Configo

Managing configuration definitions

Vladimir Ritz Bossicard (vbossica@gmail.com)

Let's first start with the elevator pitch:

Configo is a Java Swing application that let its users manage and export properties for an external system. It is targeted at ISVs that want to provide a pleasant installation experience to their users.

Configo is shipped by default as a standalone JAR archive and only needs Java 5.0 or higher to be executed.

Configo's lastest version can be downloaded from http://code.google.com/p/workingonit/wiki/ConfigoProject

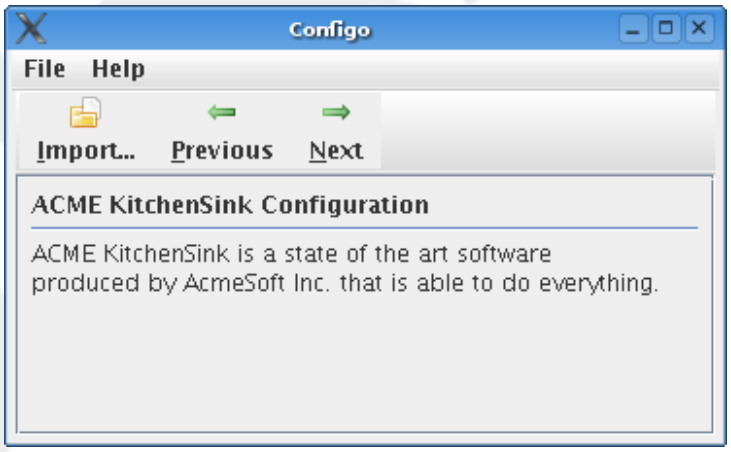
Using Configo

The easiest way to present *Configo*'s purpose in live is to walk you through a simple business case: you're the main developer of KitchenSink, ACME's flagship product and you have just published the new version of this application.

Fast forward a couple of hours... Let's follow John, the administrator that will have the pleasure to install KitchenSink.

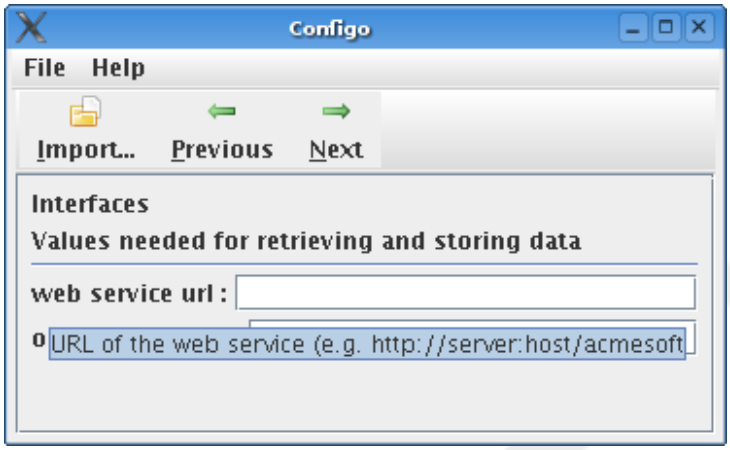
John Dow has just received ACME KitchenSink's latest release and he's making his way through the installation guide. Before KitchenSink can unleash its full power a configuration file must first be created. To ease the installation procedure and make sure that all the properties were correctly defined, AcmeSoft's engineering team (i.e. you) included *Configo* to ease the pain.

Figure 1. KitchenSink configuration application



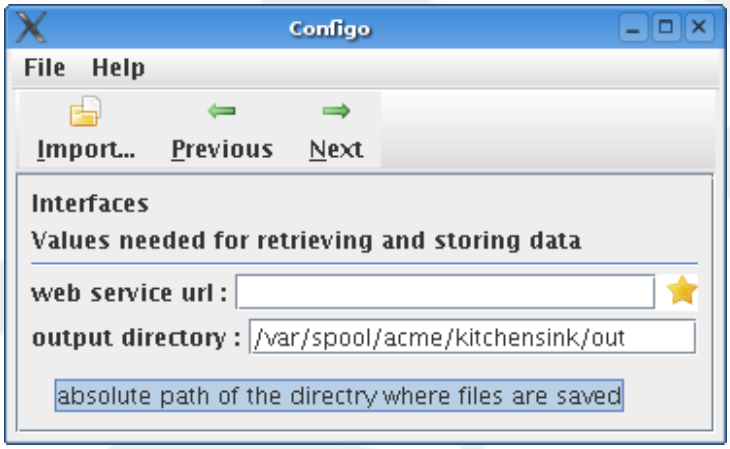
The KitchenSink application needs to access a web service and store some generated files into an external directory. Nothing too complex for Joe and when he clicked on the *Next* button, he's confronted with the following module configuration:

Figure 2. KitchenSink properties



John could fill in the blanks but since he's kind of lazy and wants to avoid typos he's simply importing the properties file from the previous installation. Then old configuration values are automatically filled in and new ones (in that case the web service url) outlines with a non twickeling little star.

Figure 3. KitchenSink imported properties



So now John is filling the missing properties and generating the file that will be use by KitchenSink to do absolutely everything what has to be done:

Figure 4. KitchenSink generated properties file

```
#
# ACME KitchenSink Configuration Properties
#
# This file was generated by Configo
# (http://code.google.com/workingonit/ConfigoProject)
#

# ------------------------------------------------------
# Interfaces
#
# Values needed for retrieving and storing data

# web service url
# URL of the web service (e.g. http://server:host/acmesoft
com.acme.kitchensink.wsurl = http://mendocino/acmesoft
# output directory
# absolute path of the directry where files are saved
com.acme.kitchensink.outputdir = /var/spool/acme/kitchensink/out
```

Customization

So now let's walk through what you'll have to do to include *Configo* into your application's release. You'll see, it's very easy.

Step 1: Configuration File Definition

The first step is to define an XML configuration file that describes not only the application but also all modules and their respective properties.

> ## Note
>
> The file *must* be named `configo-config.xml` because it is automatically looked for by that name at startup

This if for example the configuration for the KitchenSink application:

Figure 5. KitchenSink configuration file

```
<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE configuration PUBLIC
  "-//WorkingOnIt/Configo Configuration DTD 1.0//EN" "configo.dtd" >

<configuration>
  <name>ACME KitchenSink Configuration</name>
  <description>ACME KitchenSink is a state of the art software
produced by AcmeSoft Inc. that is able to do everything.
  </description>
  <modules>

    <module>
      <name>Interfaces</name>
      <description>Values needed for retrieving and storing data</description>
      <properties>

        <property>
          <name>web service url</name>
          <description>
            URL of the web service (e.g. http://server:host/acmesoft)
          </description>
          <key>com.acme.kitchensink.wsurl</key>
        </property>

        <property>
          <name>output directory</name>
          <description>
            absolute path of the directry where files are saved
          </description>
          <key>com.acme.kitchensink.outputdir</key>
        </property>
      </properties>
    </module>
  </modules>
</configuration>
```

Step 2: Configuration File Inclusion

Once the configuration file has been defined, it must be included into the *Configo* JAR archive.

Note

The file configuration file *must* be located in the `resources` folder.

If you use Apache Ant, the following script would do it for you:

Figure 6. Customization Ant target

```
<zip destfile="acme-configuration.jar" duplicate="preserve">
    <zipfileset file="custom/configo-config.xml" prefix="resources"/>
    <zipgroupfileset file="configo-[version].jar"/>
</zip>
```

That's all, there is no other step in this process: you can now ship *Configo* with your next release.

Your customers will thank you.

A. Configo DTD

Configo provides a XML DTD file located in the `configo.jar` archive to let you validate your configuration file:

Figure A.1. Configo DTD definition

```
<?xml version="1.0" encoding="UTF-8" ?>

<!ELEMENT configuration ( name, description, modules ) >

<!ELEMENT name ( #PCDATA ) >

<!ELEMENT description ( #PCDATA ) >

<!ELEMENT modules ( module+ ) >

<!ELEMENT module ( description | name | properties )* >

<!ELEMENT properties ( property+ ) >

<!ELEMENT property ( description | key | name )* >

<!ELEMENT key ( #PCDATA ) >
```