Table of Contents

If you have an application based on JVx' standard application frame, it's very easy to use ProjX. The standard application doesn't contain a workscreen manager, automatic menu/toolbar creation, and automatic logon. This missing features and much more are part of our ProjX application frame. It's a simple RemoteWorkScreen application (defined by JVx). The JVx' standard Application frame is a RemoteApplication without workscreen management. The RemoteWorkScreenApplication extends RemoteApplication and adds workscreen support.

Simply do the following:

- Add projx.jar and appsclient.jar to libs/client
- Add appserver.jar to libs/server
- Delete your custom application class or extend com.sibvisions.apps.projx.ProjX
- Change your launcher (run configuration) if you don't need a custom application and use com.sibvisions.apps.projx.ProjX as first argument (instead of your custom application)
- Add an application.xml to your application package
- Set value of Application.Login.application to your application name
- Change your launcher (run configuration) and add the full qualified resource path of your application.xml as second argument

  The launcher should have two arguments, e.g.
  `com.sibvisions.apps.projx.ProjX /com/company/shop/application.xml`
- **(optional)** Style your application:

  Application.Login.image (e.g. /com/company/shop/images/login.png),
  Application.Desktop.image (e.g. /com/company/shop/images/background.jpg),
  Application.Desktop.topimage (e.g. /com/company/shop/images/top-shadow.png),
  Application.Desktop.background (e.g. 255,255,255)
- **(optional)** Configure autologin:

  Application.Login.username
  Application.Login.password
  (and be sure that Application.authenticator contains
  `com.sibvisions.apps.auth.UserPwdAuthenticator`)

After the above steps, start the application. You'll see an exception because the workScreenAccess object wasn't found! This object should be added to your session LCO because it's responsible for the menu/toolbar configuration. The interface IWorkScreenAccess defines all relevant methods.

ProjX contains two implementations: DBWorkScreenAccess and MemWorkScreenAccess. The DBWorkScreenAccess needs some tables and views in order to work, so for a first test the MemWorkScreenAccess should be good enough. Use the following snippet and add it to your `Session.java`

```
public IWorkScreenAccess getWorkScreenAccess() throws exception
{
    IWorkScreenAccess wsac = (IWorkScreenAccess)get("workScreenAccess");

    if (wsac == null)
    {
        MemWorkScreenAccess mwsac = new MemWorkScreenAccess();

        WorkScreenConfig woscManager= new WorkScreenConfig();
```

```
woscManager.setClassName("com.company.shop.screens.ManagerWorkScreen");
        woscManager.setMenuStructure("Shop");
        woscManager.setText("Manager");

        mwsac.addScreen(woscManager);

        put("workScreenAccess", mwsac);

        return mwsac;
    }

    return wsac;
}
```

Be sure that your session class is configured as master session object in your *config.xml*:

```
...

<lifecycle>
    <mastersession>com.company.shop.Session</mastersession>
</lifecycle>
```

The next important thing is that you extend your workscreens from DataSourceWorkScreen:

ManagerWorkScreen ***extends DataSourceWorkScreen***

instead of

ManagerWorkScreen ***extends UIInternalFrame***

Use the following default constructor:

```
public ManagerWorkScreen(RemoteWorkScreenApplication pApp,
                         AbstractConnection pConnection,
                         Map<String, Object> pParameter) throws Throwable
{
    super(pApp, pConnection, pParameter);

    initializeModel();
    initializeUI();
}
```

Please remove the code from your screen which opens a new subconnection or a RemoteDataSource. The connection will be opened automatically from ProjX (see constructor parameter pConnection) and the RemoteDataSource will be created from DataSourceWorkScreen. Simply call getDataSource() to get access.

The DataSourceWorkScreen removes the boiler plate code from your screens and usually your screens don't take care of application-specific problems like save/reload or open/close. It's possible to take care but not necessary in most screens!

Start your application again and everything should work without problems. It's also possible to

configure the welcome screen for fast test iterations. To do this, set the parameter *Application.WelcomeScreen* in your application.xml.

The following ZIP archive contains templates: JVx to ProjX Templates