2008 ESRI Developer Summit

March 17–20, 2008 • Palm Springs, CA

Building and Extending Tasks for ArcGIS Server .NET Web Applications

Rex Hansen, Sentha Sivabalan

Schedule

- 75 minute session
  - 60 – 65 minute lecture
  - 10 – 15 minutes Q & A following the lecture
- Cell phones and pagers

- Please complete the session survey – we take your feedback very seriously!

Introductions

- Who are we?
  - ArcGIS Server .NET Developers
- Who are you?
  - Experience With ASP.NET?
  - Experience With AJAX?
  - New Web ADF Developers?
  - Current Web ADF Developers?

Session agenda

- ASP.NET Developer Review
- Web ADF Task Framework Overview
- Working with Task Framework controls
- Extending Web ADF Tasks
- Creating custom Web Tasks
- New 9.3 features (where applicable)
- Best practices

Web Application Development takes many forms

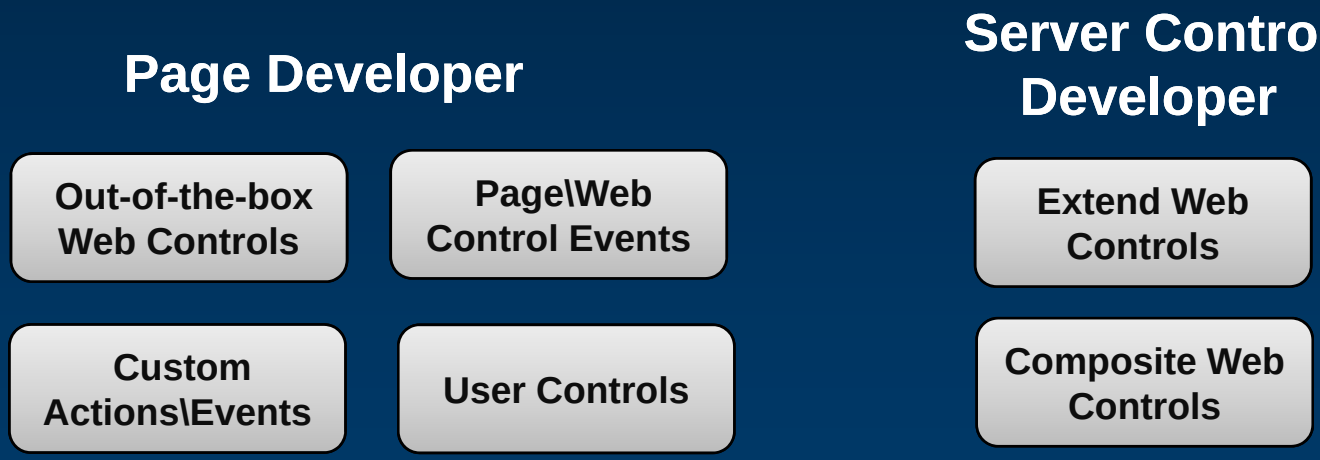
- **Server Side development**
  - **Traditional ASP.NET/JSF/Cold Fusion**
  - **Some JavaScript, most logic is on the web or server tier**
  - **At 9.2, the ADF was mostly server development**
- **Client Side development**
  - **All things happen on client - JavaScript**
  - **Only talk to services through HTTP/Get/Post/Put**
  - **Limited platform/environment**
- ***Hybrid***
  - **A well balanced combination of client and server logic**
  - **At 9.3, Web ADF “is” a Hybrid development platform**

What does it take to become an ADF Developer?
- Web Development 101 -

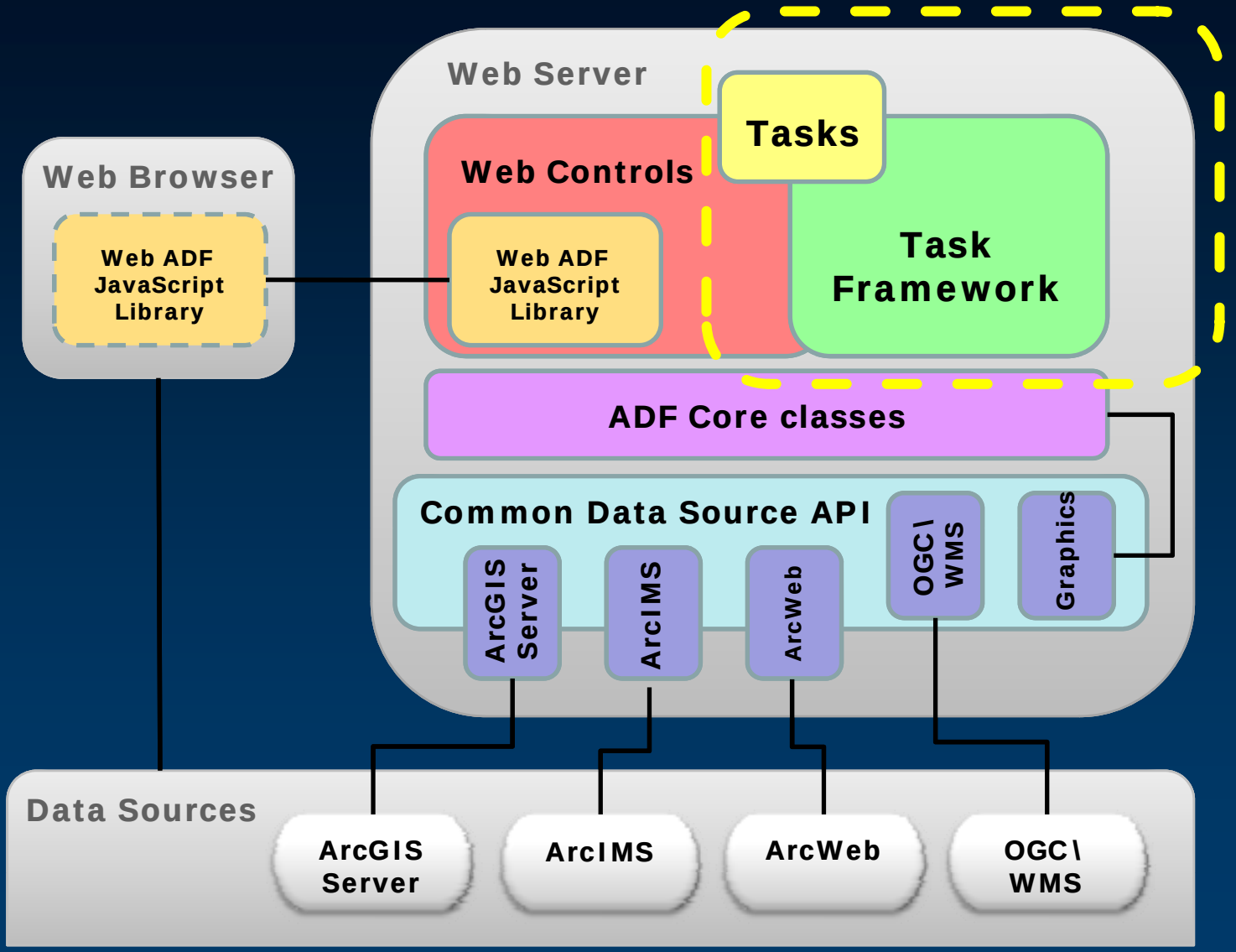
- Must be familiar with Web Development
  - Server: ASP.NET
    - C# OR VB.NET
  - Client: ASP.NET AJAX
    - JavaScript
    - Partial Post backs
    - Application Services
  - HTML/CSS
- Have Familiarity with IIS
  - Performance, Scalability, ASP.NET Worker Process

ASP.NET Page and Server Control Developers

- Add functionality to a page/User control or package in control library
- Handle user actions, process input and generate results
- Page development easier, but less flexible



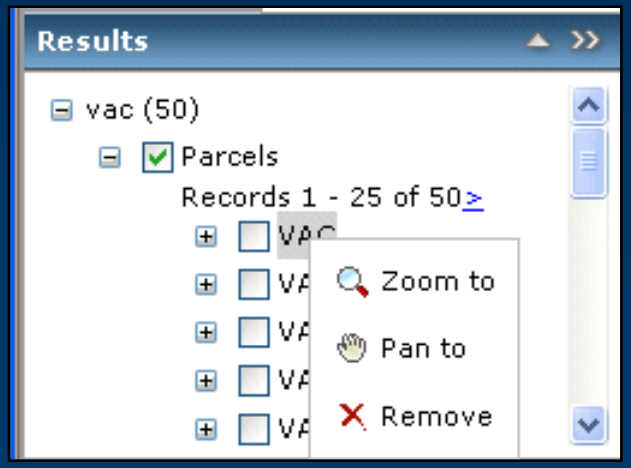
Web Application Developer Framework



Web Tasks

- Visual components designed to perform a set of related actions and generate results

- Encapsulate a workflow
  - UI for gathering input
  - Perform some action based on inputs
  - Generate results

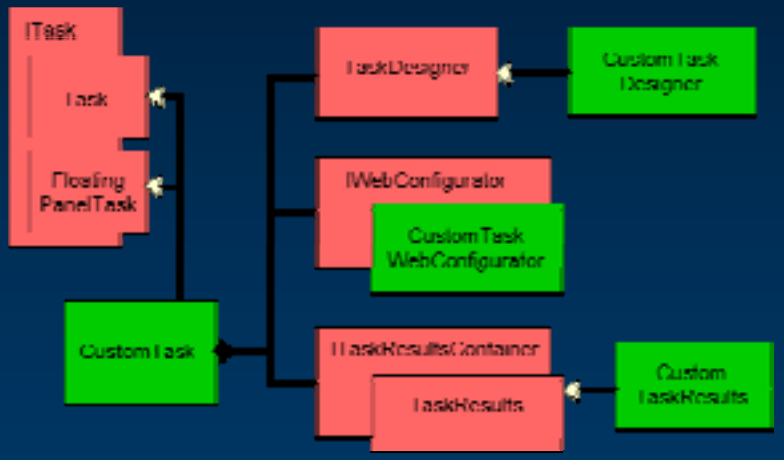


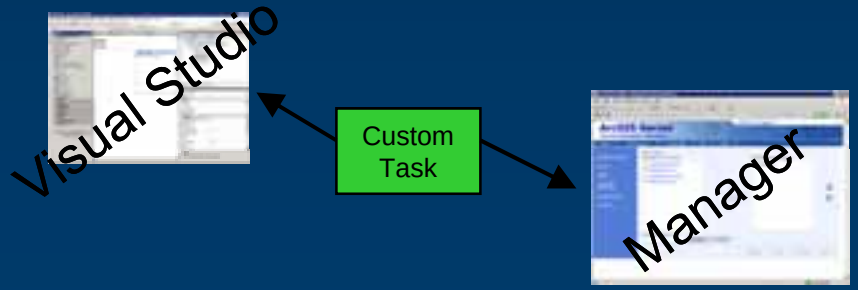
Web ADF Tasks in the box

- Search
  - Search Attributes of a layer
- Find Address
  - Use locator services to perform geocoding tasks
- Find Place
  - Gazetteer search using ArcWeb
- Query
  - Perform queries against layer attributes
- Geoprocessing
  - Gather inputs for published Geoprocessing tools and models and execute using this task
- Editor
  - Web based geodatabase editing (versioned and non-versioned edits are allowed)
- Print
    - Simple printing that provides web printing for the web application

The ADF Task Framework

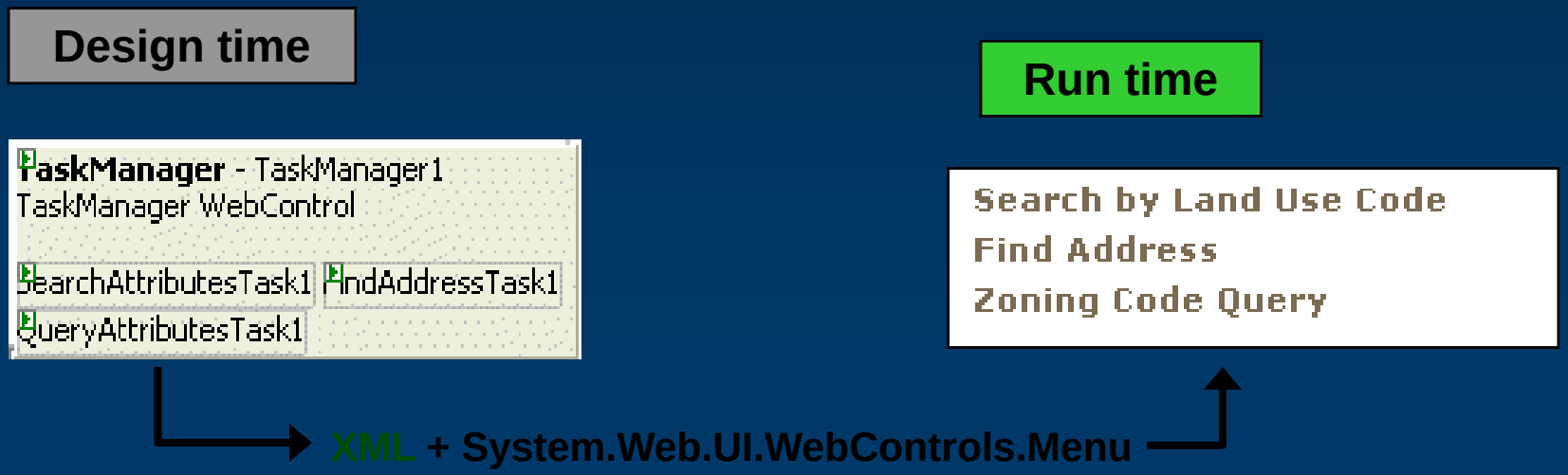
- Utilized by out-of-the-box ADF Tasks
- Extensible Framework provides for:
  - Organization of tasks
  - Feedback when a task is executing
  - Management of results
    - Display
    - Visualization on Map
    - Delete, Refresh, Re-run
  - Distribution and configuration of tasks in Visual Studio and Manager





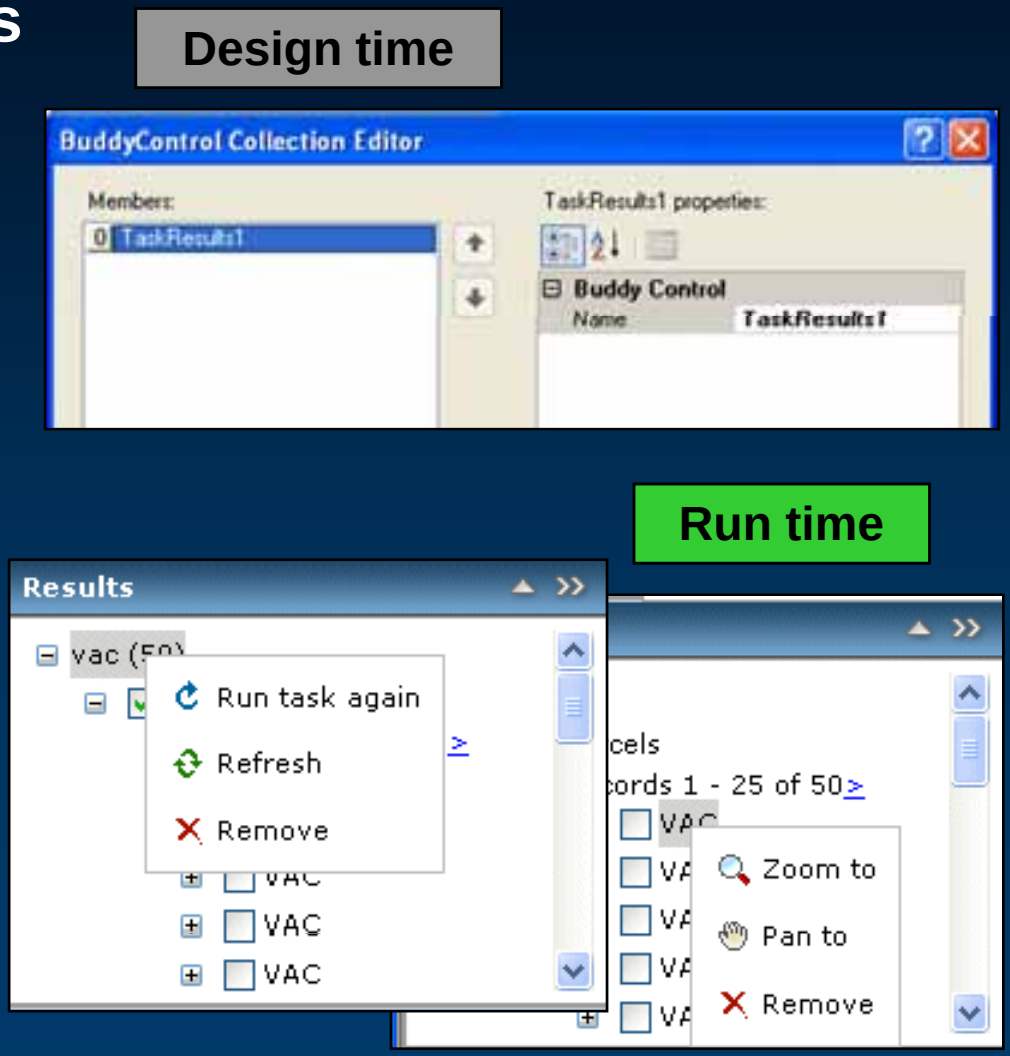
TaskManager control

- Convenient, not required
- Organizes tasks in a Web application
- Generates hierarchical XML at run time
  - XML can be bound to ASP.NET menu or tree view
  - BuddyControl property controls binding



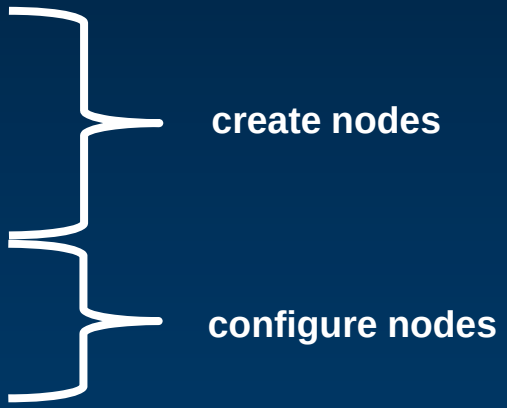
TaskResults control

- Displays results of tasks
  - ADO.NET dataset
- Run time
  - Results presented as nodes in tree view
  - Context menu for zoom or pan to a feature, highlight a feature in a result set, or remove task results



Working with TaskResults

- Use to display results, with or without a custom task
- Provides access to Map instance
- Setup results as nested nodes for display and use
- Methods
  - CreateTaskResultsNode
  - CreateDataTableNode
  - CreateAndAddDataRowNodes
  - SetupTaskResultNode
  - SetupContextMenu



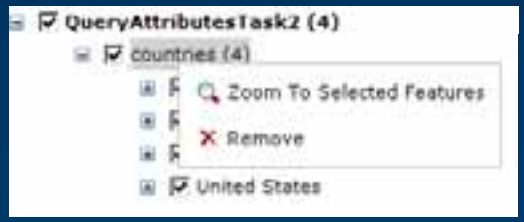
Display results in TaskResults

- Call DisplayResults() method to display results in a the TaskResults control or Map
  - Three supported result types:
    - SimpleTaskResult
      - header and body
    - DataSet
      - Each DataTable added as a node. If DataTable is GraphicsLayer, rendered in Map.
    - TaskResultNode
      - Used to package diverse result types

Using TaskResultNode

- MapResourceNode
  - Create node from a map resource definition, add to map
- GraphicsLayerNode
  - Create node from Web ADF GraphicsLayer, add to map
- FeatureNode
  - Create node from a DataRow with Web ADF geometry

- ContextMenus
  - Preconfigured for node type and content
  - Accessible via TaskResults properties
    - e.g. GraphicsLayerContextMenu



How to leverage the Task Framework

- Different Skill levels can take advantage of the Task Framework in different ways
  - Add results directly to the TaskResults container
  - Create a User Control
  - Develop a custom Task to distribute and integrate with ArcGIS Server Manager
  - Extend existing Task to modify behavior

Page and Control Development in the ADF

- Page developers – Display results or create User Control tasks
- Server Control developers – Create custom redistributable tasks

Demo Custom Tool - TaskResults

User Control Tasks

- Easiest way to create a task
  - Drag-and-drop to create UI
  - Add code for behavior
- To be released to the Code Gallery first

User Control Task Implementation

- Create user control
- Inherit from UserControlTaskPanel
  - Implement **ExecuteTask**
    - Use input parameter
    - Return result as output
  - Call **Start** on execute button click
- In your web application
  - Add a UserControlTask to your TaskManager
  - Add your user control task panel to your application
  - Point UserControlTask to your panel

User Control Demo

When to create a User Control Task

Advantages:

- Easy way to extend your Web Mapping Application

Disadvantages:

- Not easy to distribute
- Requires IDE configuration
- Not configurable in Manager

Use to create a task

- For use in one web application
- For sharing with developers who can do simple configuration in Visual Studio .NET

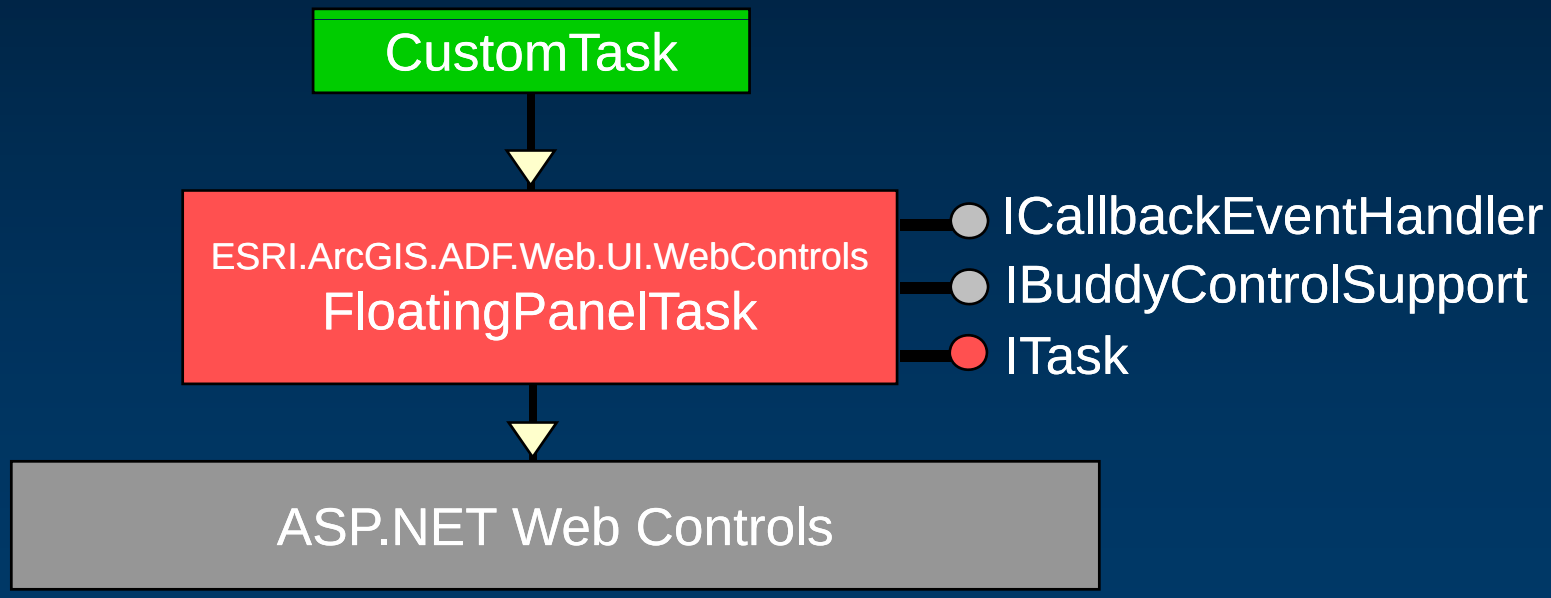
When you need to create a custom Web Task?

When you need to:

- **Create a distributable UI component**
- **Integrate into Manager**

Custom Web Task Implementation

- **You are creating a custom Web control!**
  - **Extend FloatingPanelTask abstract base class**
  - **Implements ITask which defines Web Task contract**



Key methods and properties

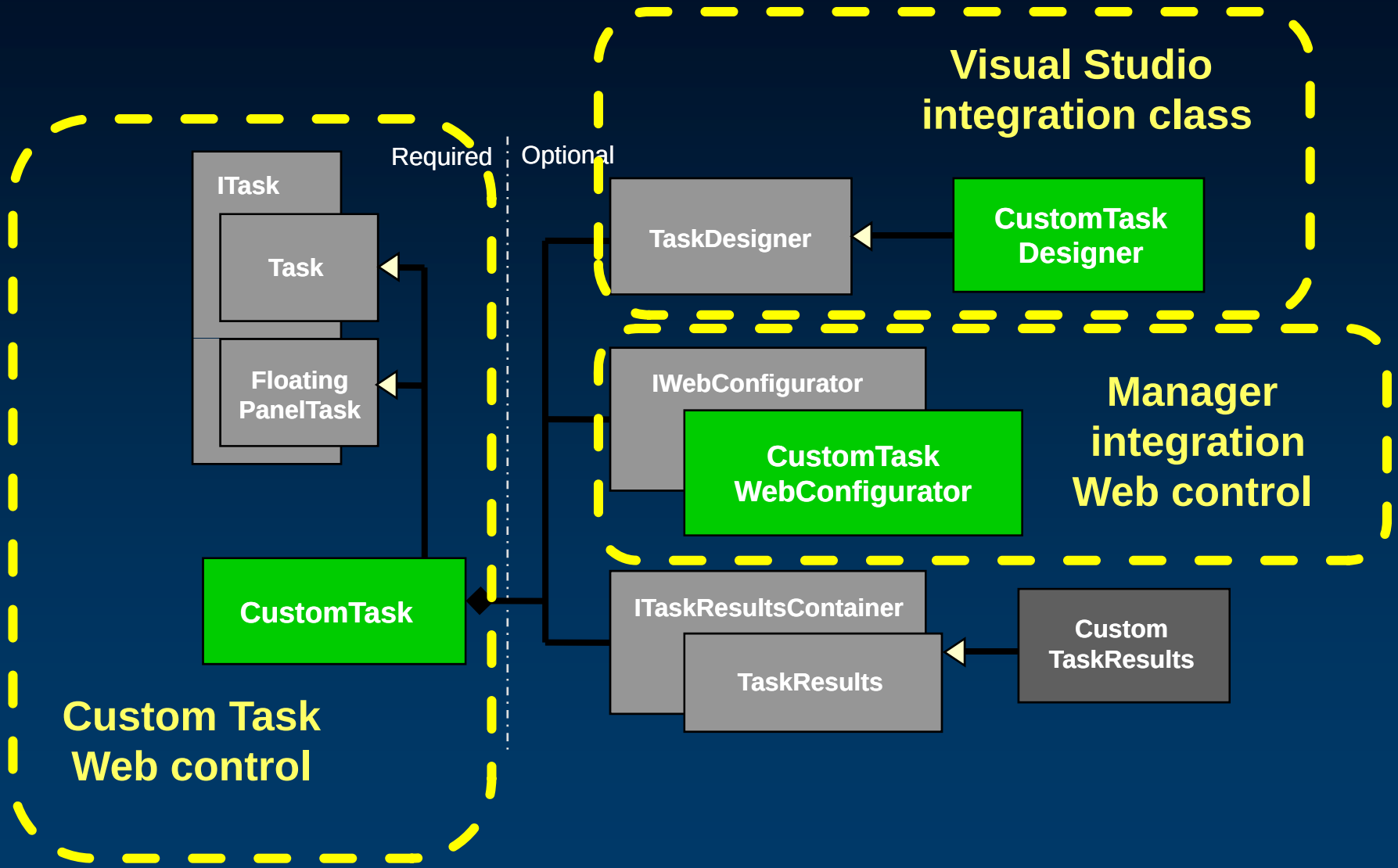
Methods

- **CreateChildControls** – create the visual interface
- **GetCallbackResult** – parse input, return callback results
- **ExecuteTask** – execute the task and generate results

Properties

- **Input** – store input data from the user
- **Results** – after task execution, generate results for use in a TaskResults control
- **CallbackResults** – collection of results designed to update browser content using Web ADF JavaScript

Custom Task Classes in this Session



Design Task Runtime Interface for user input

- Override CreateChildControls
  - Create content user will interact with in the browser
  - Called on every postback\callback during Page Load

DevTask.cs

```
[System.Web.UI.ToolboxData("<{0}:DevTask runat=\"server\" > </{0}:DevTask>")]
public class DevTask: ESRI.ArcGIS.ADF.Web.UI.WebControls.FloatingPanelTask
    {
        private System.Web.UI.WebControls.TextBox textBox = null;

        protected override void CreateChildControls()
        {
          base.CreateChildControls();
          textBox = new System.Web.UI.WebControls.TextBox();
          textBox.ID = "textBox";
          Controls.Add(textBox);
```

Utilize ADF JavaScript to execute task

- Task and FloatingPanelTask include Web ADF JavaScript to support task execution
- Manage events in browser to execute task

display_task.js

```
function executeTask(...
function
  startActivityIndicator{...
function startJob{...
```

DevTask.cs

```
protected override void CreateChildControls()
{
    . . .
    string getArgumentJS = string.Format("'textBoxValue=' +
      document.getElementById('{0}').value", textBox.ClientID);
    string onClick = string.Format("executeTask({0},\"{1}\");",
      getArgumentJS, CallbackFunctionString);
    button.Attributes.Add("onclick", onClick);
}
```

Processing user input

- Override GetCallbackResult
  - Event arguments packaged in _callbackArg variable
  - Call to GetCallbackResult in base class executes task and returns CallbackResults
- Input property stores user input
  - Storage format determined by developer

DevTask.cs

```
 public override string GetCallbackResult()
{
    System.Collections.Specialized.NameValueCollection keyValColl =
      CallbackUtility.ParseStringIntoNameValueCollection(_callbackArg);
    Input = keyValColl["textBoxValue"];
    return base.GetCallbackResult();
}
```

Generating Results

- Override **ExecuteTask** method
- **Results** property stores content to be displayed in a TaskResults control
  - SimpleTaskResult
  - Dataset
  - TaskResultNode

DevTask.cs

```
public override void ExecuteTask()
{
    string textBoxValue = Input as string;

    ESRI.ArcGIS.ADF.Web.UI.WebControls.SimpleTaskResult simpleresult =
     new ESRI.ArcGIS.ADF.Web.UI.WebControls.SimpleTaskResult(heading, detail);

    Results = simpleresult;
```

Executing the Task

- Input parameter EventArg equals “executeTask”
- FloatingPanelTask.GetCallbackResult method calls ExecuteTask

DevTask.cs

```
GetCallbackResult()
 base.GetCallbackResult

ExecuteTask()
```

FloatingPanelTask.cs

```
public override string GetCallbackResult()
{
    . . .
    string eventArg = keyValColl["EventArg"];
    . . .
    else if (eventArg == "executeTask")
    {
        ExecuteTask();
        DisplayResults(taskJobID, Input, Results);
    }
    return CallbackResults.ToString();
}
```

Working with CallbackResults

- Upon task execution, callback response string is returned from custom task GetCallbackResult method
- CallbackResults are callback messages formatted to be processed by Web ADF JavaScript
- All Web ADF controls maintain a CallbackResults collection

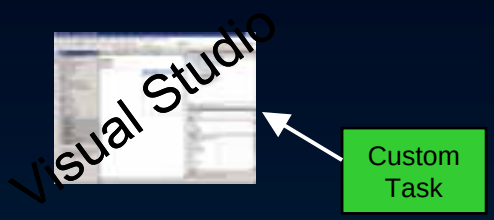
Push Web ADF control and custom callback results into task callback results

DevTask.cs

```
public override void ExecuteTask()
{
    map.CenterAt(MyPoint);
    CallbackResults.CopyFrom(map.CallbackResults);
}
```

Simple Task Demo

Visual Studio Integration



- Expose custom properties
- Extend TaskDesigner to manage smart tags
- Create a custom WinForm dialog for developer input

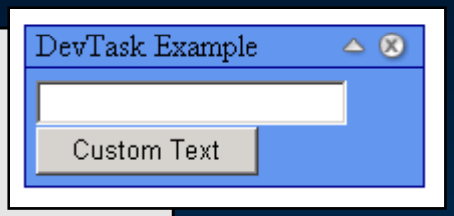
➢ Follow standard technique for integrating ASP.NET Web controls in Visual Studio

Define task properties as attributes

- **Set properties at design-time, change runtime behavior**

DevTask.cs

```
public string ButtonText
{
  get
  {
    object o = StateManager.GetProperty("buttonText");
    return (o == null) ? "Execute Task" : o as string;
  }
  set
  {
    StateManager.SetProperty("buttonText", value);
  }
}
```



MyPage.aspx (design-time)

```
<devTask:DevTask ID="DevTask1" runat="server" ButtonText="Custom Text">
</devTask:DevTask>
```

Extend TaskDesigner

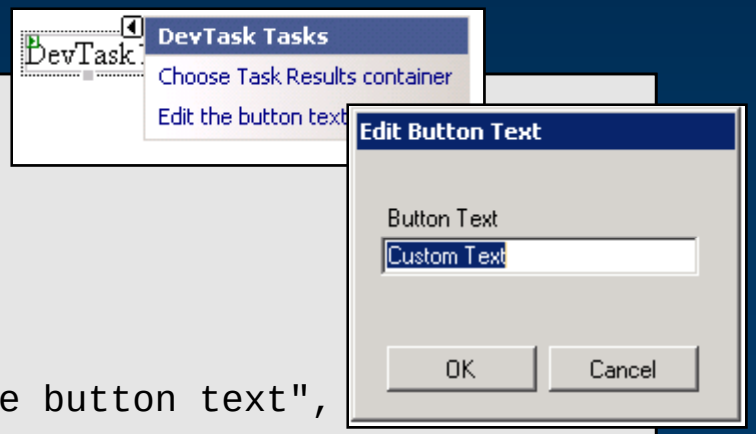
- **Access to visual interface (e.g. WinForm) for modifying task properties at design-time**

DevTask.cs

```
[System.ComponentModel.Designer(typeof(DevTaskDesigner))]
public class DevTask: ESRI.ArcGIS.ADF.Web.UI.WebControls.FloatingPanelTask
    {
```

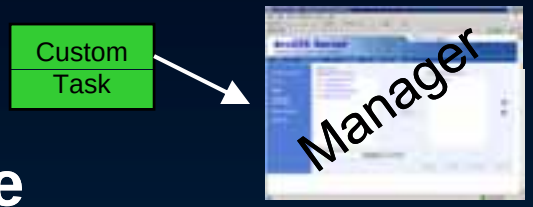
DevTaskDesigner.cs

```
 public class DevTaskDesigner : TaskDesigner
    {
        public DevTaskDesigner()
            : base()
        {
            verbs.Add(new DesignerVerb("Edit the button text",
                new EventHandler(OnEditButtonText)));
        }
```



Visual Studio Integration Demo

Manager Integration



- Create a custom Web control to be rendered at runtime in Manager
- Implement IWebConfigurator interface
  - Provides UI to configure task in Manager
  - Generates markup for configured task

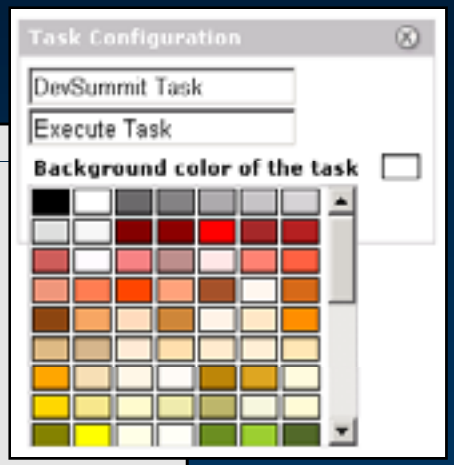
Design Web control to modify custom task

- To configure a task in Manager requires a custom Web control

DevTaskWebConfigurator.cs

```
public class DevTaskWebConfigurator:
  ESRI.ArcGIS.ADF.Web.UI.WebControls.CompositeControl
{
     private System.Web.UI.WebControls.Button okButton;

     protected override void CreateChildControls()
     {
         okButton = new System.Web.UI.WebControls.Button();
         okButton.Click += new EventHandler(okButton_Click);
```



IWebConfigurator implementation

- Properties
  - ControlToConfigure: returns an instance of the Task
  - AdditionalControls: returns other controls in same Page as the Task
  - ValidateResources: determine if a required resource is available
- Methods
  - GetGISResourceItemDependencies – reset task if dependent resource is removed
- Event Handlers
  - WebConfigurationCompleteEventHandler: write markup to page

Configure custom task properties

- **ControlToConfigure** returns an instance of the custom task
- Handle **OnWebConfigurationComplete** event

DevTaskWebConfigurator.cs

```
public class DevTaskWebConfigurator:
  ESRI.ArcGIS.ADF.Web.UI.WebControls.IWebConfigurator
{
    private DevTask TaskInstance = null;

    private void okButton_Click(object sender, EventArgs e)
    {
       TaskInstance.ButtonText = buttonText.Text;

       OnWebConfigurationComplete(new WebConfigurationCompleteEventArgs
         (TaskInstance, getDesignTimeTag()));
```

Resource validation in Manager

- Implement **ValidateResources** to determine if a valid resource is available

DevTaskWebConfigurator.cs

```
public bool ValidateResources(out string message)
{
    ArrayList mrms = new ArrayList();
    Utility.FindControls(typeof(MapResourceManager),
        AdditionalControls, ref mrms);
```

- Implement **GetGISResourceItemDependencies** to reset task when dependent resource is removed

DevTask.cs

```
public override List<GISResourceItemDependency> GetGISResourceItemDependencies()
{
   List<GISResourceItemDependency> list = new List<GISResourceItemDependency>();
   list.Add(new GISResourceItemDependency(typeof(MapResourceItem), mritem,
     typeof(MapResourceManager), mrmanager));
```

Generating Web Mapping Application output

- **Create output string of declarative content for the Web Mapping Application**

DevTaskWebConfigurator.cs

```
private string getDesignTimeTag()
{
  string openTag = string.Format("<devTask:DevTask ID=\"{0}\"
  runat=\"server\", TaskInstance.ID);

  StringBuilder trcTag = new StringBuilder();
  trcTag.Append("<TaskResultsContainers>");
  trcTag.Append("<esri:BuddyControl Name=\"TaskResults1\" />");
  trcTag.Append("</TaskResultsContainers>");

  string endTag = "</devTask:DevTask>";
  . . .
  return alltags.ToString();
```

Register with custom task and Manager

- **Add attribute to custom task**

DevTask.cs

```
[WebConfigurator(typeof(DevTaskWebConfigurator))]
public class DevTask: ESRI.ArcGIS.ADF.Web.UI.WebControls.FloatingPanelTask
{
```

- **Modify Tasks.xml**
  - **Located in <ArcGIS Instance>\Manager\App_Data**

Tasks.xml

```
<Task Name="DevTask" DisplayName="DevSummit Task"
Type="DevTaskProject.DevTask, DevTaskProject, Version=1.1.0.0,
Culture=neutral, PublicKeyToken=a284737434b9d17c"
TagPrefix="devTask" />
```

Manager Integration Demo

Extending Out of the box Tasks

- All out of the box Tasks are extensible
- Change input processing and result content
  - Graphics results rendering
  - Custom ContextMenus
  - Node click behavior
  - Custom callback results (e.g. zoom to results)
- Need to understand the key public methods and properties in a custom Task

Extending Demo

Best Practices

- Start with Task or FloatingPanelTask base classes
  - Already provides some implementation code
- Prepare for Callback and Partial Postback patterns
  - IsAsync
- Define a standard and unique tag prefix
  - Arbitrary tag prefix may conflict with other custom tasks
- Embed resources - images, JavaScript
- Sign and register (GAC) when integrating with Manager

What's New in 9.3

- Web Mapping Template using ASP.NET AJAX partial postback pattern (watch for IsCallback)
- Hover over results in TaskResults control, highlight feature in map
- Easier to declare Web Tasks which do not require Web configuration (e.g. "default markup")
- 9.3 final will include an interface\abstract class to create a User control Web task

Summary

- Web ADF Task Framework provides consistent architecture for processing user input and generating results
- Application requirements determine the extent to which the task framework is utilized
- Page developers – use the task framework to display results
- Control developers – use the task framework to encapsulate UI and execution logic in a distributable component
  - Add functionality to Manager generated applications

In Conclusion…

- All sessions are recorded and will be available on EDN
  - Slides and code will also be available
- Please fill out session surveys!
- Still have questions?
  1. Tech talk, Demo Theatres, Meet the Team
  2. "Ask a Developer" link on web page
     - www.esri.com/devsummit/techquestions