

Servlet Nodes: Using Workflow-Enabled Servlets

By Larry Smith, Co-founder, Nouveau Systems

Contents

Introduction	2
Problem Statement	2
Previous Options	2
Our Solution	2
Implementation	3
Installation	8
Demonstration	10
Customization	12
Summary	12

Introduction

This white paper describes a collection of components that allow you to rapidly create web-based applications using Nouveau Alliance Workflow servlets.

Problem Statement

The goal of this white paper is to demonstrate how to use Nouveau Alliance technologies to build web-based applications. The main challenge of this problem is to provide a solution that accesses all of the powerful modeling features of Nouveau Alliance, but is packaged in way to facilitate its use by business process modelers versus information technology developers.

Previous Options

Currently, web-based applications can be created using a variety of techniques. Solutions range from harnessing the power of application servers using embedded Business Process Modeling (BPM) enabled via a Service Oriented Architecture (SOA) using visual tools such as workflow editors, to Java Server Pages or CGI scripts that control the flow and content of the application. Generally, these implementations involve imposing technology and implementation-specific expertise.

Our Solution

With Nouveau Alliance, you can drag-and-drop predefined or user-defined components to assemble a web-based application.

In a manner similar to “story-boarding”, you can specify the sequencing and content of the web pages that compose the application

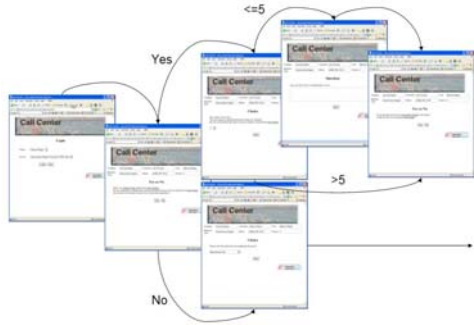


Figure 1 – Application Story Board

This paper presents a brief overview of underlying technology of the Nouveau Alliance web interface and a description of an example which packages this technology for simple and rapid implementation.

Two features of Nouveau Alliance are highlighted in this paper:

Powerful Modeling

Nouveau Systems’ FlowSpace provides a library of primitives that allow you to specify an application in a simple, yet powerful way. If some customization is needed, you can add to or modify these primitives via their properties or augment their programming via Java or a myriad of supported scripting languages. When complete, these customizations can be added to the library of built-in primitives to define new applications.

Web-based Servlet Interface

You can enter your application logic graphically by defining workflows with FlowSpace. You then can use a Workflow Servlet to create a web application. This enables domain experts to define and modify business logic without extensive knowledge of web programming.

Implementation

In order to know how Workflow-enabled Servlets are used, it is important to understand the underlying technology, including the terminology used to convey its functionality.

Technology

The following is a brief overview of the basic components that are used to build workflows and how they are ultimately accessed via servlets.

Workflow

A workflow definition is an automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules. Often shortened to just “workflow”, a workflow definition is created by connecting nodes representing pre-defined, but parameterized, functionality.

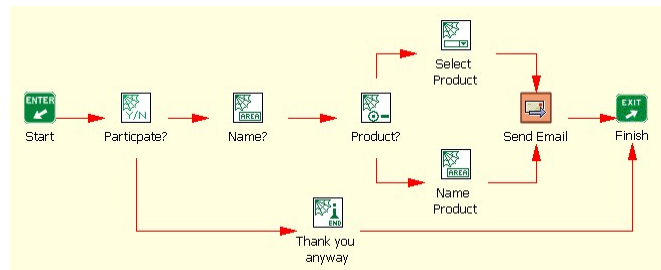


Figure 2 – Workflow Definition

Workflow Node

A workflow node is a unit of work that forms one logical step within a process. It consists of a list of activities that are executed when this node is traversed when the workflow is run.

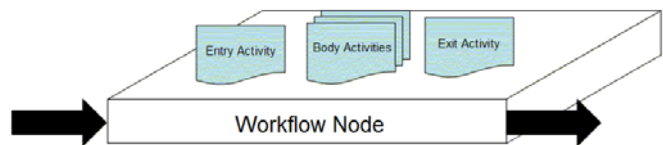


Figure 3 – Workflow Node

Workflow Activity

An activity is an arbitrary action performed by a user. Typical Nouveau Alliance activities schedule and facilitate meetings, send email, collect and set information, perform workspace operations or run external tools such as spreadsheets and word processors.

```

public class InformActivityImpl extends WorkflowActivityImpl
    implements BodyWorkflowActivity
{
    public static final AlliancePropertyDescriptor[] defaultProperties =
        new AlliancePropertyDescriptor[]
        {
            new AlliancePropertyDescriptor("message", "java.lang.String",
                "Information message", "the message to display"),
        };

    protected void performActivity() throws Exception
    {
        Object message = getActivityPropertyValue("message");
        javax.swing.JOptionPane.showMessageDialog(null, message);
    }
}
    
```

Figure 4 – Workflow Activity

This figure shows an activity that displays a dialog window with a default property that represents the message to be displayed. The activity is written in Java, but could have been written in a variety of scripting languages (Ruby, Perl, Tcl and Python to name just a few).

Workflow Property

Properties enable you to save and retrieve values within a workflow. They can be “scoped” to a workflow, workflow node, workflow activity or running instances of each of them. They can have values that include expressions referencing other properties or calling static methods defined by you or in the Nouveau Alliance API.

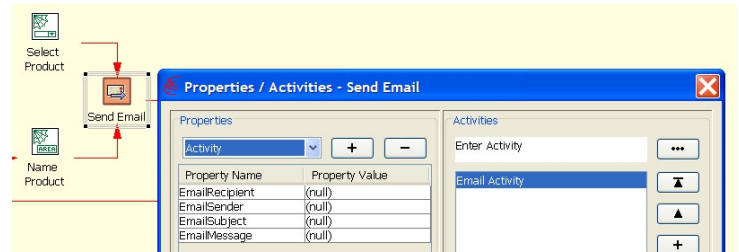


Figure 5 – Workflow Property

The figure shows the properties of the Email Activity as displayed in Properties / Activities dialog box. Using this dialog box, you can customize the properties of the selected email workflow node (email sender, recipients, subject and message). As stated previously, the property values can be constant values or expressions.

Servlet

A servlet is a Java application that typically runs in a web server and provides server-side processing such as accessing a database and e-commerce transactions. Servlets are designed to handle HTTP requests (get, post, etc.) and are the standard Java replacement for a variety of other methods, including CGI scripts and Active Server Pages. The following figure shows the Nouveau Alliance servlet architecture.

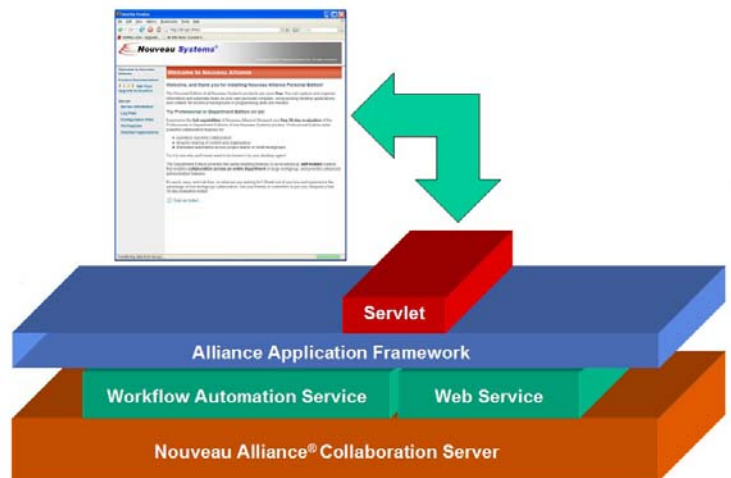


Figure 6 – Nouveau Alliance Servlet Architecture

Nouveau Alliance consists of a layer of foundation and extended services on a

collaborative server foundation. The Web Service is a foundation service which includes an embedded web server. The Workflow Automation Service is an extended service which manages the lifecycle of workflows and their related components. Through a general Application Framework, the servlet handles requests made by web applications (such as a web browser) and generates responses made back to the same web applications.

HTML Template

In Nouveau Alliance, servlets generally respond to requests by returning a standard HTML web page. This web page contains Nouveau Alliance property expressions that are filled-in at runtime.

```

<html>
<head>
<title>${INSTANCE.title}</title>
</head>
<body>
<div>
<p></p>
<img SRC="${INSTANCE.banner}" ALT="${INSTANCE.title}" BORDER=0 height=145 width=700>
</div>
<form action="${SERVLET.REQUEST.requestURI}" method="get" name="${INSTANCE.title}>
<INPUT type="hidden" name="workflowinstanceName" value="${INSTANCE.name}>
<INPUT type="hidden" name="workflowinterpreterName" value="${INTERPRETER.name}>
<center><h2>${INSTANCE.info_header}</h2></center>
<p></p>
<p>${INTERPRETER.message}</p>
<p></p>
<center><input type="submit" value="${INSTANCE.info_button}></center>
<p></p>
</form>
<div>
<p align="right"><a href="http://www.nouveausystems.com/products/flowspace/index.html">

</a></p>
</div>
</body>
</html>

```

Figure 7 – Sample HTML Template

In the figure above, there are properties (shown in bold) for the title of the page, the banner image, the name of the associated workflow instance and interpreter names and a “message” to be displayed on the page. These properties can be defined in the servlet’s configuration file (see Installation), in a workflow where it is referenced, or a property or function maintained by Nouveau Alliance.

Technology Application

The previously described technologies are the fundamental elements that compose the convenience packaging called Servlet Nodes.

Servlet Nodes

Servlet Nodes are a collection of pre-programmed workflow nodes that can be added and interconnected to build workflow-enabled, web-based applications.

Each Servlet Node has the following activity sequence:

1. Call the Set Property Activity to set the properties required by the node. A typical property is the “message” body of the HTML page.
2. Optionally call a sub-workflow activity to parse Comma Separated Values (CSV) for its input properties or execute a RDBMS SQL statement property to populate the web page.
3. Read in an HTML template and substitute the embedded property references. This typically includes the values generated by the sub-workflows.
4. Send the HTML page to the servlet for display by associated web browsers and wait for a response.
5. Process the feedback by the web browser user via another servlet request. This usually involves reading HTML form values and setting the output topology settings for the Servlet Node.

Servlet Node Palette

For ease of access, the Servlet Nodes have been packaged for installation as a FlowSpace workflow node category, to be installed in the application palette.



Figure 8 – Servlet Node Palette

Each of the Servlet Nodes in the palette is described below. Each node description consists of a list of the node's activities and default properties and a brief overview of its functionality.



Yes / No

This node is used to display a HTML message body with two buttons to prompt for a "yes" or "No" response. The resulting value is placed in an INSTANCE scoped property named "answer".

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

SetPropertyFromFileActivityImpl
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/yesno_template.html
  targetPropertyName=unformattedText

SetPropertyActivityImpl
  formattedText="$$unformattedText"
  SERVLET.RESPONSE.print=$formattedText
  close=$SERVLET.RESPONSE.close
```

```
WaitActivityImpl
  waitTime=NULL
```

```
SetPropertyActivityImpl
  TARGET=$INSTANCE.answer
```

Note: The set property sequence of "SERVLET.RESPONSE.println and \$SERVLET.RESPONSE.close, followed by the indefinite Wait, causes the templated page to be displayed in the user's default web browser. From this point on, this sequence of activities will be abbreviated by the aggregate activity label "DISPLAY_WEB_PAGE_ACTIVITIES" and their custom property settings.



Info

This node is used to display an HTML message body with a single button to move to the next node.

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/info_template.html
  ...
```



Info (End)

This node is identical to the previous Info node except for it is intended to be the last

node in the workflow to be executed. No response is elicited from user so no button is provided.

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

DISPLAY_WEB_PAGE_ACTIVITIES (no wait)
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/info_end_template.html
  ...
```



Text Area

This node provides an HTML multi-line text area INPUT element to prompt for a textual response. An HTML message body property is supplied to augment the text area. A single button is provided to move to the next node.

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/textarea_template.html
  ...
```

```
SetPropertyActivityImpl
  textResults=$INSTANCE.text
```



Text Field

This node provides an HTML single-line text field INPUT element to prompt for a short textual response. An HTML message body property is supplied to augment the text field. A single button is provided to move to the next node.

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/textfield_template.html
  ...
```

```
SetPropertyActivityImpl
  resultText=$INSTANCE.text
```



List (CSV)

This node provides an HTML multi-line SELECT element (list) to display a comma-separated list of items and their corresponding values. An HTML message body property is supplied to augment the list. A single button is provided to move to the next node.

```
SetPropertyActivityImpl
  INTERPRETER.displayList=a,b,c,d,e
  INTERPRETER.valueList=$INTERPRETER.displayList
  INTERPRETER.message=Message

SetPropertyActivityImpl
  INTERPRETER.resultsRow=<option
  value="$CONTEXT.value">$CONTEXT.display</option>
```

```
SubWorkflowActivityImpl
  workflowOID=OID for workflow "GetOptions"

SetPropertyActivityImpl
  INTERPRETER.options=$INTERPRETER.results

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/list_template.html
  ...

SetPropertyActivityImpl
  TARGET=$INSTANCE.option
```

Note: The activity sequence of two set properties, followed by a sub-workflow reference to “GetOptions” and ending with a set property of INTERPRETER.options creates HTML option select statements which are subsequently substituted into the HTML template for display to the user. The displayList/valueList properties contain the values to be displayed/selected in the HTML option fixture in the format specified by the INTERPRETER.resultsRow property. From this point on, this sequence of activities will be abbreviated by the aggregate activity label “GET_OPTION_VALUES” with their custom property settings.



List (DB)

This node is identical to the previous list node, except that the input to the list is in the form of an SQL Select statement and corresponding property mappings for the Select statement’s select expression. The property mappings allow you to specify which values are to be displayed versus selected for the list.

```
SetPropertyActivityImpl
  INTERPRETER.selectStatement=SELECT column1,
  column2 FROM table
  INTERPRETER.selectProps=INTERPRETER.display,I
  NTERPRETER.value
  INTERPRETER.resultsRow='<option
  value="$INTERPRETER.value">
  $INTERPRETER.display</option>'
  INTERPRETER.message=Message

SubWorkflowActivityImpl
  workflowOID=OID for workflow
  "GetOptionsFromDatabase"

SetPropertyActivityImpl
  INTERPRETER.options=$INTERPRETER.results

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/list_template.html
  ...

SetPropertyActivityImpl
  TARGET=$INSTANCE.option
```

Note: The activity sequence of a set property, followed by a sub-workflow

reference to “GetOptionsFromDatabase” and ending with a set property of INTERPRETER.options creates HTML option select statements which are subsequently substituted into the HTML template for display to the user. The SQL select statement and property list extract the values to be displayed/selected in the HTML option fixture in the format specified by the INTERPRETER.resultsRow property. From this point on, this sequence of activities will be abbreviated by the aggregate activity label “GET_OPTION_VALUES_FROM_DB” with their custom property settings.



Combo Box (CSV)

This node provides an HTML single-line SELECT element (combo box) to display a comma-separated list of items and their corresponding values. An HTML message body property is supplied to augment the combo box. A single button is provided to move to the next node.

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

GET_OPTION_VALUES
  INTERPRETER.displayList=1,2,3,4,5,6,7,8,9,10
  INTERPRETER.valueList=$INTERPRETER.displayList
  value="$CONTEXT.value">$CONTEXT.display</option>'
  ...

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=$$INSTANCE.website/doc/examples/
  ServletNodes/templates/combobox_template.html
  ...

SetPropertyActivityImpl
  TARGET=$INSTANCE.option
```



Combo Box (DB)

This node is identical to the previous combo box node, except that the input to the combo box is in the form of an SQL Select statement and corresponding property mappings for the Select statement’s select expression. The property mappings allow you to specify which values are to be displayed versus selected for the combo box.

```
SetPropertyActivityImpl
  INTERPRETER.message=Message

GET_OPTION_VALUES_FROM_DB
  INTERPRETER.selectStatement=SELECT column1,column2
  FROM table
  INTERPRETER.selectProps=INTERPRETER.display,INTERPRET
  ER.value
  INTERPRETER.resultsRow='<option
  value="$INTERPRETER.value">$INTERPRETER.display</opti
  on>'
  ...
```

```

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=${INSTANCE.website/doc/examples/
  ServletNodes/templates/combobox_template.html
  ...

```

```

SetPropertyActivityImpl
  TARGET=${INSTANCE.option

```



Check Box

This node provides multiple HTML check box buttons to support selection of zero or more items. Two properties with comma-separated values are used to specify the text displayed with the check box and the value that is associated with the button selection. A single button is provided to move to the next node.

```

SetPropertyActivityImpl
  INTERPRETER.message=Message

```

```

GET_OPTION_VALUES
  INTERPRETER.displayList=1,2,3,4,5
  INTERPRETER.valueList=${INTERPRETER.displayList
  INTERPRETER.resultsRow='<input type="checkbox"
  name="choice"
  value="${CONTEXT.value}">${CONTEXT.display<br>'

```

```

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=${INSTANCE.website/doc/examples/
  ServletNodes/templates/checkbutton_template.h
  tml
  ...

```

```

SetPropertyActivityImpl
  TARGET=${INSTANCE.option

```



Radio Box

This node provides multiple HTML radio box buttons to support selection of one item. Two properties with comma-separated values are used to specify the text displayed with the radio box and the value that is associated with the button selection. A single button is provided to move to the next node.

```

SetPropertyActivityImpl
  INTERPRETER.message=Message

```

```

GET_OPTION_VALUES
  INTERPRETER.displayList=1,2,3,4,5
  INTERPRETER.valueList=${INTERPRETER.displayList
  INTERPRETER.resultsRow='<input type="radio"
  name="choice"
  value="${CONTEXT.value}">${CONTEXT.display<br>'

```

```

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=${INSTANCE.website/doc/examples/
  ServletNodes/templates/radiobutton_template.ht
  ml
  ...

```

```

SetPropertyActivityImpl
  TARGET=${INSTANCE.choice

```



File

This node provides an HTML File INPUT element to prompt for a file. An HTML message body property is supplied to augment the file browser. A single button is provided to move to the next node.

```

SetPropertyActivityImpl
  INTERPRETER.message=Message

```

```

DISPLAY_WEB_PAGE_ACTIVITIES
  URLspecifier=${INSTANCE.website/doc/examples/
  ServletNodes/templates/file_template.html
  ...

```

```

SetPropertyActivityImpl
  textResults=${INSTANCE.contents

```

Installation

The Nouveau Alliance Servlet Nodes example can be located in three locations:

- A pre-installed workspace called “examples”.
- A workflow **template** called “examples”.
- A ZIP archive file named ServletNodes.zip that is available from the Nouveau Systems website.

The specific installation instructions depend on the location.

Pre-installed Workspace

If the “examples” workspace has already been installed, with workflow definitions (ServletNodes, GetOptions, GetOptionsFromDatabase and SampleBallot), no additional installation is required. If these workflow definitions do not exist, please follow the ZIP Archive instructions presented below.

Workspace Template

To install the “examples” workspace from a template, start FlowSpace and select the New Workspace option from the File menu and select the “examples” template. Once the workspace is created, confirm that the Servlet Nodes workflows are present (see

list above). If not, then follow the ZIP Archive instructions presented below.

ZIP Archive

The Servlet Nodes ZIP archive file can be downloaded from here:

<http://download.nouveausystems.com/whitepapers/ServletNodes.zip>

To install this zip file:

1. Change directory to your Alliance root directory
2. Unzip the ZIP archive file in this directory. The output of the ZIP file will be placed in the following directories:

doc/examples/ServletNodes/
etc/servlets

3. Start FlowSpace and open or create a workspace called “examples”.
4. Into this workspace, import the workflow definition archive file in this Servlet Nodes xml directory:

ServletNodes.zip

These XML-archive format files specify workflow definitions that contain all of the Servlet Nodes operations.

To load this definition into the “examples” workspace, select the “Import New Workflow” option from the FlowSpace section page number pop-up menu in the lower-right corner of the display.

Servlet Node Palette

You can install the Servlet Nodes as a workflow node category displayed on the FlowSpace palette.

This is a three step process:

- Open the previously imported “ServletNodes” workflow by selecting the “Go to Opened Workflow...” option from the page number pop-up menu

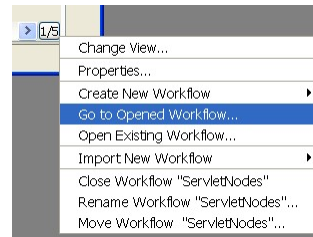


Figure 9 – Page Pop-up Menu

- Create a workflow node category by selecting the “Add A Workflow Node Category” option from the workflow node category palette’s combo box’s pop-up menu.

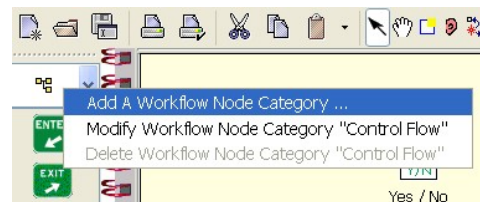


Figure 10 – Category Pop-up Menu

This menu option displays a dialog box to define the new workflow node category.

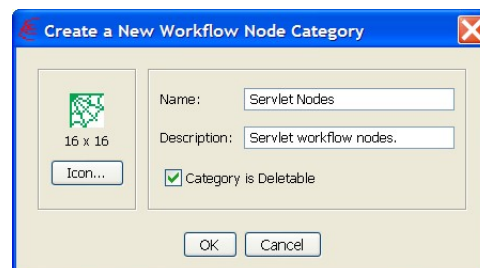



Figure 11 – Node Category Dialog Box

Fill in the dialog as shown, including the selection of the category icon () located in the following directory:

doc/examples/ServletNodes/xml/
ServletNodes_files

- Populate the Servlet Nodes category by selecting all of the nodes (as shown in the figure), hold the <CONTROL> key down and drag-and-drop the selected icons over the palette.

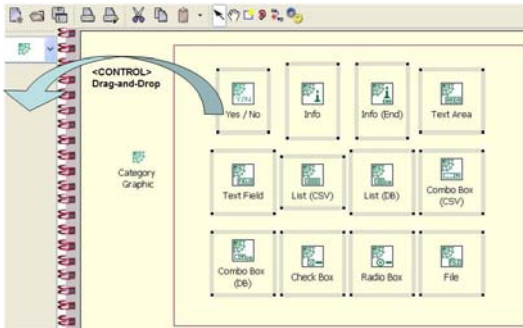


Figure 12 – Populating the palette

The icons will be added to the palette in row-major order (left-right, top-bottom).

Required License and Version

A Professional or Department license for FlowSpace version 3.2 and later is required to use the Servlet Nodes. If you're currently using Personal Edition, try FlowSpace Professional Edition *free* for 30 days now! Just point your web browser at your Nouveau Alliance server's home page, and click "Get Your Upgrade Evaluation!" to get started.

With the appropriate license, and a running Nouveau Alliance server, you can demonstrate the use of Servlet Nodes.

Demonstration

To demonstrate the use of Servlet Nodes we will create a simple workflow that guides a voter through the process of completing a sample ballot.

This Sample Ballot example shows how you can easily specify and

modify a sequence of web pages, with embedded control, to construct a web-based, workflow-backed application.

The following section presents a series of steps to construct the Sample Ballot example. If you do not wish to follow these instructions, an Import XML archive file containing the completed workflow can be found in:

`doc/examples/ServletNodes/xml/SampleBallot.zip`

To manually create the Sample Ballot application:

1. Make sure that the Nouveau Alliance Server is running.
The workflows and the servlets of the framework require this server process.
2. In the “examples” workspace, create a new workflow named “SampleBallot” by selecting the “Create New Workflow” option from the page number pop-up menu.
3. Add servlet nodes from the palette to build the workflow as shown in figure 13. Drag-and-drop each node from the palette to the workflow. Interconnect the nodes by holding down the ALT key (or SHIFT-ALT on some platforms) while depressing the mouse button over the source node and releasing it over the target node.

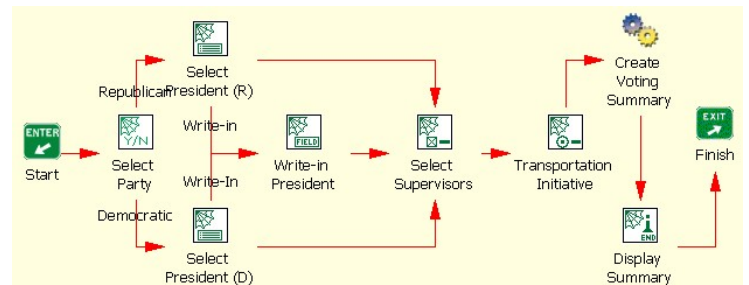


Figure 13 – Sample Ballot Workflow

4. Customize the properties of the nodes in this workflow. The properties of nodes and their constituent activities are

modified via the Properties / Activities dialog box.

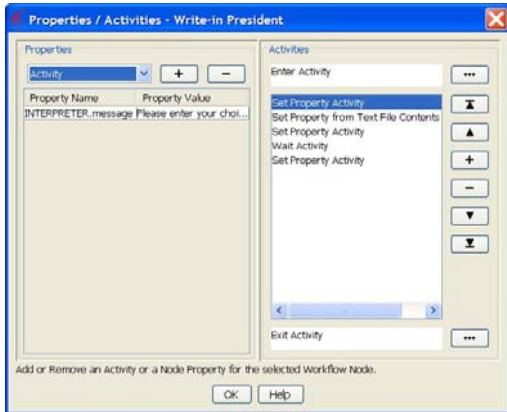


Figure 14 – Properties / Activities Dialog

This dialog is displayed when you double-click the mouse over a given workflow node.

Accordingly, please modify/add the following properties for these eight nodes:



Select Party

SetPropertyActivityImpl
 INTERPRETER.message=Please specify your political party affiliation.

SetPropertyActivityImpl (last)
 INSTANCE.Party=\$INSTANCE.answer (add)



Select President (R)

SetPropertyActivityImpl
 INTERPRETER.displayList=Sam Brownback,Rudy Giuliani, Mike Huckabee,Duncan Hunter, John McCain,Ron Paul,Mitt Romney, Tom Tancredo,Fred Thompson,Write-in
 INTERPRETER.message=Please select a candidate for President.

SetPropertyActivityImpl (last)
 INSTANCE.President=\$INSTANCE.option (add)



Select President (D)

SetPropertyActivityImpl
 INTERPRETER.displayList=Joe Biden,Hillary Rodham Clinton,Chris Dodd,John Edwards, Mike Gravel,Dennis Kucinich,Barack Obama,Bill Richardson,Write-in
 INTERPRETER.message=Please select a candidate for President.

SetPropertyActivityImpl (last)
 INSTANCE.President=\$INSTANCE.option (add)



Write-in President

SetPropertyActivityImpl
 INTERPRETER.message=Please enter your choice for President.

SetPropertyActivityImpl (last)
 INSTANCE.President=\$INSTANCE.text (add)



Select Supervisors

SetPropertyActivityImpl
 INTERPRETER.displayList=Mary Brown,Bob Johnson,Susan Jones,Joe Smith,John Thompson
 INTERPRETER.message=Please select as many as (5) Supervisors.

SetPropertyActivityImpl
 INTERPRETER.resultsRow='<input name="Supervisor" type="checkbox" value="\$CONTEXT.display">\$CONTEXT.display
'

SetPropertyActivityImpl (last)
 INSTANCE.Supervisors=\$SERVLET.REQUEST.parameterValues["Supervisor"] (add)



Transportation Initiative

SetPropertyActivityImpl
 INTERPRETER.displayList=Yes,No
 INTERPRETER.message=Should the City of San Jose issue a \$100M bond for road improvements?

SetPropertyActivityImpl (last)
 INSTANCE.Initiative=\$INSTANCE.choice (add)



Create Voting Summary

SetPropertyActivityImpl
 header=<table cellpadding=6 border=1>\n
 row1=<tr><td>Party:</td><td>\$INSTANCE.Party</td></tr>\n
 row2=<tr><td>President:</td><td>\$INSTANCE.President</td></tr>\n
 row3=<tr><td>Supervisor:</td><td>\$INSTANCE.Supervisors.toList.toString</td></tr>\n
 row4=<tr><td>Transportation Initiative:</td><td>\$INSTANCE.Initiative</td></tr>\n
 footer=</table>\n
 INTERPRETER.results=\$header\$row1\$row2\$row3\$row4\$footer



Display Summary

SetPropertyActivityImpl
 INTERPRETER.message=\$INTERPRETER.results

5. Optionally customize the node labels to reflect their names as shown in Figure 13. To change a label, just click on the label and modify the label's text.

6. Run the workflow-enabled servlet. In a web browser, type in the following location URL:

http://YOUR_HOST:8081/servlet/SampleBallot

where “YOUR_HOST” is the name to the host running the Nouveau Alliance server.

At this point, the web browser display will be updated to reflect the web page associated with the first servlet node in your workflow.



Figure 15 – Sample Ballot Display

Customization

As with any Nouveau Alliance Workflow-enabled Web Application, the Sample Ballot application can be customized by modifying its associated workflow. You can specify additional nodes representing more ballot items and you can embellish existing nodes to provide more information or more items per page. You also alter the sequencing of the nodes and the control logic which guides the voter. For example, you can add a check to the Select Supervisors node to restrict the number of selections to a subset of the total.

For the Servlet Nodes framework, you can create additional nodes that perform specialized tasks in your applications. For example, you may use a common form for registering

and authenticating users of your web site.

To add a servlet node to your library, it is simple as adding a node to a workflow, customize its activities, properties, icon and label, and Drag-and-drop it to a target palette.

Summary

This white paper describes the use of a set of pre-configured servlet nodes to build web-based applications with Nouveau Alliance. Some of the technologies used include:

- Workflow Enabled Servlets
- Flexible Workflow Modeling

These servlet nodes can be further customized and installed into a custom palette to create other applications. The paper demonstrates that you do not need a large, complex framework, or specialized skills, to rapidly construct web-based applications.