VAADIN FLOW MIT QUARKUS

MARKUS PAUER

Über 20 Jahre Erfahrung als Berater, Softwareentwickler und Trainer im Jakarta Enterprise Ökosystem

- JakartaEE
- Quarkus
- UI Technologien

Immer einen Blick auf die aktuellsten Trends

UI ENTWICKLUNG

Jakarta Faces	Vaadin Flow	Javascript-Frameworks
		
• nicht alles machbar • umständlich • Dinosaurier	**???**	• moderner • schneller • Quasi Standard
• Java • Enterprise Standard	• Java • Quarkus (Enterprise Standard)	• Javascript/Typescript • Anderes Ökosystem

VAADIN FLOW

- Webanwendungen vollständig in Java entwickeln
- HTML, CSS und JavaScript werden erstellt
- Können bei Bedarf auch direkt eingebunden werden
- Rendering der Komponenten erfolgt erst im Browser
- Komponentenbaum wird im Server gehalten
- Kommunikation zwischen Server und Client über Websockets

VAADIN FLOW QUARKUS EXTENSION

https://quarkus.io/extensions/com.vaadin/vaadin-quarkus-extension/

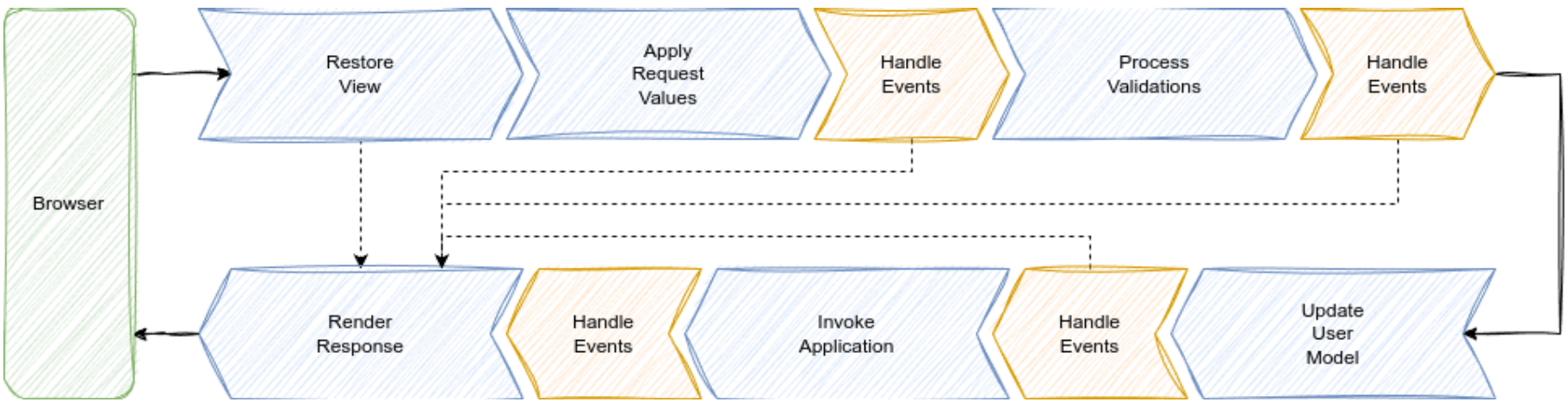
https://vaadin.com/docs/latest/flow/integrations/quarkus

pom.xml

```
1 ...
2 <dependency>
3     <groupId>com.vaadin</groupId>
4     <artifactId>vaadin-quarkus-extension</artifactId>
5     <version>${vaadin.version}</version>
6 </dependency>
7 ...
```

RENDERING - FACES

- Request-Processing-Lifecycle



RENDERING - FACES

- Komponenten (Facelets)

panel.xhtml

```
1 <h:panelGroup layout="block">
2     <h:outputText value="Hello world" />
3 </h:panelGroup>
```

- Werden auf dem Server in HTML "übersetzt"

panel.html

```
1 <div>Hello world</div>
```

RENDERING - VAADIN FLOW

- Komponenten werden vollständig in Java implementiert

PanelView.java

```java
@Route("panel")
public class PanelView extends Div {
  public PanelView() {
    add(new Text("Hello world"));
  }
}
```

RENDERING - VAADIN FLOW

- Beim Aufruf der Seite wird nur ein Grundgerüst und die Vaadin-Engine übertragen
- Die Informationen für das Rendering werden als Json "nachgeliefert"

commands.json

```
1 [...
2   {
3     "node": 4,
4     "type": "put",
5     "key": "text",
6     "feat": 7,
7     "value": "Hello world"
8   }
9 ...]
```

RENDERING - VAADIN FLOW

Nach diesen "Anweisungen" werden sie als Webkomponenten direkt im DOM gerendert.

vaadin.html

```
1 <flow-container-root-1234567 id="ROOT-1234567" style="">
2     <div>Hello world</div>
3 </flow-container-root-1234567>
```

KOMPONENTEN

- Views können (sollten) in Komponenten aufgeteilt werden
- Komponenten können in verschiedenen Views wiederverwendet werden
- Sie stellen in sich geschlossene Funktionalitäten bereit
- In Quarkus können sie als CDI-Beans implementiert werden

Toolbar.java

```
1 @RouteScoped
2 public class IssueToolbarComponent extends HorizontalLayout {
3   @Inject
4   public IssueToolbarComponent() {
5     add(new Button("Hinzufügen", new Icon(VaadinIcon.PLUS)));
6   }
7 }
```

EVENTVERARBEITUNG - FACES

- Jakarta Faces verbindet die Oberflächenkomponenten mit Methoden auf dem Server
- Bei einem Event wird normalerweise die gesamte Seite neu aufgebaut
- Mit Ajax kann man auch in Faces lediglich einzelne Komponenten aktualisieren

command-button.xhtml

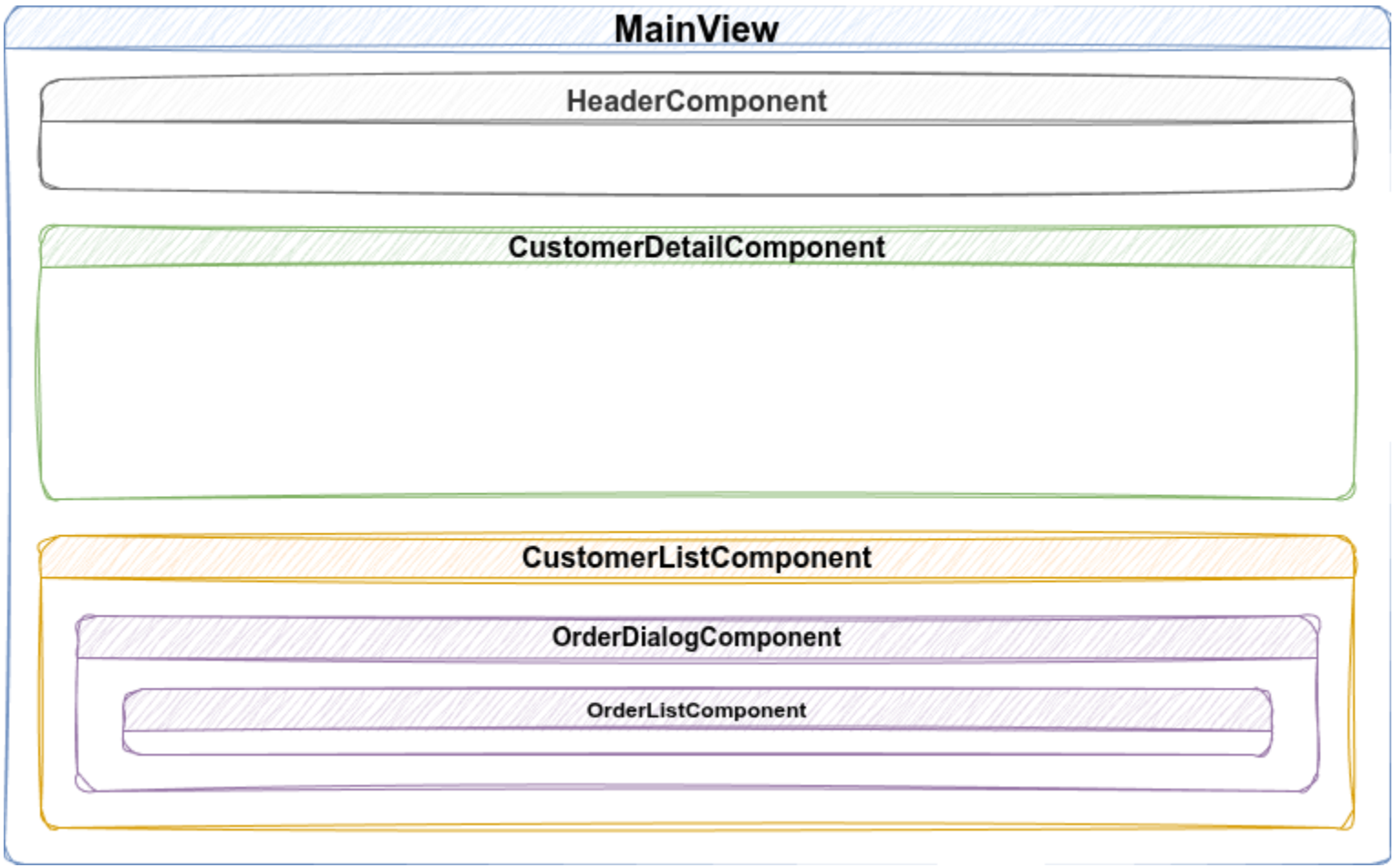
```
1 <p:commandButton value="Click me" action="#{toolbarPresenter.doSomething}"/>
```

EVENTVERARBEITUNG - VAADIN FLOW

Toolbar.java

```java
@RouteScoped
public class IssueToolbarComponent extends HorizontalLayout {
  @Inject
  public IssueToolbarComponent() {
    var button = new Button("Hinzufügen", new Icon(VaadinIcon.PLUS));
    button.addClickListener(event -> {
      // do Something
    });
    add(button);
  }
}
```

AUFBAU DER DEMO-ANWENDUNG



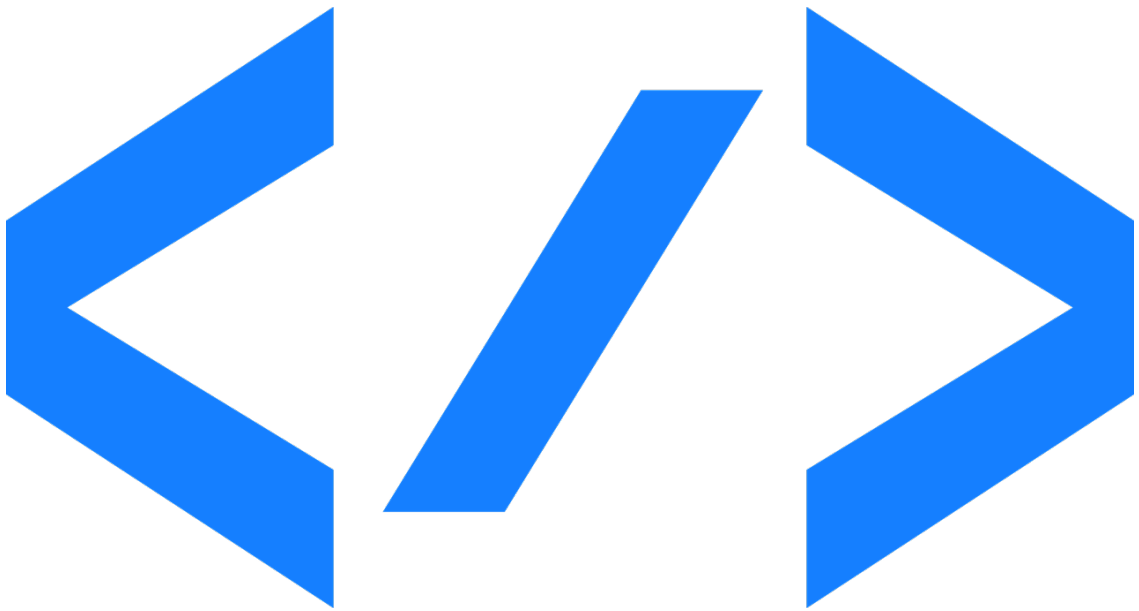
NUTZUNG DER KOMPONENTEN IN EINER VIEW

- Da die Komponenten CDI-Beans sind
- Können sie einfach per @Inject in der View verwendet werden
- Falls keine CDI-Bean notwendig ist
- Kann die Komponente auch direkt instanziiert werden

MainView.java

```java
@Route("")
public class MainView extends VerticalLayout {
  @Inject
  public MainView(CustomerDetailComponent customerDetailComponent,
      CustomerListComponent customerListComponent) {
    add(new HeaderComponent(), customerDetailComponent, customerListComponent);
  }
}
```

DEMO



FAZIT

- Vaadin Flow durchaus eine Alternative zu Javascript-Frameworks oder Jakarta Faces
- Kein Verzicht auf den Komfort des Java-Ökosystems (Maven, etc.)
- Quarkus als Grundlage für Enterprise-Anwendungen

https://github.com/GEDOPLAN/quarkus-vaadin-demo