Making Installer for component

Introduction

The following introduction and tutorial has been taken from the caGrid portal installer, this example will show how to create installer for your own component.

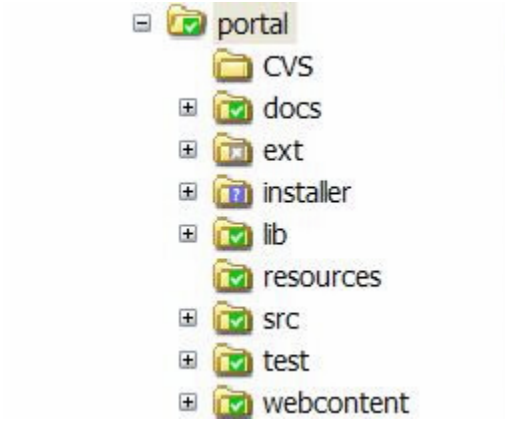
Steps

1. Download Installer.zip from the following cvs location

   http://gforge.nci.nih.gov/plugins/scmcvs/cvsweb.php/cagrid-1-0/Documentation/docs/installer/DeveloperGuide/installer.zip?cvsroot=cagrid-1-0

2. Create a folder installer and unzip the file in it.

3. The tutorial has taken portal project as an example. Copy installer folder from Step 2 at project level in the folder created. To demonstrate, I have taken caGrid/Portal tree shown in the screenshot below.



4. **Preparation:** Installer folder has following structure as depicted in the screenshot below.

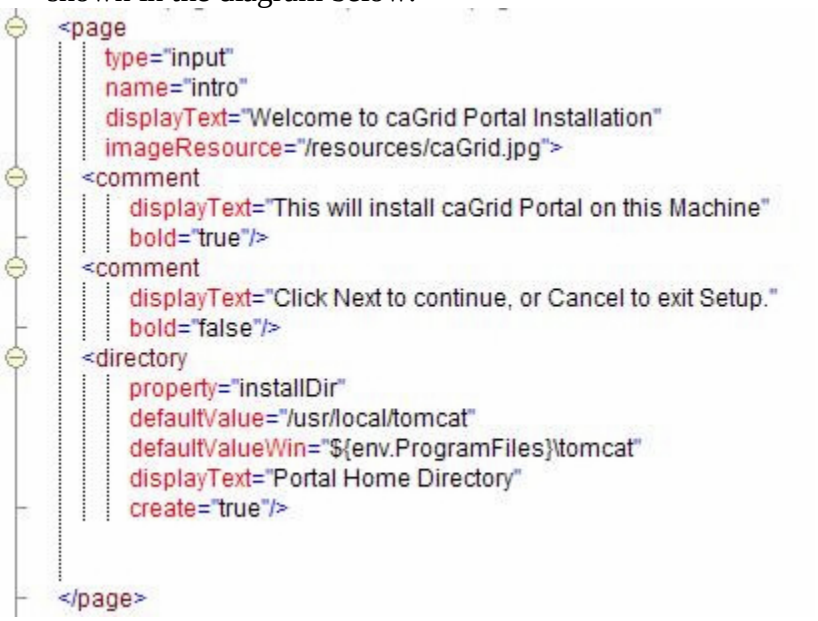
   a. Identify the properties to be configured at installer time.

b. Put all these properties with their default value in some property file (or you might have an existing project properties file, which can be used by installer)
c. In the above example, the name of the property file is portal-build.properties.
d. Choose any name for your installer. In this case, it has been named as "portalInstaller".

5. Edit build.xml file in installer folder to edit the following two properties.

```
<property name="installerName" value="portalInstaller"/>
<property name="project.property.file" value="../portal-build.properties"/>
```

6. Now open antinstall.config.xml inside antfiles folder. This file is used to define all the panels and their inputs for installer. The panels are defined between page tags. You can look at this file to understand how the panels are built declaratively.
   a. **The Welcome Page**: The welcome page panel is defined in this file as shown in the diagram below.



As can be seen on this panel above you can put comments and the directory in which this component will be installed. All the attributes of directory tag are self explanatory.

b. License Page

```
<!-- type="license" shows a license page to click through -->
<page
    type="license"
    name="license"
    displayText="License conditions"
    resource="/resources/GPL.txt"
    imageResource="/resources/caGrid.jpg">
</page>
```

As you can see that the license text file can be placed under resources folder which is inside template/installclasspath folder. At runtime the license conditions will be displayed from the license file.

c. **User Input Page**: The configuration parameters can be configured using this type of page. The properties are read from project properties file with their default values and are populated on this panel. These properties are stored in user home directory for later use. Please see the diagram below.

```
<!-- type="input" shows a list of editable options for the installer -->
<page
    type="input"
    name="properties"
    displayText="Configuration properties for database">
  <text
    property="hibernate.connection.url"
    displayText="Database URL"
    defaultValue="${custom.hibernate.connection.url}"/>
  <text
    property="hibernate.connection.username"
    displayText="User name"
    defaultValue="${custom.hibernate.connection.username}"/>
  <text
    property="hibernate.connection.password"
    displayText="Password"
    defaultValue="${custom.hibernate.connection.password}"/>
</page>
```

The important thing to note here is that the name of the property in this file should be the same as the name of the property in the project property file. Another important thing to note here is that the default values are populated using custom variable. If these properties are prefixed by custom tag then the default values will be picked up from the project properties file.

Note: There is no restriction on number of properties that you can populate using installer. As a matter of fact if the properties grow in number you can split them across multiple pages.

d. **Target page**: This panel is used to execute the selected Ant targets to perform the task. These targets are displayed as a checkbox on the panel. Only the targets that have been selected by the user will be executed. In the current example, the targets deployed such as portal, source code, documentation are shown in the figure below.

```
<page
    type="input"
    name="selector"
    displayText="Components to install">
  <comment
      displayText="The following  components will be installed"
      bold="true"/>

  <target
    displayText="caGrid Portal"
    target="deploy"
    defaultValue="true" force="false"/>
  <target
    displayText="Source code"
    target="deploySource"
    defaultValue="true" force="false"/>
  <target
    displayText="Documentation"
    target="deployDoc"
    defaultValue="true" force="false"/>

</page>
```

e. **Progress Page:** This page should not be modified and is for internal use.

```
<!-- page type="progress" shows a progress page with the install button
      Its target is set to cleanuptarget since it is the last page this target will be run last-->
<page
    type="progress"
    name="progress"
    showTargets="true"
    displayText="Installation progress"
    target="cleanuptarget">
</page>
```

7. The targets defined in step 6d, should exist in antfiles/build.xml file.

```
<target name="deploy">
<ant  dir="${basedir}/resources/release/portal" antfile="${basedir}/resources/release/build.xml" target="deployExploded">
</ant>
</target>

<target name="deploySource">
 <copy todir="${installDir}">
  <fileset dir="${basedir}/resources/release">
    <exclude name="docs/**"/>
  </fileset>
 </copy>
</target>
<target name="deployDoc">
  <copy todir="${installDir}">
    <fileset dir="${basedir}/resources/release">
      <include name="docs/**"/>
  </fileset>
  </copy>
</target>
```

Deploy source and doc targets will remain the same for all the projects but if needed developers can write custom targets here.
The deploy target will actually install the component wherever required. The deploy target will intern call the target in project build.xml file. This way the developers can reuse their targets defined for their projects and don't have to create targets specifically for installer. The following picture shows the chaining of the calls of targets.

antfiles/ant-install.xml

Cells targets in

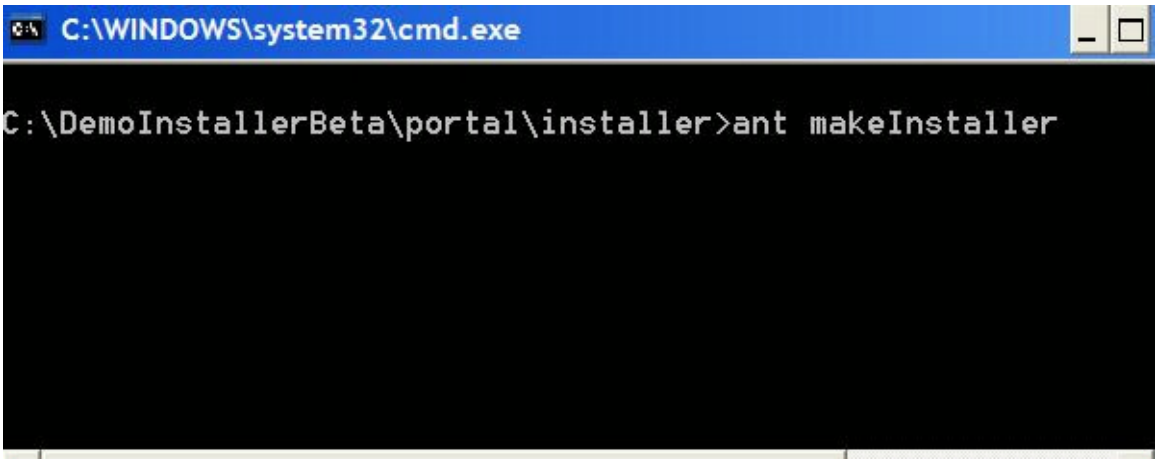
antfiles/build.xml

Cells targets in

Portal/build.xml

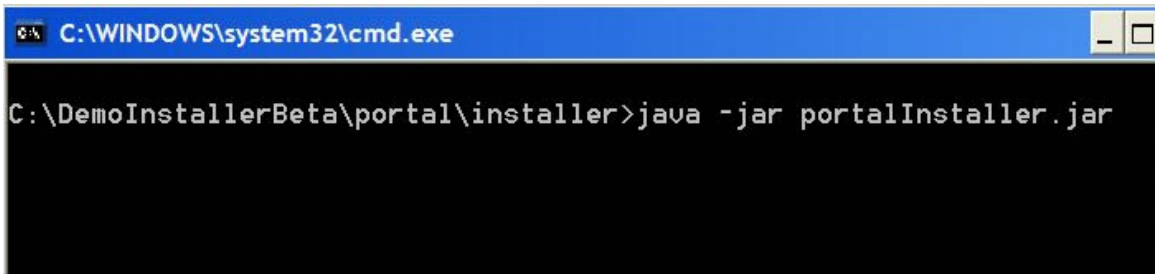
Caution: It is important to note that the configuration parameters which are captured using installer are stored in user.home directory with the same name as the project property file. It is developer's responsibility to include that file name in the Ant script file and replace the tokens wherever required.

8. Go to installer folder and bring the command prompt there and type "ant makeInstaller:". This step will build the installer.

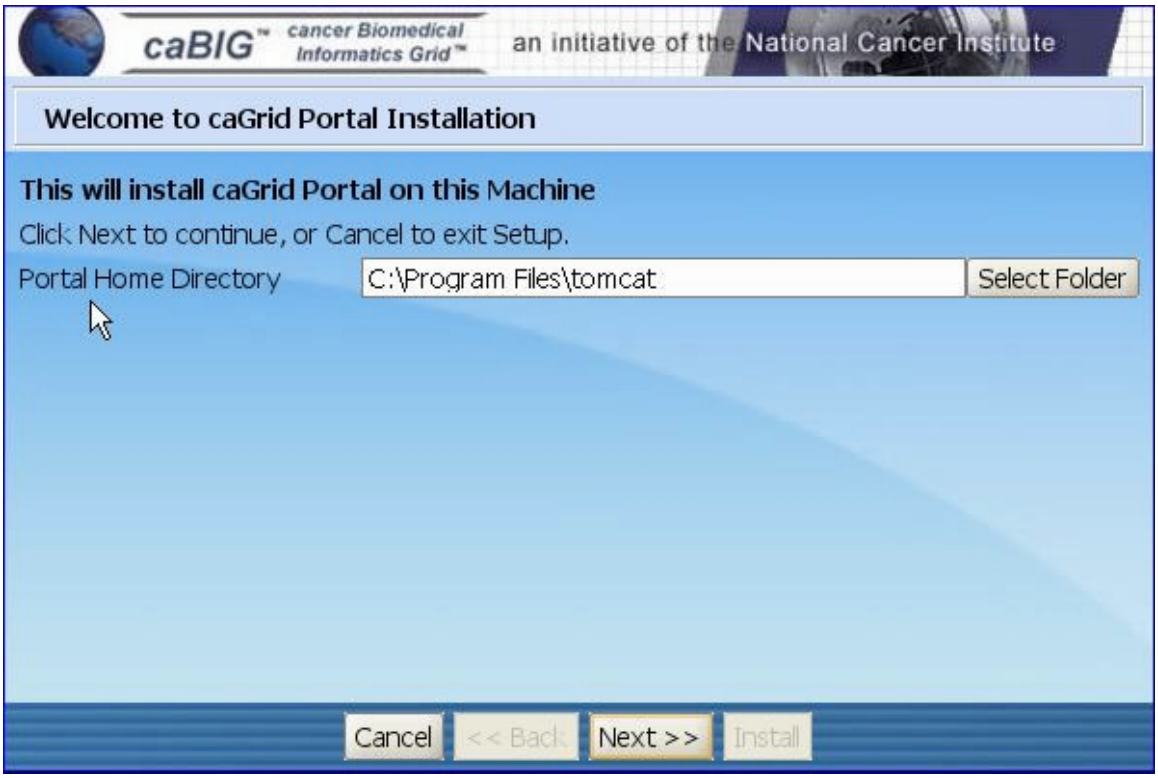


After this is executed successfully, portalinstaller.jar should appear in installer folder.

9. Using your installer: To launch your installer type the following command “java –jar portalInstaller.jar”



After executing this command, it will bring up the installer GUI.



Advance Tips

I want my installer to fail if Tomcat and/or Globus do not exist.
I want to bring up a splash page.
I want to have a conditional flow of pages.
How to install third party products?

a) **Bundle with installer**
b) **Download on the fly.**

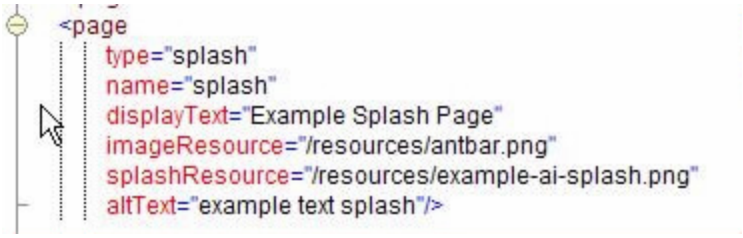
I have other questions.

I want my installer to fail if Tomcat and/or Globus both do not exist.

Build.xml file in antfiles folder has two targets called checkTomcat and checkGlobus. The init target can be made dependent on checkTomcat. If Tomcat does not exist the installation process will fail and same can be done for Globus as well.

I want to bring up a splash page.

This is the example that showcases how it can be done



I want to have a conditional flow of pages.

Please check the example script at the following location
http://gforge.nci.nih.gov/plugins/scmcvs/cvsweb.php/cagrid-1-0/Documentation/docs/installer/DeveloperGuide/exampleScripts/conditionalFlow/antinstall-config.xml?cvsroot=cagrid-1-0

How to install third party products?

c) **Bundle with installer**
d) **Download on the fly.**

Coming up

I have other questions.

Contact kumarvi@mail.nci.nih.gov