

# Popup Windows

## Table of contents

1 Introduction.....	2
2 Submit/Display Cycle in a Single Page Flow.....	2
3 Submit/Display Cycle with Nested Page Flow.....	3
4 Submit/Display Cycle with Nested Page Flow (Popup Window Enabled).....	6
4.1 The Popup Attribute.....	6
4.2 The <netui:configurePopupTag>.....	6
4.3 The <netui:retrievePopupOutput>.....	6
4.4 The _auto Global Forward.....	7

## 1. Introduction

The following topic explains how to collect data from a [nested page flow](#) and 'return' the data to the nesting/main page flow. The nested page flow can appear as a popup window if so desired.

## 2. Submit/Display Cycle in a Single Page Flow

Before we get to nested page flows and popup windows, lets first take a look at the basic page flow submission cycle. Below is a diagram of the basic cycle.

singlepageflow

Below is the code for the basic cycle: (1) `index.jsp` contains the form for collecting data, (2) the submission is handled by the method `submit()`, and (3) the submitted data is displayed by `results.jsp`.

### index.jsp

```
<netui:form action="submit">
  Name:          <netui:textBox dataSource="actionForm.name"/>
  Favorite Color: <netui:textBox dataSource="actionForm.color"/>
  <br/>
  <netui:button type="submit" value="submit"/>
</netui:form>
```

### Controller.java

```
...
@Jpf.Controller(
    simpleActions={
        @Jpf.SimpleAction(name="begin", path="index.jsp")
    }
)
public class Controller
    extends PageFlowController
{
    @Jpf.Action(
        forwards={
            @Jpf.Forward(
                name = "success",
                path = "results.jsp"
            )
        }
    )
    protected Forward submit(SubmitForm form)
    {
        return new Forward("success", "form", form );
    }
}
```

```
}  
  
public static class SubmitForm extends FormData  
{  
    private String _name;  
    private String _color;  
  
    public String getName()  
    {  
        return _name;  
    }  
  
    public void setName(String value)  
    {  
        _name = value;  
    }  
  
    public String getColor()  
    {  
        return _color;  
    }  
  
    public void setColor(String value)  
    {  
        _color = value;  
    }  
}  
}
```

#### **results.jsp**

```
Submitted Name: ${pageInput.form.name}  
<br/>  
Submitted Color: ${pageInput.form.color}  
<br/>  
<br/>  
<netui:anchor action="begin">[start over]</netui:anchor>
```

### **3. Submit/Display Cycle with Nested Page Flow**

In this section we introduce a nested page flow to help collect data from the user.

When the user clicks the "get color" button (see the `index.jsp` page below), an instance of a nested page flow is created (`GetColor.java`). When the user submits the form within the nested page flow, the data is 'returned' to populate the form within the main/nesting page flow.

The diagram below shows how the user moves into the nested page flow, and then back into the main/nesting page flow.

nested\_2

The code for the main and nested page flow follows.

### index.jsp

```
<netui:form action="submit">
  Name:          <netui:textBox dataSource="actionForm.name"/>
  Favorite Color: <netui:textBox dataSource="actionForm.color"/>
  <br/>
  <netui:button type="submit" value="get color" action="getColor"/>
  <netui:button type="submit" value="submit"/>
</netui:form>
```

### Controller.java

```
...

public class Controller
    extends PageFlowController
{
    ...

    /**
     * This action forwards to the nested page flow to gather the favorite
     color.
     */
    protected Forward getColor(SubmitForm form)
    {
        return new Forward("getColorFlow");
    }

    @Jpf.Action(
        forwards={
            @Jpf.Forward(
                name="success",
                navigateTo=Jpf.NavigateTo.currentPage
            )
        }
    )
    protected Forward colorSuccess( String color )
    {
        SubmitForm previousForm = ( SubmitForm ) getPreviousFormBean();
        previousForm.setColor( color );
        return new Forward( "success", previousForm );
    }

    ...
}
```

### results.jsp

[unchanged from above]

### getColor/index.jsp

```
<netui:form action="submitColor">
  Color:<br/>
  <netui:select dataSource="actionForm.color" size="5">
    <netui:selectOption value="red" />
    <netui:selectOption value="blue" />
    <netui:selectOption value="green" />
    <netui:selectOption value="yellow" />
    <netui:selectOption value="orange" />
  </netui:select>
  <br/>
  <netui:button type="submit" value="submit"/>
</netui:form>
```

### getColor/GetColor.java

```
package getColor;

import javax.servlet.http.HttpSession;
import org.apache.beehive.netui.pageflow.PageFlowController;
import org.apache.beehive.netui.pageflow.Forward;
import org.apache.beehive.netui.pageflow.annotations.Jpf;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionMapping;
import org.apache.beehive.netui.pageflow.FormData;

@Jpf.Controller(
    nested=true,
    simpleActions={
        @Jpf.SimpleAction(name="begin", path="index.jsp")
    }
)
public class GetColor extends PageFlowController
{
    @Jpf.Action(
        forwards={
            @Jpf.Forward(
                name="done",
                returnAction="colorSuccess",
                outputFormBeanType=String.class)
        }
    )
    protected Forward submitColor( ColorForm form )
    {
        return new Forward( "done", form.getColor() );
    }

    public static class ColorForm
    {
        private String _color;

        public String getColor()
        {
            return _color;
        }
    }
}
```

```

        public void setColor(String value)
        {
            _color = value;
        }
    }
}

```

## 4. Submit/Display Cycle with Nested Page Flow (Popup Window Enabled)

This section explains how to make the nested page flow appear as a popup window.

### 4.1. The Popup Attribute

If you want your nested page flow to run inside a popup window set the popup attribute to "true"

```

<netui:button type="submit" value="Pick Color" action="getColor"
popup="true">
    <netui:configurePopup location="false" width="550" height="150">
    </netui:configurePopup>
</netui:button>

```

This attribute is legal on all tags that can raise an action, except for [<netui:imageButton>](#).

1. [<netui:anchor>](#)
2. [<netui:imageAnchor>](#)
3. [<netui:button>](#)
4. [<netui:area>](#)

Note - Popup window support uses JavaScript behind the scenes. [<netui:imageButton>](#) is used for submitting a form *without* JavaScript, and thus cannot accept the popup attribute.

### 4.2. The <netui:configurePopupTag>

Use the [<netui:configurePopup>](#) tag to determine the shape of the popup window.

```

<netui:button type="submit" value="Pick Color" action="getColor"
popup="true">
    <netui:configurePopup location="false" width="550" height="150">
    </netui:configurePopup>
</netui:button>

```

### 4.3. The <netui:retrievePopupOutput>

To access values returned by the nested page flow use the [<netui:retrievePopupOutput>](#) tag, for example,

```

<netui:retrievePopupOutput dataSource="outputFormBean.color"
tagIdRef="colorField"/>

```

Include multiple instances of the tag to set multiple fields.

**Note** - the `<netui:retrievePopupOutput>` tag must be nested inside the `<netui:configurePopup>` tag even if the `<netui:configurePopup>` tag has no attributes

The `<netui:retrievePopupOutput>` tag has two attributes, `dataSource` and `tagIdRef`, both of which are required. Both accept expressions that can be evaluated at runtime.

**Note** - 'outputFormBean' is a special databinding context that applies in this situation. It is the "return value" of the nested page flow: It receives the value generated by the 'outputFormBean' (or 'outputFormBeanType') in the returnAction forward from the nested page flow. This databinding context is only legal when used in the `<netui:retrievePopupOutput>` tag.

#### 4.4. The `_auto` Global Forward

If you are returning data from the nested flow it is passed as a bean to the action in the nesting page flow, for example

```
@Jpf.Action()  
public Forward nestedPageFlowGotColor(GetColor.ColorForm colorForm)  
{  
    return new Forward("_auto");  
}
```

Here is what the code looks like in the nested page flow

```
@Jpf.Action(  
    forwards={  
        @Jpf.Forward(name="success",  
                    returnAction="nestedPageFlowGotColor",  
                    outputFormBeanType=ColorForm.class  
        )  
    }  
)  
public Forward done( ColorForm returnForm )  
{  
    return new Forward("success", returnForm);  
}
```

The bean returned to the nesting flow is accessed in the page using the special data binding context, `outputFormBean`