3.– 6. September 2012
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

Web-Anwendungen auf Klick

Web-Programmierung mit Apache Click

Rustam Khakimov

MATHEMA Software GmbH

Agenda

- Was ist Click Framework?
  - Überblick
  - Architektur
- Grundlegende Komponente
  - Pages
  - Controls
- Konfiguration
- Test

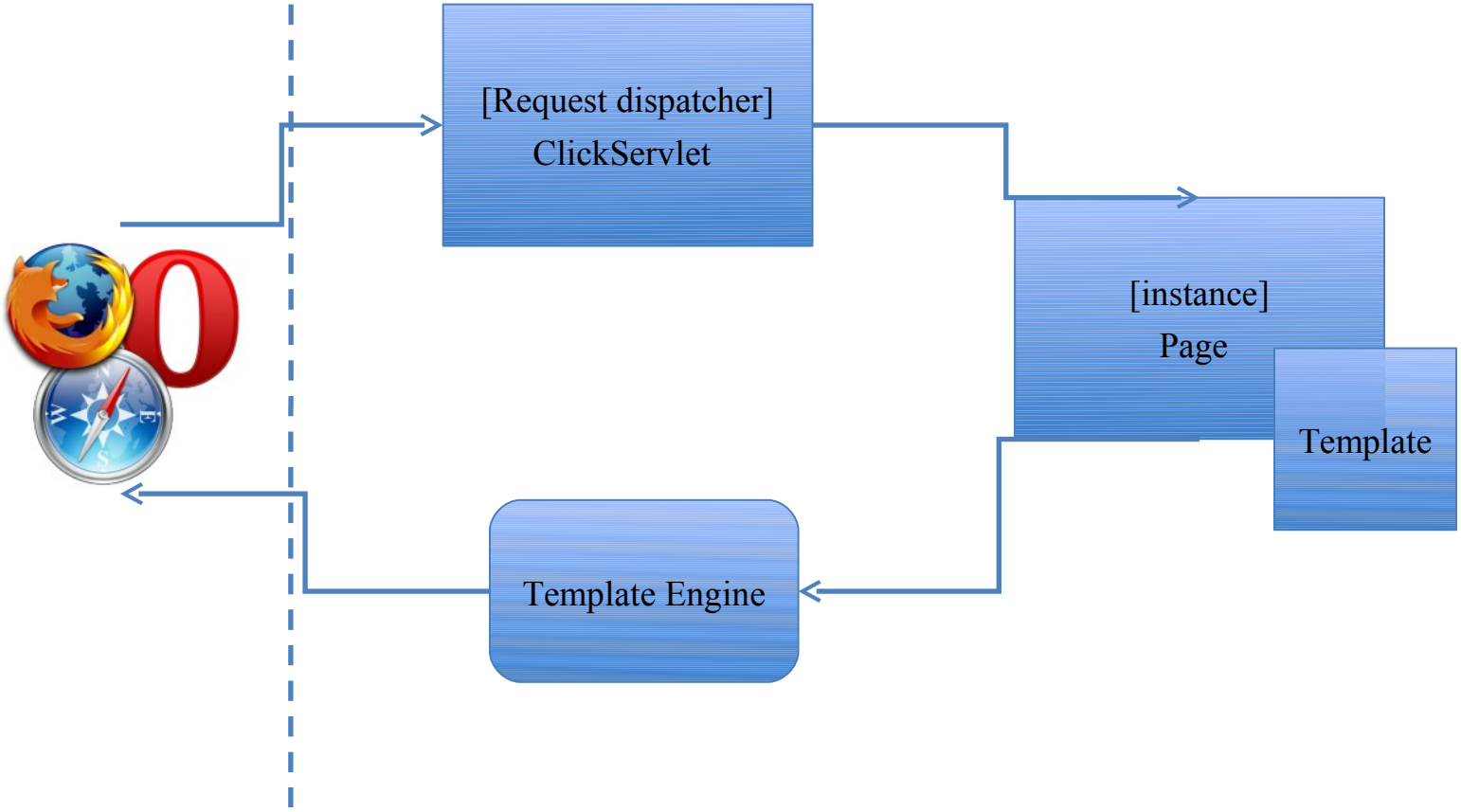
Überblick (1)

- Open Source Projekt
  - Erste Release im September 2008
  - Apache Top Level Project im Februar 2010
  - Aktuelle Release Version 2.3.0 unter Apache License
- Kein MVC
- Event basiertes Programmiermodel
- Template-Engine Apache Velocity, JSP oder Freemarker

Überblick (2)

- Client-/Serverseitige Validierung
- Serverseitige Unterstützung der Ajax Requests
- Stateless mit leichtgewichtigen stateful Komponenten
- Testgetriebene Implementierung durch Mock-API
- IDE-Unterstützung durch Eclipse Plug-in

Architektur



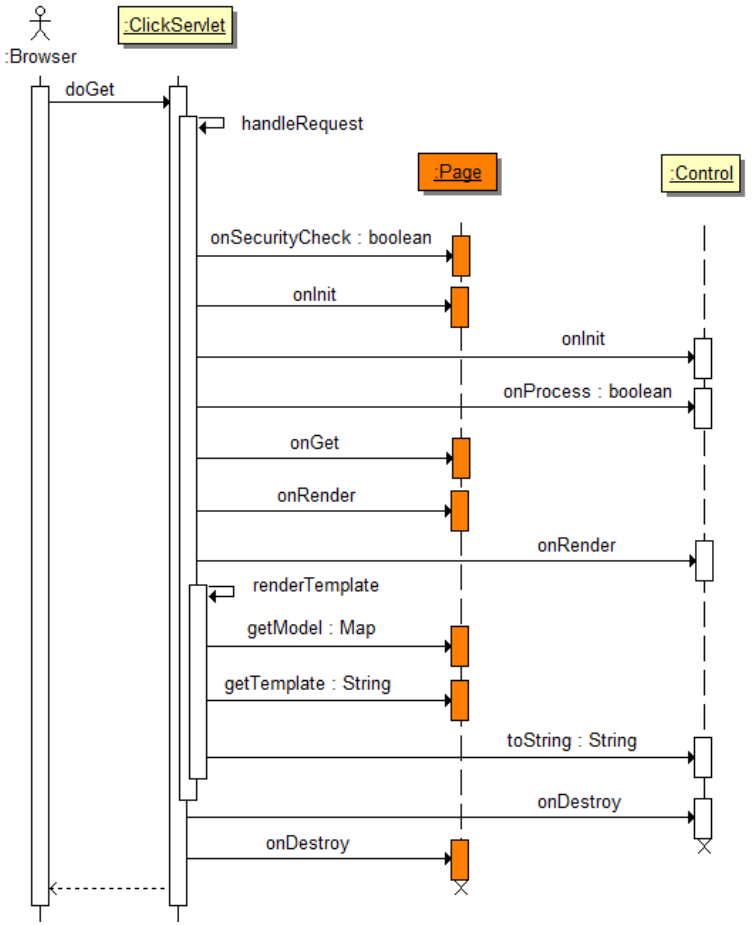
Demo

DEMO

Pages

- Kombination aus einer Java-Klasse (abgeleitet von org.apache.click.Page) und einem Template
- Verarbeitung von verschiedenen Template-Formaten:
  - Apache Velocity,
  - JSP oder
  - Freemarker
- Direct Rendering von Templates
- Stateless

Pages (Request Verarbeitung)



Quelle: Apache Click User Guide, V 2.3.0, S. 16

Pages (Security Check)

```
public class AdminPage extends Page {

    public boolean onSecurityCheck() {
        if (getContext().getRequest().isUserInRole("admin")) {
            return true;
        } else {
            setRedirect(LoginPage.class);
            return false;
        }
    }
}
```

Pages (Navigation)

- Forward
- Redirect
- Templatepfad

Pages (Navigation Forward)

```
setForward("view-customer.htm");
```

```
getContext().setRequestAttribute("customer", customer);
setForward(ViewCustomerPage.class);
```

```
ViewCustomerPage vc = getContext().createPage(ViewCustomerPage.class);
vc.setCustomer(customer);
setForward(vc);
```

Pages (Navigation Redirect)

```
setRedirect("view-customer.htm");


setRedirect(ViewCustomerPage.class);


Map<String, Integer> reqParams = new HashMap<String, Integer>();
reqParams.put("custIds", 42);
setRedirect(ViewCustomerPage.class, reqParams);
```

Pages (Navigation Templatepfad)

- Zugeordnete Template zur Laufzeit austauschen

```
setPath("view-customer.htm");
```

Pages (Page Templating)

```
public class MasterPage extends Page {

    public String getTemplate() {
        return "/MasterPage.htm";
    }
}
```

```
<html>
    <head>
        <title>$title</title>
    </head>
    <body>
        <h2 class="title">$title</h2>

        #parse($path)
    </body>
</html>
```

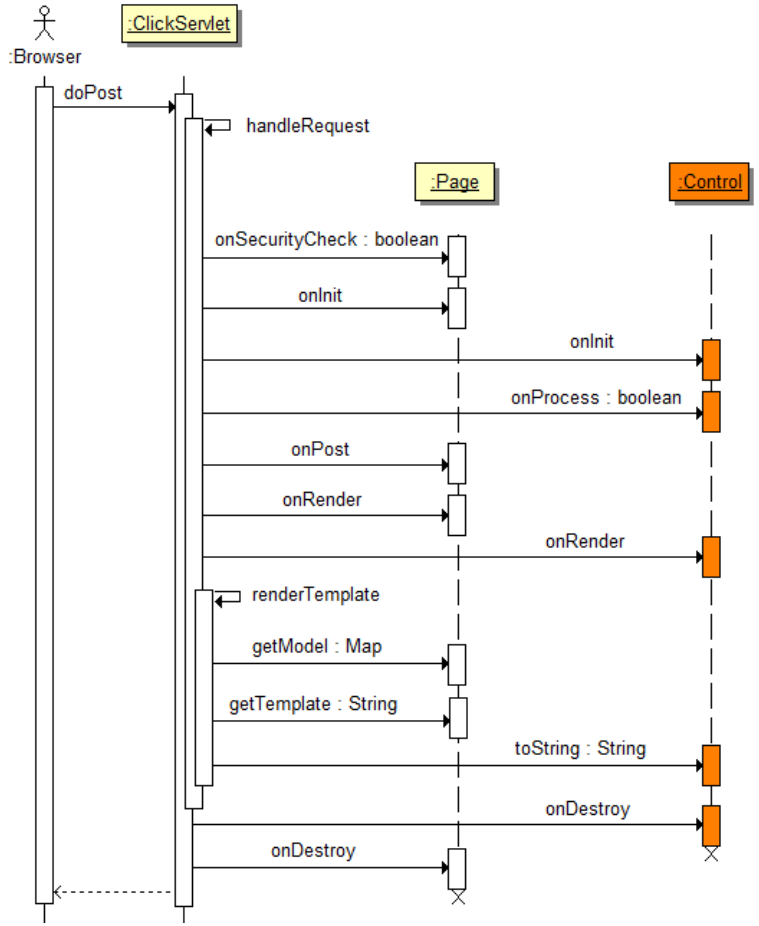
Controls

- Steuerelemente
  - Serverseitige Verarbeitung
  - Clientseitiges Rendering
- Unterstützung der ActionListener-Mechanismen (Swing)
- Client-/Serverseitige Validierung

Controls

- Form
- ActionLink
- TextField, PasswordField, CreditCardField …
- Checkbox
- Button, Submit und Reset
- …

Controls (Execution sequence)



Quelle: Apache Click User Guide, V2.3.0, S. 38

Controls (ActionListener)

```
private ActionLink link = new ActionLink();
private ActionButton button = new ActionButton();

public ActionExample() {
    link.setActionListener(new ActionListener() {
        public boolean onAction(Control source) {
            onLinkClick(source);
            return true;
        }
    });

    button.setListener(this, "onButtonClick");
}

private boolean onLinkClick(Control source) {
    return true;
}

public boolean onButtonClick() {
    return true;
}
```

Controls (Validierung)

- Clientseitig

```
Form form = new Form("form");
form.setJavaScriptValidation(true);
```

- Serverseitig

```
Form form = new Form("form");
form.setValidate(true);

if(form.isValid()) {

}
```

Controls (Bahavior)

- Interceptor-Methoden für Steuerelemente um
  - JavaScript und CSS-Elemente einzufügen und zu löschen
  - Attribute der Steuerelementen zu ändern

- Bahavior-Methoden
  - preResponse
  - preRenderHeaderElement
  - preDestroy

Demo

DEMO

Konfiguration (web.xml)

```
<web-app>

   <servlet>
         <servlet-name>ClickServlet</servlet-name>
         <servlet-class>org.apache.click.ClickServlet</servlet-class>
   <load-on-startup>0</load-on-startup>
   </servlet>

   <servlet-mapping>
         <servlet-name>ClickServlet</servlet-name>
         <url-pattern>*.htm</url-pattern>
   </servlet-mapping>


</web-app>
```

Konfiguration (click.xml)

```
<click-app charset="UTF-8" locale="de" >

    <pages package="hcampus.click" autobinding="annotation">
        <page path="index.htm" classname=„hcampus.click.Home"/>
                                ...
        <excludes pattern="/dhtml/*, /tiny_mce/*, banner.htm, about.htm"/>
    </pages>

    <headers>
            ...

    <headers>

            ...
```

Konfiguration (click.xml)

```
...

<mode value="development" />
```

- Page auto loading
- Template caching
- Click log level
- Velocity log level

Konfiguration (click.xml)

- Exception bei der Verarbeitung der Seite oder beim Rendern vom Template

```
<page path="click/error.htm" classname= " org.apache.click.util.ErrorPage"/>
```

- Seite nicht gefunden

```
<page path="click/not-found.htm" classname= " org.apache.click.Page"/>
```

Konfiguration (click.xml)

```
...

<controls>
    <control classname=" hcampus.click.control. MyField"/>
</controls>

<format classname=" hcampus.click.format. MyFormat"/>

</click-app>
```

Tests mit Click Mock API

- Unit-Tests ohne Web-Container
- MockContainer
- MockContext

```
MockContainer container = new MockContainer("c:/dev/myWebApp/web");
container.start();

            ...

container.stop();
```

Demo

DEMO

Vielen Dank!

Rustam Khakimov

MATHEMA Software GmbH