Web 2.0 / Ajax Development Primer

PC3A
John Rhodes
ADC Austin
Course Target

> Background
  - Overview of Websyedian WebClient for CA Plex
  - Technology Introduction
    - Introduction to Web Applications
    - Introduction to Web Servers
    - Introduction to Ajax

> WebClient Skills
  - Deploying a WebClient web application
  - Applying Templates to customize the web application
Module: Introduction to WebClient
WebClient in a Nutshell
WebClient Technologies

> WebClient simplifies the Web Application development process

> Allows a Plex Developer to easily create Web Applications

> WebClient is built on many technologies

> Need a high-level knowledge of these technologies to understand how it works
WebClient Technologies

> CA Plex
  ▪ The prime skill necessary to build web applications

> Development Environment
  ▪ Java
  ▪ Eclipse

> Web Environment
  ▪ Web Applications
  ▪ Web Servers
WebClient Technologies – CA Plex

- Model-based development tool
- No native web application support
- WebClient template builder takes GUI panel design and produces a WYSIWYG template
- WebClient runtime adds web browser interaction to Plex GUI screens
- WebClient can enhance user experience beyond standard CA Plex controls
WebClient Technologies – Java

> Multi-platform programming language
  - Intel, Power, IBM i/z/p/n, Linux, ...

> A commonly-used language for Web Applications via Java Servlet API

> WebClient requires CA Plex Client functions to be generated as Java

> CA Plex Server functions can still be deployed to other platforms without modification
WebClient Technologies - Eclipse

> Java IDE – Integrated Development Environment - Full development life-cycle from one tool

> Open Source, primarily written in Java.
  - Foundation for Rational Suite

> Suited to, but not restricted to, Java development

> Expandable via plug-ins
  - WebClient plugin requires Eclipse, but Eclipse can be minimal part of development
WebClient Technologies – Web Apps

- Run within a browser
- Web Server dynamically updates page content
- HTML/JavaScript to render pages
- AJAX provides interactive client controls
- WebClient provides GUI Client/Server experience as a Web Application
- One deployment, no local installation, available anywhere
HTML

> **HyperText Markup Language**
> Basis for most web pages
> Describes format of web page with defined syntax
> Rendered by web browser
> Can embed images and other objects
> Can embed scripting languages to enhance functionality
> Includes elements such as Frames, Labels or Tables
Cascading Style Sheets (CSS)

> Describes presentation of a web page
> Separates presentation from page contents
> Influences color, layout, font etc. of page elements
> Rules for precedence of element styles, hence ‘cascading’
JavaScript

> Client side scripting language
> Embedded in HTML
> Executed by web browser
> Provides enhanced functionality and user experience over static HTML
> Typically interacts with Document Object Model (DOM)
> Unrelated to Java language
Document Object Model (DOM)

> Conventions for representing and interacting with elements in a markup language such as HTML

> Platform and language independent

> Referenced by Javascript and AJAX to allow programmatic control of page objects, e.g. validating an input field before submitting to the web server
Java Platform Enterprise Edition

> Java EE, previously known as J2EE

> Services, APIs and protocols to allow web-based multi-tiered applications

> Eclipse has Java EE perspective to provide tools required for web application development

> Used by WebClient

©2009 ADC Austin
Rich Internet Application (RIA)

- Richer and more responsive than classic web applications
- Use desktop-style GUI controls
- Client-side interactions
- Asynchronous communication to the server
- Best of both client/server and web applications
- WebClient provides RIA
Ajax

> Asynchronous JavaScript and XML

> Collection of techniques to provide interactive web applications (RIA)

> Requests and retrieves information from server without needing to reload whole page, e.g. Google Maps

> Used by WebClient via Dojo toolkit
Ajax
Dojo JavaScript Toolkit

> Rich, Open Source
> Commercial support – IBM, SUN, BEA, ...
> Standards based
> Internationalization (i18n)
> Localization (110n)
> Accessibility (a11y)
> Many components support Ajax
Dojo Toolkit Dijits

> Library of JavaScript/Ajax tools to provide rich interactive web applications

> Dijits (Dojo Widgets) are prepackaged collections of HTML, JavaScript and CSS

> Wide collection of Dijits, e.g. date picker, dynamic charts

> Abstract layer around Ajax implementation to ease cross-browser compatibility

> WebClient templates based on Dijits to allow CA Plex panel controls to be implemented in a browser

> WebClient can provide templates for additional Dijits
JSON

> JavaScript Object Notation

> Lightweight data-interchange format

> Preferred to XML for Ajax applications

> Webclient uses JSON to pass and retrieve data to Dojo functions

> Stored in text format, e.g.

```json
{  "firstName": "John",  "lastName": "Smith",  "address": {   "streetAddress": "21 2nd Street",   "city": "New York",   "state": "NY",   "postalCode": 10021 },  "phoneNumbers": [ "212 555-1234", "646 555-4567" ] }
```
Java Servlet

> Java-written objects to process requests and construct responses

> Serves dynamic content to Java-based Web Servers

> Work with *Servlet Containers*, specialized web servers that support servlet execution

> WebClient uses a servlet to connect the CA Plex runtime environment to the Java EE server
WebClient Technologies - Web Servers

- Processes HTTP requests
- Serves web pages and data responses to user
- Security
- WebClient requires Java Servlet support
- Examples: Apache Tomcat, Websphere, Jboss, Jetty
Web Servers

> Software to handle HTTP/HTTPS requests and responses

> Varying levels of functionality
  - security
  - load-balancing

> Examples; Apache HTTP and IIS

> Work with servlet containers to provide dynamic pages
Servlet Containers

> Specialized web servers that support servlet execution
> Provide dynamic content
> WebClient must be deployed to a servlet container
> Examples; Tomcat, Jetty
Apache Tomcat

> Developed by Apache Foundation
> Servlet container
> Not to be confused with Apache HTTP
> Written in Java
> Widely used in servlet market
> Configuration and management available through browser interface
Jetty

> Developed by Mort Bay Consulting
> Servlet Container
> Written in Java
> Small footprint
> Shipped with WebClient Live Install
WAR Files

> Web application Archive

> JAR file used to package Web Applications

> Contains servlets, Java classes, configuration and resources

> Deploys to ‘webapps’ folder on Web Server
Exercise 1 – Build The Application in Java
Module: WebClient Overview
WebClient

- Enables the creation of Rich Internet Applications from CA Plex
- Requires only CA Plex skill set
- Use existing CA Patterns and models
- Deploy on the web with a few clicks
- Uses inheritable and customizable templates
- Can enhance interface with advanced web controls
How does WWCP work?

- Generates WYSIWYG HTML with JavaScript/Ajax directly from the Plex panel design
  - 2 modes – basic and expert
  - With expert, html templating techniques are in play

- Java based - Deploys on any J2EE web server such as IBM Websphere, JBoss or Tomcat. Integrated to Eclipse Europa / Rational

- Uses client-server action diagram techniques
  - No new statements or APIs, use what you know
  - Single code base for web and client-server / 5250
Development Environment

Java Eclipse/WDSc
Compile

WebClient Eclipse
Plug-in (Builder)

Java Source Files

HTML CSS / Templates / Panels
System Templates
User Templates

WebSphere software

©2009 ADC Austin
Template Generation Process

- Plex – Gen Function with Panel
- WC Builder – Examine Panel Inheritance
- WC Builder – Examine Panel Components
- WC Builder: Write HTML template
- WC Builder: Bind components
- Sys Template
- User Template

©2009 ADC Austin
Component Generation from Panel

```java
<!JSInitionsce>
  dojo.require("dijit.form.DateTextBox");
  <!JSInit>

  <span style="(!DefaultCSS:no:withPos="/!!Param:default
  input dojoType="dijit.form.DateTextBox" widgetId="/!
  style="(!DefaultCSS:no" name="/(!NameID)"
  value="/(!This)" />
</span>

<!ContextMenu>
```
Deployment Environment

> WebClient is a J2EE Proxy Servlet (Java)

> Uses the CA Plex Java runtime to translate panels into HTML pages

> Runs on multi-tier application server environment

> Presentation layer runs under J2EE application servers (Tomcat, JBoss, Websphere)
Development Environment

Build WebClient functions using Rational/Eclipse or direct Java build

Build .NET, RPG, or Java server functions Using standard build techniques

Enhance look and feel by CSS style sheets and html templates
Development Environment

> Business layer processes

- No change in development practices
- Generate C#, Java, or RPG the same as with client-server applications

> Presentation layer processes

- Standard CA Plex functions generate in Java
- Automated ANT scripts and Eclipse/WDSc integration provided to compile/build functions and produce HTML templates
Key Components for Development

> WebClient pattern library (WEBCLIEN)

- Group model shipped with WebClient
- Added as a library to the application group model
- Functions should inherit from patterns in this library
- Does not change business logic
- Does not change appearance of panel in client-server mode
- Contain HTML and JavaScript source code objects and control name settings
- CA Plex action diagrams are developed in exactly the same manner as client-server applications
> WebClient HTML builder

- WebClient resolves HTML templates from the panel control type and generates the HTML
- Implemented as an Eclipse plug-in called a “builder”
- Driven by settings in the panel properties and the inheritance path of the WEBCLIENT patterns employed
WebClient Group Model

> Licensed group model
> Provides basic patterns and source code for WebClient
> Does not alter functionality of existing patterns
Patterns of the WEBCLIENT Library

- WebUI
- WebShell
- Frame Property
- TabStrip
WEBCLIENT/~/WebShell

Action Diagram: [(Read-Only)] WebShell

Function: ~WebShell

Pre Point: Start initialize
- Call: LIBASIC/Meta Options

Post Point: Start initialize
- WebPlex Initialization
  - Inform the runtime of our inheritance path
- If Field: FIELDS/JAVA
  - For Each Inheritance Path Object
    - If Field: FIELDS/JAVA
      - For Each Property Target
        - FNC impl name NME
        - Define Field: FIELDS/+Name
        - Set Value To Current Field: FIELDS/+Name
        - Name Defined Field: FIELDS/+Name, Environment</Object>
      - API Call Source code: WEBCLIENT/JAVASetInhPath
      - Undefine Field: FIELDS/+Name

Panel: [(Read-Only)] WebShell

Model Editor - Function: WEBCLIENT/~/WebShell

<table>
<thead>
<tr>
<th>Function</th>
<th>is a FNC</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEBCLIENT/~/WebShell</td>
<td></td>
</tr>
</tbody>
</table>

Function
- calls: UIBASIC/Meta.Options
- impl name: WEBCLIENT/WebShell

©2009 ADC Austin
WEBCLIE~/~WebShell

> Highest level template in WebClient inheritance

> All WebClient enabled functions must inherit from this function

> Corresponds to the root template WebShell-root.html
Exercise 2 – Build The Application in WebClient
Module: WebClient Template Builder
Templating Concepts

> WebClient itself knows nothing about HTML, CSS, or JavaScript

> All aspects of the generated HTML are controlled by template files in your workspace
Root Templates

> Identified with -root suffix,

 e.g. WebShell-root.wcli

> First template to be expanded during template generation

> Panel can only have one root

> Contains ‘attach points’ to include later templates, e.g.

  /(!AttachPoint:name=MainArea)
Page Templates

> Identified by -page suffix
  e.g. WebMessages-page.wcli

> Applied by inheriting from function with same implementation name

> Can have multiple Page templates applied

> Adds contents to defined attach points in root template

/(!MainArea)

Anything here will be generated in the MainArea

/(!MainArea)
Attach Points

> Placeholders for text to be inserted from other templates
> Identified by Name
> Similar to Edit Points

<table>
<thead>
<tr>
<th>Root</th>
<th>Page1</th>
<th>Page2</th>
<th>Result</th>
</tr>
</thead>
</table>
| This is before
/(!AttachPoint:name=Demo)
This is after | /(!Demo)
This is page1
/(!Demo) | /(!Demo)
This is page2
/(!Demo) | This is before
This is page1
This is page2
This is after |

©2009 ADC Austin
Default Attach Points

> CSSInit : Stylesheet Definition
> HeadArea
> LoadScreen
> LinkArea : ‘Breadcrumb’ history
> DataArea
> JSInit : JavaScript initialization
> JSOnLoad : Page load area
> FormArea
> MainArea : Panel Controls
Evaluating Templates

> Uses implementation names to identify root and page templates

> Template builder checks implementation names of all functions in inheritance path

> Determine root template. Nearest inherited ancestor with a -root.wcli template, e.g. ~DetailPopup
Evaluating Templates

> Once the root template has been resolved, locate any ancestors with an implementation name matching a -page suffix, e.g. WebMessages-page.wcli

> Insert any attach point code at the appropriate location
Template Resolution Process

Plex – Gen Function with Panel

WC Builder – Examine Panel Inheritance

WC Builder – Examine Panel Components

WC Builder
Write html template

WC Builder
Bind components

Sys Template

User Template

©2009 ADC Austin
Template Resolution Process

1. Determine Root templates
2. Determine and apply Page templates
3. Create yourFunction-panelgen.wcli by:
   - process each control in panelresource
   - evaluate and include text for appropriate .ctrl template
4. Merge Root, Pages and yourFunction-panelgen.wcli to produce yourFunction.wcli
5. Save output template under ‘WebGenTemplates’ source folder

©2009 ADC Austin
Template Resolution
Enable Template Builder

> WebClient Eclipse plug-in provides functionality

> Select Java project to be included in WebClient, right-click and select ‘Enable Websydian Web Client for Plex’

> Adds ‘Websydian WebClient Project Builder’ (see under Properties→Builders)

> Creates ‘WebGenTemplates’ source folder for generated templates
Module: Basic Template Customization
Template Overview

> Basic Level

- Adding page headers and footers
- Stylesheets
- Enhancing control templates with minor enhancements
- Control property
  - Template, hint, stretch
How to Customize Control Templates

> Control name needs to be specified

> Control name format for all control types except grid column:

```
controlname:attachpoint:parameters
```

```
StartDate:MainArea:template=dojoDateEdit:default
```

Example Control name format for grid:

```
Grid1P:GridArea:stretch=Venue
```

> Control name format for grid column:

```
column heading: heading: attachpoint:parameters
```
## Control Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Template=(filename)</td>
<td>Use the specified template instead of the default (WebXXX). Do not include .html at the end of the filename</td>
</tr>
<tr>
<td>Seq(#)</td>
<td>Insert the component into the attachment point sequence at #</td>
</tr>
<tr>
<td>default</td>
<td>This parameter is automatically set if no attach point is specified</td>
</tr>
</tbody>
</table>
# Extra Parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Return the first value of a pair (x coordinate)</td>
</tr>
<tr>
<td>y</td>
<td>Return the second value of a pair (y coordinate)</td>
</tr>
<tr>
<td>htmlaccel</td>
<td>If the value contains a &amp; to indicate an underlined accelerator, remove the &amp; and put HTML underlining instead</td>
</tr>
<tr>
<td>nbspifempty</td>
<td>If the value is empty, return &amp;nbsp instead</td>
</tr>
<tr>
<td>html</td>
<td>The value is intended to be processed as HTML; if this is not specified, it is HTML-escaped</td>
</tr>
<tr>
<td>tight</td>
<td>Do not emit newlines or comments within the value</td>
</tr>
</tbody>
</table>
Exercise 3/4 – Customize the HTML
> http://webclientiplus.com

- Blog
- Wiki
- Support
- Software