



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 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
 - > Excellent I18N, L10N, A11Y
- 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 — Dependencies

- Shale Remoting 1.0.3 (and all of its dependencies)
 - > commons-logging-1.0.4.jar
- JSF 1.2 (and all of its dependencies)
 - > servlet-api-2.5.jar
 - > jsp-api-2.1.jar
- In other words
 - > Java EE5 + Shale Remoting + Commons Logging
- In other words, Glassfish 

DynaFaces — Entry Points

By Role

- 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

Increasing Complexity
↓

- 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
↓

Views and Partial Views

DynaFaces Usage Patterns

Using AjaxZones

- The easiest way to AJAXify an existing application
- Demarcate one or more AJAX zones within a page
- Zones will refresh via AJAX, without full page refresh.
- Action in one zone causes reaction in another zone

Zone 1

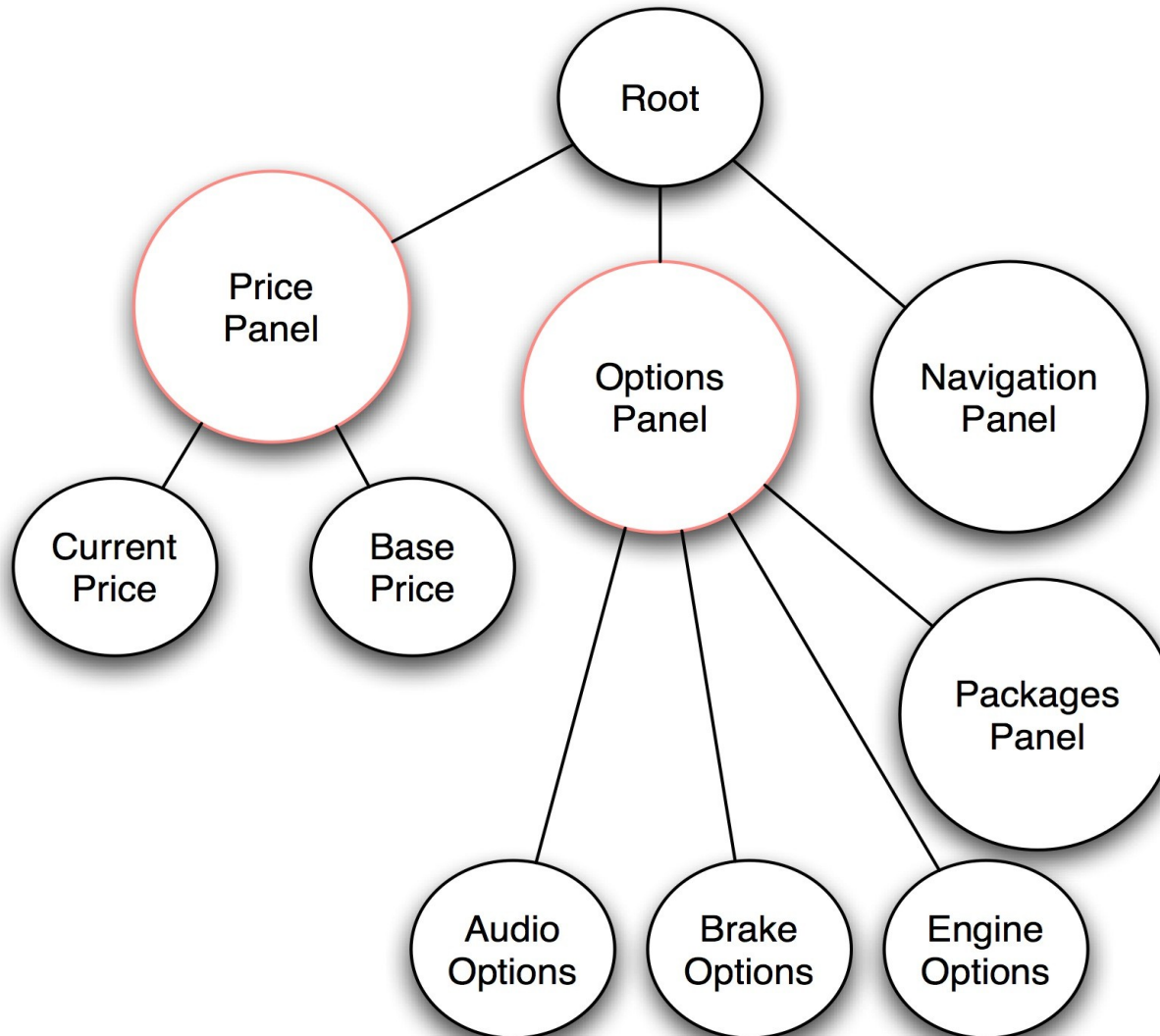
2. See update here

Zone 2

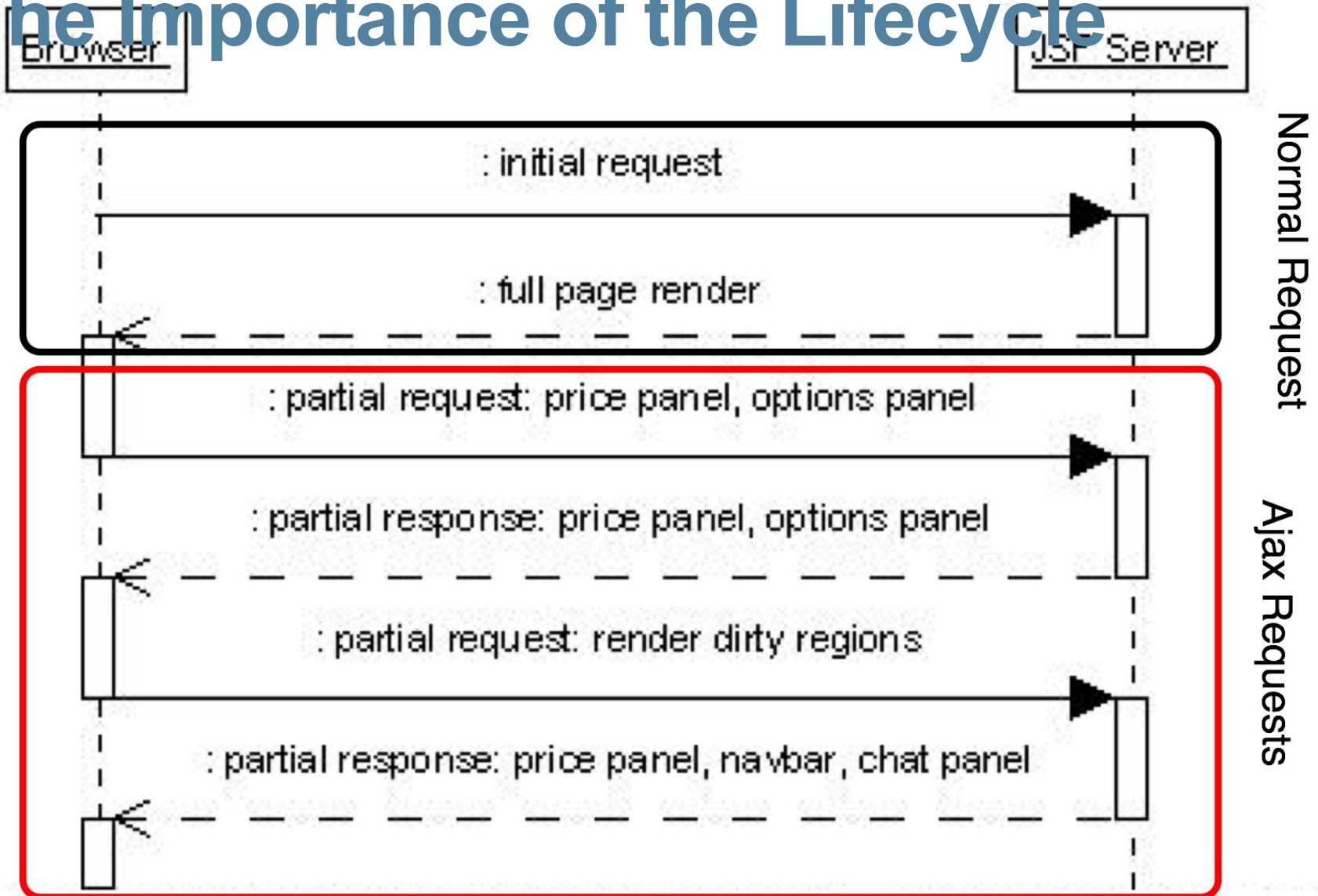
interactionType: input
eventType: onclick
eventHook: extractParams
action: #{carstore.updatePricing}

1. Click something in here

Views and Partial Views



The Importance of the Lifecycle



Demonstrations



DynaFaces Usage Patterns

Using `DynaFaces.fireAjaxTransaction`

- Defined in built-in JavaScript library.
- When called, causes an AJAX transaction to the Faces server.
- Many options for customizing the transaction.

Order Total: \$452.16

Id	Description	UOM	Qty	
1	BX Latex Surgical Gloves 3M			\$10.40
22	BX 40cc Syringe			Flownder Medical \$441.76
Id	Description	UOM	Qty	
59339	40cc Syringe	BX	22	Add Item
45439	Latex Surgical Gloves	BX	1	Add Item
46787	Bed Restraint	DZ		Add Item
54333	Small Cane Tip	EA		Add Item
78799	Large Cane Tip	EA		Add Item

Diagram illustrating the usage of `DynaFaces.fireAjaxTransaction` for each "Add Item" button:

- onclick `DynaFaces.fireAjaxTransaction(this);`
- onclick `DynaFaces.fireAjaxTransaction(this);`
- onclick `DynaFaces.fireAjaxTransaction(this);`
- onclick `DynaFaces.fireAjaxTransaction(this);`
- onclick `DynaFaces.fireAjaxTransaction(this);`



DynaFaces

Ed Burns

ed.burns@sun.com

