This doc describes the steps involved in creating a JavaWebStart(JAWS) application file from a simple AWT Swing program and deploying it in server for user download.

**A brief introduction to JAWS:**

The Java Web Start software allows you to download and run Java applications from the web. It provides an easy one-click activation of applications, guarantees that you are always running the latest version of the application, and eliminates complicated installation or upgrade procedures.

Java Web Start is an implementation of the Java Network Launching Protocol (JNLP) specification. Java Web Start works by using a special file called the JNLP descriptor, which is a small XML document that contains elements and attributes that tell Java Web Start how to run the application. That JNLP file instructs Java Web Start software to download, cache, and run the application.

**Steps to create a JAWS application:**

1. Create a simple AWT + Swing program and package it into a .jar file.
2. Sign the newly created jar by creating a keystore.
3. Create the .jnlp file of our application.
4. Place these files inside a Web Server, such as Apache Tomcat 7.
5. Access our created .jar file from the web, download and execute the application.

The following descries the steps above with a sample program.

**Step1:**

* Create a simple AWT + Swing program to add two numbers as below:

// AddNumbers.java

package com.mypack;

import java.awt.FlowLayout;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JTextField;

public class AddNumbers implements ActionListener {

JFrame frame;

JPanel panel1;

JPanel panel2;

JButton addbutton, resetbutton;

JTextField num1, num2;

JLabel headerlabel, resultlabel;

public AddNumbers() {

prepareGui();

}

public static void main(String[] args) {

AddNumbers a = new AddNumbers();

a.init();

}

public void prepareGui() {

frame = new JFrame("My Frame");

headerlabel = new JLabel("", JLabel.CENTER);

resultlabel = new JLabel("", JLabel.CENTER);

num1 = new JTextField(6);

num2 = new JTextField(6);

addbutton = new JButton("ADD");

resetbutton = new JButton("RESET");

frame.setSize(300, 350);

frame.setLayout(new GridLayout(4, 1));

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

panel1 = new JPanel();

panel1.setLayout(new FlowLayout());

panel1.add(addbutton);

panel1.add(resetbutton);

panel2 = new JPanel();

panel2.setLayout(new FlowLayout());

panel2.add(num1);

panel2.add(num2);

frame.add(headerlabel);

frame.add(panel2);

frame.add(panel1);

frame.add(resultlabel);

frame.setVisible(true);

}

public void init() {

headerlabel.setText("ADDITION");

resultlabel.setText("RESULT : ");

addbutton.setActionCommand("add");

addbutton.addActionListener(this);

resetbutton.setActionCommand("reset");

resetbutton.addActionListener(this);

}

@Override

public void actionPerformed(ActionEvent e) {

String s = new String(e.getActionCommand());

if (s.equalsIgnoreCase("add")) {

try {

int a = Integer.parseInt(num1.getText());

int b = Integer.parseInt(num2.getText());

Integer c = a + b;

resultlabel.setText(c.toString());

} catch (NumberFormatException nfe) {

nfe.getMessage();

resultlabel.setText("Invalid Input");

}

} else if (s.equalsIgnoreCase("reset")) {

num1.setText("");

num2.setText("");

resultlabel.setText("");

}

}

}

// end of file

* Package it into a .jar file by executing the following command

jar -cf TestJnlp.jar \*.\*

This will package all the Java's classes into a new jar file, named "TestJnlp.jar".

**Step2:**

* Add a new keystore named "testkeys" using the command :

keytool -genkey -keystore testKeys -alias jdc

It will ask for a keystore password, first name, last name , organization's unit...etc..just fill them all.

* Now, attach newly generated keystore "testkeys" to your "TestJnlp.jar" file.

jarsigner -keystore testKeys TestJnlp.jar jdc

It will ask password for your newly created keystore.

* Next, copy "TestJnlp.jar" to Tomcat's default web server folder, for example, in Windows - C:\Program Files\Apache\Tomcat 6.0\webapps\ROOT

**Step3:**

* Create a new testadder.jnlp file, and put this as content:

<?xml version="1.0" encoding="utf-8"?>

<jnlp spec="1.0+" codebase="http://localhost:8085/" href="testadder.jnlp">

<information>

<title>Jnlp Testing</title>

<vendor>VENDOR</vendor>

<homepage href="http://localhost:8085/" />

<description>Testing Testing</description>

</information>

<security>

<all-permissions/>

</security>

<resources>

<j2se version="1.6+" />

<jar href="TestJnlp.jar" />

</resources>

<application-desc main-class="com.mypack.AddNumbers" />

</jnlp>

**Step4:**

* Copy Test.jnlp to your Tomcat default web server folder. C:\Program Files\Apache\Tomcat 6.0\webapps\ROOT

**Step5:**

* Access URL http://localhost:8085/Test.jnlp, it will prompt you to download the Test.jnlp file, just accept and double click on it.
* Click on the "Run" button to launch the AWT program.

**Note**  
  
If jnlp has no response, put the following code in your *web.xml*, which is located in the Tomcat conf folder.

[?](http://java.dzone.com/articles/java-web-start-jnlp-hello#about)

<mime-mapping

<extension>jnlp</extension>

<mime-type>application/x-java-jnlp-file</mime-type>

</mime-mapping>