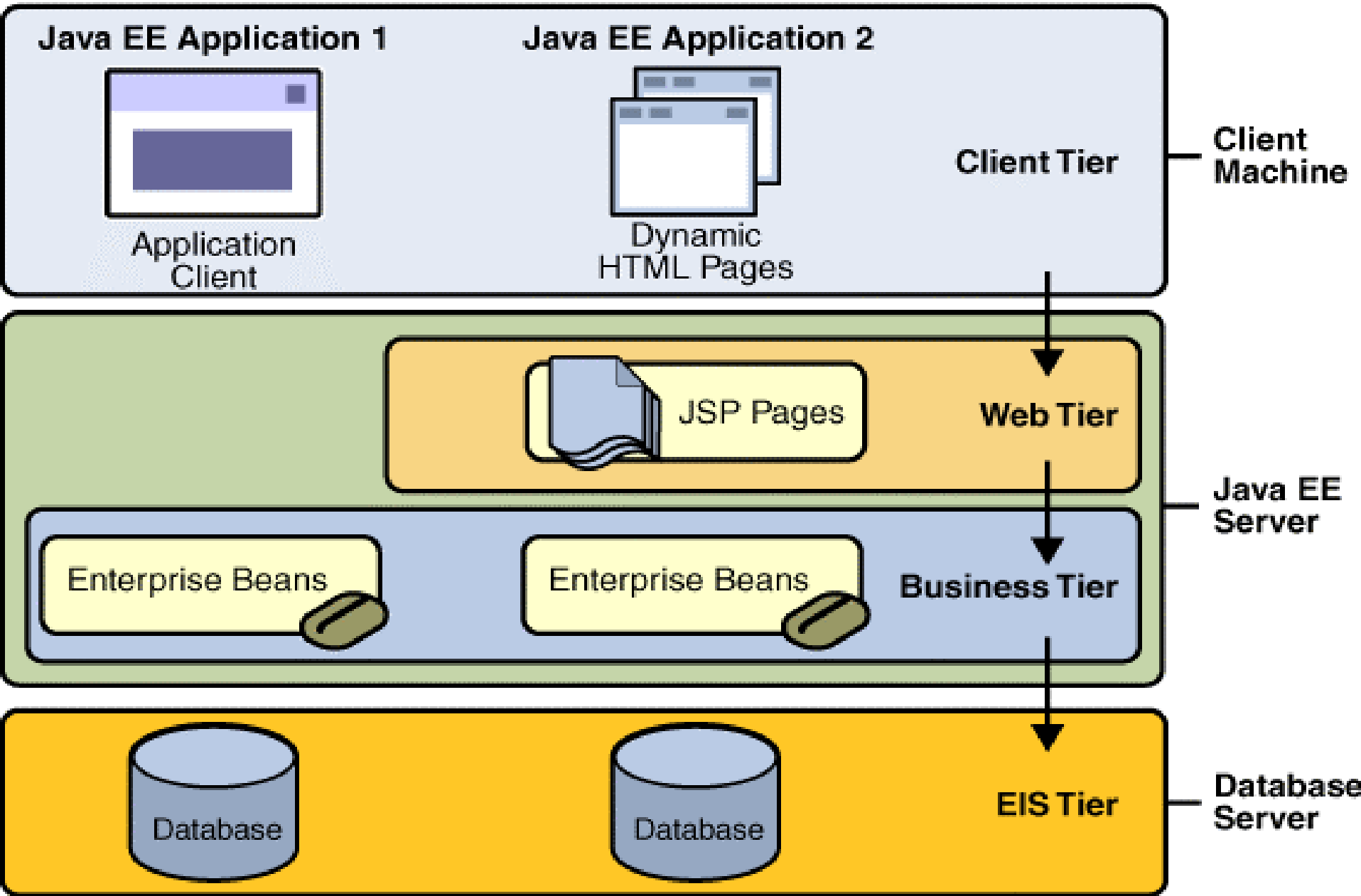
Podstawowe informacje o technologii JavaEE5

Programowanie komponentowe 2

Wielowarstwowe aplikacje w JavaEE5



Komponenty aplikacji w JavaEE5

- **komponenty typu klient** (warstwa na **maszynie klienta**): klienci aplikacji (GUI oparte na pakietach AWT/Swing), aplety
- **komponenty internetowe** działające na **serwerze aplikacji JavaEE5**
  - a) Java Servlet,
  - b) JavaServer Pages (JSP) technology
  - c) **JavaServer Faces**,
- **komponenty biznesowe**: Enterprise JavaBeans (EJB) działające na **serwerze aplikacji JavaEE5**

Kontenery aplikacji Java EE 5

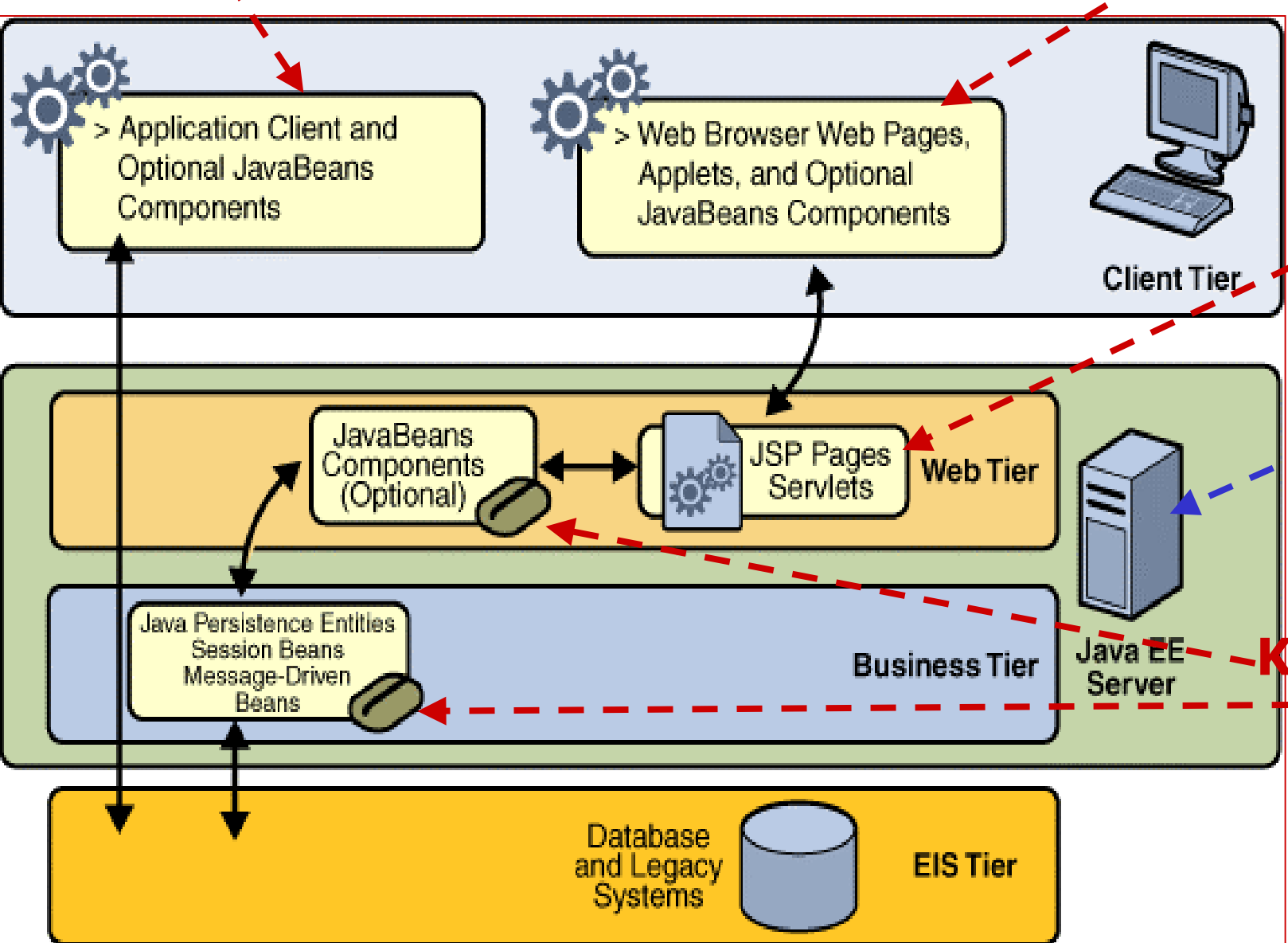
Modele usług kontenerów dla serwera aplikacji JavaEE

- **Model „security” dla komponentów internetowych i biznesowych (typu EJB)**
- **Model transakcji – wszystkie metody realizujące pojedynczą transakcję tworzą pojedynczy moduł transakcji**
- **Model usług JNDI (Naming and Directory Interface) – wyszukiwanie usług typu enterprise dla komponentów aplikacji**
- **Model zdalnych połączeń na niskim poziomie komunikacji między komponentami-klientami i komponentami biznesowymi (typu EJB)**

Architektura kontenerów

Kontener klienta aplikacji

Kontener apletu



Kontener web

Serwer aplikacji

Kontenery EJB

Tworzenie aplikacji JavaEE5

- **Build – tworzenie modułów Javy**

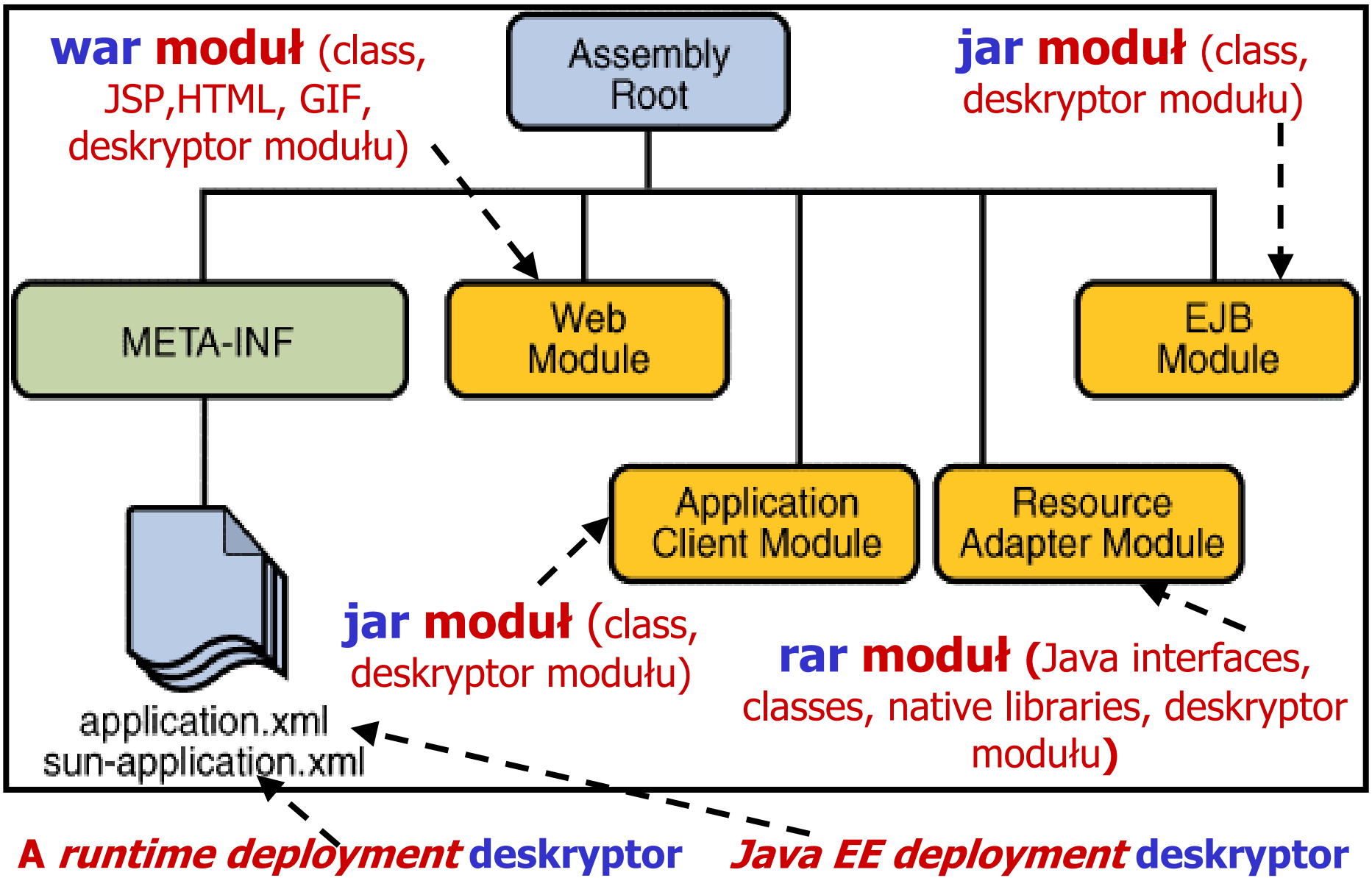
  a) tworzenie funkcjonalnych komponentów Javy (EJB, JSP page, servlet, applet, etc.)

  b) tworzenie opcjonalnego deskryptora opisującego zawartość modułu

- **Deploy: łączenie modułów z kontenerami**

  specyfikacja użytkowników oraz nazw lokalnych baz danych

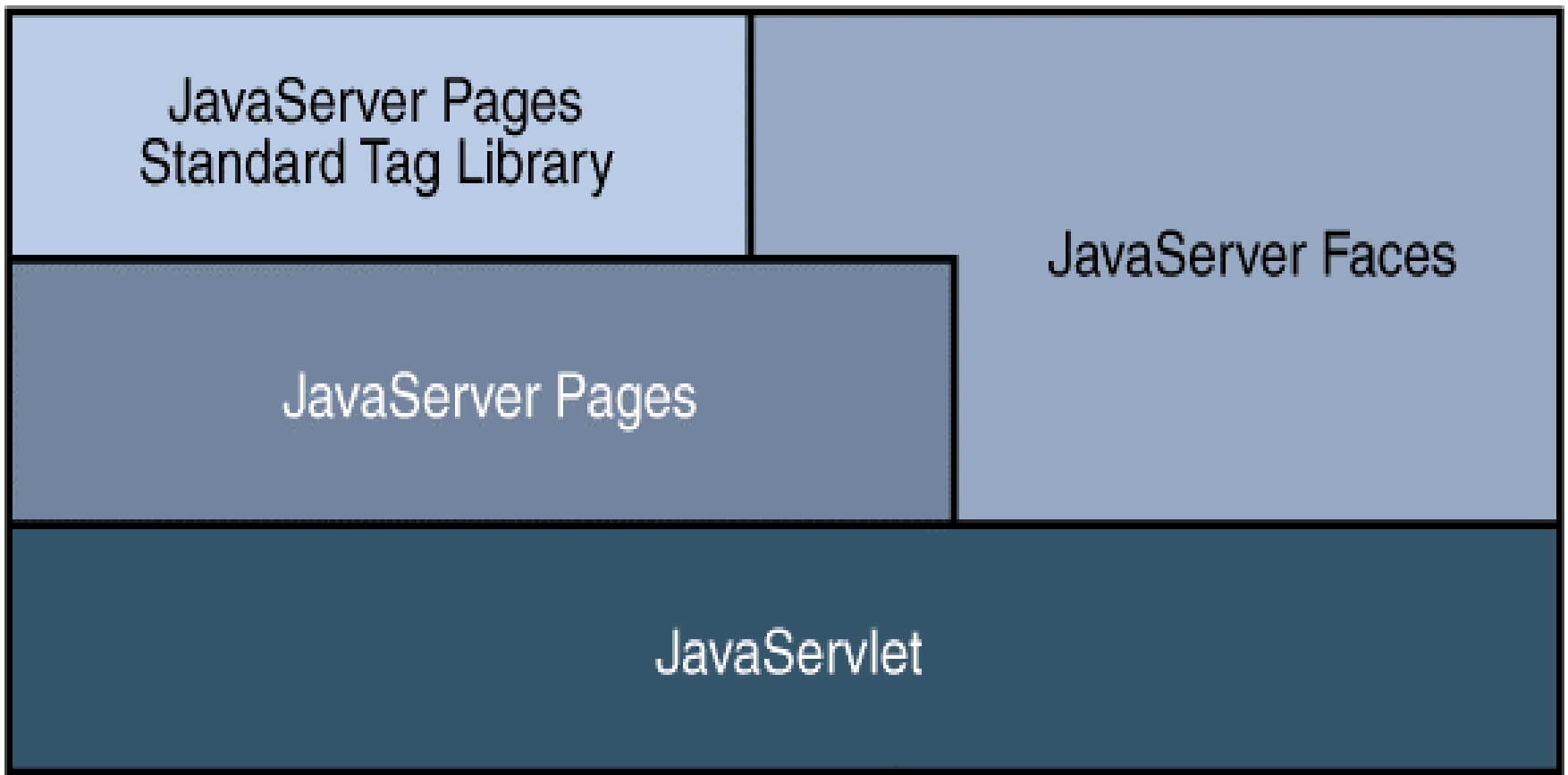
Struktura aplikacji JavaEE5– plik typu EAR



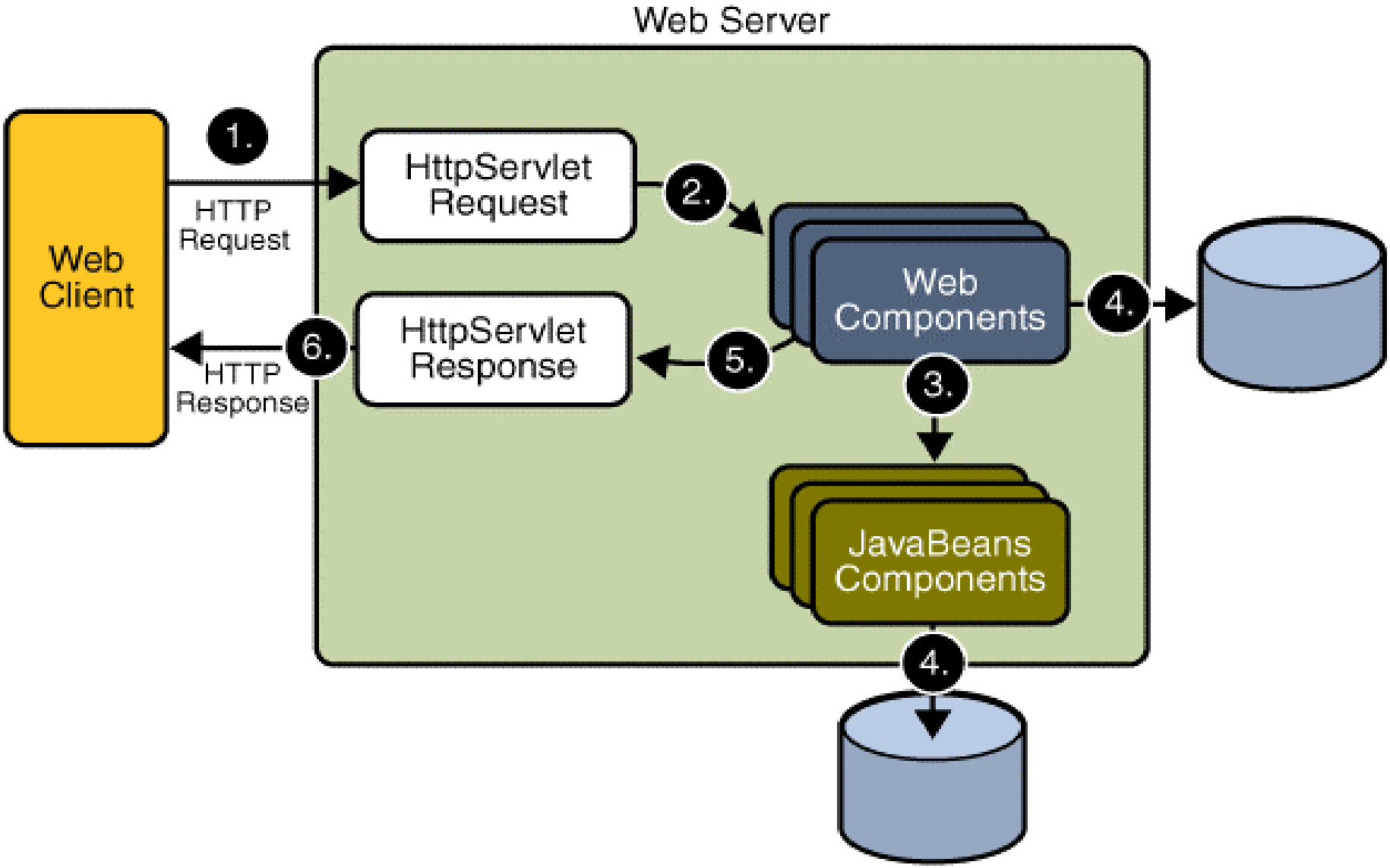
Podstawy technologii internetowych w JavaEE5 - typy aplikacji internetowych

zorientowane na prezentację

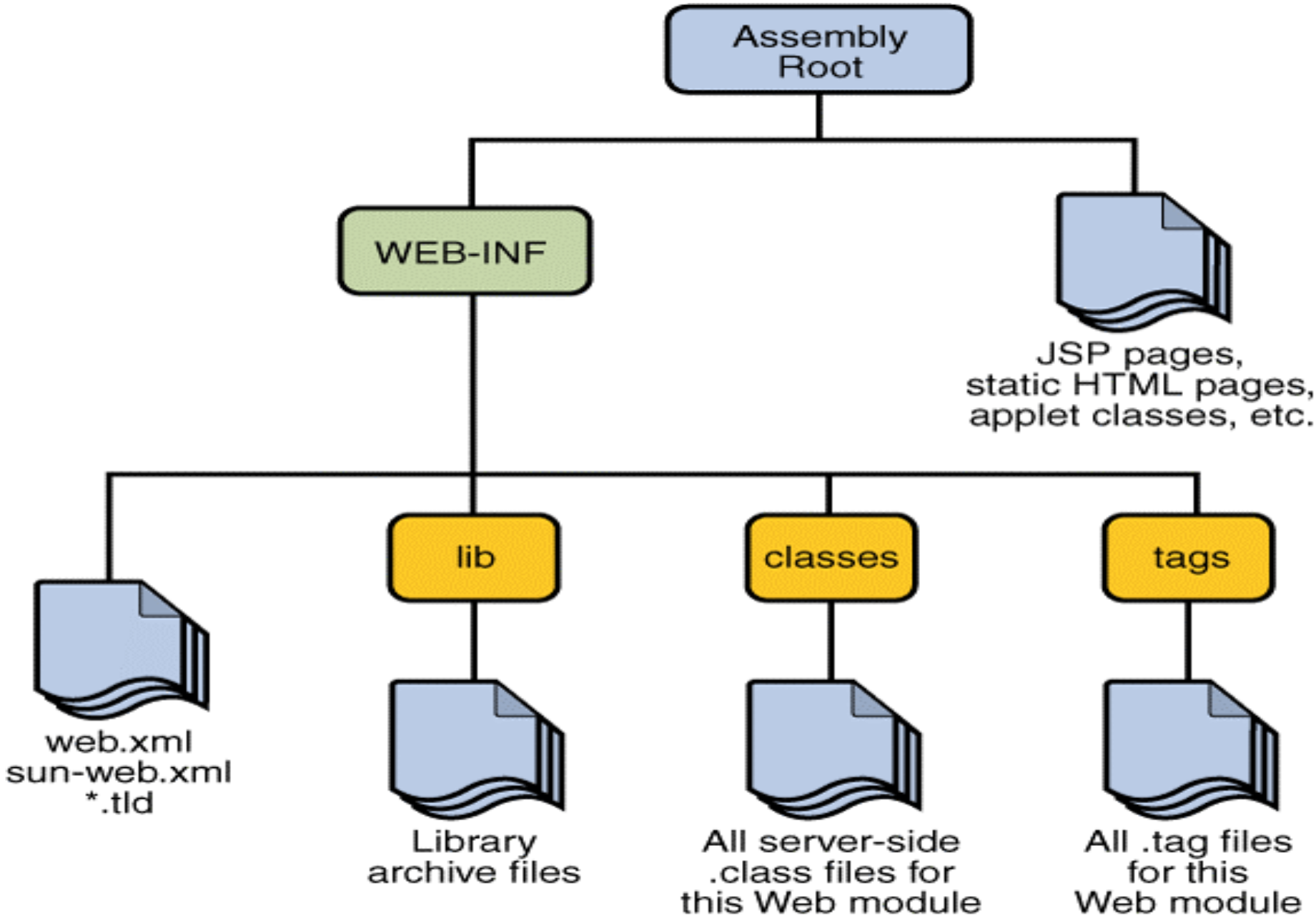
zorientowane na usługi internetowe



Podstawowa struktura działania aplikacji internetowej



Struktura modułu internetowego typu war



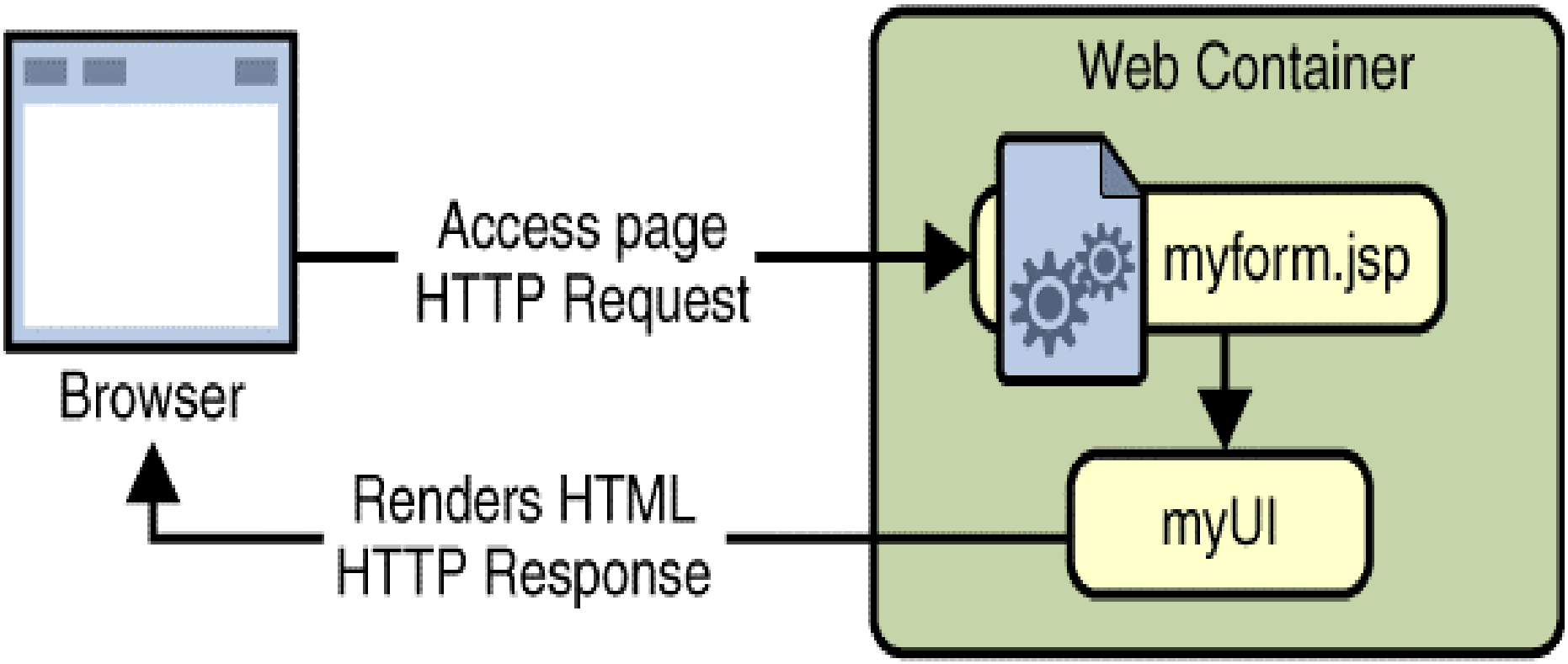
Co zawiera Java Server Faces?

- **Zbiór stron JSP** jako technologia prezentacji oraz możliwość stosowania innych technologii prezentacji
- **Zbiór *„backing beans”*,** czyli obiektów JavaBeans, które definiują właściwości i funkcje konwersji, walidacji i zdarzeń komponentów UI na stronie www
- **Plik zawierający konfigurację aplikacji**, który zawiera reguły nawigacji stron i konfiguruje ziarna (JavaBeans) i obiekty programisty np. komponenty
- **Deskryptor typu *„deployment”*** (web.xml)
- **Zbiór obiektów użytkownika**, które zawierają komponenty użytkownika, walidatory, konwertery, konwertery, „słuchacze zdarzeń”.
- **Zbiór znaczników użytkownika** reprezentujących obiekty użytkownika na stronie.

Zalety Technologii Java Server Faces

- **Separacja prezentacji od logiki aplikacji**
- **Przyjazna dla programisty koncepcja komponentów UI i warstwy internetowej**
- **Bogata architektura umożliwiająca zarządzanie stanem komponentu, danymi komponentu oraz walidacją danych użytkownika oraz obsługą zdarzeń**

Schemat obsługi komponentu UI



UI – model komponentów interfejsu użytkownika w technologii Java Server Faces

- **Zbiór klas komponentów** UI zawierających specyfikację stanu i zachowania tych komponentów
- **Model różnych sposobów odtwarzania** komponentu
- **Model zdarzeń i słuchaczy zdarzeń** do ich obsługi
- **Model konwersji danych**, określający sposób rejestrowania konwertera do komponentu UI
- **Model walidacji** danych, określający sposób rejestrowania walidatora do komponentu UI

Klasy typu UI

Znacznik klasy UI	Funkcje	Przedstawiana jako	Widok
column	kolumna danych w komponencie UIData	kolumna tabeli HTML	kolumna na stronie
commandButton	zatwierdza dane do przesłania z formularza do aplikacji	element HTML **<input type=*type*>**, gdzie *type może mieć wartość* **submit, reset**, lub **image**	przycisk
commandLink	link do innej strony lub innej części strony	element HTML **<a href>**	hyperlink
dataTable	reprezentuje opakowanie danych	element HTML **<table>**	tabela, która może zmieniać się dynamicznie
form	reprezentuje formularz zawierający inne komponenty do wprowadzania danych.	element HTML **<form>**	brak widoku
graphicImage	wyświetla obraz	element HTML **<img>**	obraz
inputHidden	pozwala autorowi strony używać ukrytych zmiennych na stronie	element HTML **<input type=hidden>**	brak widoku

inputSecret	pozwala użytkownikowi wprowadzać maskowany łańcuch znaków	element HTML **<input type=password>**	Pole tekstowe, które wyświetla wiersz znaków maskując wprowadzony łańcuch znaków
inputText	pozwala użytkownikowi wprowadzać łańcuch znaków	element HTML **<input type=text>**	pole tekstowe
inputTextarea	Allows a user to enter a multiline string.	element HTML **<textarea>**	wielowierszowe pole tekstowe
message	wyświetla konkretny komunikat	Znacznik HTML **,** jeśli użyto style	łańcuch znaków
messages	wyświetla konkretne komunikaty	Zbiór znaczników HTML **,** jeśli użyto style	łańcuchy znaków
outputFormat	wyświetla konkretny komunikat	tekst	tekst
outputLabel	wyświetla komunikat jako etykietę dla danego pola wejściowego	element HTML **<label>**	tekst

outputLink	Link do innej strony lub innej części strony bez generowania zdarzenia	element HTML **<a>**	A hyperlink
outputText	wyświetla linię tekstu.	tekst	tekst
panelGrid	wyświetla tabelę	element HTML **<table>** ze znacznikami **<tr>** i **<td>**	tabela
panelGroup	grupuje komponenty		Wiersz tabeli
selectBoolean Checkbox	umożliwia wybór typu Boolean	element HTML **<input type=checkbox>**	pole wyboru
selectItem	Reprezentuje pozycje listy w klasie UISelectOne	element HTML **<option>**	brak widoku
selectItems	Reprezentuje listę pozycji w klasie UISelectOne	A list of HTML `<option>` elements	brak widoku

selectMany Checkbox	pozwala wybrać użytkownikowi wiele pozycji ze zbioru pozycji typu checkbox i wyświetlić je jednocześnie	zbiór elementów HTML **<input>** typu **checkbox**	zbiór elementów typu checkbox
selectMany Listbox	pozwala wybrać użytkownikowi wiele pozycji ze zbioru pozycji i wyświetlić je jednocześnie	element HTML **<select>**	lista
selectManyMenu	pozwala użytkownikowi wybrać wiele pozycji ze zbioru pozycji	element HTML **<select>**	przewijana lista typu combo box
selectOne Listbox	pozwala wybrać użytkownikowi jedną pozycję ze zbioru pozycji i wyświetlić je jednocześnie	element HTML **<select>**	lista
selectOneMenu	pozwala użytkownikowi wybrać jedną pozycję ze zbioru pozycji	element HTML **<select>**	przewijana lista typu combo box
selectOneRadio	pozwala użytkownikowi wybrać jedna pozycję ze zbioru pozycji	element HTML **<input type=radio>**	zbiór przycisków typu radio

Rola „backing beans” - obsługa komponentu UI po stronie serwera przez obiekty JavaBeans

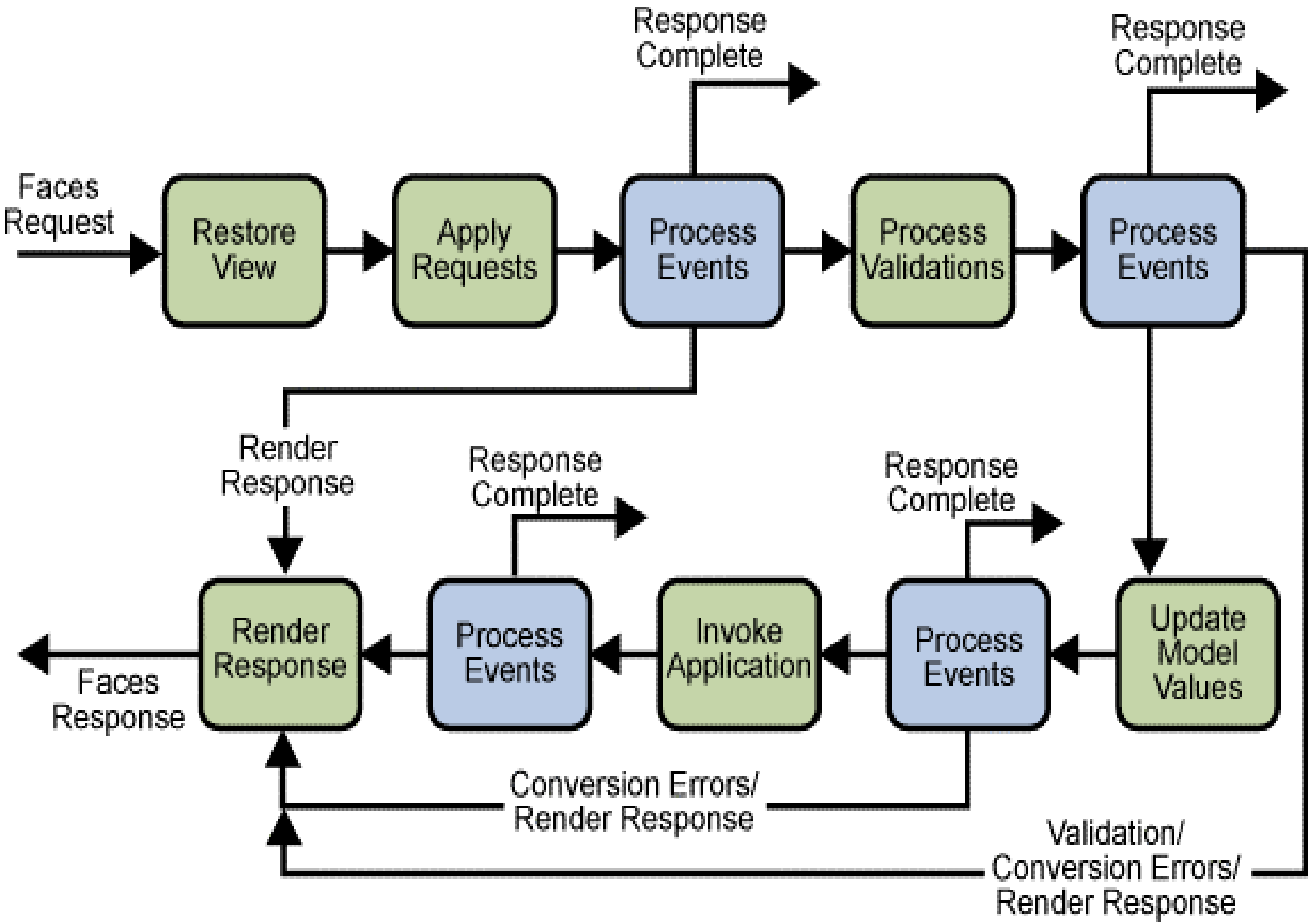
Atrybuty:

- Wartości komponentu
- Instancja komponentu
- Instancja konwertera
- Instancja słuchacza zdarzeń
- Instancja walidatora

Funkcje:

- Walidacja danych komponentu
- Obsługa zdarzeń generowanych przez komponent
- Obsługa nawigacji między stronami

Standard cyklu życia „Request-Response” dla JavaServer Faces

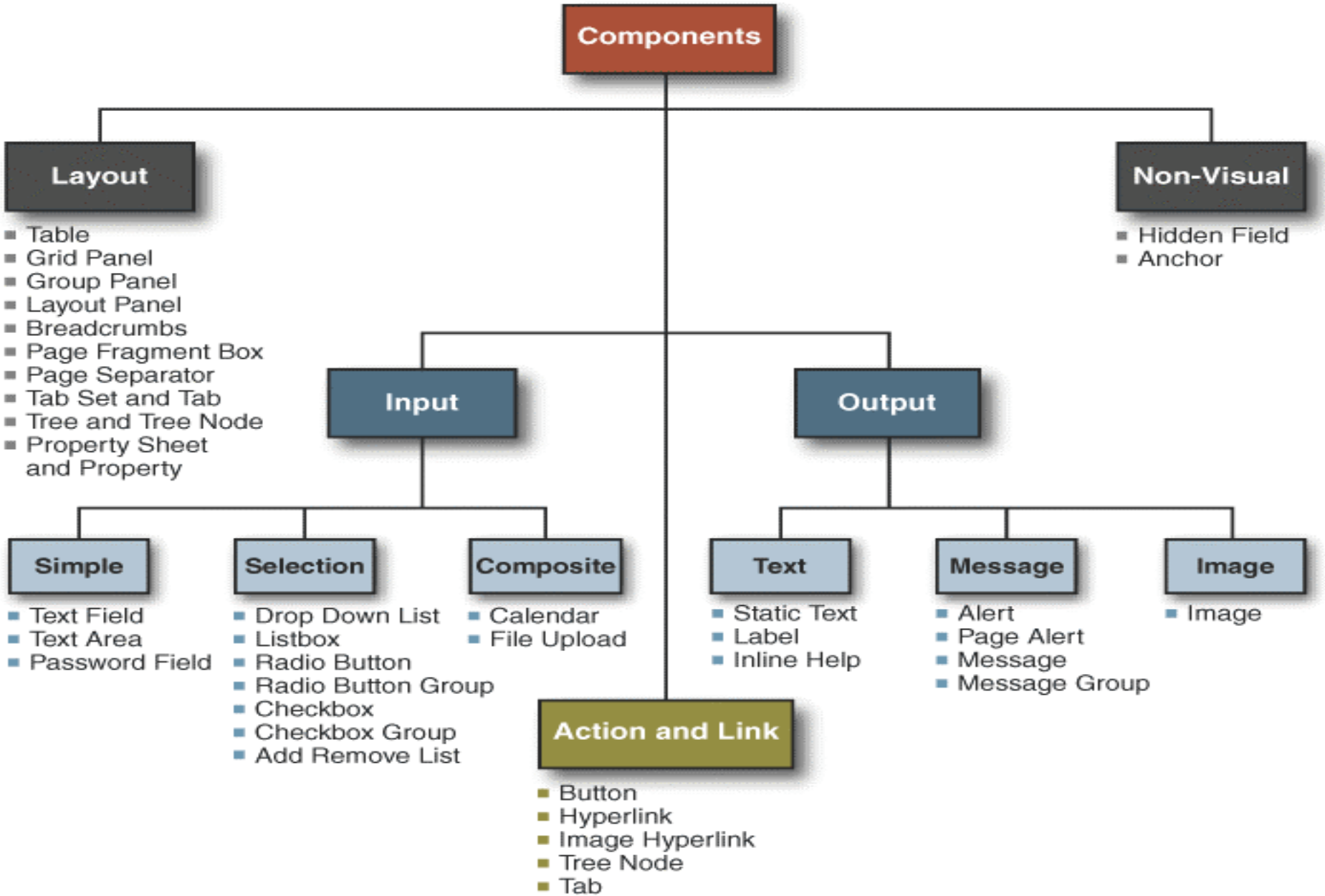


Opis faz cyklu życia JavaServer Faces

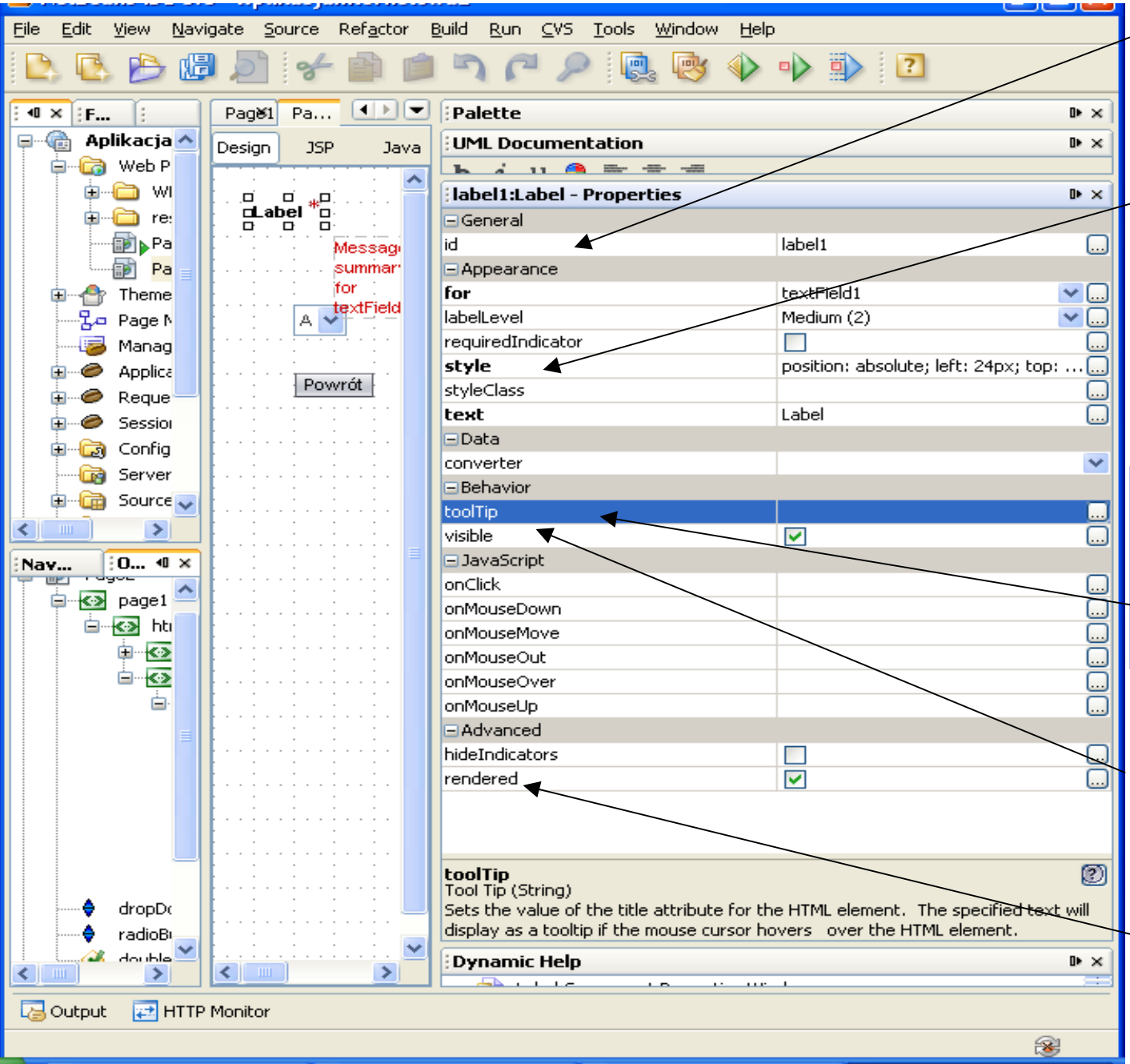
- **Dwa typy cykli życia:**
  - **initial requests** (początkowe wywołanie strony) tylko fazy RestoreView i Render Response
  - **postbacks** (obsługa formularza): wszystkie fazy
- **Akcje Response Complete** oznaczają odwołanie do innej części aplikacji, nie zawierającej komponentów Java Server Faces
- **Typy faz w przypadku pełnego cyklu życia** np. wpłata gotówki
  - **Restore View:** reakcja na zdarzenie wprowadzenia wartości na formularzu w polu typu TextField i wysłanie przez klienta strony (tworzenie **widoku strony** jako drzewa komponentów UI i łączenie ich z walidatorami, konwerterami, zapamiętanie tego widoku w FacesContext dla cyklu typu postback i pustego widoku w przypadku cyklu initial request
  - **Apply Request Values:** konwersja danych i zachowanie ich wartości, wiązanie zdarzeń ze słuchaczami, obsługa walidacji, konwersji i zdarzeń dla wartości chwilowych, możliwość akcji Response Complete, możliwość przejścia do fazy Render Response jako wynik obsługi zdarzeń
  - **Process Validations:** obsługa walidacji, konwersji i zdarzeń, zapamiętanie wartości wprowadzonych w formularzu, możliwość przejścia do fazy Render Response (obsługa błędów lub normalna reakcja) lub możliwość akcji Response Complete
  - **Update Model Values:** konwersja danych, możliwość przejścia do fazy Render Response (obsługa błędów lub normalna reakcja) lub możliwość akcji Response Complete
  - **Invoke Application:** realizacja zdarzeń np. typu submitt dla formularza lub połączenie z inną stroną, możliwość przejścia do fazy Render Response (obsługa błędów lub normalna reakcja) lub możliwość akcji Response Complete
  - **Render Response:** ustalenie zawartości strony w przypadku cyklu typu postback (komunikaty jako normalna reakcja lub komunikaty o błędach), ustalenie nowej zawartości kontenera JSP zawierającego drzewo komponentów UI – przebudowanie istniejącego **widoku strony** utworzonego podczas Restore View dla cyklu typu postbacks

Paleta komponentów UI w Visual Web Pack -technologia Java Server Faces

Drzewo komponentów w pakiecie Visual Web Pack



Formularz z właściwościami komponentu UI



identyfikator komponentu

możliwość zdefiniowania kaskadowego arkusza stylów: Font, Background, Text Block,Border, Margin, Position

napis, który ukazuje się na komponencie po najechaniu kursorem myszy

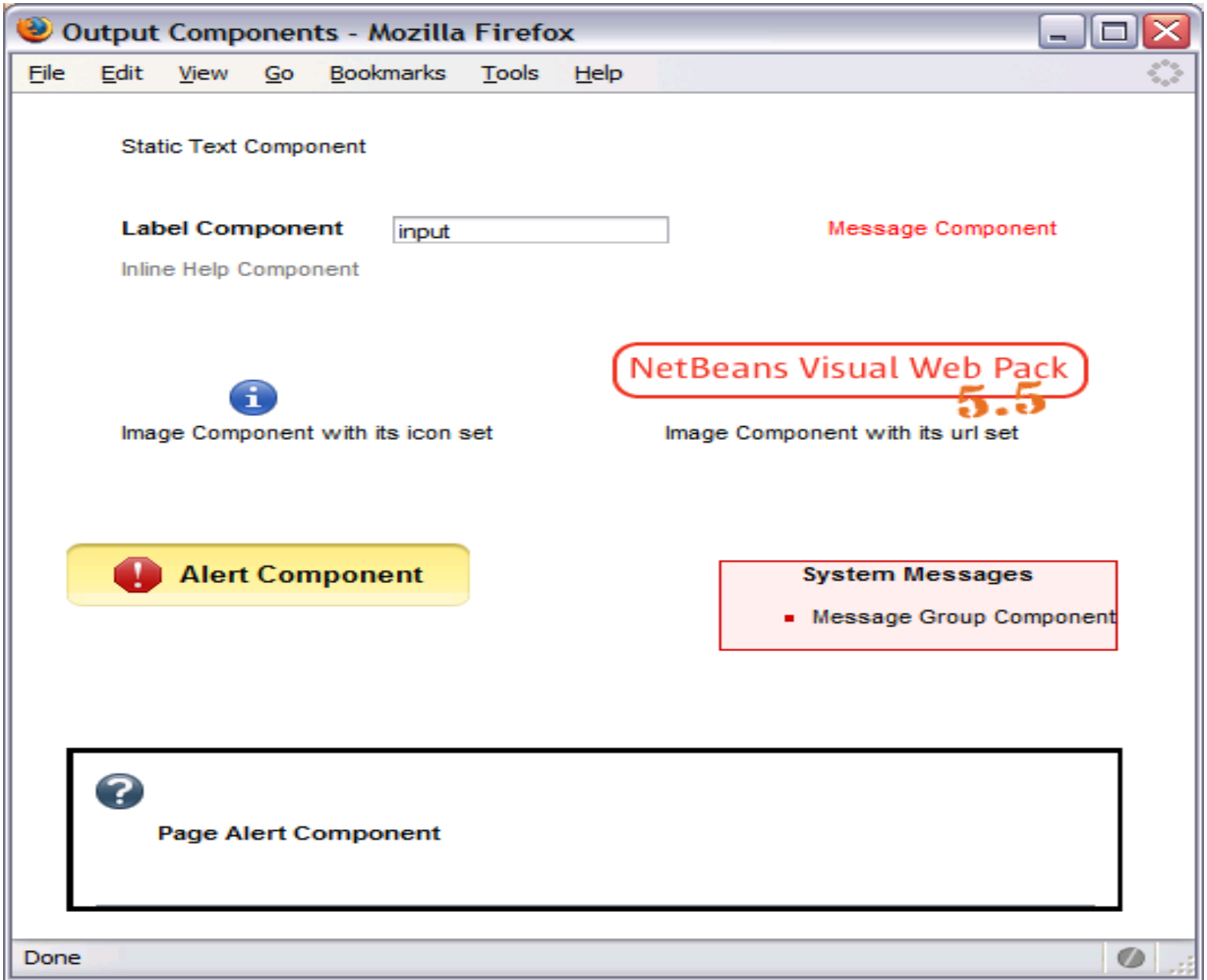
możliwość ukrywania komponentu, lecz dane komponentu są dostępne

możliwość wyłączenia dostępu do komponentu

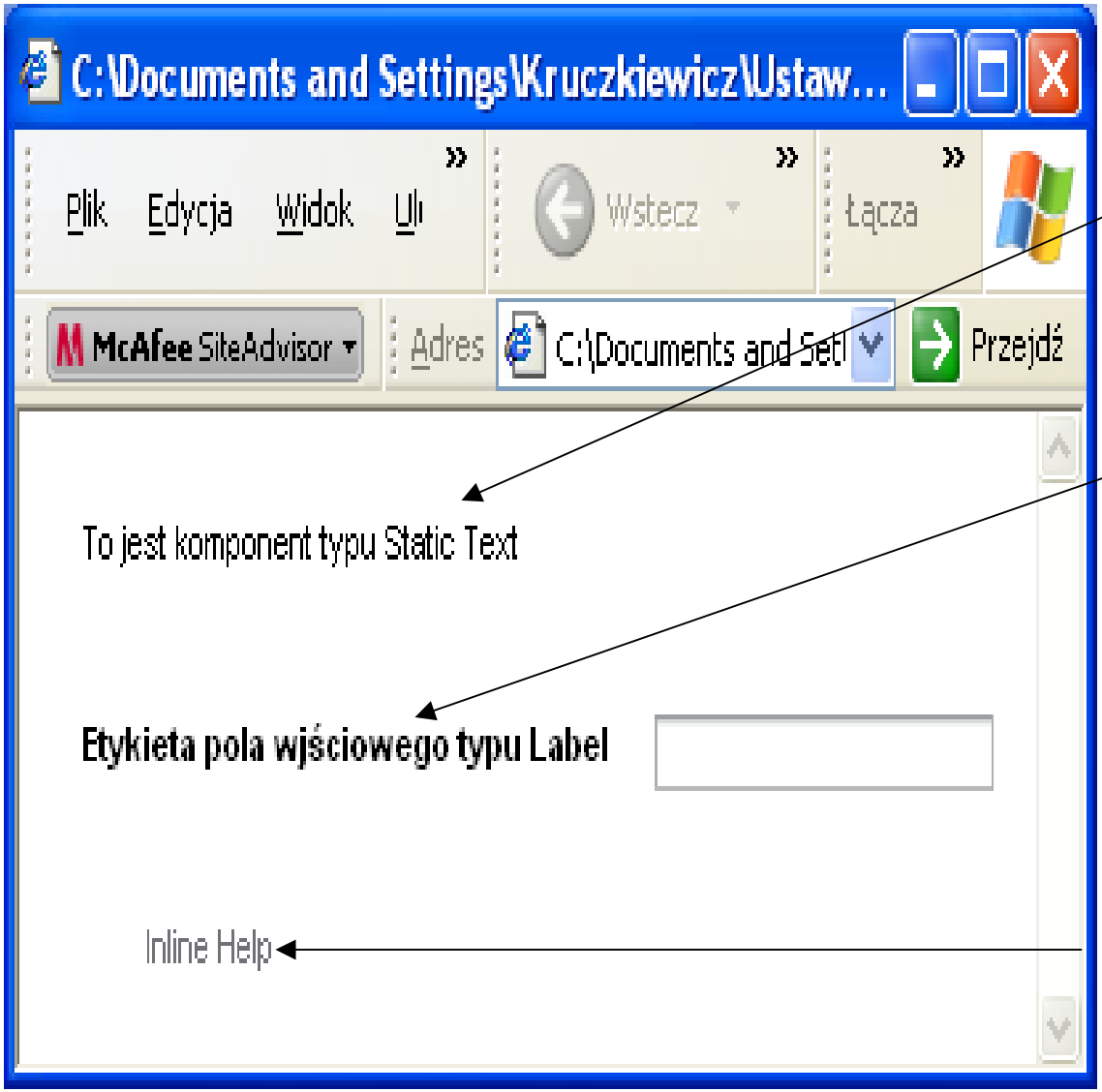
Ustalanie właściwości komponentu UI

1. Ustawianie wartości atrybutów komponentu za pomocą okna properties i bezpośrednie wpisanie wartości lub korzystanie z edytora (...)
2. Wywołanie edytora kodu źródłowego metod: preprocess, prerender, value change i action
3. Związanie komponentu albo z „providerem” danych albo obiektem aplikacji

1) Komponenty UI wyjściowe



1.1) Tekstowe komponenty wyjściowe

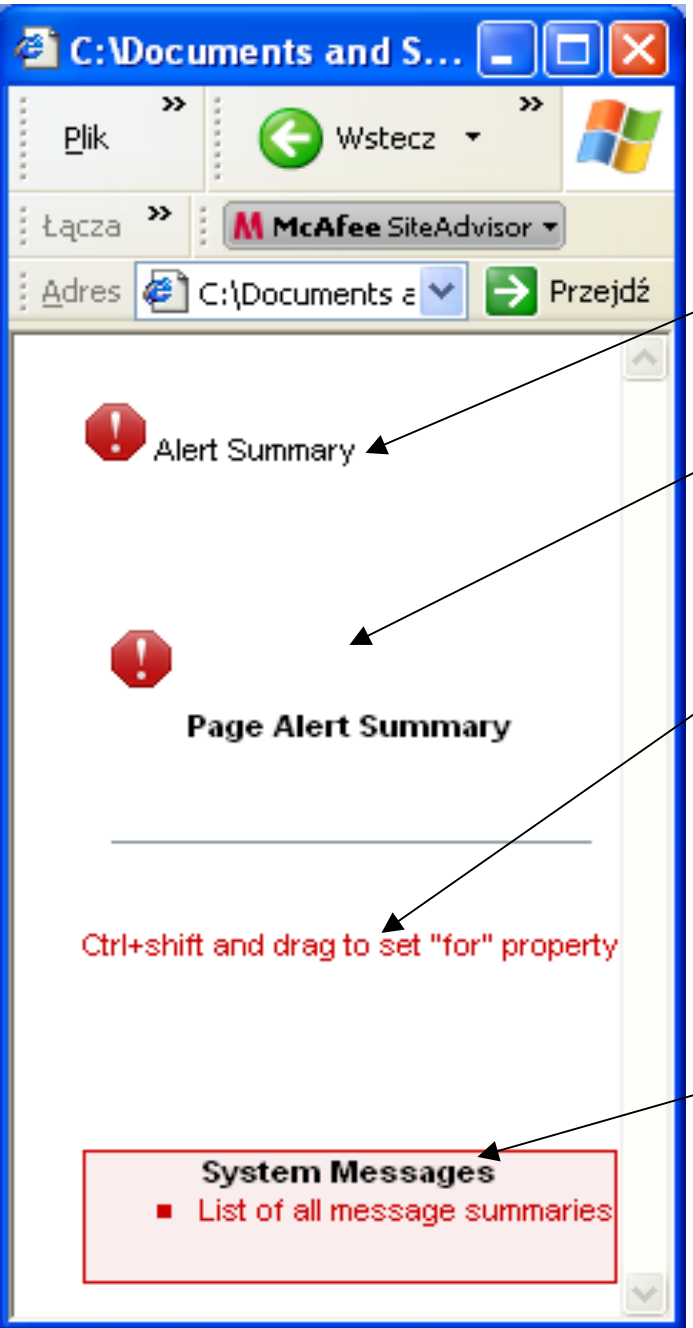


Static Text :Wyświetlanie tekstu

Label: Etykieta opisuje komponenty wejściowych (k.w.). Jej pole for pozwala ją związać z k.w.-wtedy mają wspólne własności graficzne. Po ustawieniu pola required na true w k.w. do etykiety dołączona jest *

Inline Help: Wyświetlanie krótkiej pomocy na stronie

1.2) Komponenty komunikatów wyjściowych



Alert Summary: Wyświetlanie komunikatów o błędach

type: typ ikony

summary: jeśli nadano tekst, wtedy ukaże się komunikat

detail:ustawiona wartość tego pola uniemożliwia korzystanie z linków

Ustawienie połączenia:

1) Należy ustawić pole linkText

2) Należy ustawić pole linkURL: za pomocą Page Navigation należy wykonać połączenie z wybraną stroną i nadać nazwę połączeniu np..alertOutcome

3) obsługa akcji: w trybie Visual Designer należy kliknąć dwukrotnie na komponent, po przejściu do trybu Java i napisać kod metody action, instrukcja return tej metody powinna zwracać wartość alertOutcome

Page Alert Summary: Wyświetlanie komunikatów o błędach na wybranej stronie - ustawianie własności podobne do Alert Summary

Message: Wyświetlanie komunikatów o błędach pochodzących od komponentu powiązanego z Message przez atrybut *for*

Własne komunikaty o błędach przez wywołanie wyjątków

ValidatorException(new FacesMessage(summary)).

ValidatorException(new FacesMessage(summary, detail)).

ValidatorException(new FacesMessage(severity, summary, detail)). Wartość atrybutu severity: FacesMessage.SEVERITY_INFO, FacesMessage.SEVERITY_WARN, FacesMessage.SEVERITY_ERROR, or FacesMessage.SEVERITY_FATAL.

Obsługa własna błędów przez przesłonięcie metod: requiredMessage, converterMessage i validatorMessage

Message Group: Wyświetlanie komunikatów o błędach typu runtime

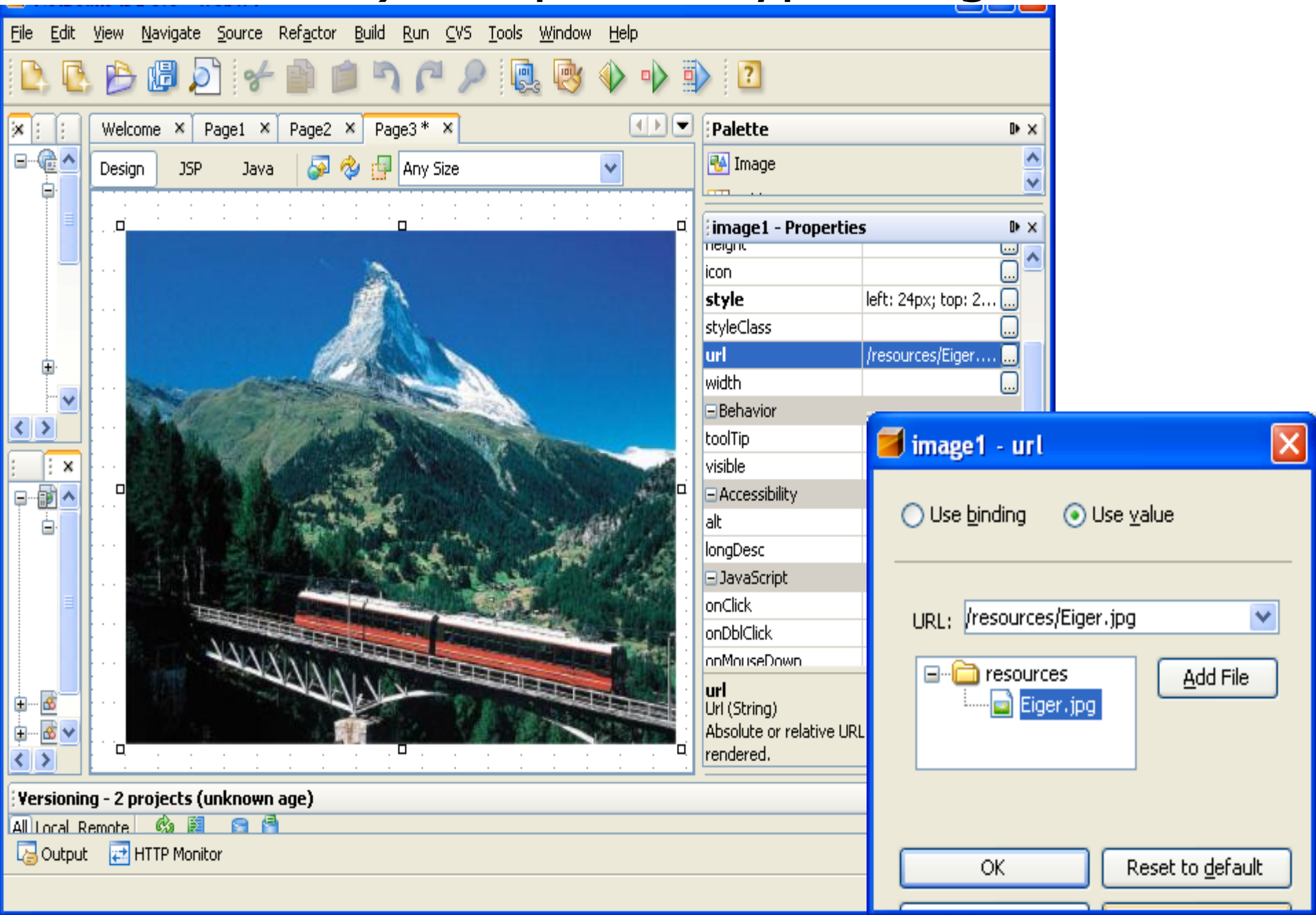
Własna obsługa błędów:

FacesContext.getCurrentInstance().addMessage(null, new FacesMessage(summary, detail))
lub
ValidatorException(new FacesMessage(severity, summary, detail));

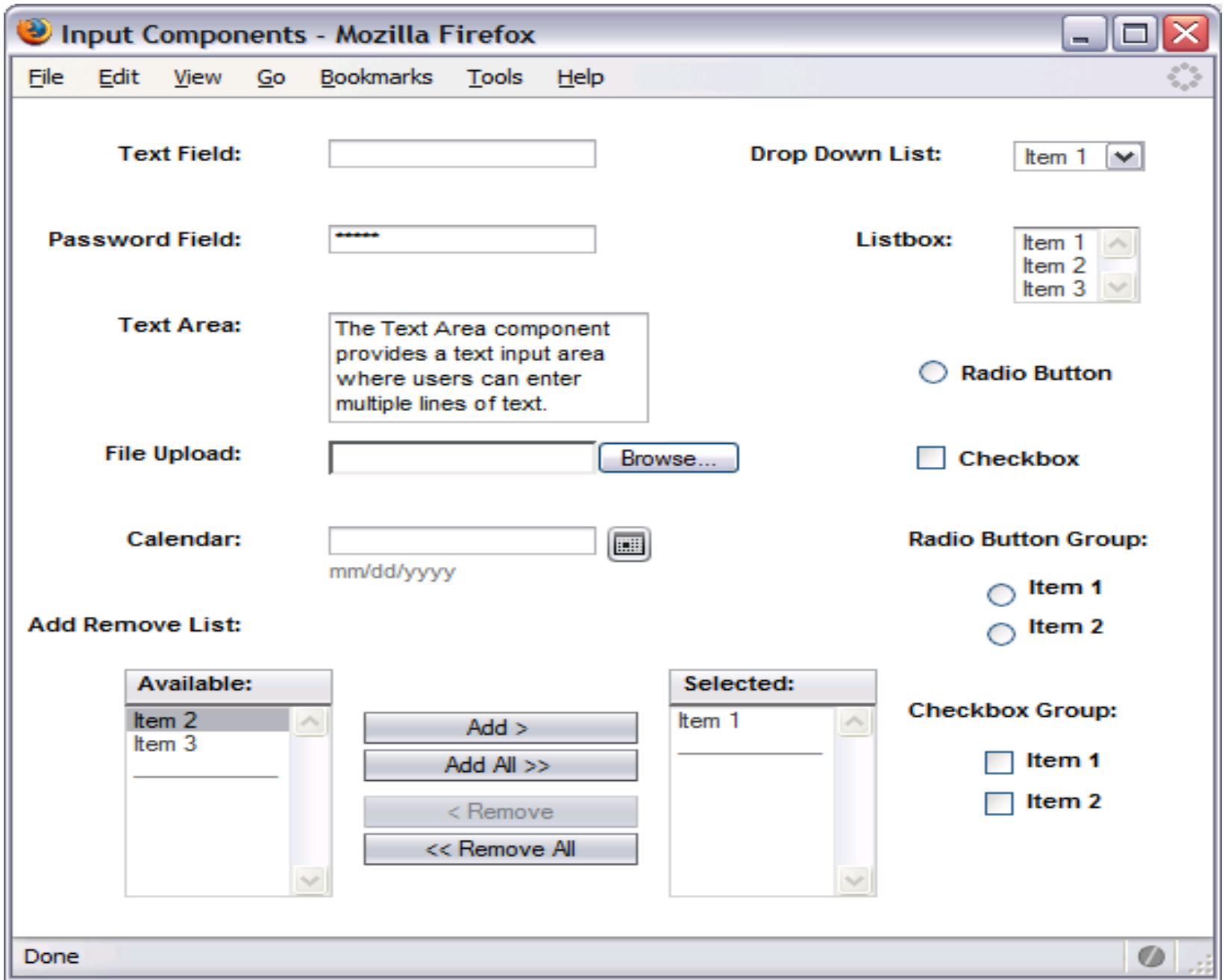
showDetail: ustawienie wartości (typ checkbox)

Tekst własnych komunikatów po wywołaniu: info(String summary), warn(String summary), error(String summary), lub fatal(String summary).

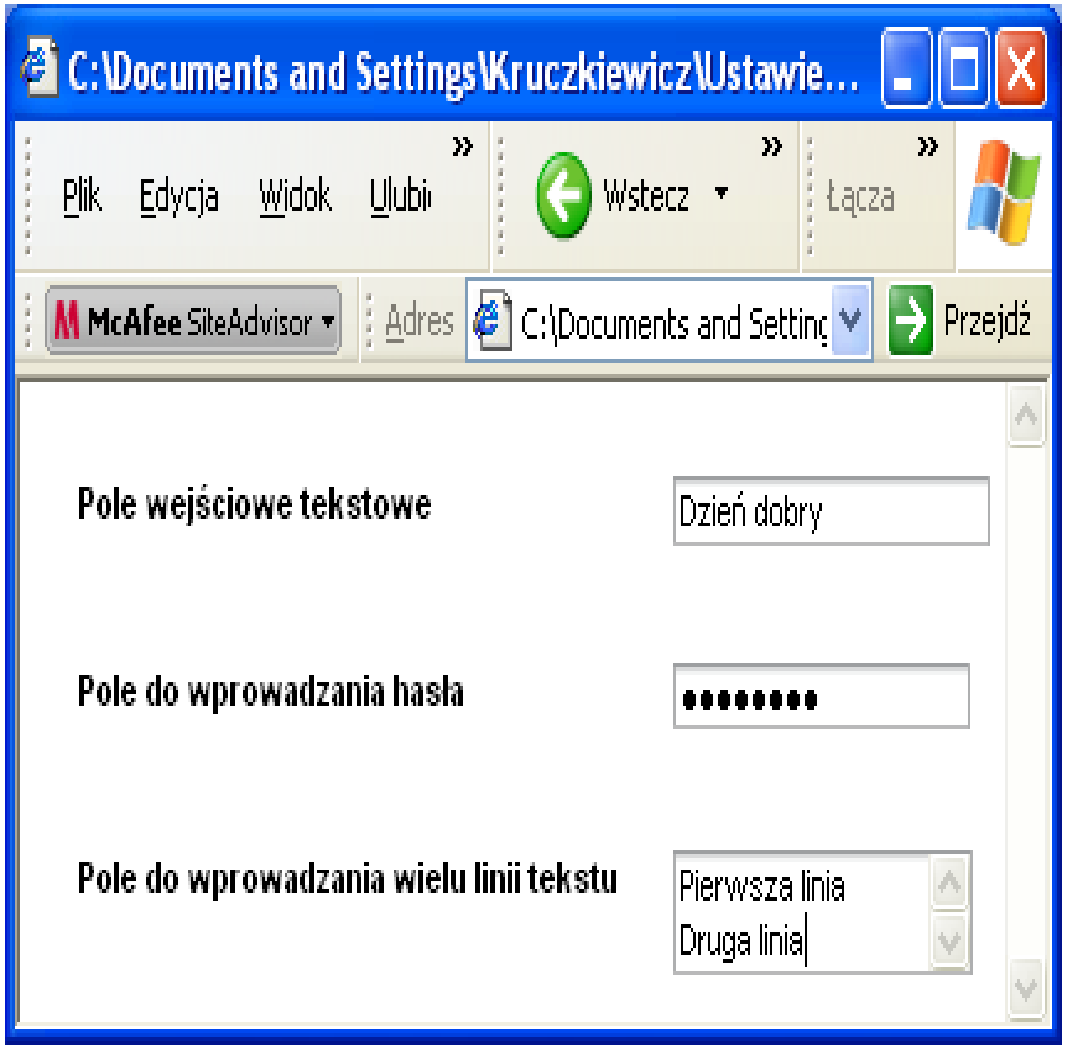
1.3) Komponent typu image



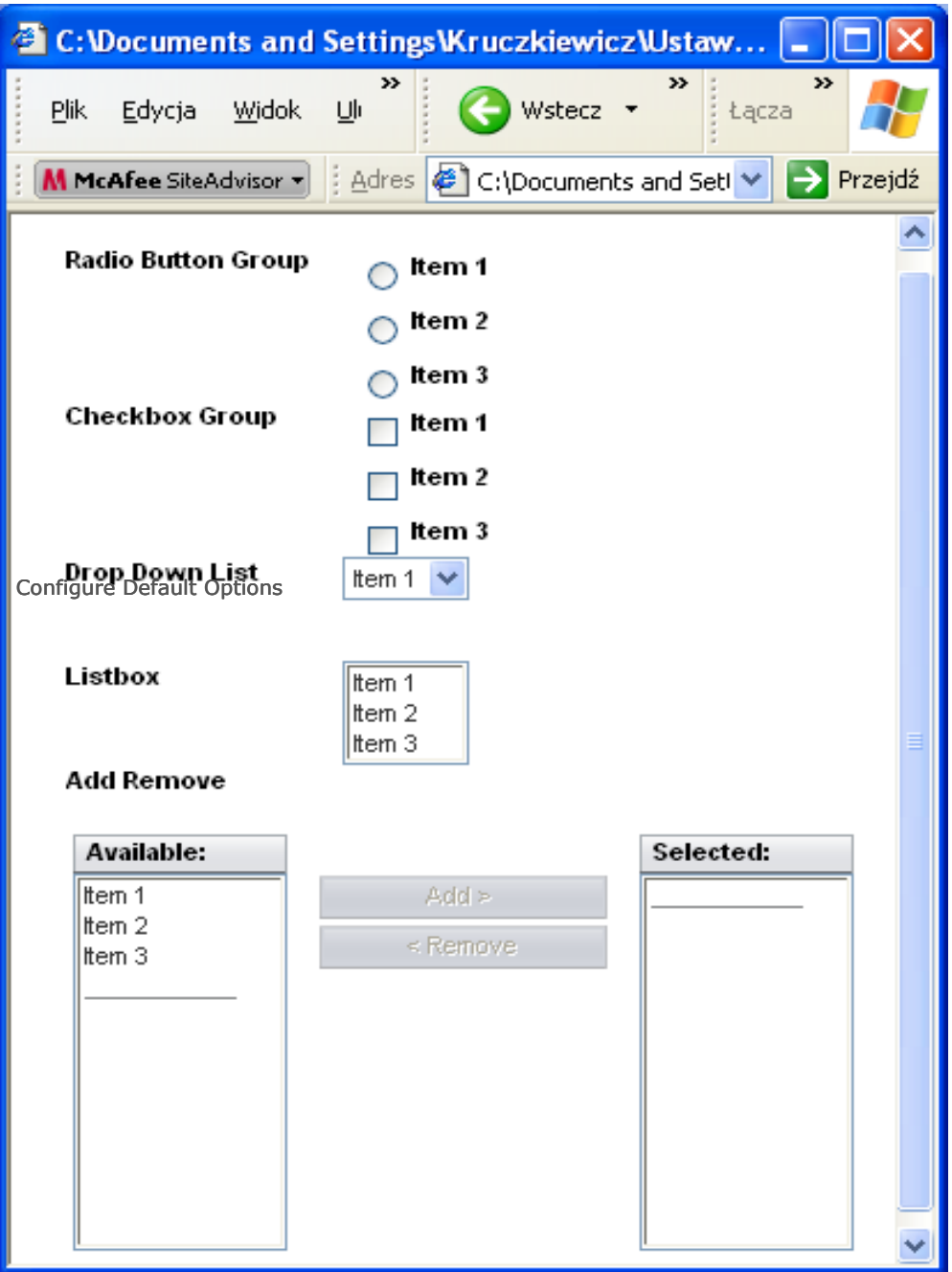
2) Komponenty UI wejściowe



2.1) Komponenty tekstowe wejściowe



2.2) Komponenty wejściowe wielu wyborów



1)Łączenie pola item z danymi:

a) wywołanie edytora Configure Default Options (po kliknięciu prawym klawiszem myszy na komponencie)

b) lub napisanie metody init

```
dropDown1DefaultOptions.setOptions
(new Option[]
 {new Option("item1", "Item 1"),
  new Option("item2", "Item 2"),
  new Option("item3", "Item 3")}
);
```

2) Łączenie pola items z tabelą

a)Przeciągnięcie wybranej tabeli na komponent

b)Po kliknięciu prawym klawiszem myszy wybór edytora Bind To Data; wybór provider'a oraz kolumny dla return value i display value

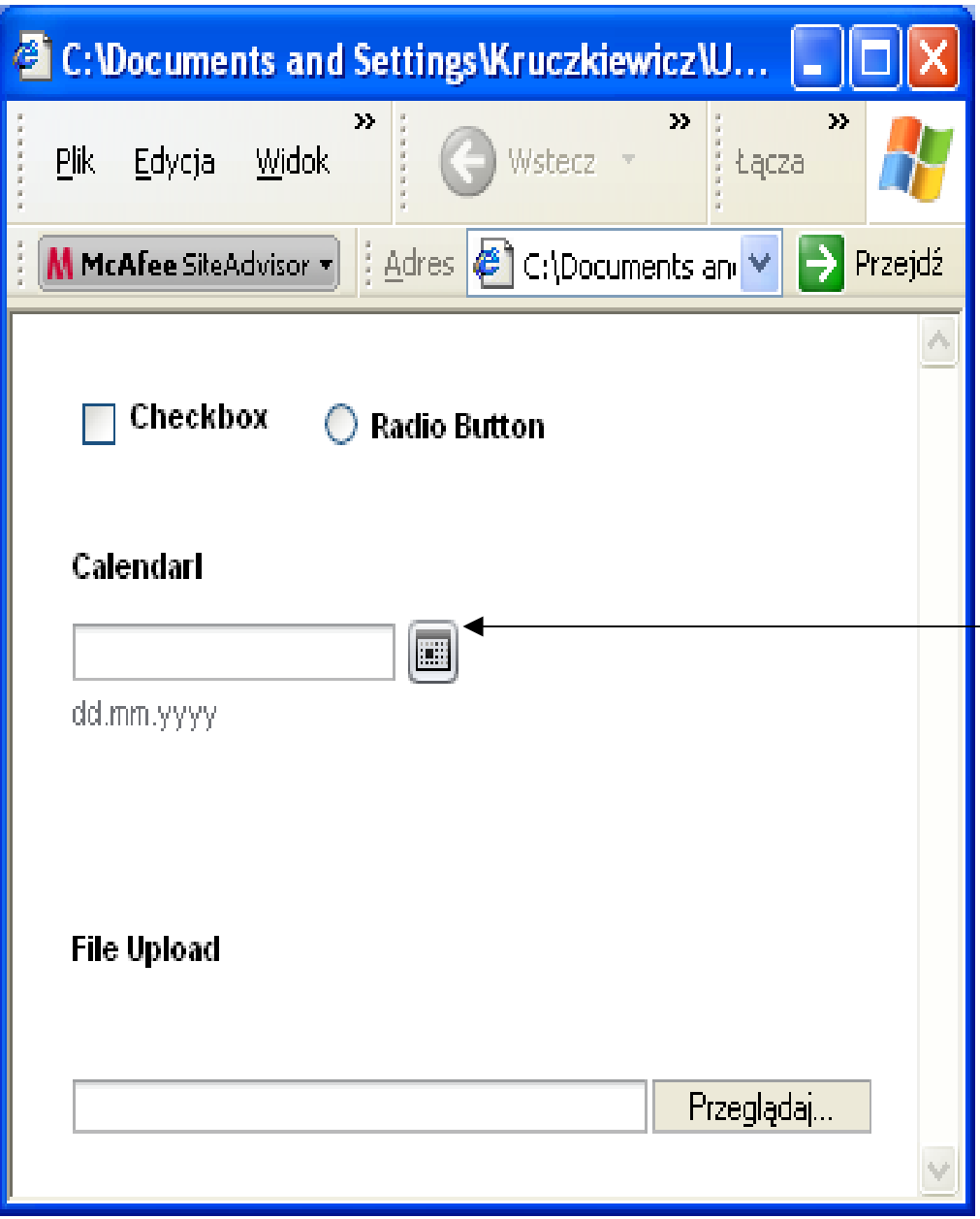
3) Łączenie pola items z tablicą com.sun.webui.jsf.model.Option

Po kliknięciu prawym klawiszem myszy wybór edytora Bind To Data; wybór obiektu skojarzonego z komponentem

4) Inicjowanie pola items poza edytorem Configure Default Options

W metodzie prerender() metodami setSelected(Object[]) lub setSelected(Object) ustawia się pola items, jeśli metoda getSelected() zwraca null

2.3) Komponenty wejściowe pojedynczego wyboru



Calendar: wprowadzanie daty

1) Pole selectedDate jest skojarzone z java.util.Date

2) Pola minDate i maxDate kojarzy się z odpowiednimi polami obiektu java.util.Date

Przykład ustawienia minimalnej daty jako daty bieżącej.

```
private Date minCalDate;
 public Date getMinCalDate()
   { java.util.Calendar date =
      java.util.Calendar.getInstance(
        FacesContext.getCurrentInstance().
          getViewRoot().getLocale());
  // Have to zero out the time because
  // the date comparison is time sensitive
   date.set(java.util.Calendar.HOUR_OF_DAY,0);
   date.set(java.util.Calendar.MINUTE, 0);
   date.set(java.util.Calendar.SECOND, 0);
   date.set(java.util.Calendar.MILLISECOND, 0);
   return date.getTime(); }
```

3) Ustawienie wzorca wprowadzania daty za pomocą edytora dateFormatPattern

4) Ustawienie podpowiedzi wzorca daty za pomocą pola dateFormatPatternHelp

Ustawianie własności komponentów wejściowych

1. Pole Label komponentu wejściowego pozwala identyfikować komponent wejściowy przy walidacji, konwersji itp. lub użycie komponentu Label skojarzonego z komponentem wejściowym
2. Wyświetlanie komunikatów o błędach przez skojarzenie komponentu Message przez pole for
3. Uniemożliwienie wprowadzania danych i ustawienie tylko do odczytu: disabled=true i readOnly =true
4. Ustawienie kolejności wybierania komponentów wejściowych: właściwość tabIndex pozwala określić tę kolejność
5. Usuwanie białych znaków: ustawienie pola trim =true
6. Ograniczenie liczby wprowadzanych znaków: maxLength
7. Użycie komponentów wejściowych do zatwierdzania przesłania danych ze strony.

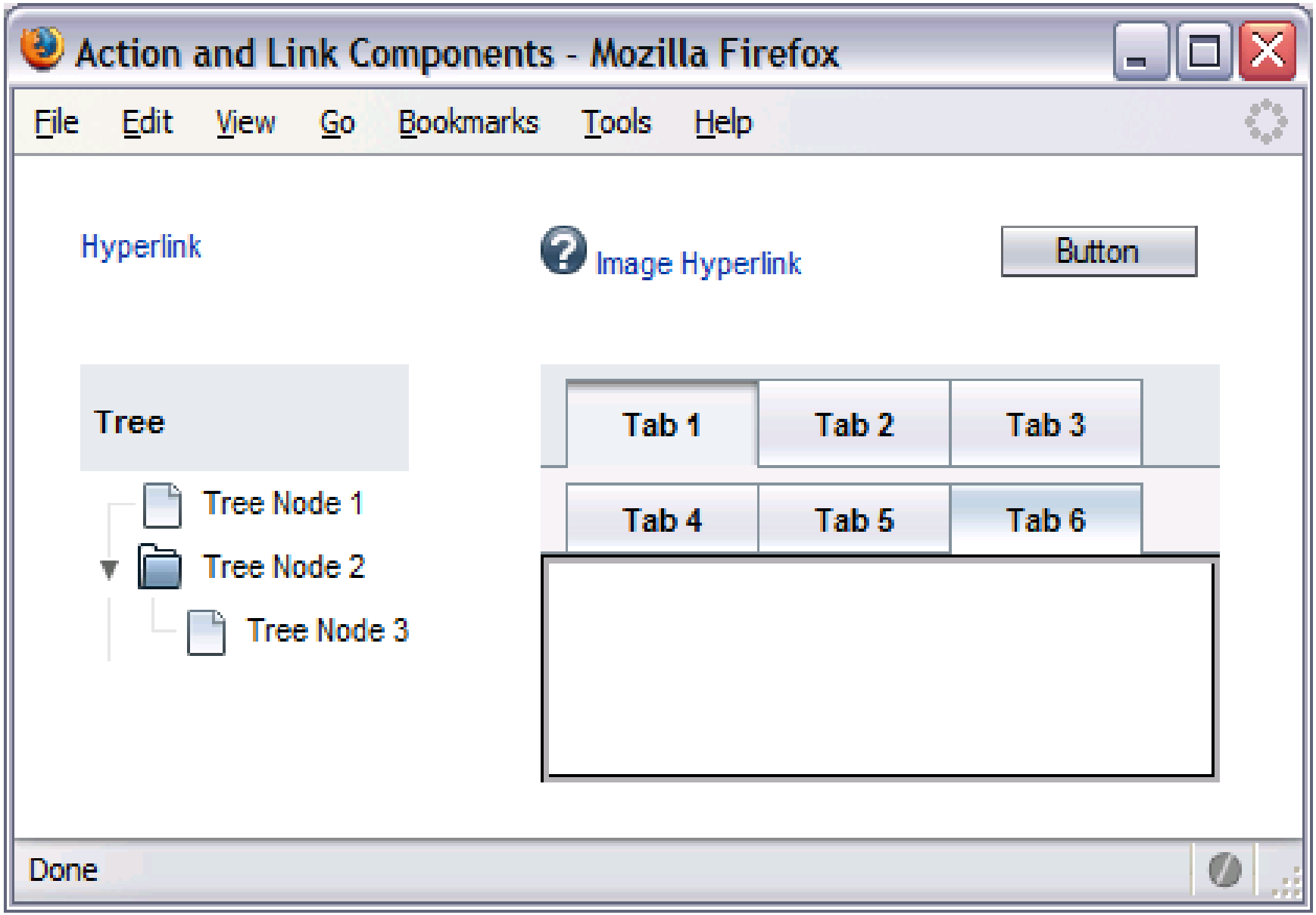
Rodzaj danych wprowadzanych do komponentów wejściowych

Komponent	Rodzaj wartości	Typ
Text Field, Text Area, Password Field	**text**	String
Drop Down List	**selected**	domyślnie (Default Options component) String
Listbox	**selected**	domyślnie (Default Options component): albo wynik typu String lub tablicę elementów typu String (pole multiply wybrane). Można zdefiniować inny typ wyniku
Radio Button (not in a group)	**selected**	Domyślnie boolean. Można jednak nadać inny typ: Boolean, Byte, Character, Double, Float, Integer, Long, Short, String, lub obiekty programisty
Radio Button Group	**selected**	Domyślnie (Default Options component) String. Można pole items zdefiniować jako inny typ danej- będzie to uwidocznione na etykiecie
Checkbox (not in a group)	**selected**	Domyślnie boolean. Można jednak nadać inny typ: Boolean, Byte, Character, Double, Float, Integer, Long, Short, String, lub obiekty programisty
Checkbox Group	**selected**	Domyślnie (Default Options component) String. Można pole items zdefiniować jako inny typ danej- będzie to uwidocznione na etykiecie
Calendar	selectedDate	java.util.Date
File Upload	uploadedFile	com.sun.rave.web.ui.model.UploadedFile
Add Remove List	**selected**	An array of Object

Konwersja, walidacja danych wejściowych i obsługa zdarzeń

1. Ustawienie obowiązku wprowadzania wartości: required-true
2. Konwersja wartości, która zawsze poprzedza walidację: skojarzenie z odpowiednim komponentem typu konwerter np. Number Converter.
3. Walidacja danych: skojarzenie z odpowiednim komponentem typu walidator np. Double Range Validator, Length Validator, Long Range Validator
4. Walidacja i konwersja już w fazie Apply Request Values, gdy immediate=true
5. Kolejność obsługi zdarzeń
   - W pierwszej kolejności następuje obsługa zdarzeń komponentów z polem immediate=true
   - Następnie obsługiwane są zdarzenia typu zmiany wartości w komponentach z polem immediate=false
   - Na końcu obsługiwane są zdarzenia typu akcje dla komponentów z polem immediate=false

3) Komponenty akcji i połączeń



Dwa sposoby wstawiania połączenia

- **Prosta nawigacja**

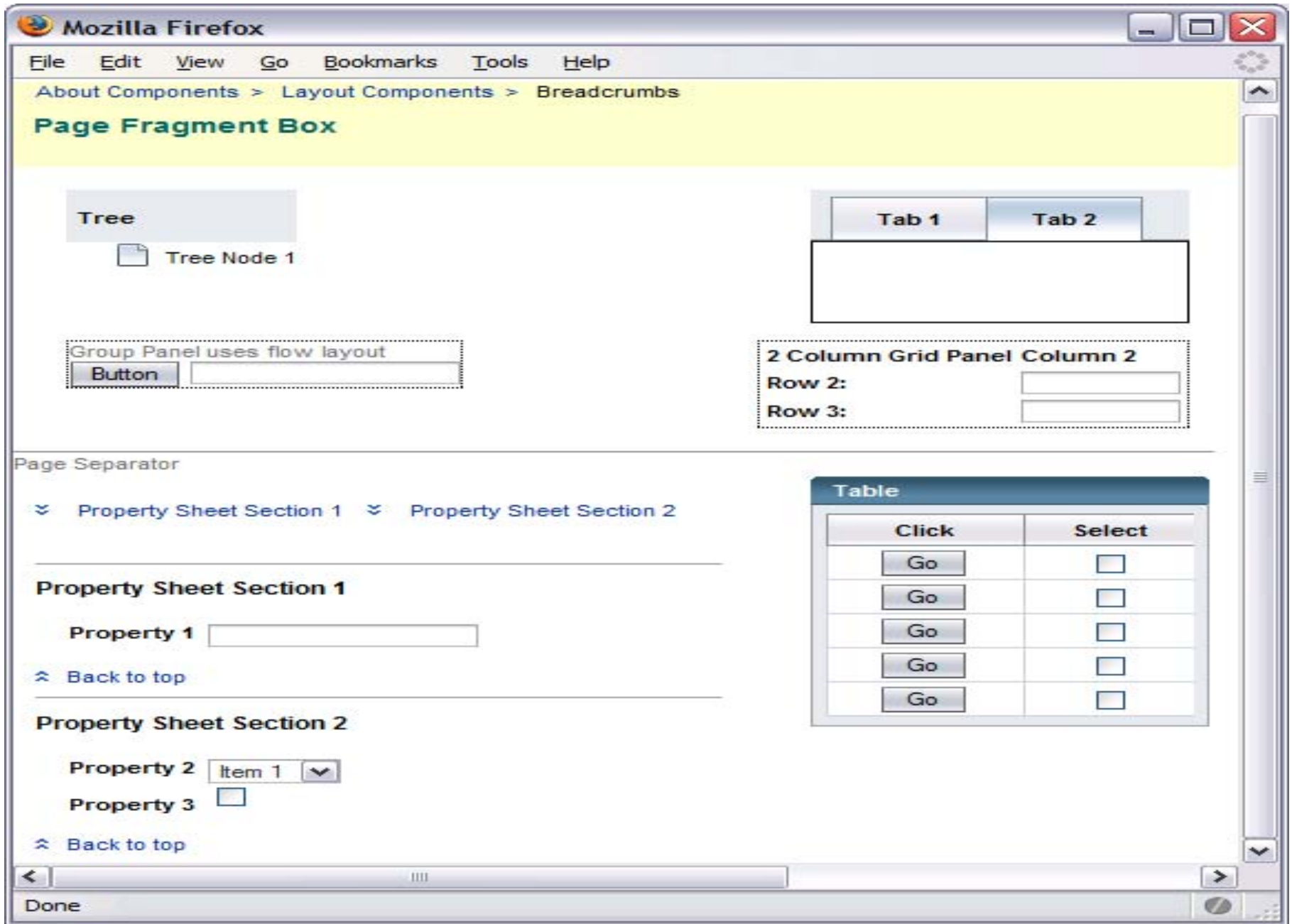
Zastosowanie pola url do wstawiania adresu

URL-obsługa połączenia bez obsługi zdarzeń

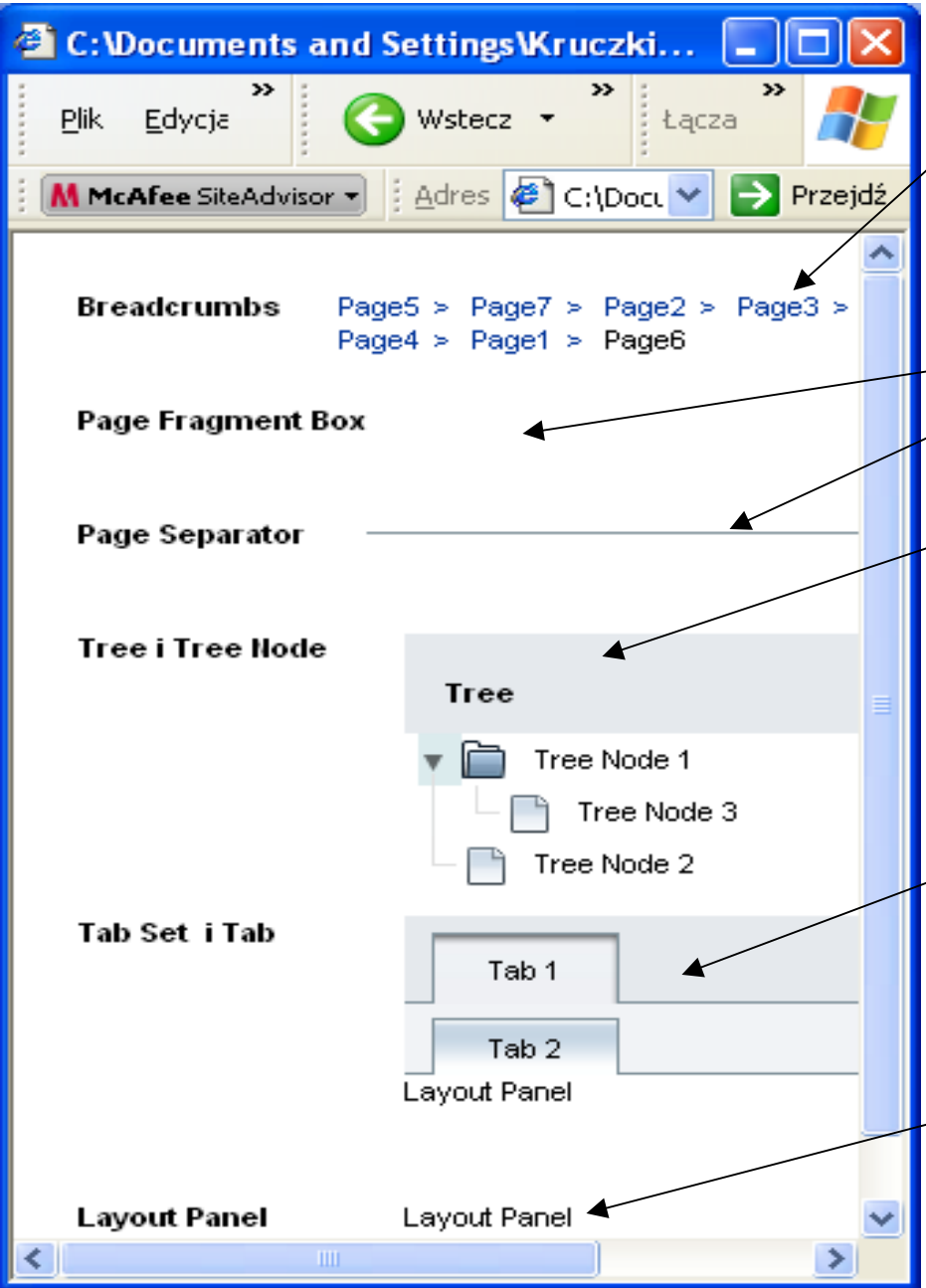
- **Dynamiczna nawigacja**

Zastosowanie edytora Page Navigation pozwala wyspecyfikować reguły nawigacji odwzorowujące akcje i decyzje przy nawigacji między stronami. W edytorze należy wybrać komponent łączący i przeciągnąć go do strony wyjściowej oraz nadać nazwę połączeniu. Należy również napisać kod metody **action**, która reprezentuje akcje towarzyszące połączeniu. Instrukcja **return** tej metody zwraca nazwę połączenia

4) Komponenty typu Layout



4.1) Komponenty typu Layout



Breadcrumbs: Grupa komponentów łączących ze stronami za pomocą zbioru automatycznie dołączonego zbioru komponentów typu **Hyperlink**. Alternatywą może być tablica komponentów typu **Hyperlink**. Każdy Hyperlink musi mieć nadaną właściwość typu text i url lub action

Page Fragment Box: Nadanie takiego samego kontekstu kilku stronom

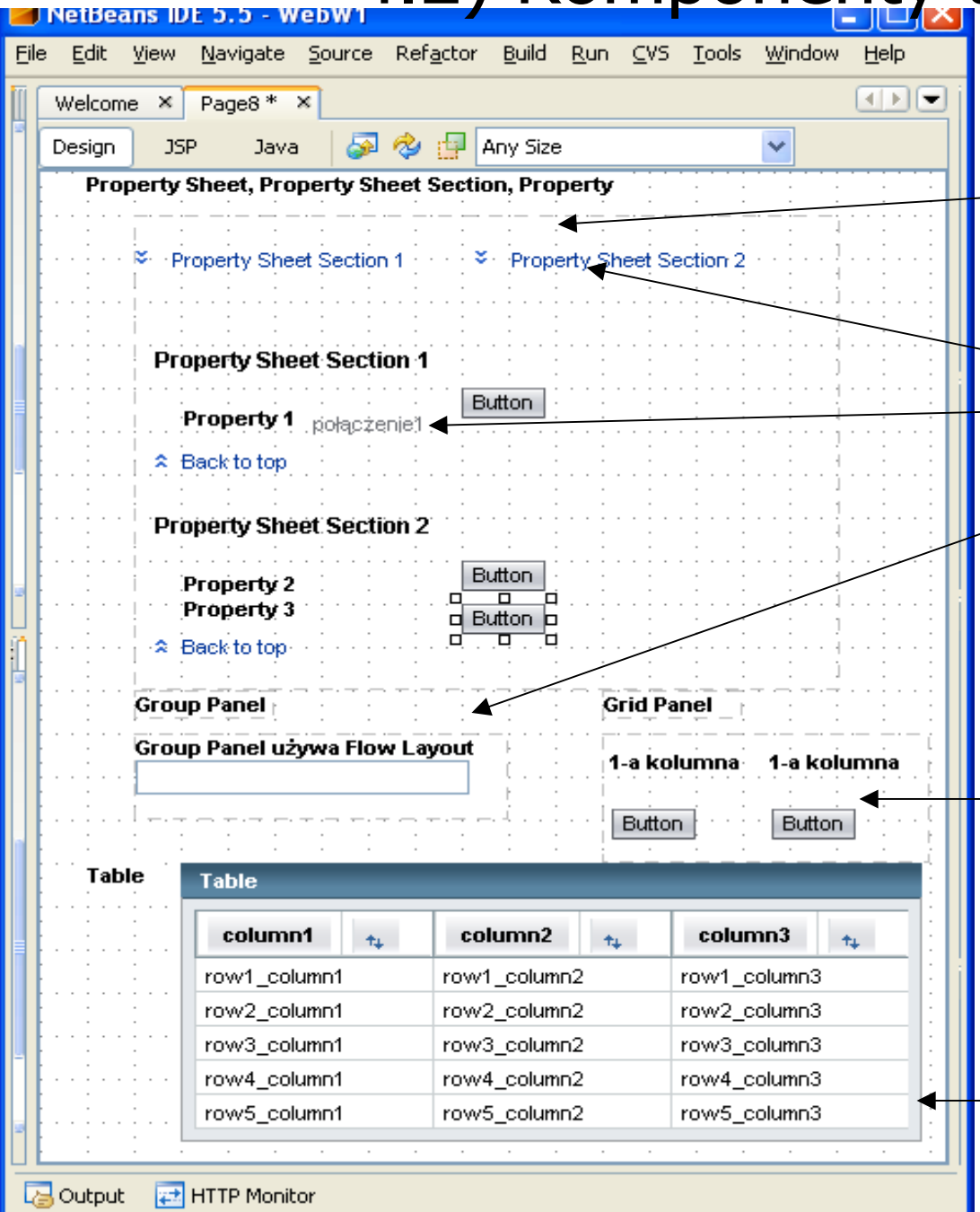
Page Separator: Separator oddzielający komponenty na stronie

Tree, Tree Node: Komponent używany do tworzenia hierarchii połączeń lub utworzenia hierarchicznego dostępu do plików. Pole **action** pozwala ustalić obsługę zdarzeń wyboru węzła drzewa (Tree Node). Pole **actionListener** pozwala ustalić obsługę zamykania i otwierania węzła (Tree Node).

Tab Set, Tab: Komponenty używane do nawigacji (opcje nawigacji są realizowane przez narzędzia nawigacji – np.. Page Navigation) lub prezentowania różnych Layout na stronie

Layout Panel: Nadania pewnemu obszarowi własności **flow** (komponenty są rozmieszczane jeden za drugim, w kolejnych wierszach) lub **grid** (komponenty są umieszczane tam, gdzie są położone)

4.2) Komponenty typu Layout



Property Sheet, Property Sheet Section, Property: komponenty pozwalają na szybkie rozmieszczenie zbiorów komponentów strony nadając im etykiety –tutaj jako Property (ustawiono pole **jumplink** w **Property Sheet ;** ustawiono pole **helpText** w **Property**)

Group Panel: Komponent używany jest do rozmieszczania komponentów w trybie **flow** - jeden za drugim w kolejnym wierszu zgodnie z rozmiarami komponentów. Przy zmianie rozmiaru okna komponenty zmieniają swoje położenie

Grid Panel: Komponent używany do rozmieszczania komponentów w wierszach i kolumnach. Pole **columns** pozwala określenie liczby kolumn wierszu. Komponenty umieszczane są w kolejnych kolumnach w wierszu. Kolejny wiersz jest dodawany, jeśli zachodzi taka konieczność

Table: Komponent używany do tabelarycznej prezentacji danych.