Swing and Java Server Faces – How does it fit?

It might sound a bit strange:

- **Java Swing** is an approved, widely used standardized User Interface framework for rendering beautiful screens at the user's frontend.

  Java Swing has shown that it is fast, robust and compatible throughout various operating systems. Java Swing is easily installable on client-side desktops - web start and applet technology is widely used.

- **Java Server Faces (JSF)** is an approved, widely used standardized server-side framework for binding client-side UI processing to server-side application logic.

  Though being mainly used in the area of HTML web applications, JSF is designed to serve multiple User Interface technologies. Java Server Faces is the standard for decoupling server-side processing from client-side rendering.

So it makes sense to combine both frameworks - in order to come up with an efficient, standardized way to build power user applications for mission critical enterprise applications.

Browser-based Swing Client

Imagine the CaptainCasa Enterprise Client being a browser: it sends http requests ("URLs") to the server, and receives back layout definitions.

Well, the layout definitions are not HTML, of course. They are XML layout definitions, in which various controls are assembled to form pages – just like HTML, but with a different set of controls.

The client interprets these layout definitions and renders them. And, while rendering, the client very smartly finds out what has already been rendered, what has to be changed, and what has to be removed. The result: page updates are very fast!

The user updates data within the screen and causes events that trigger the client to communicate back to the server. Communication is done for events which are marked as being relevant for synchronization.

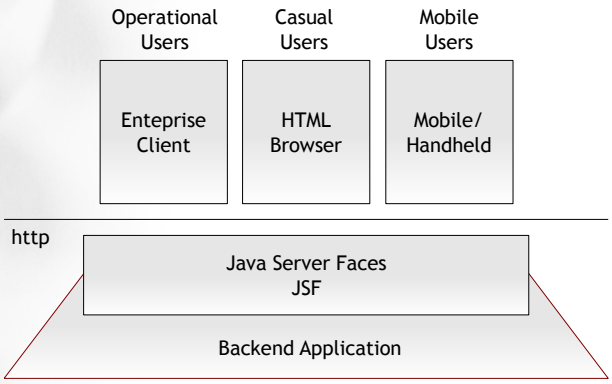
Default JSF Server Processing

On server-side, events and requests from the client are processed within JSF in exactly the same way as JSF processes requests from an HTML browser.

Indeed, there is no difference in server-side programming between "HTML Browser" and "Enterprise Client":

- Layouts are defined (if desired) in JSP pages.
- The component tree is processed, and it can be manipulated at any time.
- Data is passed to managed beans, which reflect the logical part of the processing.

Advantages



Why does the combination of "Swing-based browser" and "JSF processing" make sense?

- It allows to plug a native rendering technology, that is designed to serve power users and operational users, into a web application – without changing the programming model on the server side.
- It provides a simple and efficient way to implement rich client frontends for server-side enterprise applications. Application developers declaratively define screens by XML which are rendered in the Java Swing based client.

CaptainCasa GmbH
Hindemithweg 13
D – 69245 Bammental
+49 6223 484147

www.CaptainCasa.com
info@CaptainCasa.com