

Meet The New Workflow

Windows Workflow Foundation 4

Maurice de Beijer



DEVELOPMENTOR

DEVELOPING PEOPLE WHO DEVELOP SOFTWARE

Logistics

- **Please mute your microphone**
- **Twitter: #dmwf4**
- **Live Meeting Q&A**



Objectives

- **What is the goal of WF4**
- **Designing workflows**
- **Running workflows**
- **Creating custom activities**
- **Long running workflows**



Windows Workflow Foundation Goals

- **Software is about solving business problems**
 - Make sure all orders are processed
 - Keep the inventory levels at a reasonable level
 - Make sure all invoices are paid
- **Code is about solving computer problems**
 - Insert the order into a database table and retry the transaction if it fails the first time
 - Send various SOAP messages to the suppliers systems
 - Resend messages if the other system is unavailable
 - Check the database for invoice records older than x days with no related payment records



Windows Workflow Foundation Goals

- **There is a clear mismatch between business problems and the code that tries to solve them**



Windows Workflow Foundation Goals

- Windows Workflow Foundation creates a separation of concerns
- **Workflows** are all about **WHAT** needs to be done
- Workflows are composed of activities
- Activities are the basic blocks and should model business activities
- Activities are either composed out of other activities or build using code.
- **Activities** (code) all are about **HOW** something needs to be done.



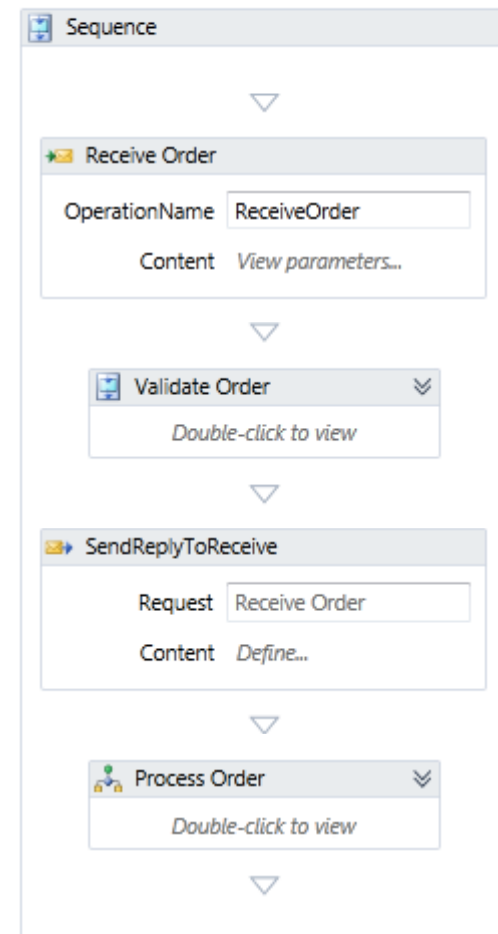
Designing Workflows

- **All workflows are just activities**
- **Two basic styles:**
 - Sequential
 - Flowchart
- **The WF3 StateMachine is missing!**
- **Either XAML or Code**
- **The designers produce XAML**



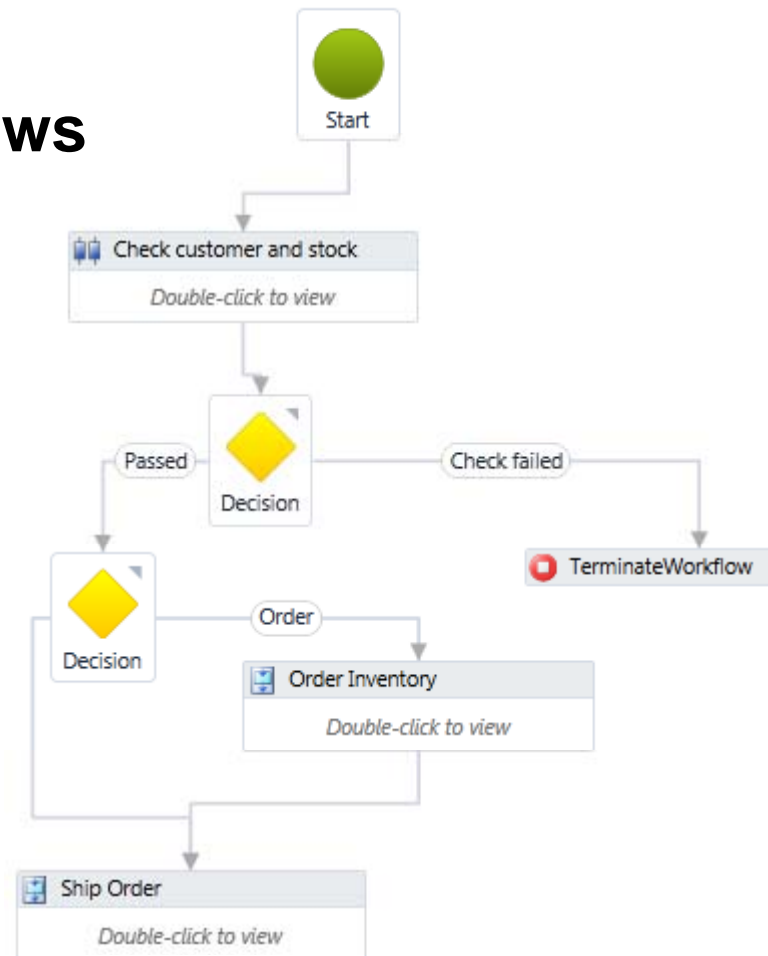
Designing Workflows - Sequence

- **Very structured**
- **Execution is explicit**
- **Loops and branches must be explicitly modeled**
- **Suitable for rigid processes**



Designing Workflows - Flowchart

- **Very flexible**
- **Execution directed by arrows**
- **Control is very visual**
- **Can jump anywhere**
- **Use all activities**
- **Suitable for human interaction processes**



Designing Workflows & Data

- **Argument to get data into or out of an activity**
 - InArgument
 - OutArgument
 - InOutArgument
- **A Variable is used to store data in an activity**



Executing Workflows

- Different options depending on need
- **WorkflowInvoker**
- **WorkflowApplication**
- **WorkflowServiceHost**



WorkflowInvoker

- **Simple**
- **Synchronous**
- **Great for unit testing**
- **Runs on calling thread**

```
var workflow = new HelloWorldWorkflow();  
workflow.FirstName = "Maurice";  
  
var output = WorkflowInvoker.Invoke(workflow);  
  
PrintOutput(output);
```



WorkflowApplication

- **Powerful**
- **Asynchronous**
- **Uses callbacks to indicate status changes**
- **Set SynchronizationContext to control threading**

```
var workflow = new HelloWorldWorkflow();  
workflow.FirstName = "Maurice";  
  
var wa = new WorkflowApplication(workflow);  
wa.Completed = e => PrintOutput(e.Outputs);  
wa.Run();
```



WorkflowServiceHost

- Exposes a workflow as a WCF service
- Self hosting or IIS/WAS
- Runs on thread provided by WCF

```
var workflow = new HelloWorldService();  
var baseAddress = new Uri("http://localhost:8080/HelloWorkflow");  
var wsh = new WorkflowServiceHost(workflow, baseAddress);  
  
wsh.Open();  
Console.WriteLine("The WorkflowServiceHost is listening.");  
Console.ReadLine();  
wsh.Close();
```



Creating Custom Activities

- **Use custom activities to model business activities**
- **Create composite activity compose existing activities (XAML)**
- **Use Code Activity for basic activities**
- **Base class depends on needs**
 - CodeActivity for simple work
 - AsyncCodeActivity for asynchronous IO
 - NativeActivity for full control



CodeActivity

- Use InArgument or OutArgument to pass data
- Use the CodeActivityContext to read/write these properties

```
public class MyWriteLine : CodeActivity
{
    public InArgument<string> Text { get; set; }

    protected override void Execute(CodeActivityContext context)
    {
        var text = Text.Get(context);
        Console.WriteLine("MyWriteLine: {0}", text);
    }
}
```



NativeActivity

- **Both synchronous and asynchronous**
- **Can schedule child activities**
 - `context.ScheduleActivity()`
- **Can run asynchronously**
 - `context.CreateBookmark("MyBookmark", MyCallback)`
- **Finished when all child activities and bookmarks are done**



Long Running Workflows

- **Save workflow state to disk**
- **Implementation for SQL Server in the box**
- **Can create custom implementations**
- **Use 2 SQL scripts to create the database**
- **Host is responsible for mapping workflow type**

```
osql -E -S .\sqlexpress -Q "Create Database WorkflowInstanceStore"  
osql -E -S .\sqlexpress -d WorkflowInstanceStore -i SqlWorkflowInstanceStoreSchema.sql  
osql -E -S .\sqlexpress -d WorkflowInstanceStore -i SqlWorkflowInstanceStoreLogic.sql
```



Long Running Workflows

```
private static Guid CreateAndPersist()
{
    var wa = CreateWorkflowApplication("Maurice");
    wa.PersistableIdle = e => PersistableIdleAction.Unload;
    wa.Run();

    return wa.Id;
}
```

```
private static void LoadAndResume(Guid instanceld)
{
    var wa = CreateWorkflowApplication();
    wa.Completed = e => PrintOutput(e.Outputs);
    wa.Load(instanceld);
}
```



Summary

- **Workflow allows us to solve business problems at a high level**
- **Flowcharts give us a lot of flexibility**
- **We need to create activities that model our business activities**
- **It's easy to create workflows that will stick around for months**



More Information

- **My blog**
<http://msmvps.com/blogs/theproblemsolver/>
- **Browse Develop.com**
<http://browse.develop.com/workflow/>
- **MSDN**
[http://msdn.microsoft.com/en-us/library/dd489441\(VS.100\).aspx](http://msdn.microsoft.com/en-us/library/dd489441(VS.100).aspx)
- **E-mail**
mauricedb@develop.com

