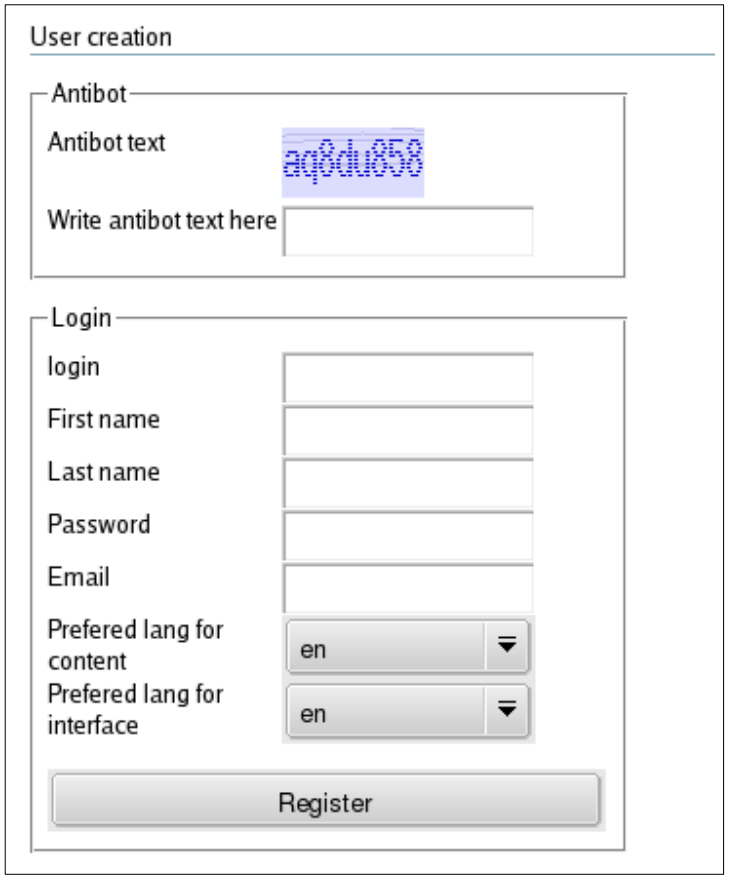
Java Server Faces
narzędzie do implementacji w-wy prezentacji

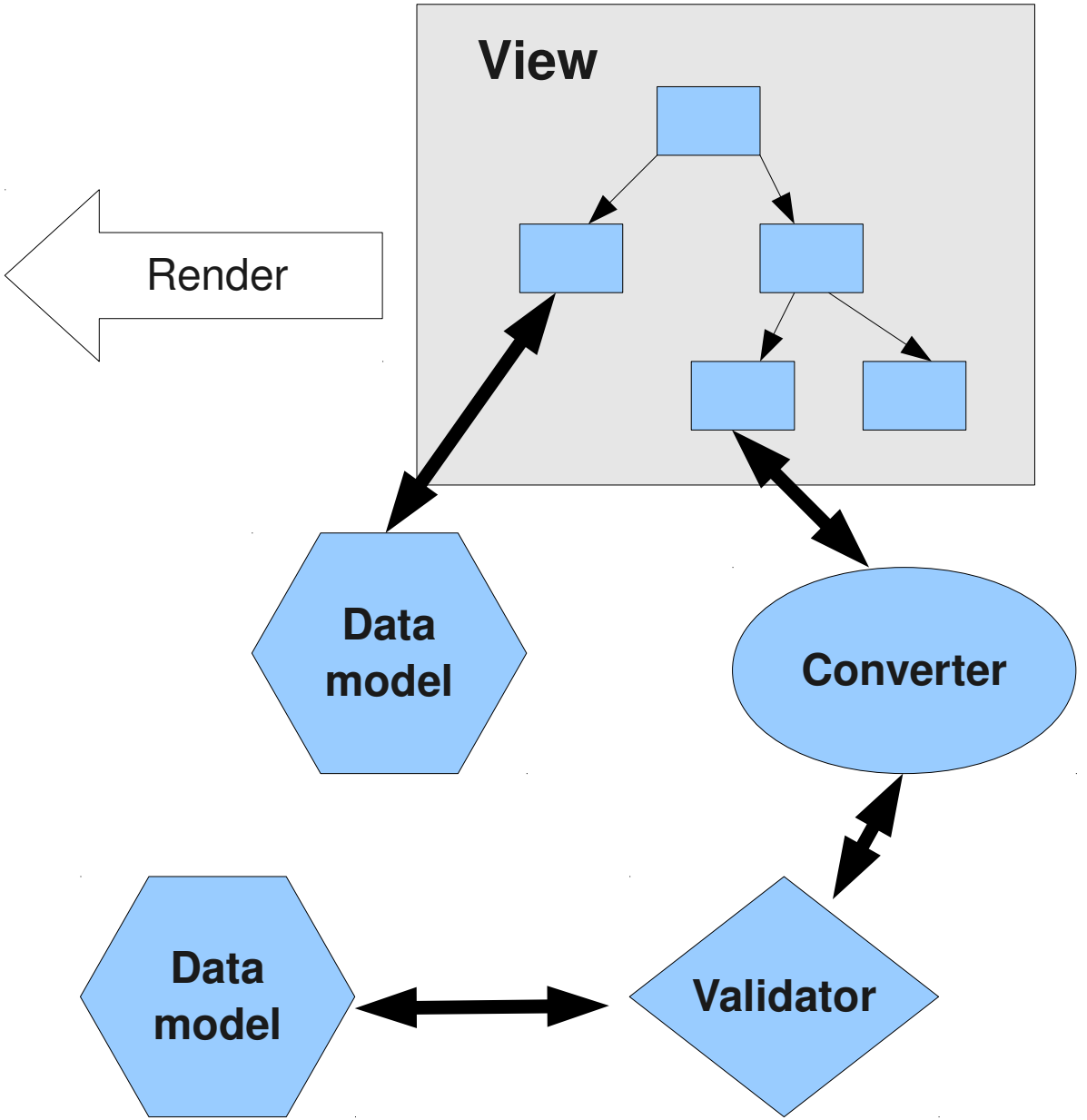
- pojęcie komponentu
- powiązanie z modelem danych
- widok (View) jako drzewo komponentów
- obiekty pomocnicze: konwertery, walidatory, obsługa zdarzeń

Zadania JSF

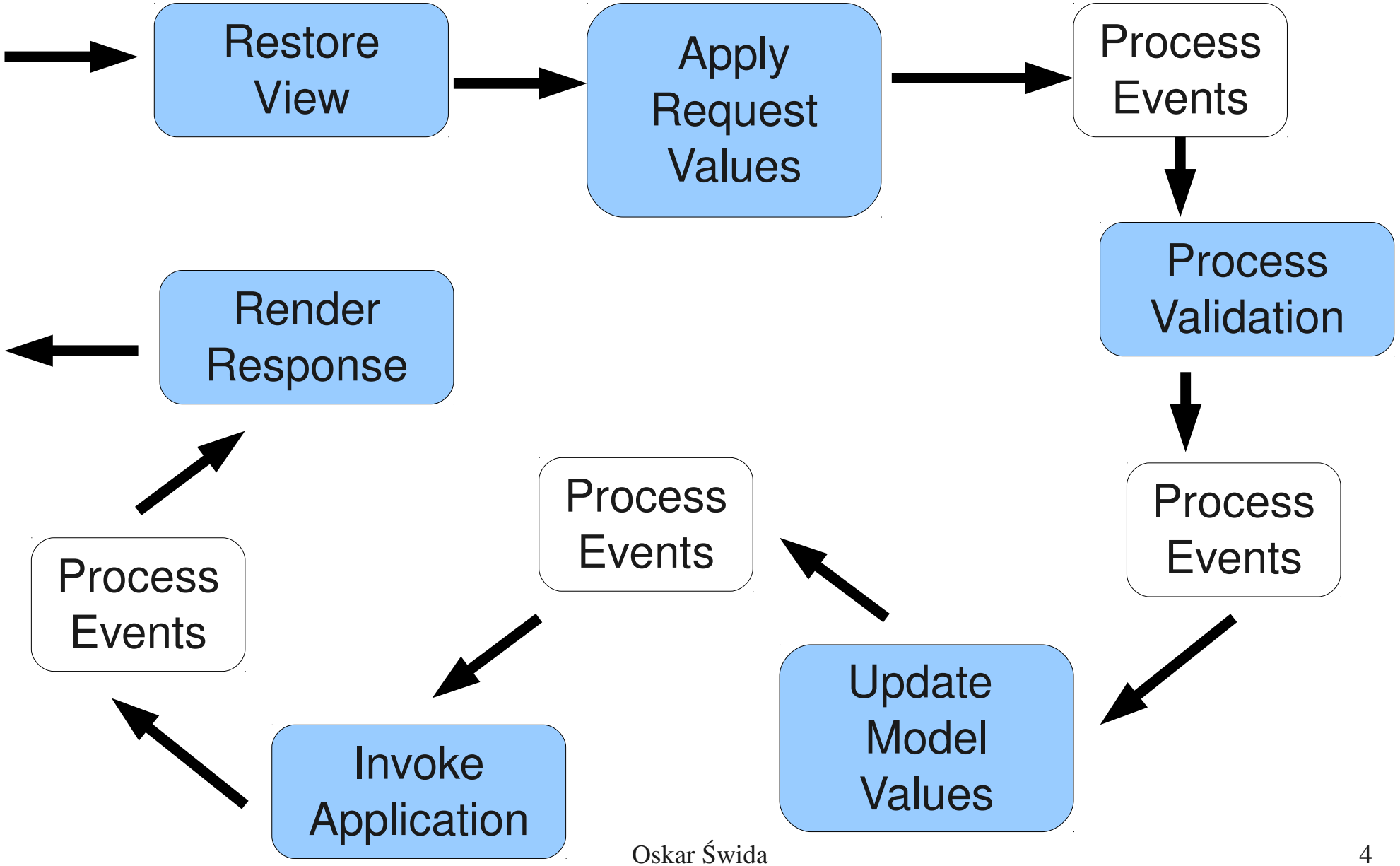
- zarządzanie stanem komponentów pomiędzy wywołaniami
- obsługa wielu klientów (przeglądarek)
- przetwarzanie formularzy (wielostronnicowe)
- zdarzenia generowane przez klienta obsługiwane po stronie serwera
- walidacja danych
- konwersja danych
- przejrzysty system obsługi błędów i wyjątków
- system nawigacji

Schemat środowiska JSF





Schemat przetwarzania żądań



Schemat przetwarzania żądań

- **Restore View**: szukamy widoku (*UIViewRoot)*
  - znaleziono - sprawdzamy "binding"
  - nie znaleziono - określamy typ żądania (initial, postback)
- **Apply Request Values**: zapis danych z żądania w węzłach widoku
  - komponenty typu *ActionSource* kolejkują zdarzenia (w fazie InvApp lub po aktualnej jeśli *immediate*)
  - komponenty typu EditableValueHolder wykonują konwersję i walidację jeśli *immediate*

Schemat przetwarzania żądań

- **Process Validations**: konwersja i walidacja z ewentualnym umieszczeniem komunikatu w kolejce
- **Update Model Values**: przekazanie danych do modelu wykorzystywanego przez aplikację
- **Invoke Application**
- **Render Response**
  - generowanie odpowiedzi
  - zapis stanu odpowiedzi dla późniejszych wywołań (State Management)

Obejrzyjmy przetwarzanie

- javax.faces.lifecycle - klasy odpowiedzialne za cykl życia żądania i odpowiedzi

faces.config.xml

```
<lifecycle>
 <phase-listener>wyklad.Sledz</phase-listener>
</lifecycle>
```

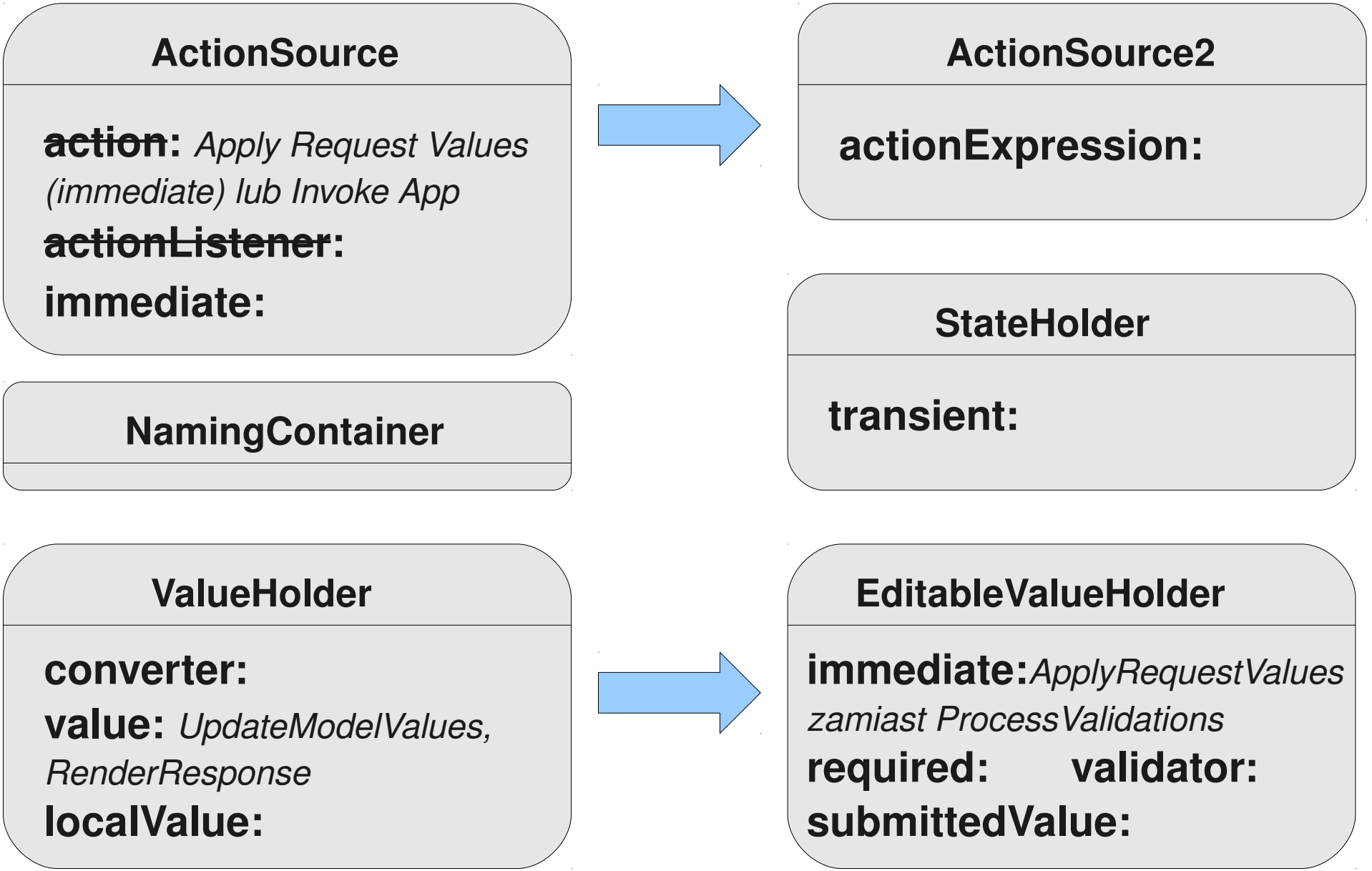
```
import javax.faces.event.*;

public class Sledz implements PhaseListener {
    public PhaseId getPhaseId() {  return PhaseId.RESTORE_VIEW;    }
    public void beforePhase(PhaseEvent event) {    }
    public void afterPhase(PhaseEvent event) {   }}
```

Interfejs zbudowany z komponentów

- Każdy element na stronie jest reprezentowany przez komponent
- Komponenty tworzą strukturę hierarchiczną (tak jak znaczniki)
- Aspekt (facet) - element zależny ale nie dziecko
- Własności komponentu niezależne od wizualizacji: *id, parent, rendered, rendererType, rendersChildren, transient*
- Dowiązywanie komponentu do obiektu po stronie serwera (binding)

Interfejs zbudowany z komponentów



Klasy wspomagające

- Konwertery: *Converter(getAsObject, getAsString)*
- Zdarzenia: ActionListener, ValueChangeListener
- Walidatory: *Validator (validate)*
- Standardowe walidatory:
- *DoubleRangeValidator* - dowolny typ numeryczny z zakresu
- *LengthValidator* - napis o wyznaczonej długości
- *LongRangeValidator* - liczba całkowita z zakresu
- *MethodExpressionValidator* - metoda obiektu

Obiekty zarządzane (Managed beans)

- Rola MB w aplikacji JSF
- Własności zarządzane (managed property)
- Zakres widzialności: none, application, session, request
- Obiekty niejawne (implicit):
- request: *view, facesContext, header, request*
- session: *session, sessionScope*
- application: *application, initParam*
- Język wyrażeń a obiekty zarządzane (wartości i metody)

Obiekty zarządzane c.d.

```
<h:inputText value="#{ob.wlasnosc}" />
<h:commandButton action="#{ob.metoda}" />
```

```
<managed-bean>
   <managed-bean-name />
   <managed-bean-class />
   <managed-bean-scope />
   <managed-property>
      <property-name />
      <value />
   </managed-property>
</managed-bean>
```

```
<managed-property>
   <property-name />
   <map-entries>
   <key-class />
   <value-class />
   <map-entry>
      <key />
      <value />
   </map-entry>
   </map-entries>
</managed-property>
```

Inne elementy aplikacji JSF

Akcja

public
bez parametrów
zwraca ***Object*** (outcome)

Navigation Handler

odpowiedzialny
za nawigację

View Handler

Obsługa faz
RestoreView oraz
RenderResponse

State Manager

Zapis i odczyt
widoku pomiędzy
wywołaniami
javax.faces.STATE_SAVING_METHOD

Nawigacja

```
<navigation-rule>

  <from-view-id />

  <navigation-case>
    <from-action />

    <from-outcome />

    <to-view-id />
  </navigation-case>
</navigation-rule>
```

Strona wyjściowa konkretnej reguły

Wyrażenie wskazujące na wykonywaną akcję

Wartość zwracana z akcji

Strona docelowa

Przykład

- **Temat:** tablica ogłoszeń
- **Realizacja**
  - faza 1 - ogłoszenia anonimowe
  - faza 2 - ogłoszenia imienne
- **Dane:** przechowywane w pliku (tytuł, treść, data dodania/ważności, autor, kategoria)
- **Funkcjonalność:** dodawanie, usuwanie, edycja, przeglądanie/przeszukiwanie ogłoszeń

Modyfikacja obsługi drzewa komponentów - Facelets

- Konfiguracja:
  - **javax.faces.DEFAULT_SUFFIX**
  - <application><view-handler>
    com.sun.facelets.FaceletViewHandler
  - Przestrzeń nazewnicza:
    http://java.sun.com/jsf/facelets
- Atrybut jsfc
- Znaczniki: *composition, decorate, insert, define, param, repeat*