Introducing Apache Pivot

Greg Brown
7/12/2010
Bio

- Greg Brown
  - Senior Software Architect
  - 15 years experience developing client and server applications in both services and R&D
  - Apache Pivot Project Founder
What is Apache Pivot?

• Open-source platform for building rich internet applications in Java (or any JVM scripting language: Groovy, JavaScript, Scala, etc.)

• Similar to Adobe Flex or Microsoft Silverlight, but based on the JVM rather than Flash or Silverlight player

• Pivot applications can be run as an applet or as stand-alone desktop application (installed or launched via Web Start)
What is Apache Pivot?

- Like other RIA platforms, includes features that make building modern GUI applications much easier:
  - Declarative XML-based UI markup language ("WTKX")
  - Themes (aka "skins")/styling
  - Data binding
  - Effects and transitions (animations)
  - Web services integration (JSON/REST)
Why RIA?

• Functional requirements for many web applications have begun to scale beyond the capabilities of the browser

• Difficult to create a user experience in HTML that is truly on par with that of a desktop application
Why RIA?

• RIA platforms bridge the gap between the web and desktop experiences

• Allow developers to build applications that look and feel more like native desktop applications but are deployable via the web

• Often incorporate visual effects intended to enhance the overall user experience (animations and other dynamic behaviors)
Why Pivot?

1. Provide a viable option for developers who want to build rich Internet applications in Java
   - Flex: ActionScript
   - Silverlight: C#/JavaScript
   - JavaFX: JavaFX Script
Why Pivot?

2. Provide a truly open alternative for RIA developers

- Flex, Silverlight, and JavaFX are all proprietary platforms
- Pivot is completely open source and driven entirely by the software development community
Platform Overview

- Pivot classes are grouped into the following libraries:
  - *pivot-core-1.5.jar* - common, non-UI utility classes (collections, event processing, localization, threading, I/O, etc.)
  - *pivot-web-1.5.jar/pivot-web-server-1.5.jar* - REST client/server APIs
  - *pivot-wtk-1.5.jar/pivot-wtk-terra-1.5.jar* - WTK/Terra L&F
  - *pivot-charts-1.5.jar* - charting components (requires charting provider; currently based on JFreeChart)
Platform Overview

WTK Class Hierarchy
“Kitchen Sink” Demo

“Kitchen Sink” Demo Application
“Hello WTKX!”

Source code for “Hello World” in Pivot
Pivot Compared to Swing

- Swing can also be used to build RIAs
- Both Pivot and Swing use Java2D under the hood
- Pivot offers numerous advantages that make it a more compelling, modern alternative
Pivot Compared to Swing

- Pivot advantages:
  - Provides XML markup language for simplifying user interface construction
  - Built-in support for JSON and REST-based data services
  - Built-in data binding support
  - Platform-level support for visual effects and transitions
  - Takes advantage of newer Java language features: generics, enums, for..each loops, varargs, and annotations
Pivot Compared to JavaFX

- Pivot allows developers to build applications in Java, vs. JavaFX scripting language
- Slightly different emphasis: “Application” vs. “Rich” (media delivery) in “RIA”
- Not mutually exclusive!
Pivot Compared to GWT

- GWT also allows developers to use Java to write web-based applications
- Runtime environment for a GWT application is the browser, not a JVM:
  - Code executes as interpreted JavaScript, not bytecode
  - Doesn’t support full Java API (no I/O, networking, threading, reflection, XML, etc.) - basically, only Java language
  - Presentation performed via CSS and DOM manipulation rather than 2D drawing API
“Stock Tracker” Tutorial Application

- Simple but practical sample application
- Highlights key platform features and development best practices
Stock Tracker

Key Features

- UI markup using WTKX
- Event handling
- Web queries
- Data binding
- Localization
UI markup using WTKX

- Pivot UI often defined in WTKX
- Hierarchical structure of XML parallels the component hierarchy, makes it easy to visualize the resulting output
- Developers are familiar with markup metaphor
- Can still be defined in code - WTKX is just a “shortcut”
- Not compiled - serialized representation of object graph
  - Generally loaded at runtime from application JARs
  - Can load dynamically (from server, for example)
UI markup using WTKX

```xml
<stocktracker:StockTrackerWindow title="%stockTracker" maximized="true"
 xmlns:wtkx="http://pivot.apache.org/wtkx"
 xmlns:content="org.apache.pivot.wtk.content"
 xmlns:stocktracker="org.apache.pivot.tutorials.stocktracker"
 xmlns="org.apache.pivot.wtk">
  <content>
    <TablePane styles="{padding:8, horizontalSpacing:6, verticalSpacing:6}">
      <columns>
        <TablePane.Column width="1*" />
      </columns>
      <rows>
        ...
        <TablePane.Row height="1*">
          <SplitPane splitRatio="0.4">
            <left>
              ...
            </left>
            <right>
              <Border styles="{padding:6, color:10}">
                <content>
                  <wtkx:include wtkx:id="detailPane" src="detailPane.wtkx"/>
                </content>
              </Border>
            </right>
          </SplitPane>
        </TablePane.Row>
        <TablePane.Row height="-1">
          <BoxPane styles="{horizontalAlignment:'left', verticalAlignment:'center'}">
            <Label text="%symbol" styles="{font: [bold:true]}" />
            <TextInput wtkx:id="symbolTextInput" textSize="10"
              maximumLength="8" />
          </BoxPane>
        </TablePane.Row>
      </rows>
    </TablePane>
  </content>
</stocktracker:StockTrackerWindow>
```
UI markup using WTKX

- Quick WTKX primer:
  - Elements
    - Uppercase = class instance
    - Lowercase = property/listener list
  - Attributes = property/event listener
  - Namespaces = Java packages
  - “wtkx” prefix (IDs, includes, etc.)
  - Script code (logic)
UI markup using WTKX

- Resolution operators:
  - Used in WTKX attribute values
  - ‘%’ = resource resolution (localization)
  - ‘@’ = location resolution (relative URL)
  - ‘$’ = variable resolution
UI markup using WTKX

- WTKX binding:
  - Maps objects defined in WTKX to Java member variables ("dependency injection")
  - wtkx:id maps to @WTKX annotation
UI markup using WTKX

• Implementing Bindable interface ensures that bindings are processed

• Resources argument allows bound instance to retain reference to the resource bundle used to process the WTKX file
Event Handling

• WTKX = structure, code = behavior

• Logic generally executed in response to an “event” (button pressed, selection changed, etc.)

• Event listeners often wired up in Bindable#initialize()

• Can also be registered in inline script, similar to HTML
Event Handling

• Actions:
  • Extend abstract org.apache.pivot.wtk.Action class
  • Defines abstract perform() method
  • Used to attach application behaviors to multiple UI elements (e.g. toolbar button, menu item, etc.)
  • Can be enabled/disabled; attached components reflect state
Web Queries

- Pivot’s native means of server communication
- Part of “Web” class library
- Similar to XMLHttpRequest in web browser
- Facilitate communication with and implementation of REST services
- Use JSON by default, but can use any data format (XML, CSV, Java serialization, etc.)
Web Queries

- Quote data returned by HTTP GET request to http://download.finance.yahoo.com/d/quotes.csv/
- Query string arguments specify symbols and fields to retrieve, returns CSV file:

```
"AAPL", "APPLE INC", 171.06, 169.59, 172.17, 166.00, +2.88, 12995603
"AMZN", "AMAZON.COM INC", 72.54, 72.35, 73.83, 70.52, +1.10, 2748930
"EBAY", "EBAY INC", 27.09, 27.35, 27.44, 27.04, -0.02, 3426369
```
Web Queries

- Stock Tracker uses an instance of `org.apache.pivot.web.GetQuery` to retrieve the data
- POST, PUT, and DELETE also supported
- Uses an instance of `org.apache.pivot.serialization.CSVSerializer` to deserialize the data
- Returns the quotes as an instance of `org.apache.pivot.collections.List` which is used as the model data for the table view
Web Queries

- By default, CSVSerializer returns an ArrayList of HashMaps
- Untyped - all data are strings
- Can be configured to return instances of any Java Bean type
- Stock Tracker uses a StockQuote bean class to convert strings to numbers (for sorting)
Web Queries


- Abstract (generic) base class for executing background operations

- Defines a single abstract execute() method that returns the result of the operation

- GetQuery returns Object (in this case, the result data)
Web Queries

- `execute()` is synchronous - blocks UI
- Task provides an overload that takes an instance of `org.apache.pivot.util.concurrent.TaskListener`
- Caller is notified asynchronously via callback when task has succeeded or failed
- UI remains responsive
Data Binding

- Maps values between a set of user interface elements and a data structure, called the “bind context”
- Eliminates tedious boilerplate code for manually populating field data
Data Binding

- Uses a load/store model:
  - load() populates UI with values from context
  - store() populates context with values from UI
- Maps well to REST-based applications:
  - GET - load()
  - POST/PUT - store()
Data Binding

• Bind context is either an instance of org.apache.pivot.collections.Dictionary or a Java Bean that can be wrapped in org.apache.beans.BeanAdapter (which implements Dictionary)

• Easy to bind to JSON data returned by web query - JSON Objects are returned as instances of HashMap, which implements Dictionary
Data Binding

- Stock Tracker uses binding to populate quote detail form:
Data Binding

- "textKey" property associates Label text with bind key

- Bind context is an instance of the StockQuote bean returned by GetQuery/CSVSerializer

- Uses "bind mapping" to transform data during binding:

```java
public class ValueMapping implements Label.TextBindMapping {
    private static final DecimalFormat FORMAT = new DecimalFormat("$0.00");

    @Override
    public String toString(Object value) {
        return Float.isNaN((Float)value) ? null : FORMAT.format(value);
    }

    @Override
    public Object valueOf(String text) {
        throw new UnsupportedOperationExcep()}
Localization

• Translatable text and other resources stored in “resource bundles”

• In Pivot, resource bundles are JSON files rather than .properties files

• Use UTF-8 natively, vs. ISO-8859

• May be hierarchical, vs. flat
Localization

- Stock Tracker resource bundles (default and ‘fr’):

```
01 {  
02    stockTracker: "Pivot Stock Tracker",
03    symbol: "Symbol",
04    companyName: "Company",
05    value: "Value",
06    openingValue: "Open",
07    highValue: "High",
08    lowValue: "Low",
09    change: "Change",
10    volume: "Volume",
11    addSymbol: "Add symbol",
12    removeSymbol: "Remove selected symbols",
13    lastUpdate: "Last Update",
14    dataProvidedBy: "Data provided by",
15    yahooFinance: "Yahoo! Finance"
}
```

```
01 {  
02    stockTracker: "La Bourse Pivot",
03    symbol: "Code",
04    companyName: "Société",
05    value: "Cours",
06    openingValue: "Ouverture",
07    highValue: "+ Haut",
08    lowValue: "+ Bas",
09    change: "Variation",
10    volume: "Volume",
11    addSymbol: "Ajouter un code",
12    removeSymbol: "Enlever codes sélectionnés",
13    lastUpdate: "Dernier échange",
14    dataProvidedBy: "Données fournies par",
15    yahooFinance: "Yahoo! Finance"
}
```

```
Localization

- Quote detail form uses localized form labels:
Localization

• Et voilà!

<table>
<thead>
<tr>
<th>Code</th>
<th>Cours</th>
<th>Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAPL</td>
<td>$259,85</td>
<td>-3,27</td>
</tr>
<tr>
<td>AMZN</td>
<td>$126,71</td>
<td>-2,05</td>
</tr>
<tr>
<td>EBAY</td>
<td>$22,57</td>
<td>+0,39</td>
</tr>
<tr>
<td>GOOG</td>
<td>$505,70</td>
<td>+0,10</td>
</tr>
<tr>
<td>IBM</td>
<td>$126,56</td>
<td>-1,40</td>
</tr>
<tr>
<td>MSFT</td>
<td>$26,40</td>
<td>-0,46</td>
</tr>
<tr>
<td>ORCL</td>
<td>$22,73</td>
<td>-0,11</td>
</tr>
</tbody>
</table>

La Bourse Pivot

Code | Cours | Variation |
-----|-------|-----------|
AAPL | $259,85 | -3,27     |
AMZN | $126,71 | -2,05     |
EBAY | $22,57  | +0,39     |
GOOG | $505,70 | +0,10     |
IBM  | $126,56 | -1,40     |
MSFT | $26,40  | -0,46     |
ORCL | $22,73  | -0,11     |

Apple Inc.

<table>
<thead>
<tr>
<th>Code</th>
<th>Cours</th>
<th>Variation</th>
<th>Ouverture</th>
<th>Haut</th>
<th>Bas</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$259,85</td>
<td>-$3,27</td>
<td>$258,47</td>
<td>$261,90</td>
<td>+257,10</td>
<td>10 813 601</td>
</tr>
</tbody>
</table>

Dernier échange 4 juin 2010 12:02:20
Données fournies par Yahoo! Finance
Summary

• Pivot is a platform for building modern GUI applications in Java that can be deployed via the web or to the desktop

• Stock Tracker tutorial demonstrates some key features and is a great quick-start example
Further Information

- http://pivot.apache.org
- http://pivot.apache.org/demos/
- http://pivot.apache.org/tutorials/
- http://pivot.apache.org/1.5/docs/api/
Q & A