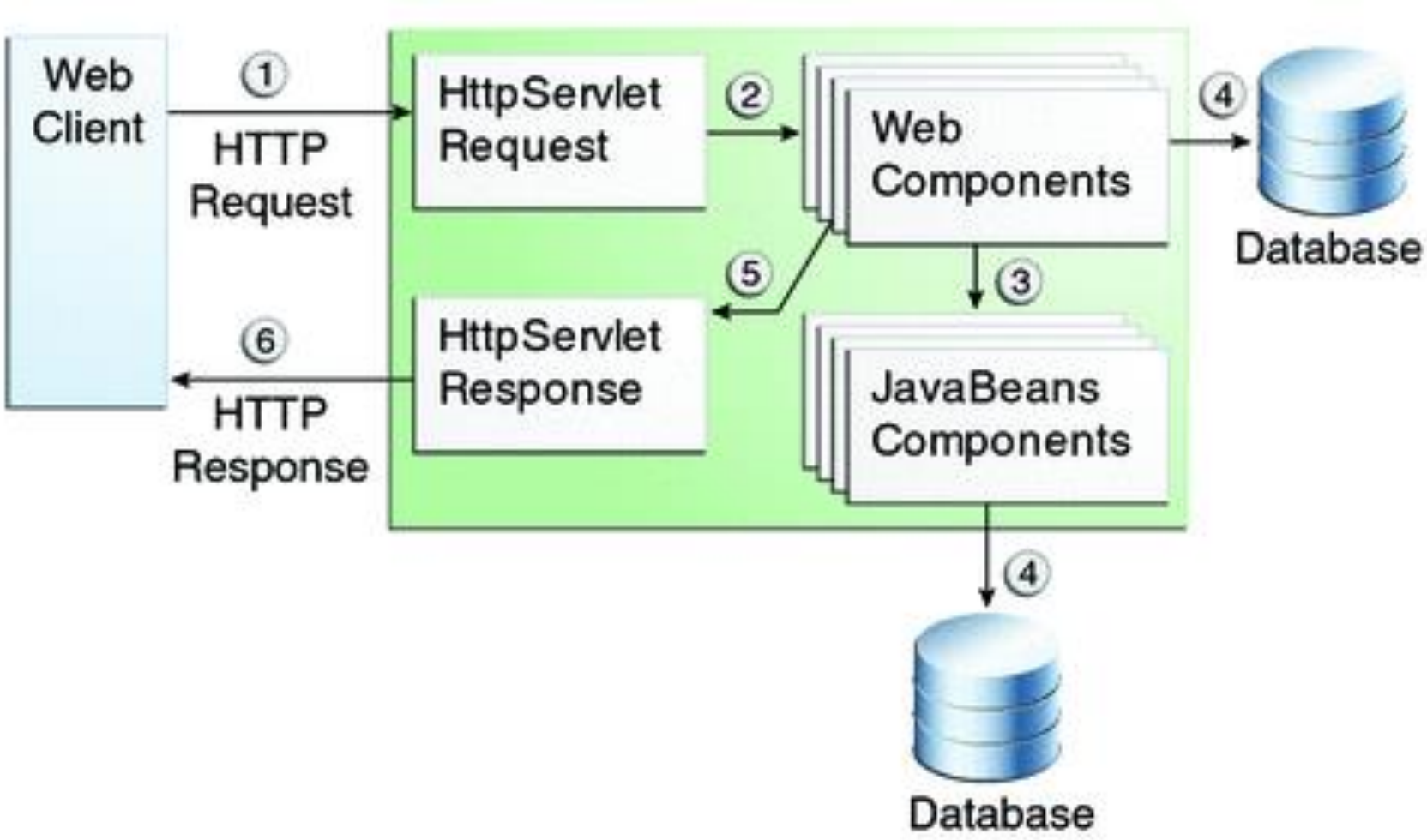
Cykl życia aplikacji JSF

wg

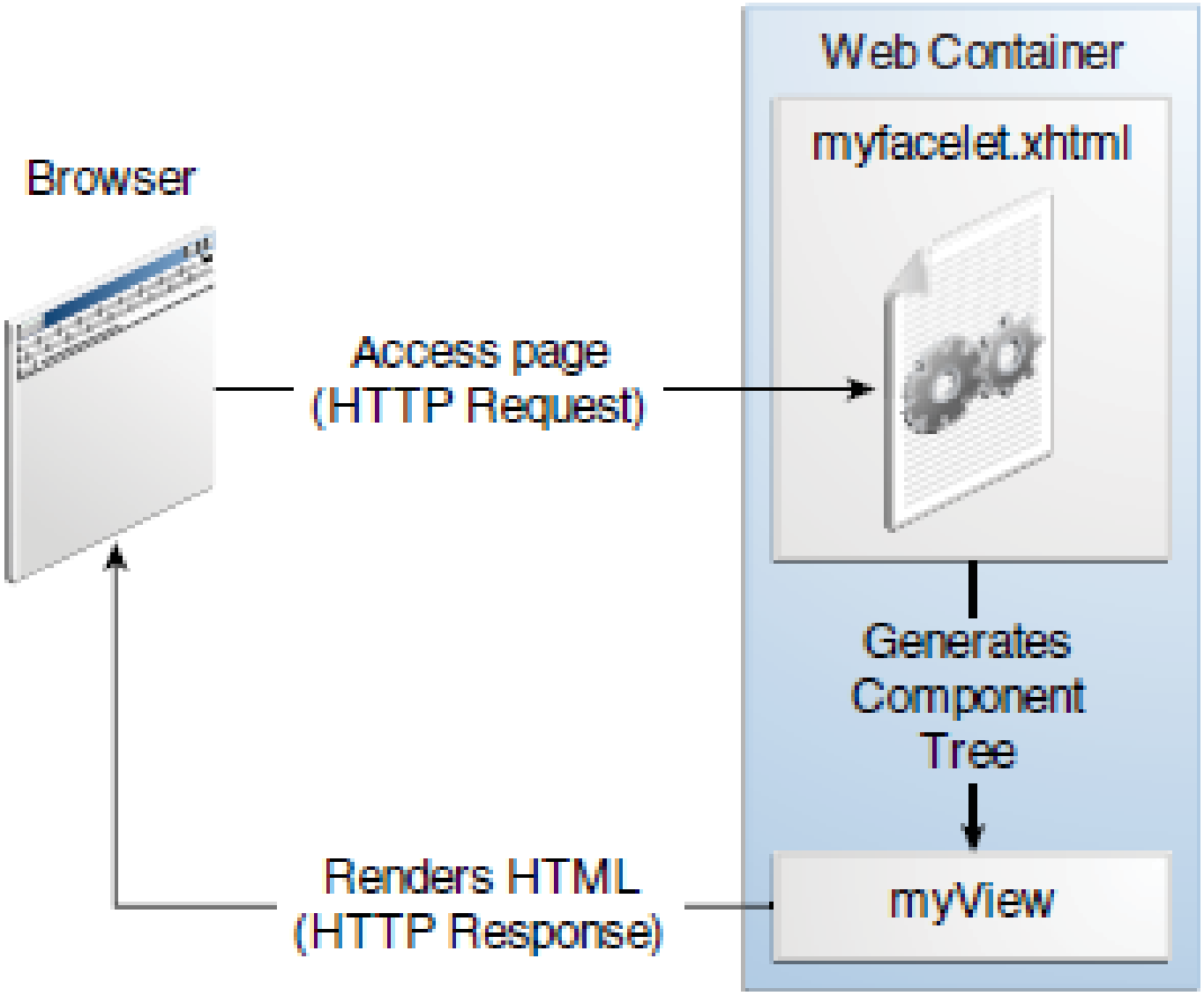
https://docs.oracle.com/javaee/7/JEETT.pdf

Programowanie komponentowe 3_1

Standard cyklu życia „Request-Response” dla JavaServer Faces



Proces „żądanie-odpowiedź” w technologii JavaServer Faces



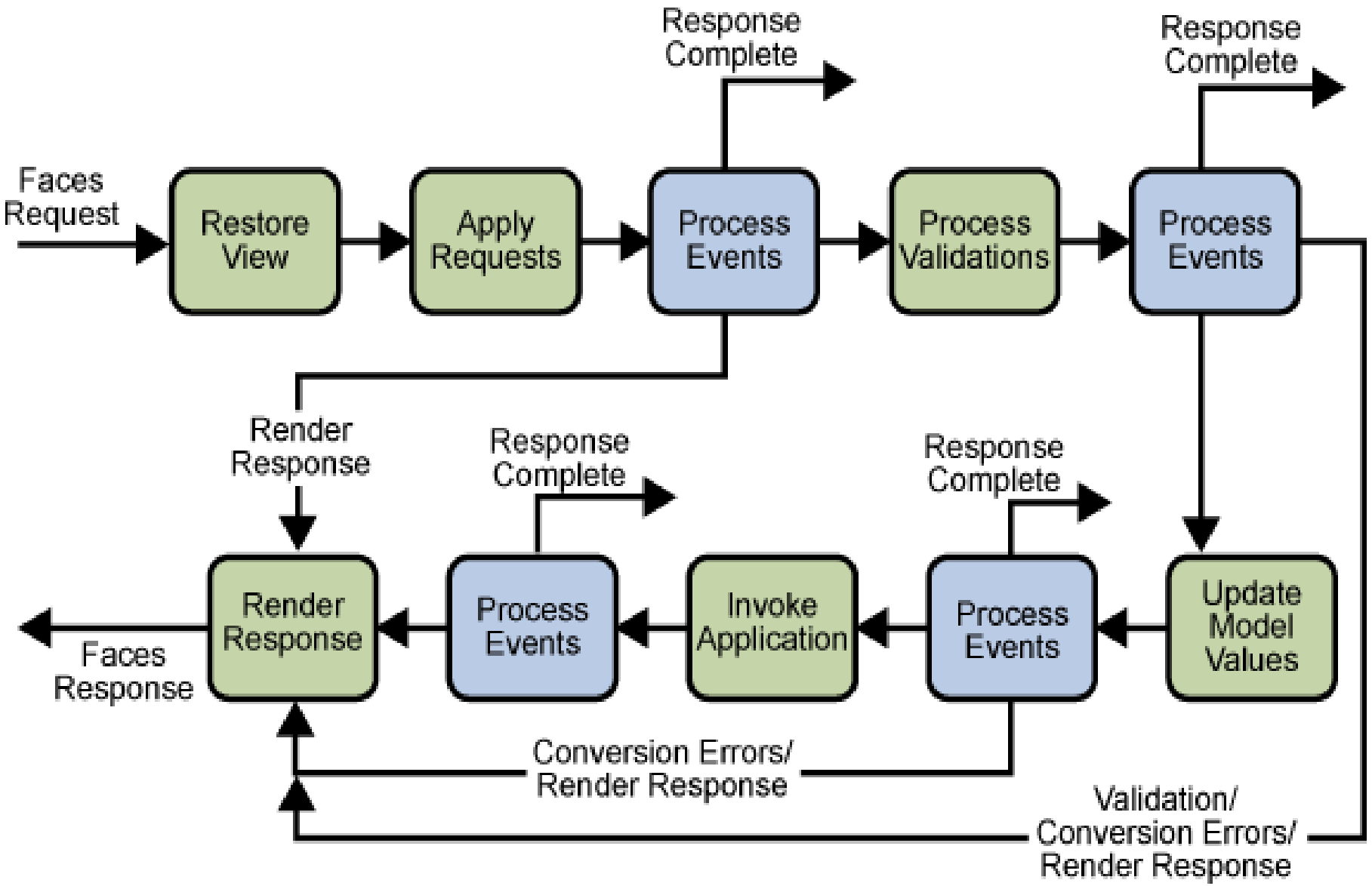
Zadania realizowane podczas cyklu życia aplikacji typu JavaServer Faces

- Obsługa żądań
- Dekodowanie parametrów
- Modyfikacja i zachowanie stanu
- Renderowania stron internetowych i wysłanie ich do przeglądarki internetowej

Framework aplikacji typu JavaServer Faces zarządza **fazami cyklu życia:**

- **automatycznie w przypadku prostych aplikacji**,
- **ręcznie w przypadku złożonych aplikacji.**

Standard cyklu życia „Request-Response” dla JavaServer Faces



Opis faz cyklu życia JavaServer Faces (1)

- **Dwa typy cyklu życia:**
  - **initial request** (pierwsze wywołanie strony) – realizowane są tylko fazy RestoreView i Render Response
  - **postbacks** (obsługa formularza): realizowane są wszystkie fazy.
- **Akcje Response Complete** oznaczają odwołanie do innej części aplikacji, nie zawierającej komponentów Java Server Faces
- **Dwie główne fazy cyklu życia**:
  - **execute** – składa się z podfaz, wspierających obsługę drzewa komponentów: w zakresie konwertowania i walidacji danych, obsługi zdarzeń komponentów oraz przekazania danych komponentów do obiektów typu Managed Bean w sposób uporządkowany
  - **Render**
- **Drzewo komponentów** reprentuje stronę JSF, zwaną **view (widok)**.
- W przypadku, gdy komponent typu JavaServer Faces **w procesie typu submit przekierowuje żądanie do innej strony**, gdzie w ustalony sposób rezalizowane są fazy cyklu życia (w celu konwersji i walidacji, zmiany stanu modelu i generowania widoku)

Opis faz cyklu życia JavaServer Faces (2)

- **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 i słuchaczami zdarzeń
    - aktualizacja tego widoku w **javax.faces.context.FacesContext** dla cyklu typu **postback**
    - lub zapamiętanie pustego widoku w przypadku cyklu **initial request**. Pusty widok w fazie RenderResponse jest renderowany z wykorzystaniem komponentów powiązanych ze znacznikami strony. **Pierwsze żądanie** powoduje utworzenie nowego drzewa komponentów i przechowanie go w obiekcie typu **javax.faces.context.FacesContext**, który reprezentuje wszystkie informacje związane z procesem obsługi żądania i tworzenia odpowiedzi

Opis faz cyklu życia JavaServer Faces (3)

- **Apply Request Values:**
  - konwersja danych i zachowanie ich wartości, wiązanie zdarzeń ze słuchaczami,
  - obsługa konwersji, walidacji i zdarzeń w przypadku, gdy atrybut **immediate** ma wartość true, wtedy konwersje, walidacje oraz zdarzenia powiązane z takim komponentem zachodzą w tej fazie - możliwość akcji **Response Complete**,
  - możliwość przejścia do fazy **Render Response** jako wynik obsługi błędów.
  - W przypadku żądania przekierowania do innego zasobu aplikacji internetowej np. do usług internetowych lub generowanie odpowiedzi, która nie zawiera komponentów typu JavaServer Faces, wówczas program w celu pominięcia fazy RenderResponse musi wywołać metodę **facesContext.responseComplete**
  - Na końcu fazy komponenty zawierają dane zaktualizowane
  - Częściowy cykl życia aplikacji typu JSF oraz częściowy rendering

Opis faz cyklu życia JavaServer Faces (4)

– **Process Validations:**

- obsługa walidacji za pomocą walidatorów umieszczonych w komponentach widoku – użycie metody validate (processValidators), konwersji i zdarzeń,
- zapamiętanie wartości wprowadzonych w formularzu,
- możliwość przejścia do fazy **Render Response** (obsługa błędów lub normalna reakcja) – jeśli metoda **validate** zostanie wywołana przez walidator lub słuchacze zdarzeń wywołają metodę **renderResponse** od bieżącego obiektu typu FacesContext. W przypadku błędu, do obiektu typu FacesContext zostaną wstawione komunikaty o błędach i renderowanie strony z informacją o błędzie
- W przypadku żądania przekierowania do innego zasobu aplikacji internetowej np. do usług internetowych lub generowanie odpowiedzi, która nie zawiera komponentów typu JavaServer Faces, wówczas program w celu pominięcia fazy RenderResponse musi wywołać metodę **facesContext.responseComplete**
- Częściowy cykl życia aplikacji typu JSF oraz częściowy rendering

Opis faz cyklu życia JavaServer Faces (5)

–Update Model Values:

- Jeśli dane są poprawne, wtedy wybrane właściwości komponentów w widoku są zaktualizowane - w przeciwnym wypadku następuje przejście do fazy **RenderResponse** i renderowanie strony z informacją o błędzie
- Jeśli jakikolwiek komponent wywoła metodę **updateModels** lub **renderResponse** od aktualnego obiektu typu **FacesContext** następuje przejście do fazy **RenderResponse**
- W przypadku żądania przekierowania do innego zasobu aplikacji internetowej np. do usług internetowych lub generowanie odpowiedzi, która nie zawiera komponentów typu JavaServer Faces, wówczas program w celu pominięcia fazy RenderResponse musi wywołać metodę **facesContext.responseComplete**

Opis faz cyklu życia JavaServer Faces (6)

– **Invoke Application:**

- realizacja zdarzeń np. typu submitt formularza – często obsługa dotyczy jedynie rekonstrukcji widoku
- lub połączenie z inną stroną,
- możliwość przejścia do fazy Render Response (obsługa błędów lub normalna reakcja)
- W przypadku żądania przekierowania do innego zasobu aplikacji internetowej np. do usług internetowych lub generowanie odpowiedzi, która nie zawiera komponentów typu JavaServer Faces, wówczas program w celu pominięcia fazy RenderResponse musi wywołać metodę **facesContext.responseComplete**

Opis faz cyklu życia JavaServer Faces (7)

– **Render Response:**

- ustalenie zawartości strony w przypadku cyklu typu **postback** - komunikaty jako normalna reakcja
- lub komunikaty o błędach, pochodzące z faz ApplyRequest, ProcessValidations lub UpdateModel. Jeśli strona zawiera znaczniki h:message lub h:messages, błądy wyświetlane są za pomocą renderowania typ znaczników
- ustalenie nowej zawartości drzewa komponentów UI – przebudowanie istniejącego **widoku strony** utworzonego podczas **Restore View** dla cyklu typu **postbacks**

Częściowy cykl życia aplikacji typu JSF oraz częściowy rendering

Takie przetwarzanie jest możliwe np podczas przetwarzania asynchronocznego z użyciem np technologii **Javaserver Faces Ajax**. Renderowane są wtedy fragmenty strony np wybrany komponent reprezentujący fragment strony (TINT, wykład 7)