Resources

Resources

anderScore trust in competence

GFU Cyrus AG

Grundlagen

- Resources: CSS, JS, PDF, Images, …
- OO-Repräsentation: ResourceReference
- Statische Resourcen
  - Package Resources
  - HeaderContributors
  - Resource Bundles
  - NIO Resources
- Dynamische Resourcen
  - Custom Resources
  - JAX-RS-Annotationen
- Mounting an URL möglich
- Shared Resources

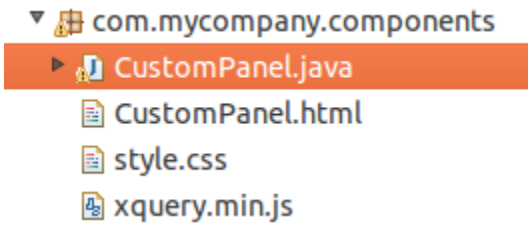
Resources

anderScore trust in competence

GFU Cyrus AG

Package Resources

- Resourcen in gleichem Package wie HTML + Java
- Distribution mit Komponenten (→ Abgeschlossenheit)
- PackageResourceReference
- Images: Responsive Design
- URL
  - <*Applikation*>/wicket/resource/<*Klasse*>/<*Name*>-<ver-<*id*>>[.*Dateiendung*]
  - ./wicket/resource/net.gfu.CustomPanel/calendar-ver-1297887542000.jpg



Resources

Package Resources

- HTML

```
<html>

<head></head>

<body>

<wicket:panel>
    Package resource image: <img wicket:id="packageResPicture"/>
</wicket:panel>

</body>

</html>
```

Resources

Package Resources

- Java

```
public class CustomPanel extends Panel {

    public CustomPanel(String wicketId) {
        super(id);

        Image image = new Image("packageResPicture",
                new PackageResourceReference(getClass(), "large.jpg"),
                new PackageResourceReference(getClass(), "medium.jpg"),
                new PackageResourceReference(getClass(), "small.jpg"));
        image.setXValues("1024w", "640w", "320w");
        image.setSizes("(min-width: 36em) 33.3vw", "100vw");

        add(image);
    }
}
```

Resources

Header Contributors

- Interface *IHeaderContributor*
- Einfügen von Header Items in umgebende Page
  - CSSHeaderItem
  - JavaScriptHeaderItem
  - MetaDataHeaderItem
  - OnDomReadyHeaderItem
  - PriorityHeaderItem
  - …
- Jede Komponente ist selbst HeaderContributor!

Resources

Header Contributors

- Java

```
public class MyComponent extends Component{

@Override
public void renderHead(IHeaderResponse response) {
    super.renderHead(response);

    response.render(CssHeaderItem.forReference(
    new CssResourceReference(Index.class, "PATH_TO_CSS_FILE")));

    response.render(JavaScriptHeaderItem.forReference(
    new JavaScriptResourceReference(Index.class, "PATH_TO_JSS_FILE")));
    }
}
```

Resources

Resource Bundles

- Zusammenfassung von Ressourcen (vgl. Grunt)
- Nicht zu verwechseln mit Java Resource Bundles!
- Registrierung in Application-Klasse
- Applikationsweite Verwendung

Resources

Resource Bundles

- Java

```
@Override
public void init() {
    super.init();

    getResourceBundles().addJavaScriptBundle(WicketApplication.class,
            "jqueryUiJs",
            jqueryJsReference,
            jqueryUiJsReference);

    getResourceBundles().addCssBundle(WicketApplication.class,
            "jqueryUiCss",
            jqueryCssReference,
            jqueryUiCssReference);

}
```

Resources

anderScore trust in competence

GFU Cyrus AG

NIO Resources

- Zugriff auf Resourcen mittels Java NIO (seit Wicket 7.2.0)
- Laden von Daten aus dem Dateisystem

- HTML

```
<video wicket:id="video"/>
```

- Java

```
URI uri =
URI.create("jar:file://videosFolder/videos.zip!/folderInZip/MyVideo.mp4");

Path path = FileSystemResourceReference.getPath(uri);
FileSystemResourceReference ref = new FileSystemResourceReference("myvideo", path);
Video video = new Video("video", ref);
add(video);
```

Resources

Custom Resources

- Einbindung dynamisch zu erstellender Inhalte
- Erzeugung bei Bedarf durch WriteCallback
- Direkter Zugriff auf ResourceResponse

Resources


```
public class RSSProducerResource extends AbstractResource {

    @Override
    protected ResourceResponse newResourceResponse(Attributes attributes) {
        ResourceResponse resourceResponse = new ResourceResponse();
        resourceResponse.setContentType("text/xml");
        resourceResponse.setTextEncoding("utf-8");

        resourceResponse.setWriteCallback(new WriteCallback() {

            @Override
            public void writeData(Attributes attributes) throws IOException
            {
                OutputStream outputStream = attributes.getResponse().getOutputStream();
                Writer writer = new OutputStreamWriter(outputStream);
                SyndFeedOutput output = new SyndFeedOutput();
                try {
                    output.output(getFeed(), writer);
                } catch (FeedException e) {
                    throw new WicketRuntimeException("Problems writing feed to response...");
                }
            }
        });

        return resourceResponse;
    }
    // Methode getFeed()…
}
```

Resources

Shared Resources

- Applikationsweit verfügbare Ressourcen
- Registrierung in Application-Klasse
- Einbindung nach Bedarf

Resources

Shared Resources

- Application-Klasse

```
@Override
public void init(){
        RSSProducerResource rssResource = new RSSProducerResource();
        // …
        getSharedResources().add("globalRSSProducer", rssResource);
}
```

- Einbindende Komponente

```
add(new ResourceLink("globalRssLink", new
SharedResourceReference("globalRSSProducer")));
```