Table of Contents

The application (ProjX) implements the **IExceptionListener** interface and registers itself as listener:

```
protected void initApplication(UILauncher pLauncher) throws Throwable
{
    ExceptionHandler.addExceptionListener(this);

    ...
}
```

All exceptions will be handled in the listener method:

```
public void handleException(Throwable pThrowable)
{}
```

So, it's very easy to change the default error handling which shows a popup dialog:

with the error message and some details (stack trace).

As an example of a custom error handling, we'll show the exceptions directly in the workscreens. So, every workscreen should be able to show error messages. To make this possible, we introduce the following interface:

```
public interface IErrorHandler
{
    public void showErrorMessage(String pMessage);
}
```

and all our workscreens should implement the interface, e.g.:

```
public class FirstWorkScreen extends DataSourceWorkScreen implements
IErrorHandler
{
    /** labelMain. */
    private UILabel.labelMain = new UILabel();

    private void initializeUI() throws Throwable
    {
        labelMain.setText("ErrorArea");
        labelMain.setForeground(UIColor.red);

        ...

        add(labelMain, UIBorderLayout.SOUTH);
    }

    public void onLoad() throws Throwable
    {
        super.onLoad();
```

```
        labelMain.setVisible(false);
    }

    public void showErrorMessage(String pMessage)
    {
        labelMain.setText(pMessage);
        labelMain.setVisible(true);
    }
}
```

So, our workscreen example contains a new label in the SOUTH area (the bottom of the screen). The label will be hidden initially in **onLoad()**.
Our screen is now ready to show error messages, but our application must be changed. To do this, we need a custom application.

Our application contains the following code:

MyCustomApplication.java

```
public class MyCustomApplication extends ProjX
{
    private List<Throwable> liErrors = null;

    public MyCustomApplication(UILauncher pLauncher) throws Throwable
    {
        super(pLauncher);
    }

    public void handleException(Throwable pThrowable)
    {
        if (liErrors == null)
        {
            liErrors = new ArrayUtil<Throwable>();

            invokeLater(this, "showErrorMessage");
        }

        liErrors.add(pThrowable);
    }

    public void showErrorMessage()
    {
        String message =
ExceptionUtil.getRootCause(liErrors.get(0)).getMessage();

        for (IWorkScreen screen : getWorkScreens())
        {
            if (screen instanceof IErrorHandler)
            {
                ((IErrorHandler)screen).showErrorMessage(message);
            }
```

```
            }

            liErrors = null;
        }
    }
```

We collect all occurred exceptions in a list - **liErrors** - and show one message at the end, see **showErrorMessages**.

In **showErrorMessages**, we iterate through all opened workscreens and check if our interface is implemented. In our example, every workscreen has to implement the interface **IErrorHandler**. A better solution would be a custom workscreen base class, e.g.

```
public class MyCustomWorkScreen extends DataSourceWorkScreen implements
IErrorHandler
{
}
```

and all our workscreens should extend this base screen:

```
public class FirstWorkScreen extends MyCustomWorkScreen
```

And the result will look like the following: