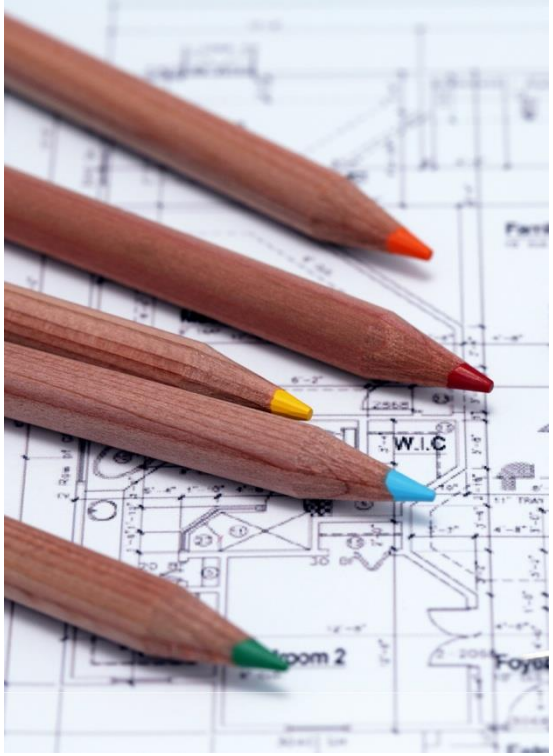


ARS

Java Server Faces

Event Handling

ARS

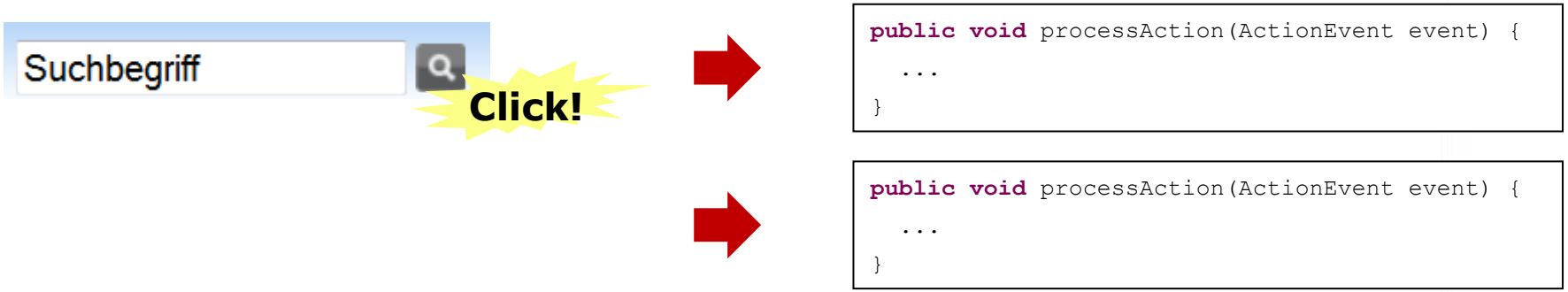
Überblick

- JSF Event-Modell
- Event-Arten
  - Phase Events
  - Faces Events
    - Action Events
    - Value Change Events
  - System Events
- Events im Request Lifecycle
  - Abbruch der Verarbeitung
  - Immediate Event Handling

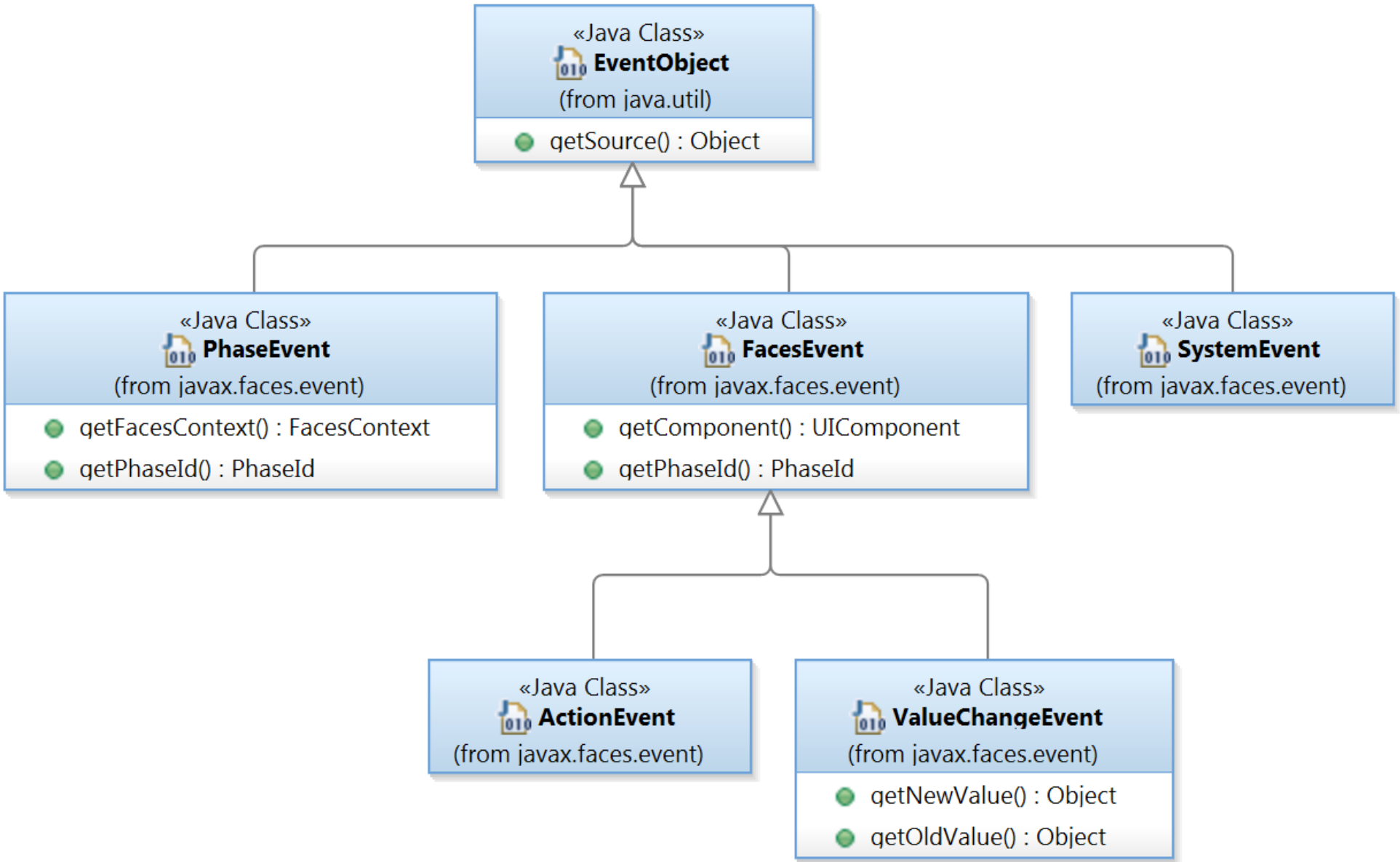
JSF Event-Modell

Konzept

- Basierend auf Event-Modell der JavaBeans-Spezifikation
  - Event-Objekt mit Informationen zum Ereignis
  - Event-Listener mit Callback-Methode (Event-Objekt als Parameter)
    - Registrierung bei auslösender Komponente
- Analogie zu Event-Modell von Swing u.a. Desktopoberflächen
- Verarbeitung in den JSF-Lebenszyklus integriert



Event-Arten



ARS

Event-Arten – Phase Events

Phase Events

- Auslösen von Events zu Beginn und Ende jeder Phase der Requestverarbeitung
- Auslöser: Komponentenbaum (UIViewRoot)

Erstellung

- Implementierung von `javax.faces.event.PhaseListener`
  - `beforePhase(PhaseEvent)`
  - `afterPhase(PhaseEvent)`
  - `getPhaseId()`
- Alternativ: Implementierung einer Methode
  - `public void doSomething(PhaseEvent)`

Event-Arten – Phase Events

Registrierung

- Datei `faces-config.xml` (gesamte Applikation)

```
<lifecycle>
  <phase-listener>de.ars.jsf.MyPhaseListener</phase-listener>
</lifecycle>
```

- Facelet und programmatisch (pro Seite)
  - Aufruf in allen Phasen außer „Restore View“

```
<f:view afterPhase="#{myBean.doSomething}">
  <f:phaseListener type="de.ars.jsf.MyPhaseListener"/>
</f:view>
```

```
FacesContext.getCurrentInstance().getViewRoot().addPhaseListener(
  new MyPhaseListener()
);
```

ARS

Event-Arten – Faces Events

Faces Events

- Auslösen von Events durch Oberflächenkomponenten
  - Benutzerinteraktionen

Arten

- Action Event
  - Klick auf Button (allgemein: `ActionSource`)
  - Auslösen einer Aktion
- Value Change Event
  - Ändern von Werten bei Eingabekomponenten
  - Aufruf serverseitig während der Requestverarbeitung

ARS

Event-Arten – Faces Events

Erstellung von Action Event Handlern

- Implementierung von `javax.faces.event.ActionListener`
  - `processAction(ActionEvent)`
- Alternativ: Implementierung einer Methode
  - `public void doSomething(ActionEvent)`

ARS

Event-Arten – Faces Events

Registrierung von Action Event Handlern

- Gesamte Applikation
  - Aufruf vor Phase „Process Action“
  - Beschränkung auf einen ActionListener

```
<application>
  <action-listener>de.ars.jsf.MyActionListener</action-listener>
</application>
```

(faces-config.xml)

```
FacesContext.getCurrentInstance().getApplication().setActionListener(
  new MyActionListener()
);
```

(programmatisch)

- Pro Komponente

```
<h:commandButton actionListener="#{myBean.doSomething}">
  <f:actionListener type="de.ars.jsf.MyActionListener"/>
</h:commandButton>
```

(Facelet)

```
myButton.addActionListener( new MyActionListener() );
```

(programmatisch)

Event-Arten – Faces Events

Aktion vs. Action Event Handler

- Ausführung jeweils in Phase „Process Action“
- Aktion
  - Max. eine Aktion pro Request
  - Navigation durch Rückgabewert der Aktionsmethode
- Action Event Handler
  - Ausführung mehrerer Handler pro Komponente möglich
  - Kein Rückgabewert

```
<h:commandButton
  actionListener="#{myBean.doSomething}"
  action="#{person.sprechen}"/>
```

???

ARS

Event-Arten – Faces Events

Erstellung von Value Change Event Handlern

- Implementierung von `javax.faces.event.ValueChangeListener`
  - `processValueChange(ValueChangeEvent)`
- Alternativ: Implementierung einer Methode
  - `public void doSomething(ValueChangeEvent)`

Registrierung von Value Change Event Handlern

- Pro Komponente

```
<h:inputText valueChangeListener="#{myBean.doSomething}" value="#{book.title}">
  <f:valueChangeListener type="de.ars.jsf.MyValueChangeListener"/>
</h:commandButton>
```

(Facelet)

```
myTextField.addValueChangeListener( new MyValueChangeListener() );
```

(programmatisch)

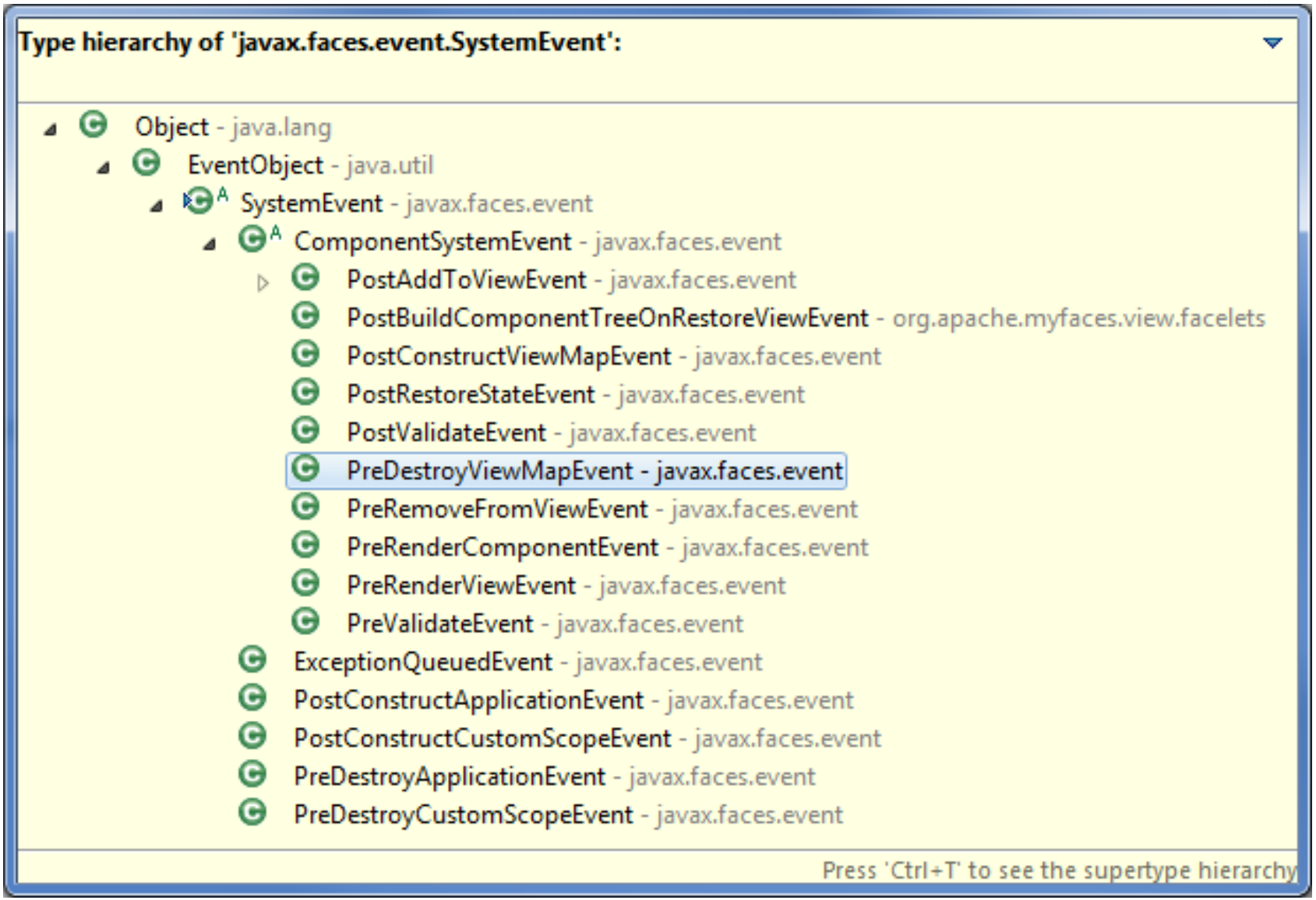
ARS

Event-Arten – System Events

System Events

- Ereignisse zu bestimmten Zuständen des Systems
- Auslöser
  - Applikation
  - Oberflächenkomponente (Component System Event)

Event-Arten – System Events



ARS

Event-Arten – System Events

Erstellung

- Implementierung von `javax.faces.event.SystemEventListener`
  - `processEvent(SystemEvent)`
- Alternativ: Implementierung einer Methode
  - `public void doSomething(ValueChangeEvent)`

Event-Arten – System Events

Registrierung

- Gesamte Applikation

```
<system-event-listener>
  <system-event-listener-class>de.ars.jsf.MySystemListener </system-event-listener-class>
  <system-event-class>javax.faces.event.PreDestroyApplicationEvent</system-event-class>
</system-event-listener>
```

(faces-config.xml)

```
FacesContext.getCurrentInstance().getApplication().subscribeToEvent(
  PreDestroyApplicationEvent.class, new MySystemListener()
);
```

(programmatisch)

- Pro Komponente

```
myButton.subscribeToEvent(...);
```

(programmatisch)

ARS

Event-Arten – System Events

Registrierung – Problem bei Annotation

- Javadoc:
  „The default implementation must support attaching this annotation to UIComponent or Renderer classes.”
- Umsetzung in MyFaces und Mojarra

```
@ListenerFor(systemEventClass=PreDestroyApplicationEvent.class)
public class MySystemListener implements SystemEventListener {...}
```

Events im Request-Lifecycle

Abbruch der Requestverarbeitung

- Direkter Übergang in Phase „Render Response“
- Programmatisch, u.a. in Event Handler

```
public class MyActionListener implements ActionListener {

  @Override
  public void processAction(ActionEvent event)
      throws AbortProcessingException {
    [...]
    FacesContext.getCurrentInstance().renderResponse();
  }

}
```

ARS

Events im Request-Lifecycle – Immediate Events

Problemstellung

- Komponentenbezogene Events (Value Change, Action) nach Konvertierung und Validierung aller Eingaben
- Gesucht: Mechanismus zur Ereignisbehandlung einer Komponente vor/ohne Konvertierung/Validierung aller Eingabefelder

Lösung

- Attribut `immediate="true"` einer Komponente
- Vorgezogenes Konvertieren/Validieren/Auslösen von Events für einzelne Komponente (während Phase „Apply Request Values“)
  - Werte noch nicht im Model
  - Werte anderer Komponenten noch nicht aktualisiert
- Anschließend Behandlung der anderen Komponenten

Kontrollfragen

- Welche Arten von Events gibt es in JSF?
- Wie werden Event Handler in JSF erstellt?
- Was bedeutet Immediate Event Handling?