Project Dynamic Faces™ World Class AJAX for JavaServer™ Faces Technology

- Ed Burns
- Senior Staff Engineer
- Enterprise Java Platforms

Agenda

- Why JavaServer Faces?
- Why AJAX?
- Why JSF and AJAX?
- DynaFaces Introduction
  - > What is it?
  - > History and Current Status
- DynaFaces Details
  - > Entry Points
  - > Managing Complexity
- Demonstrations

Why JavaServer Faces?

You probably know why already, but just in case you don't...

- Web apps have more reach than non web apps
- JSF makes it easy to build Java Web apps
  - > Markup based UI construction
    - > Allows Browser Layout (CSS, HTML)
    - > Several view description languages to choose from: JSP, Facelets (XHTML or HTML)
  - > Hassle free data integration and transactionality
  - > Large selection of off the shelf components
  - > Use with or without tools

Why AJAX?

- Web apps may have more reach than rich client apps, but traditionally have suffered in the richness department.
- AJAX brings more richness without sacrificing very much reach.
- It's the hot buzzword now, so it has to be good, right?

Why JSF and AJAX?

- OO Design of JSF was ready for AJAX when AJAX wasn't cool.
- Key Features of JSF that make it AJAX friendly
  - > Flexible and extensible component model
  - > Well defined Request Processing Lifecycle
  - > Flexible and extensible rendering model
- Concepts that enable AJAX
  - > Encapsulation: ability to hide JavaScript from the page author, but show it to the component author
  - > State Management: easily keep client and server state in synch

Project Dynamic Faces (DynaFaces)

What is it?

- Incremental improvement in JSF 1.2 runtime to enable first class AJAX support in JSF.
- Extends the JSF lifecycle to work on AJAX requests.
- Drop in JAR to any JSF 1.2 compliant container
- One line of additional configuration in web.xml
- Ready to use JavaScript library included

DynaFaces — History, Current Status

- Started as Avatar: an idea on Jacob Hookom's blog in September 2005.
- Refined by Ed Burns, Jacob Hookom, and JSF developer community since then.
- DynaFaces is an implementation of the Avatar idea.
- Still experimental, currently in 0.1 SNAPSHOT release.

DynaFaces — Dependencies

- Shale Remoting 1.0.3 (and all of its dependencies)
  - commons-beanutils-1.7.0.jar
  - commons-chain-1.0.jar
  - commons-codec-1.2.jar
  - commons-collections-2.1.jar
  - commons-digester-1.6.jar
  - commons-el-1.0.jar
  - commons-fileupload-1.0.jar
  - commons-logging-1.0.4.jar
  - xml-apis.1.0.b2.jar
- JSF 1.2 (and all of its dependencies)
  - servlet-api-2.5.jar
  - jsp-api-2.1.jar

DynaFaces — Entry Points

By Role

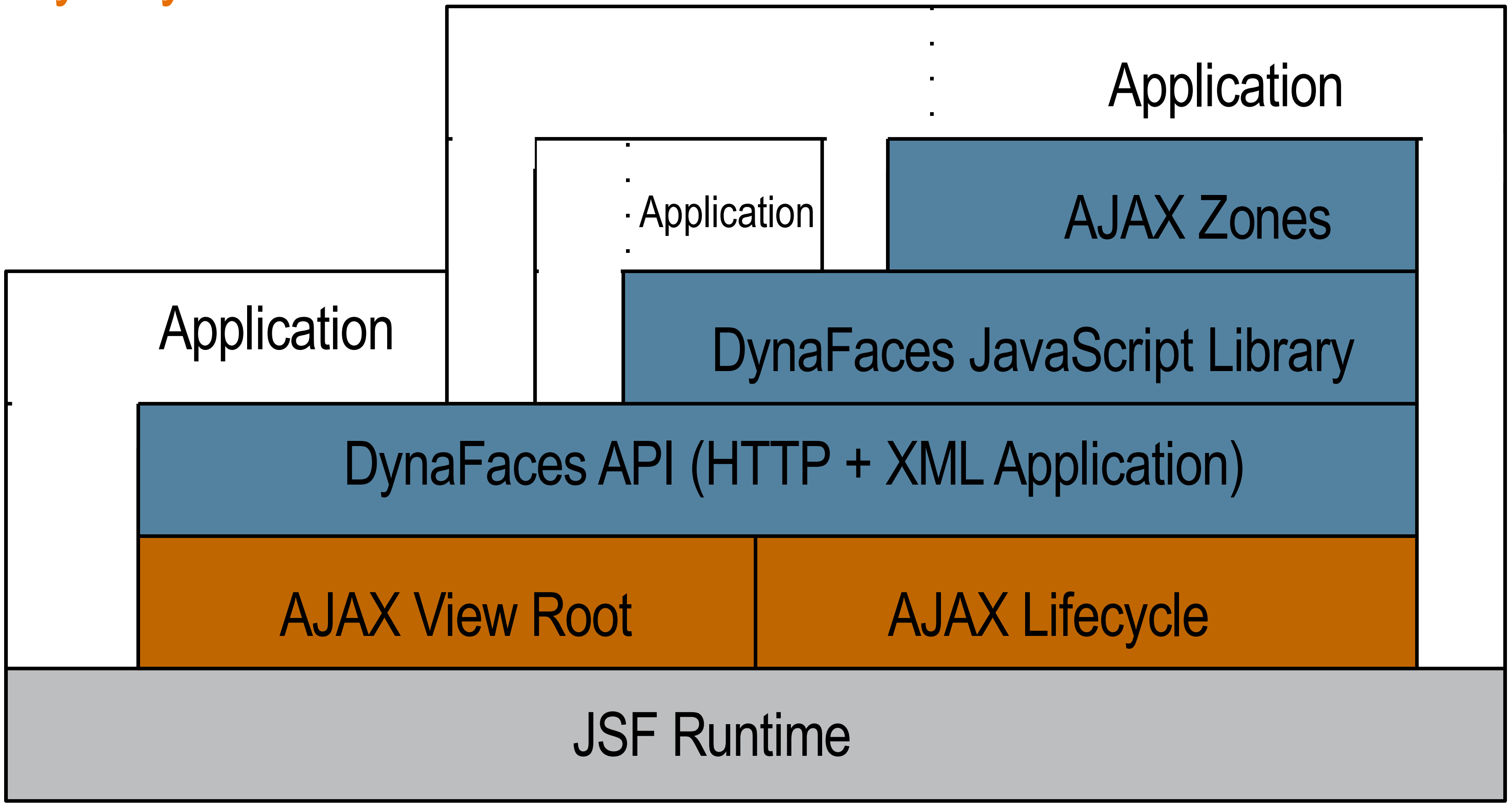
Increasing Complexity

- Page Author
  - Use AJAX enabled components
  - Use ajaxZone tag to AJAXify regions of the page
  - Use provided JavaScript library to AJAXify page elements and components
- Component Author
  - Build composite components with AjaxZones
  - Use provided JavaScript library in custom components
  - Write your own JavaScript that talks directly to the HTTP protocol and the XML application defined by DynaFaces

Increasing Complexity

DynaFaces — Entry Points

By Layer



Demonstrations

DynaFaces

Ed Burns
ed.burns@sun.com