Contents

- 1 PiSi Package Building
  - 1.1 Structure of a PiSi 1.1 Package
  - 1.2 Building a PiSi Package
    - 1.2.1 pspec.xml
  - 1.3 Installation of Additional Files from the Files Of the Source Tree
    - 1.3.1 actions.py

PiSi Package Building

This article tells only the basics of pisi package building. For a detailed version see: Making Pisi Packages

Structure of a PiSi 1.1 Package

A pisi package is essentially a zipped file. Let's download one and examine it:

```
$ wget http://packages.pardus.org.tr/pardus-2009/knazar-1.1-10-4.pisi
$ unzip knazar-1.1-10-4.pisi -d apackage
Archive:  knazar-1.1-10-4.pis
  inflating: apackage/metadata.xml
  inflating: apackage/files.xml
  inflating: apackage/install.tar.lzma
$ cd apackage
$ ls
files.xml  install.tar.lzma  metadata.xml
```

- files.xml contains locations, types, sizes and sha1sums of all files in the package
- metadata.xml contains general information like package name, homepage, packager, etc.
- install.tar.lzma contains compressed files that we can extract with those commands

```
$ tar xvf install.tar.lzma
```

Building a PiSi Package

In order to build a pisi package we need to prepare at least two files by hand: *pspec.xml* and *actions.py*

pspec.xml

This file is an XML file containing at least 3 child nodes: *Source, Package, History*

There can be multiple Package nodes in here.

In other words, one source package may generate multiple binary packages. Pisi is very flexible :)

Let's look at our example here:
http://svn.pardus.org.tr/pardus/2009/stable/desktop/kde/addon/knazar/pspec.xml

As you can see in the example,

- *Source* node contains general package and packager information, sha1sum, type and location of source archive, Patches and BuildDependencies.

- *Package* node contains RuntimeDependencies and locations of different file types, AdditionalFiles and Comar scripts if needed.
- *History* node simply contains information about package history.

You may want to examine the dtd file for pisi here: http://www.pardus.org.tr/projeler/pisi/pisi-spec.dtd

Installation of Additional Files from the Files Of the Source Tree

The Package may contain the AdditionalFiles tag, which can be used to copy files from the files subdirectory of your source tree into the .pisi.

E.g., suppose I have a structure like so:

```
myproject/
myproject/files/somefile.config
myproject/actions.py
myproject/pspec.xml
```

Then, (one of) the Package(s) can have the following:

```
<Package>
    <Name>myproject</Name>
    <Summary>Core of MyProject.</Summary>
    <RuntimeDependencies>
        <Dependency>some-lib</Dependency>
    </RuntimeDependencies>
    <Files>
        <Path fileType="http://en.pardus-wiki.orgexecutable"http://en.pardus-wiki.org>/usr/bin<
    </Files>
    <AdditionalFiles>
        <AdditionalFile target="http://en.pardus-wiki.org/etc/path/to/install"http://en.pardus-
                        owner="http://en.pardus-wiki.orgroot"http://en.pardus-wiki.org>somefile
    </AdditionalFiles>
</Package>
```

actions.py

This file contains python codes that would compile and install the source package into a specific InstallDIR (in our example it is /var/pisi/knazar-0.2-3-3/install/)

http://svn.pardus.org.tr/pardus/2009/stable/desktop/kde/addon/knazar/actions.py

In this file, we use actionsapi that comes with pisi. Actions API has all functions for us to compile and install our package.

If you know python, you may want to have a look at sources here:
http://svn.pardus.org.tr/uludag/trunk/pisi/pisi/actionsapi/

After preparing pspec.xml and actions.py you can easily form a pisi package by typing:

```
sudo pisi build pspec.xml
```

in the console. To compile my example you may type:

```
sudo pisi bi http://svn.pardus.org.tr/pardus/2009/stable/desktop/kde/addon/knazar/knazar/pspec.xml
```

Finally, you can examine other official pisi source packages here: http://svn.pardus.org.tr/pardus/