Module 06 – Java File IO

Danairat T.
Line ID: Danairat
FB: Danairat Thanabodithammachari
+668-1559-1446

Fundamental Java Programming

The Course Outline

Module 01 – Introduction to Java

Module 02 – Basic Java Programming

Module 03 – Control Flow and Exception Handling

Module 04 – Object Oriented in Java

Module 05 – Java Package and Access Control

Module 06 – Java File IO

Module 07 – Java Networking

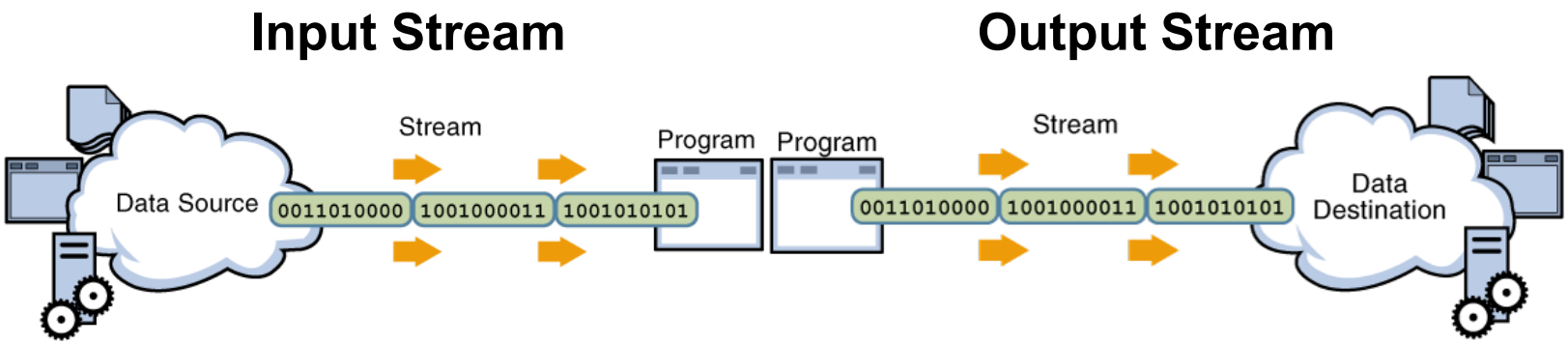
Module 08 – Java Threading

Module 06 – Java File IO

- IO Stream
- Byte Stream
- Character Stream
- Listing Directory Objects (Directory and File)
- Creating Directory and File
- Deleting Directory and File
- Java Console Stream

I/O Streams

An *I/O Stream* represents an input source or an output destination. A stream can represent many different kinds of sources and destinations, including disk files, devices, other programs, and memory arrays.



Byte Streams

The low level File IO process is using Byte Stremes; FileInputStream and FileOutputStream

```
package com.mycompany.fileio;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class CopyBytes {
    public static void main(String[] args) throws IOException {
        FileInputStream in = null;
        FileOutputStream out = null;
        try {
            in = new FileInputStream("inputfile.txt");
            out = new FileOutputStream("outagain.txt");
            int c;
            while ((c = in.read()) != -1) {
                out.write(c);  System.out.println(c);
            }
        } finally {
            if (in != null) {
                in.close();
            }
            if (out != null) {
                out.close();
            }
        }
    }
}
```

Character Stream with Line Reader

```
package com.mycompany.fileio;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.StringTokenizer;

public class CopyLinesBuffered {
   public static void main(String[] args) throws IOException {
      BufferedReader inputStream = null;
      BufferedWriter outputStream = null;
      try {
         inputStream =
            new BufferedReader(new FileReader("inputfile.txt"));
         outputStream =
            new BufferedWriter(new FileWriter("lineoutput.txt"));
         String l;
```

```
         while ((l = inputStream.readLine()) != null) {
                  outputStream.write(l);
                  System.out.println("line data->"+l);
                  StringTokenizer st = new StringTokenizer(l," ");
                  while (st.hasMoreTokens()) {
                        System.out.println(st.nextToken()+"|");
                     }
               }
            } finally {
               if (inputStream != null) {
                  inputStream.close();
               }
               if (outputStream != null) {
                  outputStream.close();
               }
            }
      }
}
```

Listing Directory Objects

```
import java.io.File;
public class ListDirectoryObjects {
    public static void main(String[] a) {
        File myFile = new File("C:" + File.separator);

        // reture String
        for (String s : myFile.list()) {
            System.out.println(s);
        }

        // return File obj for next iterative
        for(File s: myFile.listFiles()){
          System.out.println(s);
        }
    }
}
```

Creating Directory and File

```
import java.io.File;
public class FileDemo {
    public static void main(String[] a)throws Exception {
      File file = new File("d:\\JavaDemo\\JavaDemoSub");
      file.mkdirs();
      file = new File("d:\\JavaDemo\\JavaDemoSub\\test.txt");
      file.createNewFile();
    }

}
```

Deleting Directory and File

```
import java.io.File;

public class DeleteFile_Dir_Demo {
    public static void main(String args[]) {
        File f = new File("D:" + File.separator + "temp4" + File.separator + "a.txt");
        if (f.exists()) {
            f.delete();
        }
    }
}
```

Delete File Recurrsively

```
import java.io.File;
public class DeleteDirectoryTree {
    public static void main(String args[]) {
      deleteDirectory(new File("v:\\delete_temp\\delete_demo"));
    }
    static public boolean deleteDirectory(File path) {
      if( path.exists() ) {
        File[] files = path.listFiles();
        for(int i=0; i<files.length; i++) {
           if(files[i].isDirectory()) {
             deleteDirectory(files[i]);
           }
           else {
             files[i].delete();
           }
        }
      }
      return( path.delete() );
    }
}
```

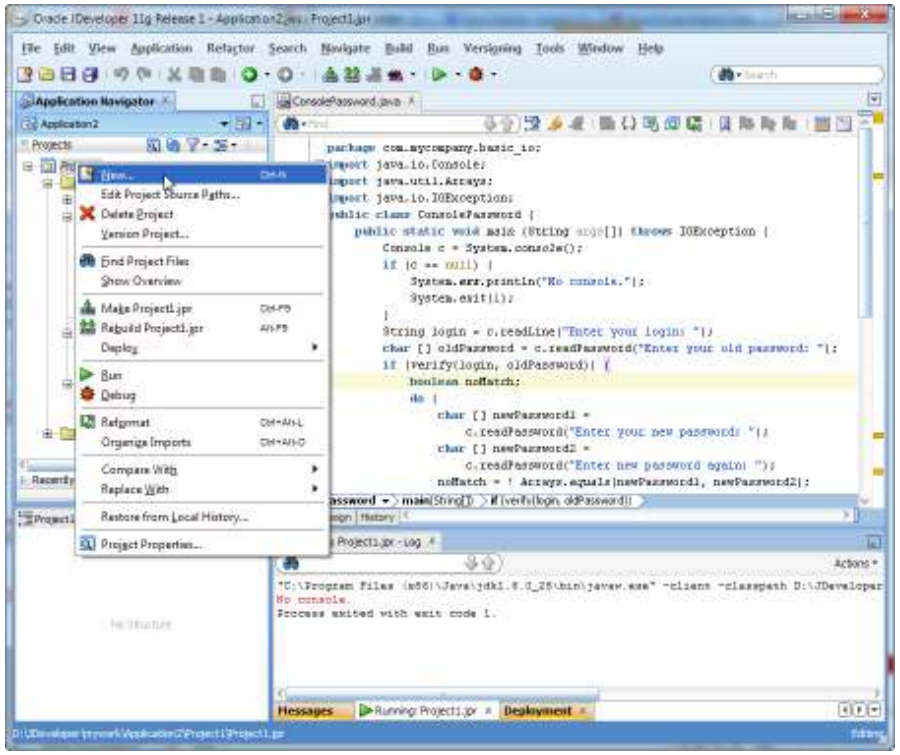
Java Console

Start from Java SE6, It is a predefined object of type Console that has most of the features provided by the Standard Console Streams.

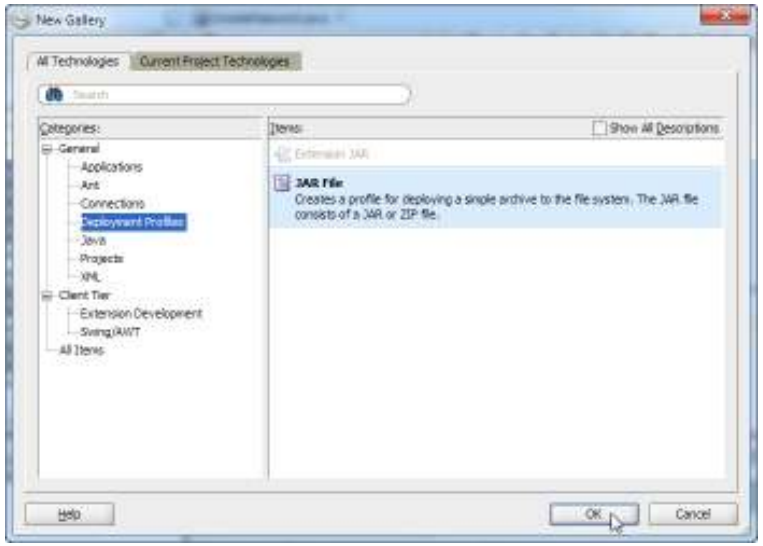
```
import java.io.Console;
import java.util.Arrays;
import java.io.IOException;
public class ConsolePassword {
   public static void main (String args[]) throws IOException {
      Console c = System.console();
      if (c == null) {
         System.err.println("No console.");
         System.exit(1);
      }
      String login = c.readLine("Enter your login: ");
      char [] oldPassword = c.readPassword("Enter your old
password: ");
      if (verify(login, oldPassword)) {
         boolean noMatch;
         do {
            char [] newPassword1 =
               c.readPassword("Enter your new password: ");
            char [] newPassword2 =
               c.readPassword("Enter new password again: ");
            noMatch = ! Arrays.equals(newPassword1,
newPassword2);
            if (noMatch) {
                  c.format("Passwords don't match. Try
again.");
               } else {
                  change(login, newPassword1);
                  c.format("Password changed.", login);
               }
               Arrays.fill(newPassword1, ' '); // clear data
               Arrays.fill(newPassword2, ' '); // clear data
            } while (noMatch);
         }
         Arrays.fill(oldPassword, ' '); // clear data
      }
      //Dummy verify method.
      static boolean verify(String login, char[] password) {
         return true;
      }
      //Dummy change method.
      static void change(String login, char[] password) {}
}
```

Java Console – Deploy and Test

1.) Select “New…” from project menu

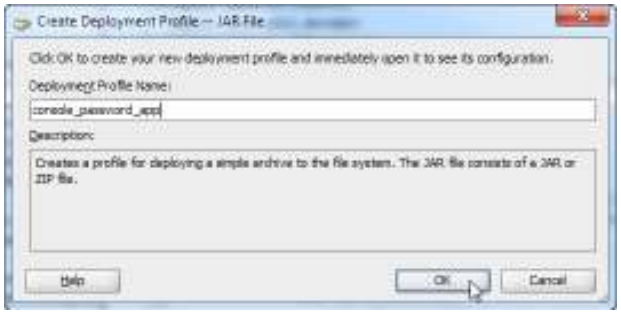


2.) Select “Deployment Profiles” -> “JAR File”

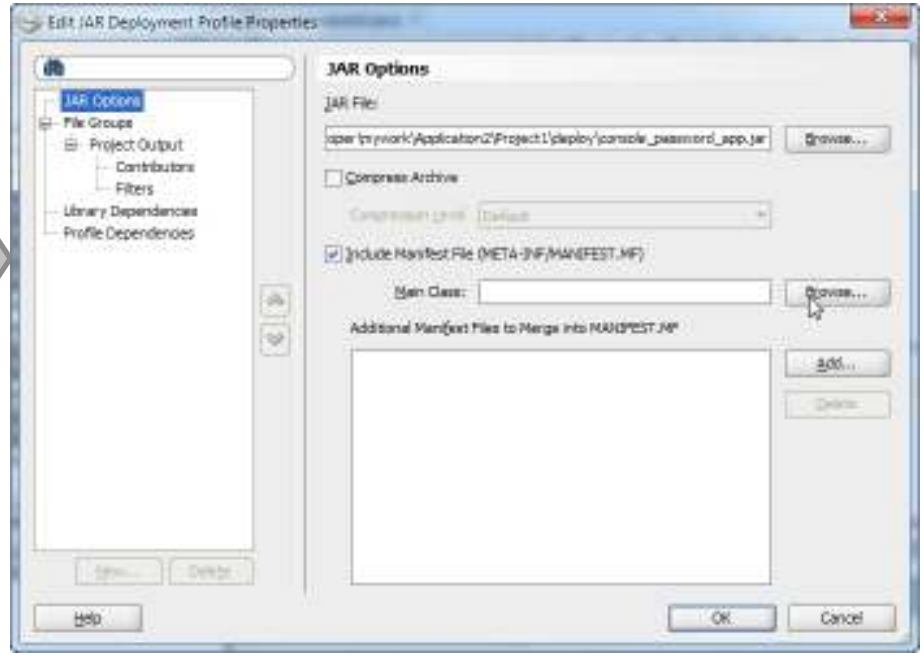


Java Console – Deploy and Test

3.) Enter Application Name "console_password_app"

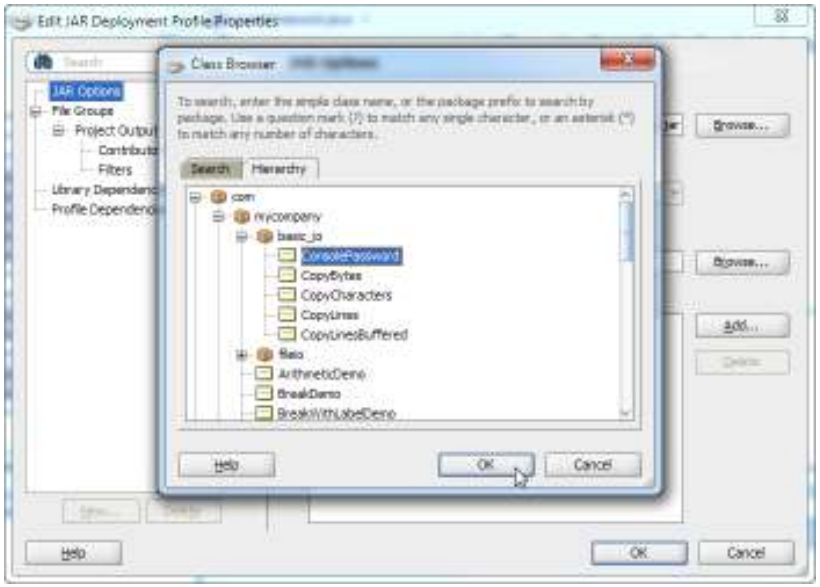


4.) Click "Browse" to select the start class file

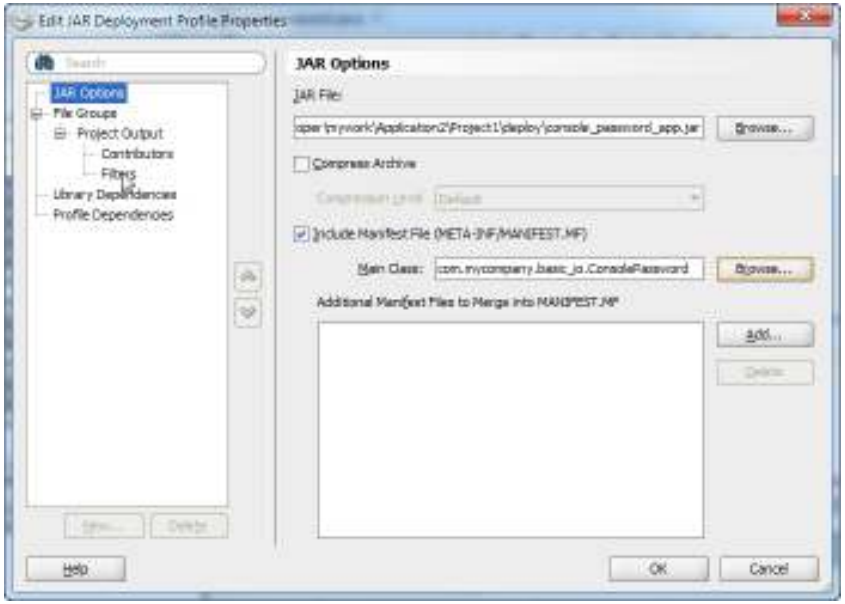


Java Console – Deploy and Test

5.) Select the starting class “ConsolePassword”

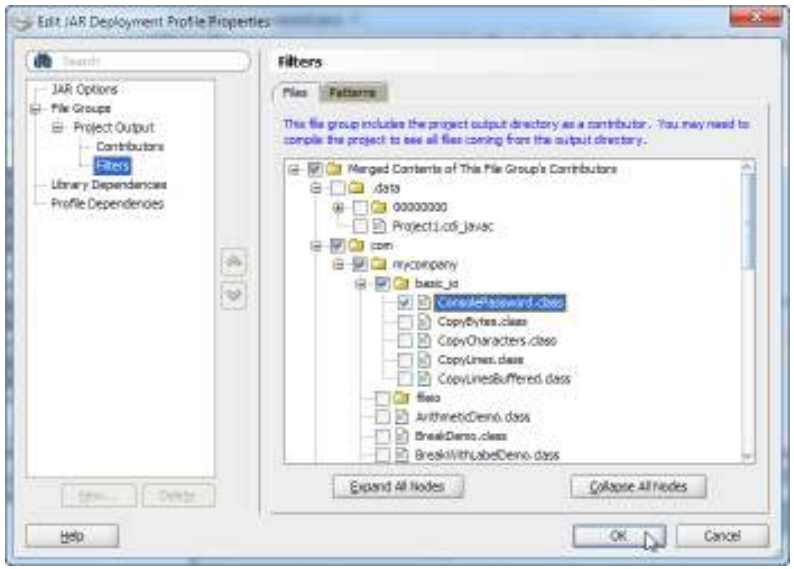


6.) Click “Filters”

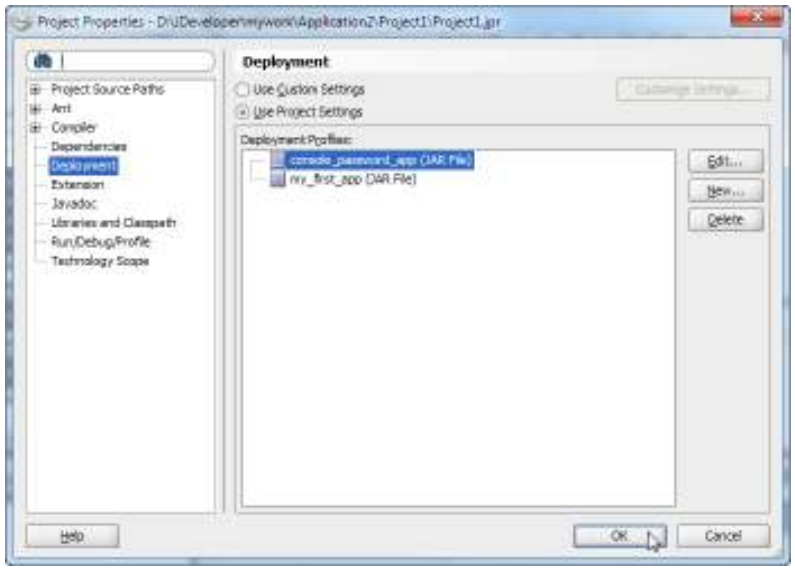


Java Console – Deploy and Test

7.) Check only the required class. “ConsolePassword.java”

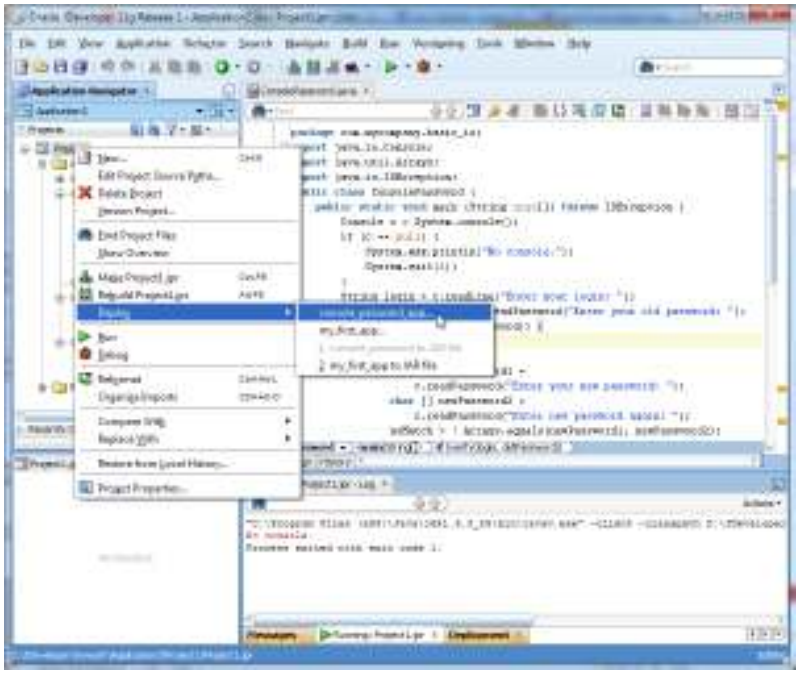


8.) Click “OK”

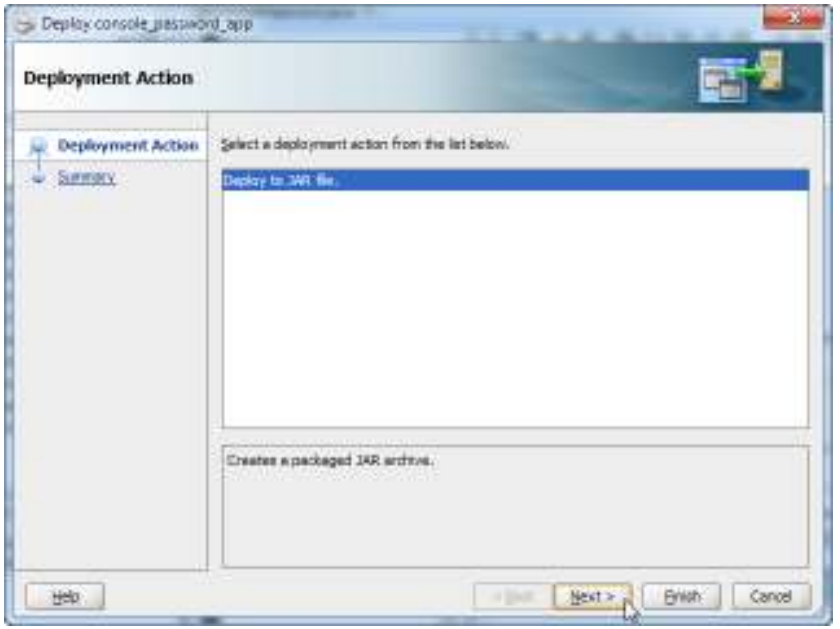


Java Console – Deploy and Test

9.) Click “Deploy” and select “console_password_app”

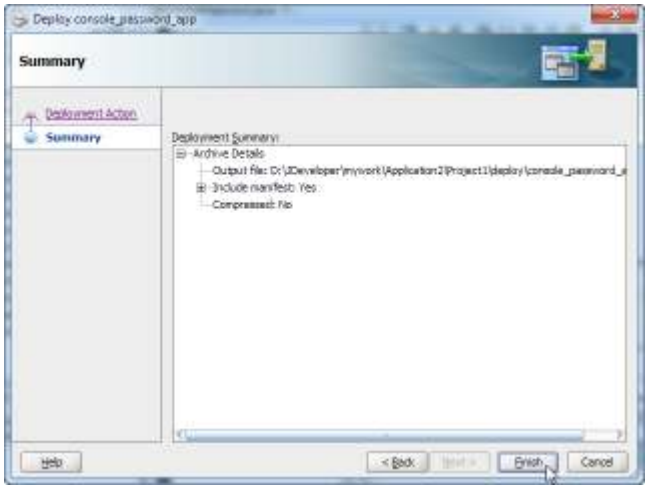


10.) Click “Next”

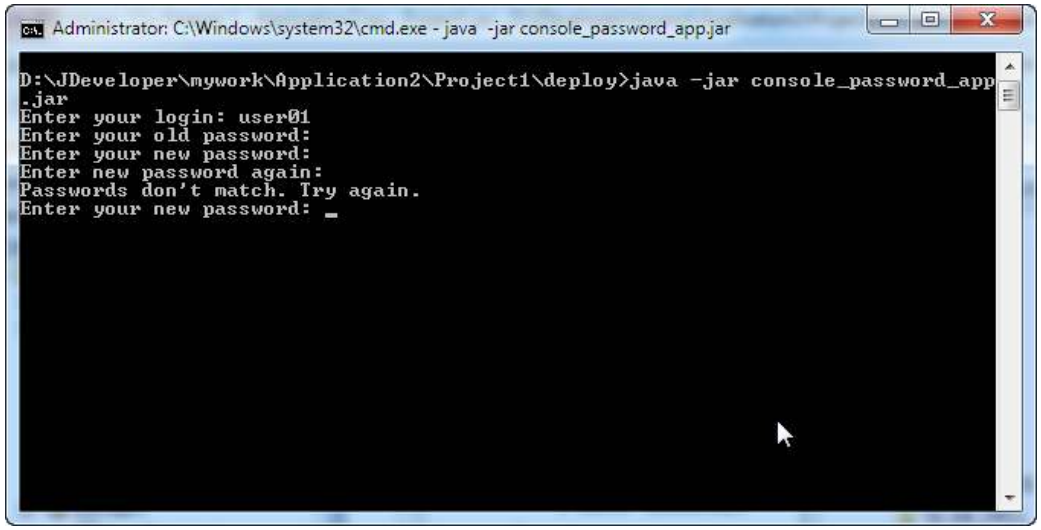


Java Console – Deploy and Test

11.) Click “Finish”



12.) Execute application from command line mode.
java –jar console_password_app.jar



Thank you

Danairat T.
Line ID: Danairat
FB: Danairat Thanabodithammachari
+668-1559-1446