




JavaServer Faces (JSF) Fundamentals



An overview of core concepts,
lifecycles, and best practices



What is JSF?

- A Java-based web framework for building component-driven UIs
- Part of Java EE (now Jakarta EE)
- Simplifies MVC architecture with reusable UI components

Merits and Demerits of JSF

- Merits:

- - Component-based: promotes reuse
- - Integrated with Java EE: built-in support for validation, navigation
- - Rich set of UI libraries (PrimeFaces, RichFaces)

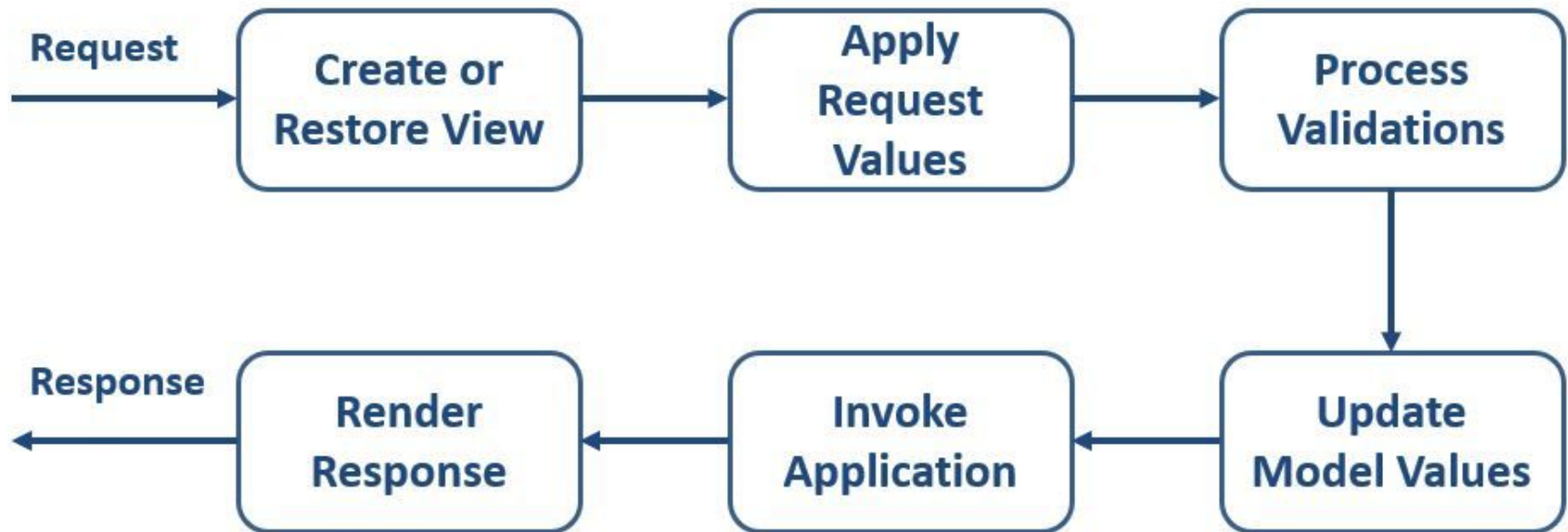
- Demerits:

- - Can be heavyweight: learning curve, performance overhead
- - Complex configuration in early versions

JSF Lifecycle

- 1. Restore View: Build component tree
- 2. Apply Request Values: Populate components
- 3. Process Validations: Convert & validate data
- 4. Update Model: Push values to backing beans
- 5. Invoke Application: Execute actions
- 6. Render Response: Generate HTML output

JSF Lifecycle



Features of JSF

- UI Component Model: built-in & custom tags
- Managed Beans: Java objects as controllers
- Navigation Handling: XML or annotations
- Event Handling: Listeners & Ajax support
- Templating: Facelets for view composition

Managed Bean in JSF

- A Java class annotated (`@ManagedBean`) or CDI (`@Named`)
- Holds UI state & action methods
- Configured via annotations or `faces-config.xml`

Bean Scopes in JSF

- `@RequestScoped`: per HTTP request
 - - Lightweight, no state retention
- `@ViewScoped`: per view, survives postbacks
 - - Maintains state within same page
- `@SessionScoped`: per user session
 - - Good for multi-page flows, memory cost
- `@ApplicationScoped`: shared across all users
 - - Global config, be cautious with state

AJAX in JSF

- AJAX = Asynchronous JavaScript and XML
- Benefits:
 - - Partial page updates, improved UX
 - - Reduced bandwidth usage
- Using AJAX in JSF:
- `<p:ajax>` tag for event-based updates
- Built-in support in PrimeFaces, RichFaces