vbug

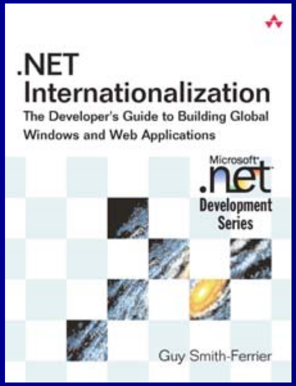
An Introduction To Windows Workflow Foundation

Microsoft CERTIFIED Professional

Guy Smith-Ferrier
guy@guysmithferrier.com
Blog: http://www.guysmithferrier.com

About…

- Author of .NET Internationalization
  - Visit http://www.dotneti18n.com to download the complete source code


.NET Internationalization
The Developer's Guide to Building Global Windows and Web Applications
Microsoft .net Development Series
Guy Smith-Ferrier

- 25% of 4 Chaps From Blighty
  - http://www.4chapsfromblighty.com
  - Podcasts and blog about development issues and the UK developer community

Agenda

- Overview, Resources, Books
- Creating Workflows
- Activities
  - Looping, Conditional, Composite
- Custom Activities
- Exception Handling
- Communication Between Workflow And Host
- Persistence
- State Machines
- Hosting The Designer

Overview

- Windows Workflow Foundation is:-
  - A library of workflow activity classes
  - A library of workflow service classes
  - A workflow engine
  - A workflow designer
- Runs on Vista, Windows XP, Windows Server 2003

Resources

- http://wf.netfx3.com/
  - .NET Framework 3.0 (including Windows Workflow Foundation)
  - Visual Studio 2005 Extensions For Windows Workflow Foundation
  - Windows SDK for Vista and the .NET Framework 3.0
- http://msdn.microsoft.com/workflow
- Mike Taulty's Windows Workflow Foundation Nuggets
  - http://wf.netfx3.com/files/folders/screencasts/default.aspx
- Webcasts
  - http://blogs.msdn.com/pandrew/articles/460630.aspx
- Windows Workflow Foundation Virtual Labs
  - http://msdn.microsoft.com/virtuallabs/windowsworkflow/
- Windows Workflow Foundation Wiki
  - http://wiki.windowsworkflowfoundation.eu/default.aspx/WF/WindowsWorkflowFoundationWiki.html

Books

- Essential Windows Workflow Foundation
  - Dharma Shulka and Bob Schmidt, Addison-Wesley
- Foundation of WF
  - Brian R. Myers, Apress
- Presenting Windows Workflow Foundation
  - Paul Andrew, James Conard and Scott Woodgate, Sams Publishing

Demo

Creating a simple workflow

Demo

The Workflow Debugger

Examples Of Workflow (Why Do You Care ?)

- Order Processing
- Stock Management
- Timesheet Processing
- Collaborative Documents
- Bug Tracking
- Wizards

Code Only Workflows

```
public class Workflow1 : SequentialWorkflowActivity
{
    public Workflow1()
    {
        CodeActivity codeActivity1 = new CodeActivity();
        codeActivity1.ExecuteCode +=
            delegate { Console.WriteLine("Hello World");};
        Activities.Add(codeActivity1);

        DelayActivity delayActivity1 = new DelayActivity();
        delayActivity1.TimeoutDuration = new TimeSpan(0, 0, 2);
        Activities.Add(delayActivity1);

        CodeActivity codeActivity2 = new CodeActivity();
        codeActivity2.ExecuteCode +=
            delegate { Console.WriteLine("Goodbye Cruel World"); };
        Activities.Add(codeActivity2);
    }
}
```

Code Only Workflows (continued)

```
static void Main(string[] args)
{
    WorkflowRuntime runtime = new WorkflowRuntime();

    runtime.StartRuntime();

    WorkflowInstance instance =
        runtime.CreateWorkflow(typeof(Workflow1));

    instance.Start();

    Console.ReadLine();
}
```

Demo

Code only workflows

XOML Workflows

```
<SequentialWorkflowActivity x:Class="ConsoleApplication1.Workflow1"
x:Name="Workflow1" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/workflow">
  <CodeActivity x:Name="activity1" ExecuteCode="activity1_ExecuteCode" />
    <x:Code><![CDATA[
    void activity1_ExecuteCode(object sender, EventArgs e)
    {
        Console.WriteLine("Hello World");
    }
    ]]></x:Code>
  <DelayActivity TimeoutDuration="00:00:02" x:Name="delayActivity1" />
  <CodeActivity x:Name="activity2" ExecuteCode="activity2_ExecuteCode" />
    <x:Code><![CDATA[
    void activity2_ExecuteCode(object sender, EventArgs e)
    {
        Console.WriteLine("Goodbye Cruel World");
    }
    ]]></x:Code>
</SequentialWorkflowActivity>
```

Compiling XOML Workflows Using The Command Line Compiler

- Workflows can be compiled using the command line Workflow Compiler (wfc.exe)

```
"C:\Program Files\Microsoft SDKs\Windows\v6.0\Bin\"wfc HelloWorld.xoml
```

Compiling XOML Workflows Using WorkflowCompiler Class

```
static Type CompileWorkflow(string xomlfile, string workflowName)
{
    WorkflowCompiler compiler = new WorkflowCompiler();
    WorkflowCompilerParameters parameters =
        new WorkflowCompilerParameters();
    parameters.GenerateInMemory = true;
    WorkflowCompilerResults results =
        compiler.Compile(parameters, new string[] { xomlfile });
    if (results.Errors.Count > 0)
    {
        foreach (CompilerError error in results.Errors)
        {
            Console.WriteLine(error.ErrorText);
        }
        return null;
    }
    return compilerResults.CompiledAssembly.GetType(workflowName);
}
```

Demo

XOML only Workflows

Demo

Looping, Conditionals And Composite Activities

Writing Custom Activities

- Inherit from System.Workflow.ComponentModel.Activity and override the Execute method

```
public class WriteLineActivity: Activity
{
    protected override ActivityExecutionStatus
        Execute(ActivityExecutionContext executionContext)
    {
        System.Console.WriteLine(Text);
        return ActivityExecutionStatus.Closed;
    }

    private string text;
    public string Text
    {
        get { return text; }
        set { text = value; }
    }
}
```

Properties vs. Dependency Properties

- Properties have storage allocated regardless of whether storage is used
- Dependency properties only have storage allocated if they are assigned a value
- Dependency properties support change notification
- Dependency properties support binding to other workflow activities

Dependency Properties

```
public static DependencyProperty TextProperty =
    DependencyProperty.Register(
    "Text", typeof(string), typeof(WriteLineActivity));

public string Text
{
    get { return (string) base.GetValue(TextProperty); }
    set { base.SetValue(TextProperty, value); }
}
```

Activity Validators

```
public class TextValidator : ActivityValidator
{
    public override ValidationErrorCollection
        ValidateProperties(ValidationManager manager, object obj)
    {
        ValidationErrorCollection errors =
            base.ValidateProperties(manager, obj);
        WriteLineActivity activity = (WriteLineActivity)obj;
        if (activity != null && activity.Parent != null)
        {
            if (String.IsNullOrEmpty(activity.Text))
                errors.Add(new ValidationError("Text is empty", 1));
        }
        return errors;
    }
}
```

```
[ActivityValidator(typeof(TextValidator))]
```

Exception Handling

- Add a CodeActivity to throw an ArgumentException
- Run the workflow and show that the exception is handled by the WorkflowRuntime's WorkflowTerminated event
- Change the Workflow Designer to View Fault Handlers
- Add a FaultHandlerActivity to the FaultHandlersActivity and set FaultType to System.ArgumentException
- Add a CodeActivity to the FaultHandlerActivity with the following ExecuteCode

```
Console.WriteLine("Handled by workflow fault handler");
```

Exception Handling (continued)

- Run the workflow and show that the exception is handled by the Workflow instance's FaultHandlerActivity
- Add a SequenceActivity to the workflow and move codeActivity1 into the SequenceActivity
- Right click the SequenceActivity and select View Fault Handlers
- Add a FaultHandlerActivity to the FaultHandlersActivity and set FaultType to System.ArgumentException

Exception Handling (continued)

- Add a CodeActivity to the FaultHandlerActivity with the following ExecuteCode

```
Console.WriteLine("Handled by sequence fault handler");
```

- Run the workflow and show that the exception is handled by the Sequence's FaultHandlerActivity

CallExternalMethodActivity

- The CallExternalMethodActivity provides a means for the workflow to call methods exposed by the host
- Create an interface that is decorated with the ExternalDataExchange attribute

```
[ExternalDataExchange]
public interface IExternalDataExchange
{
    bool SendEmail(string address, string body);
}
```

CallExternalMethodActivity (continued)

- Create an implementation of the interface

```
public class ExternalDataExchangeImplementation:IExternalDataExchange
{
    public bool SendEmail(string address, string body)
    {
        return true;
    }
}
```

- Make the implementation available to the workflow runtime

```
ExternalDataExchangeService edes = new ExternalDataExchangeService();
workflowRuntime.AddService(edes);
edes.AddService(new ExternalDataExchangeImplementation());
```

CallExternalMethodActivity (continued)

- Use the implementation in the workflow
  - Add a CallExternalMethodActivity to the workflow
  - Set the InterfaceType to IExternalDataExchange
  - Set the MethodName to SendEmail
  - Bind the parameters (body, address) and ReturnValue to corresponding properties on the workflow class

Workflow Persistence - Database Setup

- Create a SQL Server 2005 Express database (e.g. WorkflowPersistence)
- Run the following scripts in C:\WINDOWS\Microsoft.NET\Framework\v3.0\Windows Workflow Foundation\SQL\EN
  - SqlPersistenceService_Schema.sql
  - SqlPersistenceService_Logic.sql

Workflow Persistence – Adding The Persistence Service

- Add the following code to the WorkflowRuntime initialization code:-

```
string connectionString = @"Data Source=CAPLAPTOPTOSH1\SQLEXPRESS;"+
    "Initial Catalog=WorkflowPersistence;Integrated Security=True";

SqlWorkflowPersistenceService persistenceService =
    new SqlWorkflowPersistenceService(connectionString, true,
    new TimeSpan(10, 0, 0), new TimeSpan(0, 0, 2));

workflowRuntime.AddService(persistenceService);
```

Workflow Persistence – Events

```
workflowRuntime.WorkflowPersisted +=
    delegate{Console.WriteLine("Persisted");};

workflowRuntime.WorkflowLoaded +=
    delegate { Console.WriteLine("Loaded"); };
```

Demo

State Machine

Hosting The Designer - Resources

- Everything About Re-Hosting the Workflow
  - Includes a downloadable working example of hosting the designer
  - http://msdn2.microsoft.com/en-us/library/aa480213.aspx
- Create And Host Custom Designers With The .NET Framework 2.0
  - http://msdn.microsoft.com/msdnmag/issues/06/03/DesignerHosting
- Tailor Your Application by Building a Custom Forms Designer with .NET
  - http://msdn.microsoft.com/msdnmag/issues/04/12/CustomFormsDesigner/default.aspx
- Mike Taulty
  - http://mtaulty.com/CommunityServer/blogs/mike_taultys_blog/archive/2006/08/18/5935.aspx

Summary

- Overview, Resources, Books
- Creating Workflows
- Activities
  - Looping, Conditional, Composite
- Custom Activities
- Exception Handling
- Communication Between Workflow And Host
- Persistence
- State Machines
- Hosting The Designer