
        env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
        env.put(Context.PROVIDER_URL, "ldap://localhost:10389/dc=example,dc=com");
        env.put(Context.SECURITY_AUTHENTICATION, "simple");
        env.put(Context.SECURITY_PRINCIPAL, "uid=admin, ou=system");
        env.put(Context.SECURITY_CREDENTIALS, "secret");

        /*Create context for the environment*/
        Context ctx = new InitialContext(env);

        /*Retrieve an object from the LDAP server*/
        TicketMachine app = (TicketMachine)ctx.lookup("cn=TicketMachine");

        /*Use the object*/
        int ticket = app.getTicket();
        System.out.println("Got ticket number: "+ ticket);
    }

}
```

```
<terminated> Customer [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_65.jdk/Contents/Home/bin/java (Feb 27, 2017, 1:34:56 AM)
Got ticket number: 1000
```