Controller Lifecycle Handler

Error CSS Stylesheet macro - Import URL 'http://felix.apache.org/ipojo/site/superfish.css' is not on the allowlist. If you want to include this content, contact your Confluence administrator to request adding this URL to the Allowlist.
Error CSS Stylesheet macro - Import URL 'http://felix.apache.org/ipojo/site/style.css' is not on the allowlist. If you want to include this content, contact your Confluence administrator to request adding this URL to the Allowlist.

Lifecycle Controller Handler

The lifecycle controller allows a component implementation to participate to the instance lifecycle. So, you can immediately decide to stop an instance if the configuration is incorrect (correct properties, accessible resources...). The licecyel controller impacts the instance lifecycle, if you want to impact only the registered service, have a look to the service controller (service providing).

- iPOJO instance lifecycle & Lifecycle controller
- An example

iPOJO instance lifecycle & Lifecycle controller

Once started, iPOJO instances can be either valid or invalid. The decision comes from handlers. An instance is valid if every plugged handler are valid. Basically it means that all required services are available. As soon as one handler becomes invalid, the instance becomes invalid.

The lifecycle controller just monitors a field inside the POJO class. When this field becomes `false`, the handler becomes invalid, and so the instance becomes invalid. When the field get the `true` value, the handler becomes valid, and if all handlers are valid, the instance becomes valid.

An example

Imagine the following component :

```
@Component
public class LifecycleControllerTest {

    @Controller
    private boolean m_state;

    @Property
    public void setConf(String newConf) {
        System.out.println("setConf : " + newConf);
        if (newConf.equals("a correct value")) {
            m_state = true; // update controller value.
        } else {
            m_state = false;  // update control value
        }
    }
}
```

If you don't want to use annotations, the following snippet does the same job using XML:

```
<component
   classname="org.apache.felix.ipojo.test.scenarios.component.LifecycleControllerTest">
               <controller field="m_state"/>
               <properties>
                       <property name="conf" method="setConf"/>
               </properties>
</component>
```

The component requires the `conf` property. iPOJO checks if this property is inside the pushed configuration, but cannot checks if the configuration is correct according to the component semantic. When the instance is created, the `setConf` method is called with the given value. If the given `conf` property is "valid" the `m_state` field (i.e. the controller) is set to `true`. Else, the `m_state` is set to `false`. It means that the lifecycle controller handler becomes invalid and as a consequence, the instance becomes invalid.