Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

Conversations and Page Flows on the JavaServer™ Faces Platform

Dan Allen
JBoss, a division of Red Hat, Inc.
Senior Software Engineer

Springing JSF from its social shell

Today you are going to learn two ways to implement a multi-page dialog in JSF

Seam

Spring Web Flow

Agenda

- Introduction to conversations and page flows
- Conversations and page flows in Seam
- Flows with Spring Web Flow and Spring Faces
- Finding peace with Java persistence
- Dealing with the “back” button
- Summary

Scope inventory

A brief look at where state is stashed

- Servlet API
  - Request – single request
  - Session – all requests by same browser session
  - Application – all requests
- JSF
  - View (Page) – UI component tree
- Query string
  - Hidden form fields
  - URL rewriting

The fallacy of state in JSF

- Stateful UI
- Lacks stateful model to support UI
- Can stash state in UI component tree
  - View (page) scope (e.g., <t:saveState>)
  - Does not survive navigation event
  - No predictable removal point
- Session is our crutch
  - Leaks memory / cross streams

What is a conversation?

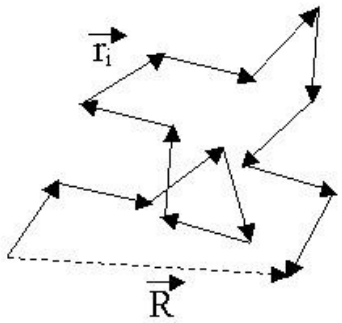
- *A long-running context whose boundaries are dictated by the application logic*

Conversation aspects

- Long-running context
  - request ≤ conversation << session
- Retains object identity
- Can exist in parallel
  - No interference between tabs/windows
  - Can become idle and later continued

Conversation navigation styles

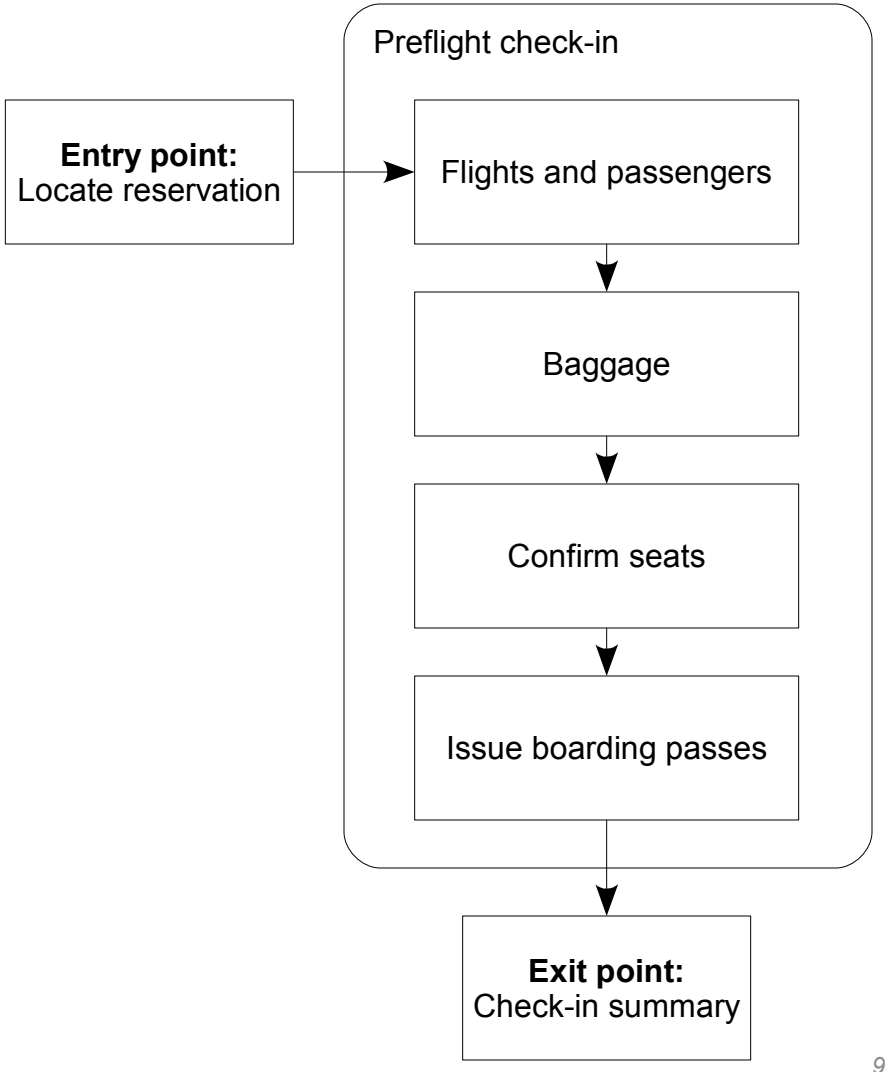
- Ad-hoc (Seam only)
  - User decides where to go next
  - Useful when goal is vague
  - Less work to setup



- Constrained
  - Guided by a page flow
  - Ideal when goal is well defined

What is a page flow?

- *A progressive series of states (i.e., pages) with constrained transitions modeling a single-user process*



Going with the flow

- Have a definitive START and END
- Backed by a conversation
- Each state represents a page view (wait state)
- User events trigger transition to next state
- Transitions
  - Route – can be decision-based
  - Invert control
  - Can invoke auxiliary behavior

Seam

Preflight Check In

A conversation and page flow demo

Conversations in Seam

- Central aspect of framework – *no setup required!*
- Conversation styles
  1) Ad-hoc
  2) Constrained
- Controlled declaratively
- Not explicitly tied to a navigation model
  - Page flow is an optional feature
- Available on every request
  - Application dictates if it lives on

Conversation propagation
Keeping the conversation in play

- Conversation token
  - Synthetic numeric value (default): cid=1
  - Natural business key: airportCode=BWI
- Token propagation (typically automatic)
  - Faces request (postback) – stored in UI view root
  - Non-faces request – passed as request parameter
- Token not shared across windows or tabs

Seam page flows

- Driven by jBPM execution engine
- Flows defined in jPDL
  - XML-based process language for jBPM
  - Page nodes map to JSF view IDs
  - Resembles Seam stateless navigation descriptor
  - Uses JBoss EL to resolve expressions
- A page flow is an extension to a conversation
- Subflows only partially supported

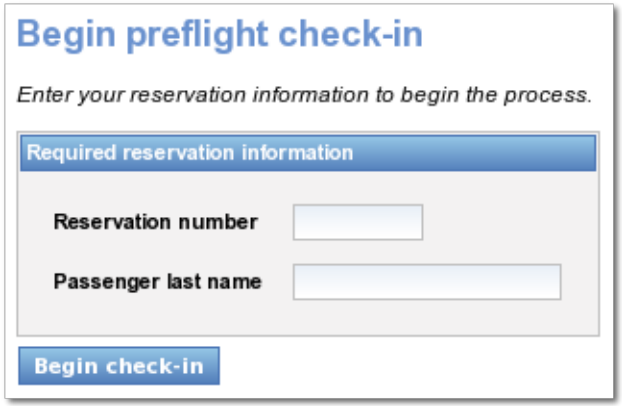
Registering the page flow
/WEB-INF/components.xml

```
<components ...>
    <bpm:jbpm>
        <bpm:pageflow-definitions>
            <value>check-in.jpdl.xml</value>
        </bpm:pageflow-definitions>
    </bpm:jbpm>
</components>
```

Seam page flow definition
/WEB-INF/classes/check-in.jpdl.xml

```
<pageflow-definition ... name="check-in">

    <start-state name="begin">
        <transition name="begin" to="flights"/>
    </start-state>

    <page name="flights" view-id="/checkIn/flights.xhtml">
        <redirect/>
        <transition name="cancel" to="cancel"/>
        <transition name="continue" to="declareBaggage"/>
    </page>

</pageflow-definition>
```

Initiating the page flow
/start.xhtml




```
<h:form>
    <h:panelGrid columns="2">
        <h:outputLabel for="number"
            value="Reservation number"/>
        <h:inputText id="number"
            value="#{reservationIdentifier.reservationNumber}"/>
        <h:outputLabel for="lastName"
            value="Passenger last name"/>
        <h:inputText id="lastName"
            value="#{reservationIdentifier.passengerLastName}"/>
    </h:panelGrid>
    <h:commandButton value="Begin check in"
        action="#{checkInAssistant.locateReservationForCheckIn}"/
>
</h:form>
```

Beginning the conversation / page flow

CheckInAssistant.java

```
@Name("checkInAssistant")
@Scope(ScopeType.CONVERSATION)
public class CheckInAssistant implements Serializable {

    @In private ReservationIdentifier reservationIdentifier;
    @Out(required = false) private Reservation reservation;
    @DataModel private List<BoardingPass> boardingPasses;

    @Begin(pageflow = "check-in", flushMode =
FlushModeType.MANUAL)
    public String locateReservationForCheckIn() {
        reservation = ...;
        if (reservation == null) return null;
        boardingPasses = ...;
        return "located";
    }
}
```

Protecting page flow views
/WEB-INF/pages.xml

- Page flow views served via normal JSF life cycle
  - JSF isn't aware of page flows
- Seam can require conversation to render page
  - Not part of page flow definition
  - Doesn't say *which* conversation is required

```
<page view-id="/checkIn/*" conversation-required="true"
    no-conversation-view-id="/start.xhtml"/>
```

Workspace management
For when the user strays

- Workspace
  - A conversation with a description (continuable)
  - User can have parallel workspaces (akin to tabs)
- Nested conversation
  - Related, yet independent conversation
  - Parent conversation restored when ended
- Conversation switcher
  - Only one active workspace per window at a time
  - UI control used to select and resume a workspace

Spring Web Flow

Preflight Check In

A conversation and page flow demo

Spring web flows

- Spring Framework module
- Conversation styles allowed
  1) Constrained
- Flows defined in a DSL (XML)
  - Expressions evaluated with JBoss EL (in flow only)
- Has distinct scopes for top-level flow and subflow
  - conversation scope – visible to flow and subflows
  - flow scope – visible to current flow only
- Incorporates partial page updates into flow

Getting Spring in the flow

- Lots of copy-paste configuration to get started!
- Thankfully, it's a "set and forget" configuration
  - Flows discovered based on convention
- Flow is in full control

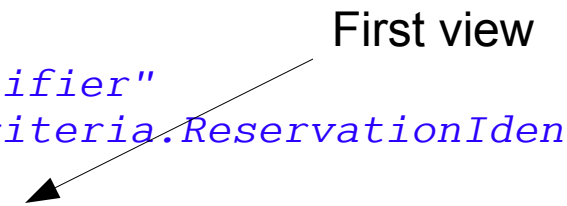
/preflight /spring /checkIn ?execution=e1s1

application path MVC handler flow name flow token

- Can model flow with DSL
  - UI strictly focused on input and output
  - All scoped data must be Serializable!

Defining a top-level flow

```
<flow ...>

    <var name="reservationIdentifier"
        class="org.preflight.criteria.ReservationIdentifier"/>

    <view-state id="enterReservationId">
        <transition on="locate" to="locateReservation"/>
    </view-state>

    <action-state id="locateReservation">
        <evaluate expression="checkInService.
            locateReservationForCheckIn(reservationIdentifier)"
            result="conversationScope.reservation"/>
        <transition on="${reservation!=null}" to="beginCheckIn"/>
        <transition on="${reservation==null}"
to="enterReservationId"/>
    </action-state>

</flow>
```



Branching to a subflow

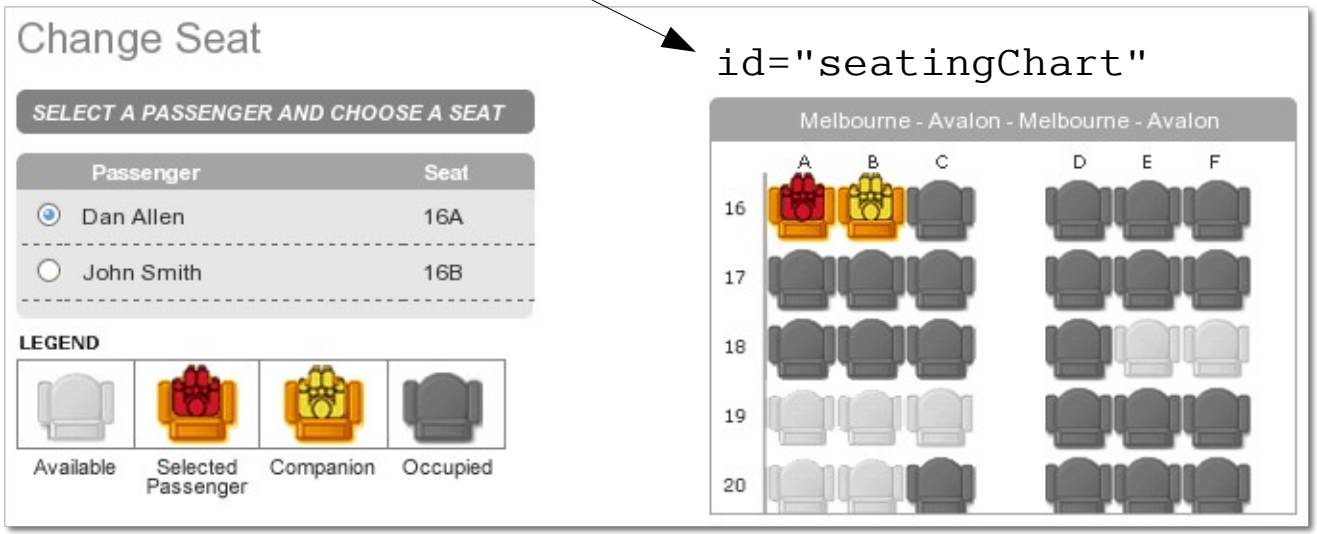
```
...
<subflow-state id="beginCheckIn" subflow="checkIn">
    <input name="checkInGroup" type="boolean"
        value="reservationIdentifier.checkInGroup"/>
    <transition on="confirmed" to="finish"/>
    <transition on="canceled" to="enterReservationId"/>
</subflow-state>
...
```

A persistence conscious subflow

```
<flow ...>
    <persistence-context/>
    <input name="checkInGroup" type="boolean" required="true"/>

    <on-start>
        <evaluate expression="checkInService.
            refetchReservation(reservation)"
            result="flowScope.reservation"/>
        <evaluate expression="checkInService.
            locateBoardingPasses(reservation, checkInGroup)"
            result="flowScope.boardingPasses"
            result-type="dataModel"/>
    </on-start>

    <view-state id="flights">...</view-state>

    <end-state id="confirmed" commit="true">
    <end-state id="canceled"/>
</flow>
```

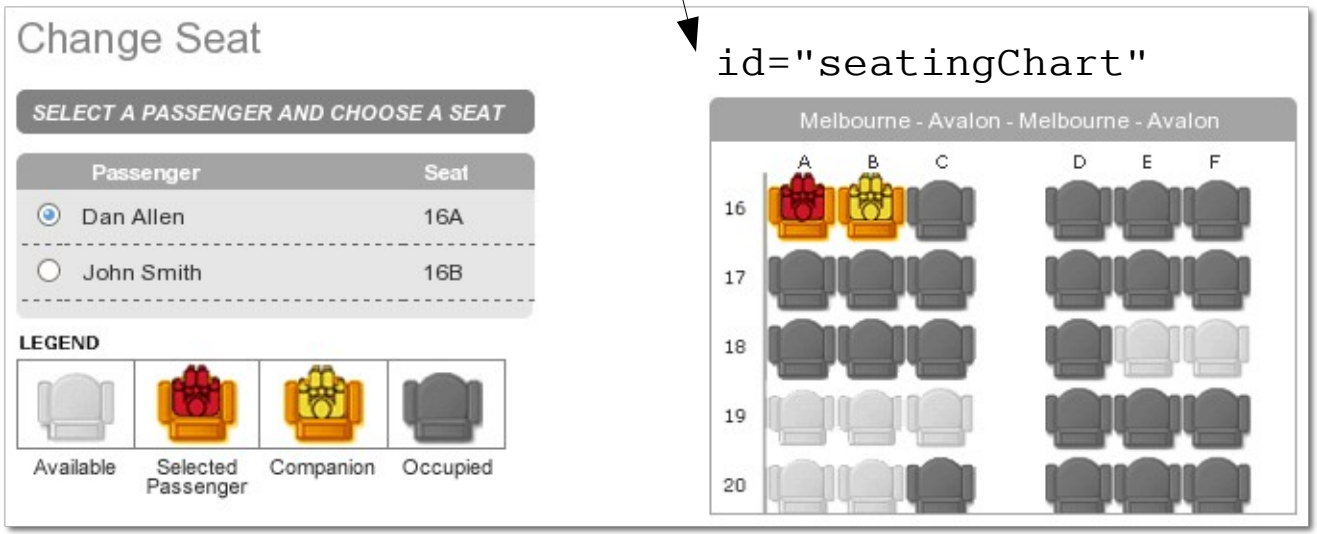
Partial page update transition

```
<view-state id="confirmSeats">
    <transition on="change">
        <evaluate expression="checkInService.changeSeat(
            boardingPasses.selectedRow,seatMatrix.selectedSeat)"/
>
        <render fragments="seatingChart"/>
    </transition>
    ...
</view-state>
```



How was it done in Seam?

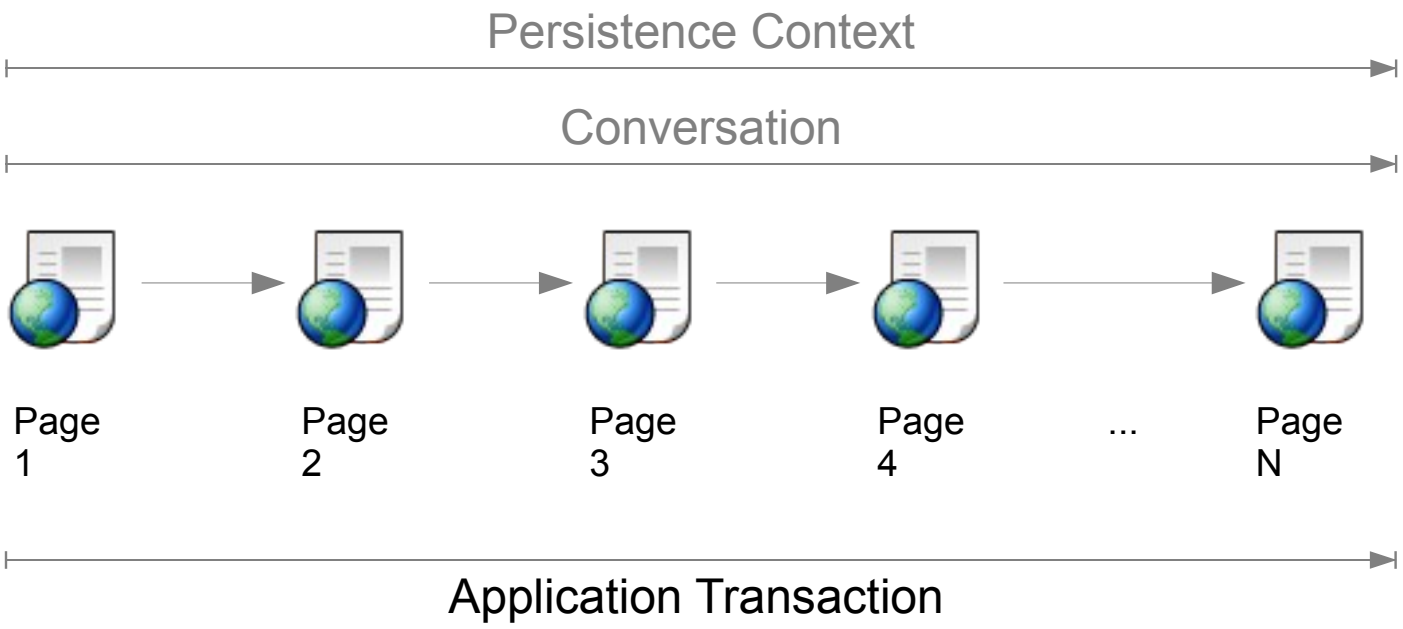
```
<rich:subTable var="_col" value="#{_colgroup.columns}">
    <rich:columns var="_row" id="#{_section.rows}" index="r">
        <a:commandLink action="#{seatSelector.select(
            _col.seatSelections[r])}" reRender="seatingChart">
        </a:commandLink>
    </rich:columns>
</rich:subTable>
```



Persistence mismanagement
...or who put the LIE in my Hibernate?

- Java persistence manager provides:
  - Unique cache of objects per database id
  - Lazy fetching of associated entities / collections
  - Automatic state detection (dirty checking)
    - Transitive persistence
    - Optimistic locking
- All bets are off when session is closed
  - All loaded entity instances become *detached*
  - The LazyInitializationException reigns!

Extending the persistence context



Don't commit changes until the user says so!

Making the changes stick

- Save == Flush persistence context

```
entityManager.flush();
```

```
<end-state commit="true"/>
```

Contending with evil browser buttons

- Detects out of sequence request
  - Blocks action
  - Routes to current page
- Attempt to use conversation that has ended will fail

Disclaimer:

Cannot prevent browser from revisiting cached page

A better “state” of affairs

What conversations and page flows provide

- Correlate sequential requests
- Maintain long-running state
- Guide and enforce navigation “flow”
- Support for multi-tasking

Conclusion

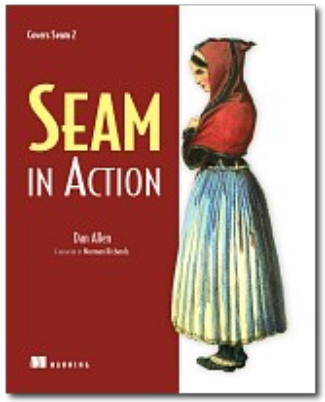
Seam

Spring Web Flow

Both frameworks offer a strong choice for implementing multi-page dialogs in a web application

Questions?

Let’s keep the conversation going!



Dan Allen
dan.allen@mojavelinux.com

http://mojavelinux.com
http://in.relation.to/Bloggers/Dan
http://code.google.com/p/seaminaction/source
http://delicious.com/seaminaction