SimpleComponentExample

Writing a Simple Avalon Component

In this example we'll look at a very simple Avalon component.

The Service Interface

```
package org.apache.avalon.examples.simple;

/**
 * a simple service interface
 * @avalon.version 0.1
 */
public interface Simple {

  /**
   * @return the name or a message from the service
   */
  public String getName();

}
```

A Simple Implementation

```
package org.apache.avalon.examples.simple.impl;

import org.apache.avalon.examples.simple.Simple;

/**
 * a Simple service implementation that returns 'Hello World!'
 * @avalon.component name="simple" lifestyle="singleton"
 * @avalon.service type="org.apache.avalon.examples.simple.Simple"
 */
public class SimpleHelloImpl implements Simple
{

    public String getName()
    {
        return "Hello World!";
    }

}
```

Adding Lifecycles

```
package org.apache.avalon.examples.simple.impl;

import org.apache.avalon.examples.simple.Simple;

import org.apache.avalon.framework.logger.AbstractLogEnabled;

import org.apache.avalon.framework.context.Contextualizable;
import org.apache.avalon.framework.context.Context;

import org.apache.avalon.framework.service.Serviceable;
import org.apache.avalon.framework.service.ServiceManager;
import org.apache.avalon.framework.service.ServiceException;

import org.apache.avalon.framework.activity.Initializable;
import org.apache.avalon.framework.activity.Startable;
import org.apache.avalon.framework.activity.Disposable;
```

```
import org.apache.avalon.framework.configuration.Configurable;
import org.apache.avalon.framework.configuration.Configuration;
import org.apache.avalon.framework.configuration.ConfigurationException;

import org.apache.avalon.framework.parameters.Parameterizable;
import org.apache.avalon.framework.parameters.Parameters;

/**
 * a Simple service implementation that implements the basic lifecycles.
 * @avalon.component name="lifecycle" lifestyle="singleton"
 * @avalon.service type="org.apache.avalon.examples.simple.Simple"
 */
public class SimpleLifecycleImpl
    extends AbstractLogEnabled
    implements Simple, Contextualizable, Serviceable,
               Configurable, Parameterizable, Initializable,
               Startable,  Disposable
{

    protected String m_name = "Avalon Lifecycle Example";
    protected ServiceManager m_serviceManager;

    public String getName()
    {
        return m_name;
    }


    public void contextualize(Context context) {
      if(getLogger().isInfoEnabled())
        getLogger().info("contextualizing simple component");
    }


    public void service(ServiceManager manager){
        if(getLogger().isInfoEnabled())
            getLogger().info("servicing simple component");
    }


    public void configure(Configuration config)  throws ConfigurationException
    {
      if(getLogger().isInfoEnabled())
        getLogger().info("configuring simple component");
    }

    public void parameterize(Parameters parameters) {
      if(getLogger().isInfoEnabled())
        getLogger().info("parameterizing simple component");
    }


    public void initialize(){
        if(getLogger().isInfoEnabled())
            getLogger().info("initializing simple component");
    }

    public void start(){
        if(getLogger().isInfoEnabled())
            getLogger().info("starting simple component");
    }

    public void stop(){
        if(getLogger().isInfoEnabled())
            getLogger().info("stopping simple component");
    }

    public void dispose(){
        if(getLogger().isInfoEnabled())
            getLogger().info("disposing simple component");
```

```
    }

}
```

Adding A Dependency

Adding Meta-Data

Adding Configuration

Container Deployment