Table of Contents

Sometimes it's important to store/cache temporary information. It's no problem to create or use a variable to save such information. This is also the recommended solution because it makes the code readable and references can be found. However, sometimes it's good to have an alternative to clean code because sometimes you want to add information to existing objects quickly and easily.

Instance Values

As an example, you could open a work-screen and add additional metadata like read-only or allowed users. To solve this problem, you could save the screen in a HashMap with metadata as values. This information should be saved in the application because the screen has access to the application. This makes things a little bit complicated.

To make everything easier, every UI resource (e.g., a component) has the methods:

```
public Object pubObject(String pName, Object pValue);
public Object getObject(String pName);
public Collection<String> getObjectNames();

public ResourceHandler eventResourceChanged();
public ResourceHandler eventResourceChanged(String pObjectName);
```

So, it's super easy to add temporary information to any UI resource. We recommend to use constants as object names in order to find references easily. If you use hardcoded strings, it will be hard to find the usage in your whole application.

Here's a short example:

```
public void doOpenStatus()
{
    IWorkScreen wosc =
getApplication().openWorkScreen(StatusWorkScreenc.class.getName());

    ((WorkScreen)wosc).putObject(IConstants.STATE, ScreenState.New);
}
```

We save a simple **“state”** for the screen directly in the screen instance.

It's also possible to use the launcher or application, e.g.,

```
@Override
protected void afterLogin()
{
    super.afterLogin();

    getLauncher().putObject(IConstants.APPMODE, AppMode.AUTHENTICATED);
}
```

We save the **“application mode”** in the launcher and this information is available in every screen or any other class which has access to the launcher. It's not possible to add variables to the launcher or to change the class directly, so it's a good alternative to use custom objects to add metadata and to

avoid HashMap for simple caching.

Global Values

We have another alternative for the above instance values. It's also possible to use our ObjectCache, e.g.,

```
//available for 1 minute
ObjectCache.put(IConstants.APPMODE, AppMode.AUTHENTICATED, 60000);
```

The ObjectCache is a static cache and can be used to use values without application context. It's also possible to use an **ObjectCacheInstance** which isn't static.

From:
https://doc.sibvisions.com/ - **Documentation**

Permanent link:
https://doc.sibvisions.com/jvx/client/gui/temporary_values

Last update: **2021/02/01 12:47**