Class Binding

For each page, there must be one adapter class. The name of the adapter class is given inside the PAGE tag of a page by the corresponding `model` property. Several page definitions can point to one adapter class, for example, when you have several page variants to display the same logical content.

The Application Designer runtime can create an adapter instance of a page in two ways:

1. **Direct Class Binding**
   The Application Designer runtime directly tries to create an adapter instance from the name in the `model` property of the PAGE tag of the page.

2. **Generic Class Binding**
   If the direct class binding does not succeed, the Application Designer runtime tries to create a generic adapter instance that has to be made accessible by the application.

Option 1 is the typical one. Option 2 is a solution if you require an adapter to be used very generically inside your framework.

The following topics are covered below:

- Direct Class Binding
- Generic Class Binding

Direct Class Binding

The following page definition forces the Application Designer runtime to look for a `Test1Adapter` class inside the package `com.softwareag.cis.test`:

```
<page model="com.softwareag.cis.test.Test1Adapter">
    ...
    ...
    ...
</page>
```

The class itself is derived typically from the class `com.softwareag.cis.server.Adapter`:

```
package com.softwareag.cis.test;

import com.softwareag.cis.server.Adapter;

public class Test1Adapter extends Adapter
{
    // ------------------------------------------------------------------------
    // constructor - without parameters
    // ------------------------------------------------------------------------

    public Test1Adapter()
    {
    }

    // ------------------------------------------------------------------------
    // public access
    // ------------------------------------------------------------------------

    /** The init message is called when an object is created and all
     *  runtime aspects are correctly set inside the adapter. */
    public void init()
    {
        ...
        ...
    }
}
```

The default constructor is required (without any parameters).

Generic Class Binding

If the runtime does not find the class, it tries to find a generic one. The name of the generic class is created in the following way:

- The package name is taken from the `model` property of the PAGE tag of your page definition.
- The class name is "GenericAdapter".

Example: if you bind a page to the class `com.softwareag.cis.test.Test2Adapter` and the runtime system cannot locate the class `Test2Adapter`, the system tries to load the class `GenericAdapter` in the same package as the class that could not be found:

```
<page model="com.softwareag.cis.test.Test2Adapter">
    ...
    ...
    ...
</page>
```

The generic adapter is just a normal adapter which typically supports the dynamic binding of properties (see below).

```
package com.softwareag.cis.test;

import com.softwareag.cis.server.Adapter;

public class GenericAdapter extends Adapter
{
    /** */
    public void init()
    {
        System.out.println("My original class name is " + this.m_modelName);
    }
}
```

Each adapter object can access the `m_modelName` member. This member is set after the initialisation of the object. It holds the original adapter name to which the page refers.