

Apache Wicket

Niels Sthen Hansen
Trifork Projects Copenhagen A/S
nsh@trifork.com

Apache Wicket

- The Wicket approach
- Brief tour of key Wicket concepts
- Lots of code examples!
 - Hello World'ish snippets
 - Samples from a real-life Wicket app
- Thoughts on modelling & design, productivity and maintainability

About Us

- Trifork A/S
 - Finance
 - Public
 - Academy
 - 110+ employees in Århus, Copenhagen & Zürich
- Niels Sthen Hansen
 - Java technology since 2002
 - Trifork Projects Copenhagen A/S

Today's Real-life Wicket App

- The Lessor Portal
 - Allows employees to register payroll-related data on-line and to access personal information
 - Integrates to 5 (and counting) payroll-systems
 - Denmark, Norway and Germany
- Lessor A/S
 - Payroll and HR systems
 - Large and growing market share

Example

- The Lessor Portal
 - Absence registration approval

Web Applications

- More mission critical than ever
 - Time-to-market is crucial
 - ... and so is maintainability
- Users have come to expect more from wepapps
 - More functionality
 - More responsive (less page reloads)
 - Why can't all webapps be like, say, gmail ?

Web Application Frameworks

Echo Cocoon Millstone OXF Struts SOFIA Tapestry WebWork
RIFE Spring MVCCanyamo Maverick JPublish JATO Folium
Jucas Verge Niggle Bishop Barracuda Action Framework Shocks
TeaServlet wingS Espresso Bento jStatemachine jZonic OpenEmcee
Turbine Scope Warfare JWAA Jaffa Jacquard Macaw Smile
MyFaces Chiba Jbanana Jeenius JWarp Genie Melati Dovetail
Cameleon JFormular Xoplon
Japple Helma Dinamica WebOnSwing Nacho Cassandra Baritus
Stripes Click GWT ...

Wicket Design Goals

- Easy (simple / consistent / obvious)
- Reusable
- Non-intrusive
- Safe
- Efficient / scalable
- Complete

Goal: Simple/Consistent/Obvious

- POJO-centric
- All code written in Java ala Swing
- Avoid overuse of XML configuration files
- Fully solve back button problem
- Easy to create bookmarkable pages
- Maximum type safety, compile-time diagnosis
- Maximum diagnosis of run-time problem

Goal: Non-intrusive

- HTML or other markup not polluted with programming semantics
- Only one simple tagging construct in markup
- Compatible with any ordinary HTML editor
- Easy for graphics designers to recognize and avoid framework tagging
- Easy to add tagging back to HTML if designers accidentally remove it

Goal: Safe, Efficient/Scalable

- Only explicitly bookmarkable links can expose state in the page or URL
- All logic in Java with maximum type safety
- Easy to integrate with Java security
- Efficient & lightweight, but not at the expense of other goals
- Clustering through sticky sessions preferred
- Clustering via session replication is easy to accomplish and easy to tune by working with detachable models

Example: Hello Wicket World

```
mvn archetype:create -DarchetypeGroupId=org.apache.wicket  
-DarchetypeArtifactId=wicket-archetype-quickstart -DarchetypeVersion=1.4.8  
-DgroupId=com.trifork -DartifactId=hello
```

```
import org.apache.wicket.markup.html.WebPage;  
import org.apache.wicket.markup.html.basic.Label;  
  
public class HelloWorld extends WebPage  
{  
    public HelloWorld()  
    {  
        add(new Label("message", "Hello World!"));  
    }  
}
```

```
<html>  
<body>  
    <span wicket:id="message">Message goes here</span>  
</body>  
</html>
```

Wicket Application & Life-cycle

- Application
 - Defines (surprise) application-wide settings
- Wicket Session
 - Are created on first request and exists for the duration of the HTTP session
 - Can be subclassed (Example: PortalSession)
- RequestCycle
 - Handles request, holds information on HTTP request

Wicket Components

```
public abstract class Component implements ...
{
    ...

    protected final void renderComponentTag(ComponentTag tag) {
        ...
    }

    protected final void replaceComponentTagBody(
        final MarkupStream markupStream,
        final ComponentTag tag, final CharSequence body) {
        ...
    }

    ...
}
```

Wicket Components

- Are Java classes
- Extends other components and can be extended themselves
- Defines HTML mark-up
 - In file `ComponentClassName.html`
 - ... or inherit from superclass
 - ... or both!

Wicket MarkupContainers

- A class hierarchy of Components
- Can contain other components
- Also other containers
- When rendered, all child-components are also rendered
- Example:
 - Panels

EJB & Spring Integration

- @SpringBean
- @EJB
- Example:
 - Spring

Localization

- StringResourceModel
 - Looks up in resource files on-demand
- It is possible to override resource lookup
 - Example: Lessor Portal Localization

Some HTML Tags & Attributes

- Attribute wicket:id
- Attribute wicket:message="attr:resource_id"
 - `<input type="submit" wicket:message="value:page.search"/>`
- `<wicket:link>`
- `<wicket:panel>`
- `<wicket:extend>` and `<wicket:child>`
- `<wicket:remove>`
- `<wicket:enclosure>`

RepeatingViews and ListView

- Names are pretty self-explanatory
 - For table rows, bullet lists etc. etc. ...
- ListView when a list is available
- RepeatingView just needs a loop
- Examples: Calendar

Wicket Pages

- Pages are Components
- Pages are also MarkupContainers
- Wicket keeps pages in a PageMap
 - To handle that annoying back-button
 - Watch your session size!
- Certain default pages can be specified
 - Home page
 - Error page
 - Page to show when session is expired

Behaviors

- Implements IBehavior
- Are added to Components
- Modifies the Components .. well, behavior
 - Add or modify attributes on Component tag
 - Require elements in HTML header
 - Ajax events
- Example:
 - Tooltip header contributor

Wicket models

- Implements `IModel<T>`
- Encapsulates data
- The value (model object) can be
 - A constant
 - Calculated on request
 - Fetched for rendering and then discarded to minimize session size (`LoadableDetachableModel`)
- Components have a default model
 - And may define more

Wicket Models

- Example
 - LoadableDetachableModel

Wicket and Ajax

- Ajax Behaviors can be added to components
- The `AjaxRequestTarget`
 - Produces an Ajax response envelope
- When Components are added to the `AjaxRequestTarget` with `.addComponent()`
 - The Component will be rendered
 - The resulting markup will be included in the ajax response
 - .. and inserted in DOM, replacing existing mark-up

Examples

- A simple AjaxEventBehavior
- Portal
 - CRUDForm.formSaved – refresh grid
 - Absence approval revisited – advanced ajax

Working with Wicket

- Nice things
 - Navigating and understanding model and behavior made easy by type safety
 - Subclassing facilitates a consistent UI – maps well to the users perception of the application
 - Simple things are easy, difficult things are possible
- However, all this also means ...
 - ... that you should be just as careful in designing your front-end as you would with your application back-end

Working with Wicket

- The Lessor Portal
 - Project started from scratch September 2008
 - Appr. 2 developers at any given time
 - In production May 2009 (pilot-scale)
- One year after
 - Entire HTML layout changed August 2009
 - Still a clear vision of design
 - Lots of refactoring

Thank you!

- Questions & Discussion
- Niels Sthen Hansen, Trifork
 - nsh@trifork.com
- Product placement
 - Trifork Wicket courses!
 - Remember JA00 (<http://jaoo.dk>)