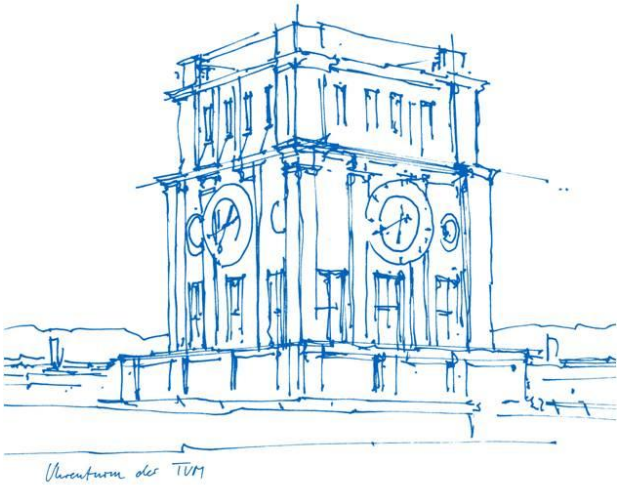
Praktikum XML-Technologien

Team XLink

Technische Universität München

Fakultät für Informatik

Garching, 17.10.2019



Andreas Binder, Christian Lang, Julian Feldmeier, Moritz Münker | TUM

Abschlusspräsentation

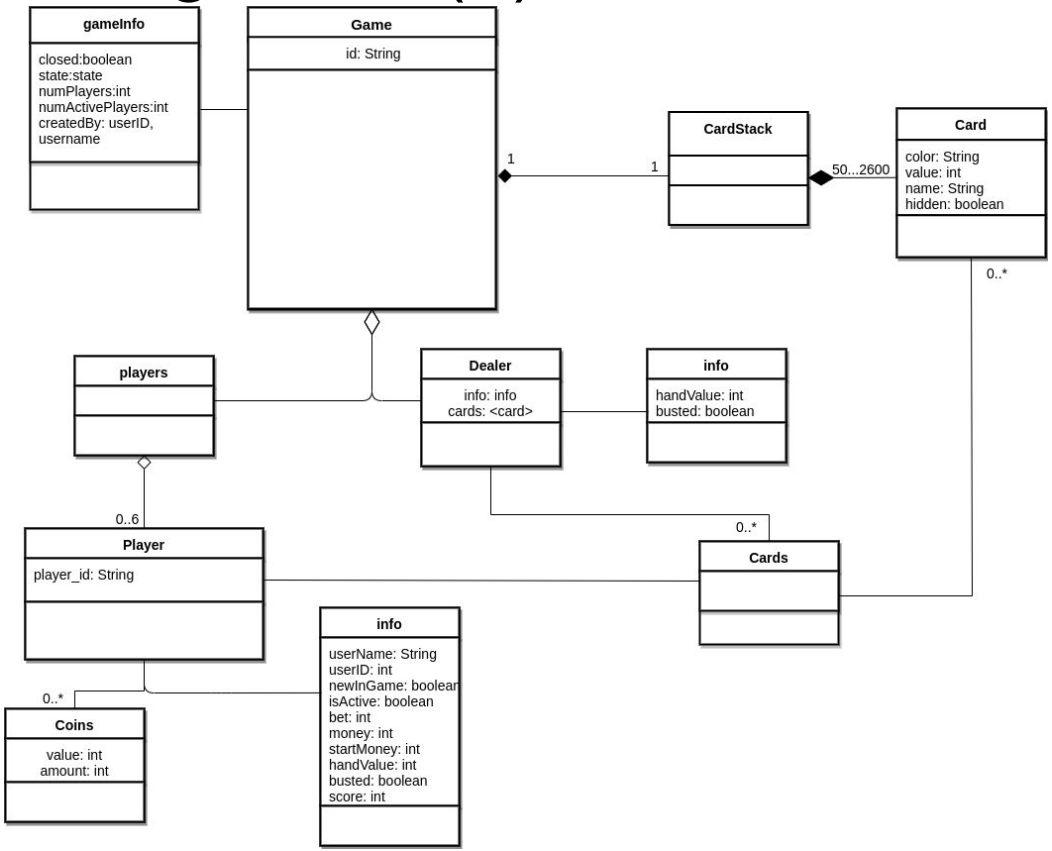
Projektaufgabe: Entwurf und Implementierung des Spiels BlackJack auf Multiclient Basis mit Hilfe des X-Stacks

Gliederung

1. Aufbau des Datenmodells
2. Grafische Komponenten
3. Umsetzung der Multi Client Architektur
4. Überblick über die Spiellogik
5. Show Case
6. Vorstellen des DocBooks
7. Resume zum Praktikum

1. Aufbau des Datenmodells

UML-Klassendiagramm (1)



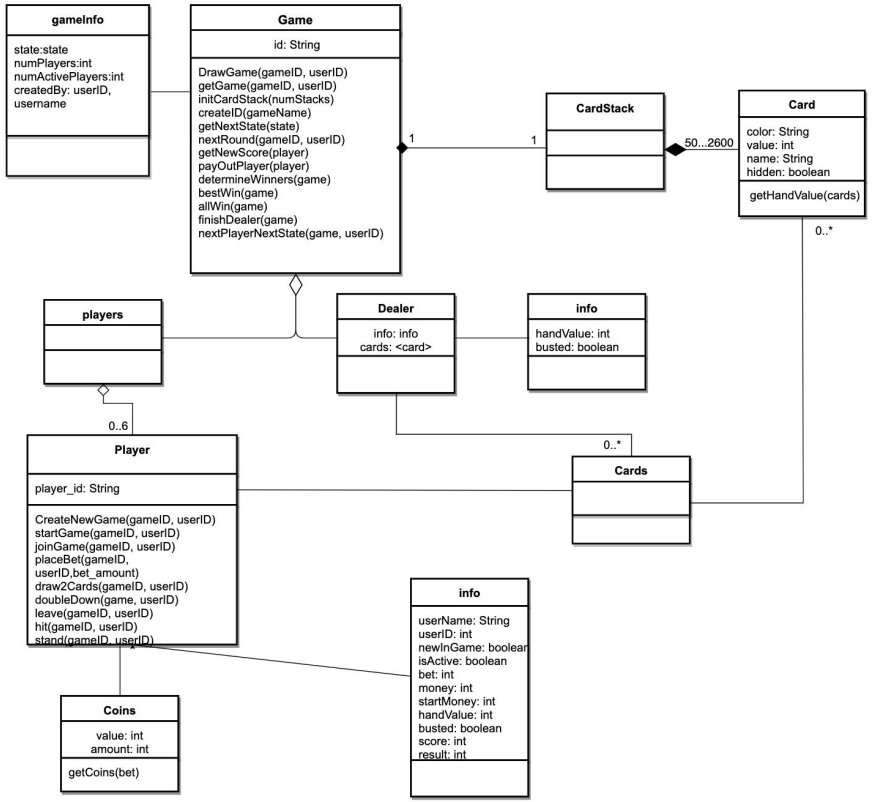
XML Model eines Games...

```xml
<game id="">
  <gameInfo>
    <closed></closed>
    <state></state>
    <numPlayers></numPlayers>
    <numActivePlayers></numActivePlayers>
    <createdBy>
      <userID></userID>
      <username>
        <username></username>
      </username>
    </createdBy>
  </gameInfo>
  <dealer>
    <info>
      <handValue></handValue>
      <busted></busted>
    </info>
    <cards/>
  </dealer>
```

...XML Modell eines Games

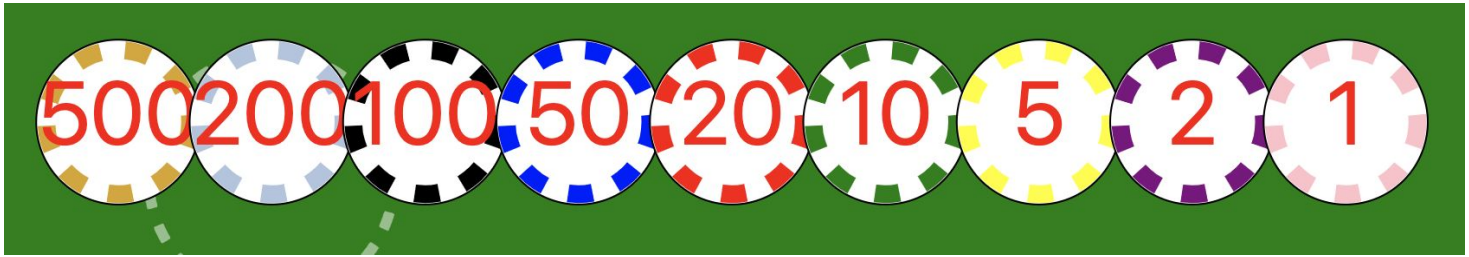
```
<players>
  <player player_id="">
    <info>
      <username></username>
      <userID></userID>
      <newInGame></newInGame>
      <isActive></isActive>
      <bet></bet>
      <money></money>
      <startMoney></startMoney>
      <handValue></handValue>
      <busted></busted>
      <score></score>
      <result></result>
    </info>
    <cards/>
    <coins>
      <coin value="" amount=""/>
    </coins>
  </player>
</players>
<cardStack>
  <card color="" value="" alt_value="" name=""/>
</cardStack>
</game>
```

UML-Klassendiagramm (2)



2. Die graphischen Komponenten

Die Coins




```
<g id="coin_1" >
    <circle r="4%" fill="white" stroke="black" />
    <circle r="3.5%" fill="white" stroke="pink" stroke-width="10" stroke-dasharray="1.2%,1.2%" />
    <text  fill="red" font-size="50" dominant-baseline="middle" text-anchor="middle">1</text>
</g>
```

Die Farben...

```svg
<svg id = "club">
    <g>
        <circle r="30" cx = "75" cy = "90" fill = "black"></circle>
        <circle r = "30" cx = "55" cy = "120" fill = "black"></circle>
        <circle r="30" cx = "95" cy = "120" fill = "black"></circle>
        <polygon points = "75,135 55,180 95,180"></polygon>
    </g>
</svg>
```

```svg
<svg id = "heart">
    <g>
        <circle cx="50" cy="90" r="27" style="fill:red;stroke:red"/>
        <circle cx="100" cy="90" r="27" style="fill:red;stroke:red"/>
        <polygon points="25,100 125,100 75,175" style="fill:red;stroke:red"/>
    </g>
</svg>
```

… die Farben

```svg
<svg id = "diamond">
    <g>
        <polygon points="75,52 35,112 75,172 115,112 75,52"
        style="fill:red;stroke:black;stroke-width:0"/>
    </g>
</svg>
```

```svg
<svg id ="spade">
    <g>
        <circle cx="55" cy="140" r="23" />
        <circle cx="95" cy="140" r="23" />
        <polygon points="75,55 35,130 115,130" />
        <polygon points="75,135 55,180 95,180" />
    </g>
</svg>
```

Karten

Definition des cardStacks per XML

```
<card color='club' value='1' alt_value='11' name='A' />
<card color='club' value='2' name='2' />
<card color='club' value='3' name='3' />
<card color='club' value='4' name='4' />
<card color='club' value='5' name='5' />
<card color='club' value='6' name='6' />
<card color='club' value='7' name='7' />
<card color='club' value='8' name='8' />
<card color='club' value='9' name='9' />
<card color='club' value='10' name='10' />
<card color='club' value='10' name='J' />
<card color='club' value='10' name='Q' />
<card color='club' value='10' name='K' />

<card color='diamond' value='1' alt_value='11' name='A' />
<card color='diamond' value='2' name='2' />
<card color='diamond' value='3' name='3' />
<card color='diamond' value='4' name='4' />
<card color='diamond' value='5' name='5' />
<card color='diamond' value='6' name='6' />
<card color='diamond' value='7' name='7' />
<card color='diamond' value='8' name='8' />
<card color='diamond' value='9' name='9' />
<card color='diamond' value='10' name='10' />
<card color='diamond' value='10' name='J' />
<card color='diamond' value='10' name='Q' />
<card color='diamond' value='10' name='K' />
```

Zusammensetzen einer Karte - Theorie

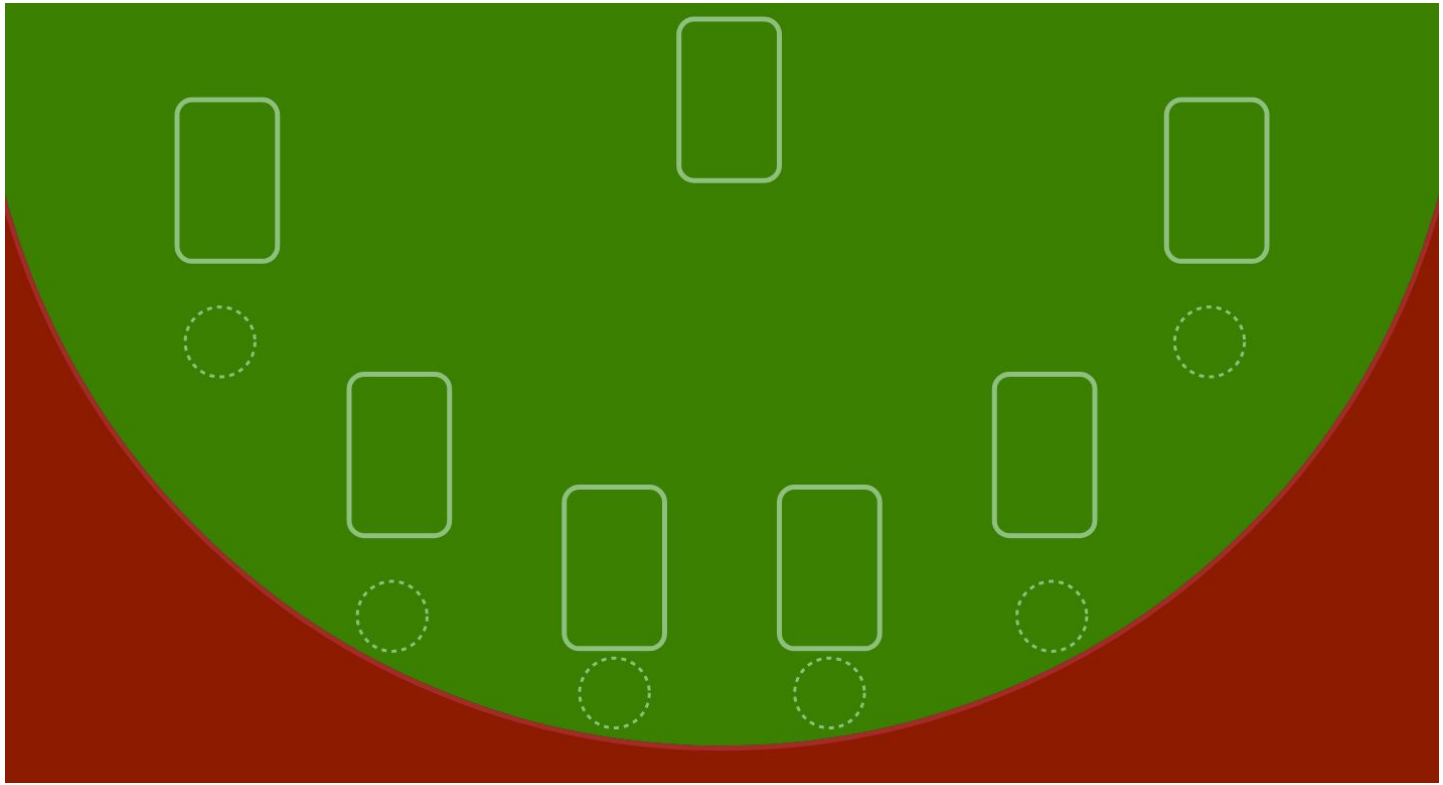
```svg
<!-- Card shape -->
<svg id = "cardShape">
<g>
    <rect width="9%" height="25%" fill = "white" stroke="black" stroke-width = "2.5" stroke-linejoin="round" rx="20" ry="20"></rect>
</g>
</svg>
```

```xsl
<g id="PlayerCards">
    <xsl:for-each select='game//player'>
        <xsl:variable name='player_id' select='@player_id' />
        <xsl:for-each select='.//card' >

            <use href="#cardShape"/>

            <use href="#{@color}" height="100%" width="100%"/>

            <xsl:if test="@color='diamond' or @color='heart' " >

                <text color="red" fill="red" font-size="250%" transform="scale(0.5)">
                    <xsl:value-of select="@name"/>
                </text>
                <text color="red" fill="red" font-size="250%" transform="scale(0.5)">
                    <xsl:value-of select="@name"/>
                </text>
            </xsl:if>

            <xsl:if test="@color='club' or @color='spade' " >
                <text color="black" fill="black" font-size="250%" transform="(0.5)">
                    <xsl:value-of select="@name"/>
                </text>

                <text color="black" fill="black" font-size="250%" transform="scale(0.5)">
                    <xsl:value-of select="@name"/>
                </text>
            </xsl:if>

        </xsl:for-each>
    </xsl:for-each>
</g>
```

Dynamisches Zusammensetzen der Karten mit Komponenten:

- Kartenrahmen
- Farbzeichen
- Wert

Der Tisch...



… der Tisch

```
    <!-- Tischform -->
    <circle cx="50%" cy="0%" r="64%" fill="green" stroke="brown" stroke-width="5"/>

    <!-- Plätze für Karten -->
    <use xlink:href="#CardArea" id="CardArea1" x="12%" y="12%"/>
    <use xlink:href="#CardArea" id="CardArea2" x="24%" y="46%"/>
    <use xlink:href="#CardArea" id="CardArea3" x="39%" y="60%"/>
    <use xlink:href="#CardArea" id="CardArea4" x="54%" y="60%"/>
    <use xlink:href="#CardArea" id="CardArea5" x="69%" y="46%"/>
    <use xlink:href="#CardArea" id="CardArea6" x="81%" y="12%"/>
    <use xlink:href="#CardArea" id="CardAreaDealer" x="47%" y="2%"/>

    <!-- Plätze für Coins -->
    <use xlink:href="#CoinArea" id="CoinArea1" x="15%" y="42%"/>
    <use xlink:href="#CoinArea" id="CoinArea2" x="27%" y="76%"/>
    <use xlink:href="#CoinArea" id="CoinArea3" x="42.5%" y="85.5%"/>
    <use xlink:href="#CoinArea" id="CoinArea4" x="57.5%" y="85.5%"/>
    <use xlink:href="#CoinArea" id="CoinArea5" x="73%" y="76%"/>
    <use xlink:href="#CoinArea" id="CoinArea6" x="84%" y="42%"/>
</g>
```

3. Umsetzung der Multi-Client Architektur

Modell

Client

- über Webbrowser, HTML & Javascript

Server

- BaseX Datenbank
- Jetty Server

Kommunikation

- WebSockets mit 2 verschiedenen subscription modellen
  - “/xlink/lobby/{userID}
  - “/xlink/game/{gameID}/{userID}

Websocket Subscriptions

- Subscription URLs bestehen aus verschiedenen Parametern
  - “/xlink/game/new_game_123/12345”
  - “/xlink” = group
  - “/game” oder “/lobby” = position
  - “/new_game_123” = gameID
  - “/12345” = userID

- Nach jedem Spielzug wird der aktive Spieler wieder zurück zum Spiel redirected und subscribed wieder. Durch diese neue Subscriptions werden an alle anderen Clients im Spiel das neue Spiel geschickt und angezeigt.

```
return (update:output(web:redirect(concat('/xlink/game/', $gameID, '/', $userID))),
```

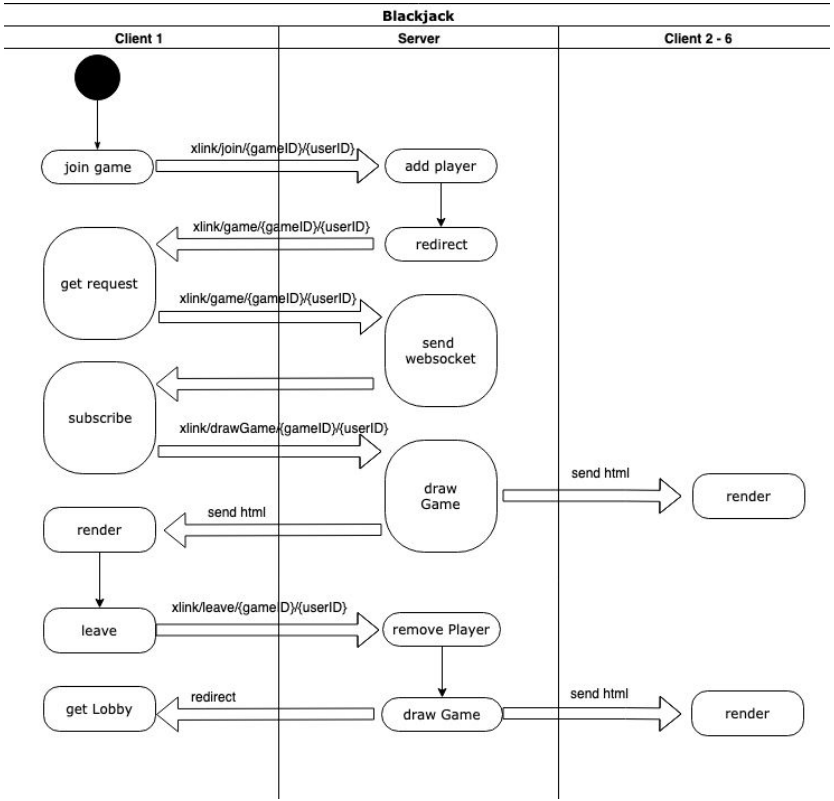
Verbindung zum Server

```
declare
%ws-stomp:subscribe("/xlink/game")
%ws:header-param("param0", "{$group}")
%ws:header-param("param1", "{$position}")
%ws:header-param("param2", "{$gameID}")
%ws:header-param("param3", "{$userID}")
%updating
function xlinkws:subscribeGame($group, $position, $gameID, $userID){
    websocket:set(websocket:id(), "group", $group),
    websocket:set(websocket:id(), "position", $position),
    websocket:set(websocket:id(), "gameID", $gameID),
    websocket:set(websocket:id(), "userID", $userID),
    update:output(trace(concat("/xlink/game ",$gameID, "with id ",
        ws:id(), " and userID ", $userID, " subscribed to ", $group, "/", $position)))
};
```

```
>>> SUBSCRIBE
param0:xlink
param1:game
param2:new_game_212021958
param3:211947472
id:id1
destination:/xlink/game
```

Ein User abonniert einen Channel und die key/values werden entsprechend gespeichert. Über diese Channel kann dem User später neue Inhalte geschickt werden.

Websocket Activity



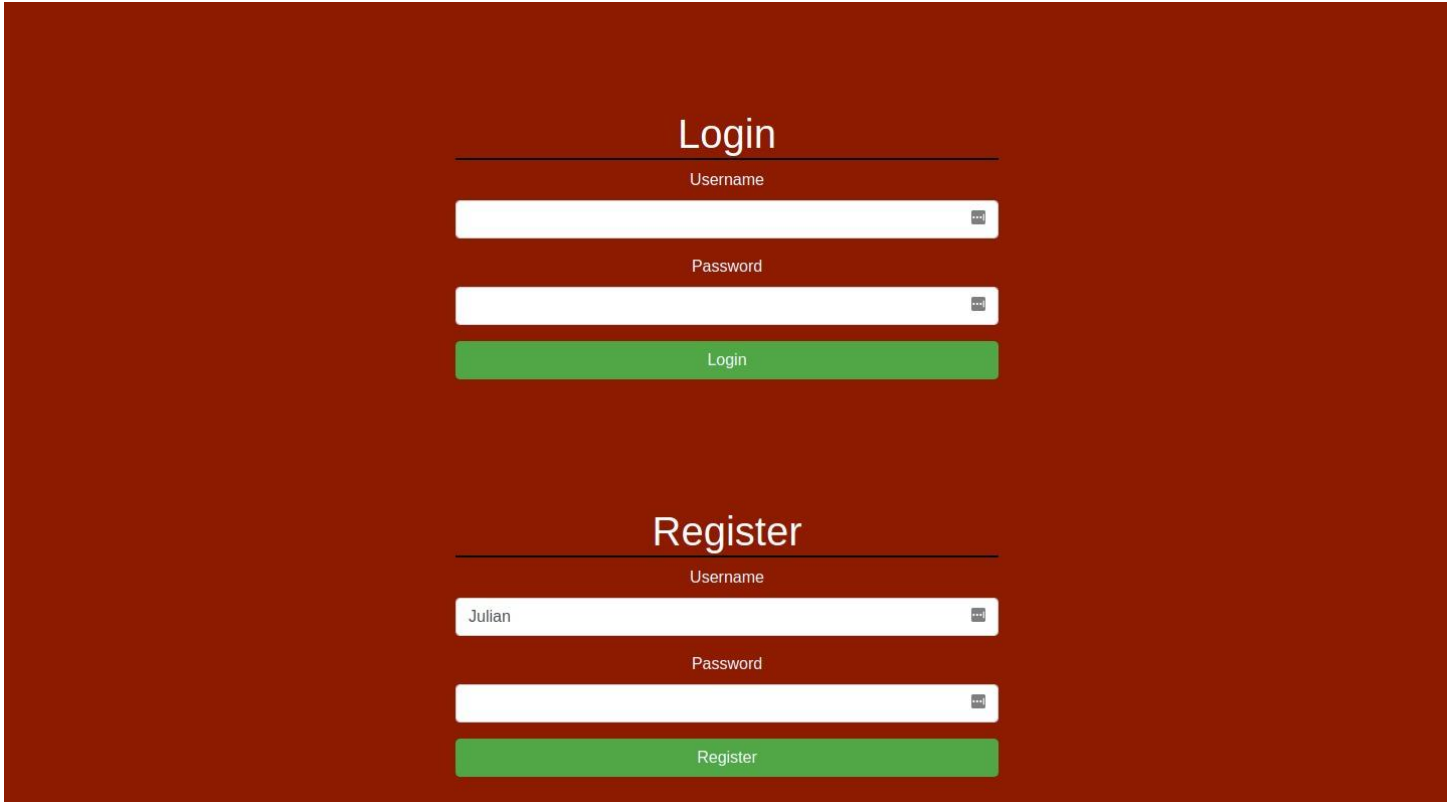

```
▼<body style="background-color: darkred">
  ▼<ws-stream id="xlink_blackjack" url="ws://localhost:
  8984/ws/xlink/game" subscription="/xlink/game/
  new_game_212021958/211947472" geturl="http://localhost:
  8984/xlink/drawGame/new_game_212021958/211947472">
```

Bei jedem neuen Laden “drawGame()” wird das neue Spiel an alle Spieler mit key/value *position = “game”* und *gameID = gameID* gesendet

```
let $wsIDs :=
    for $wsID in ws:getIDs()
    where ws:get($wsID, 'position') = 'game'
        and ws:get($wsID, 'gameID') = $gameID
    return ws:get($wsID, 'userID')

for $wsID in $wsIDs
    let $userID := $wsID
    let $isActive := $game//player[info/userID = $userID]/info/isActive
    let $username := helper:getUsername($userID)
    let $params := map {'userID' : $userID, 'username' : $username, 'isActive' : $isActive}
    let $template := fn:doc('./XSL/BlackJack.xsl')
    let $content := xslt:transform($game, $template, $params)

    let $destinationPath := concat('/xlink/game/', $gameID, '/', $userID)
    return (ws:send($content, $destinationPath))
```

4. Überblick über die Spiellogik

Von der login.xsl...



… zu der lobby.xsl

Verwendung von Bootstrap für besseres UI (1)

Sowohl beim Einloggen...

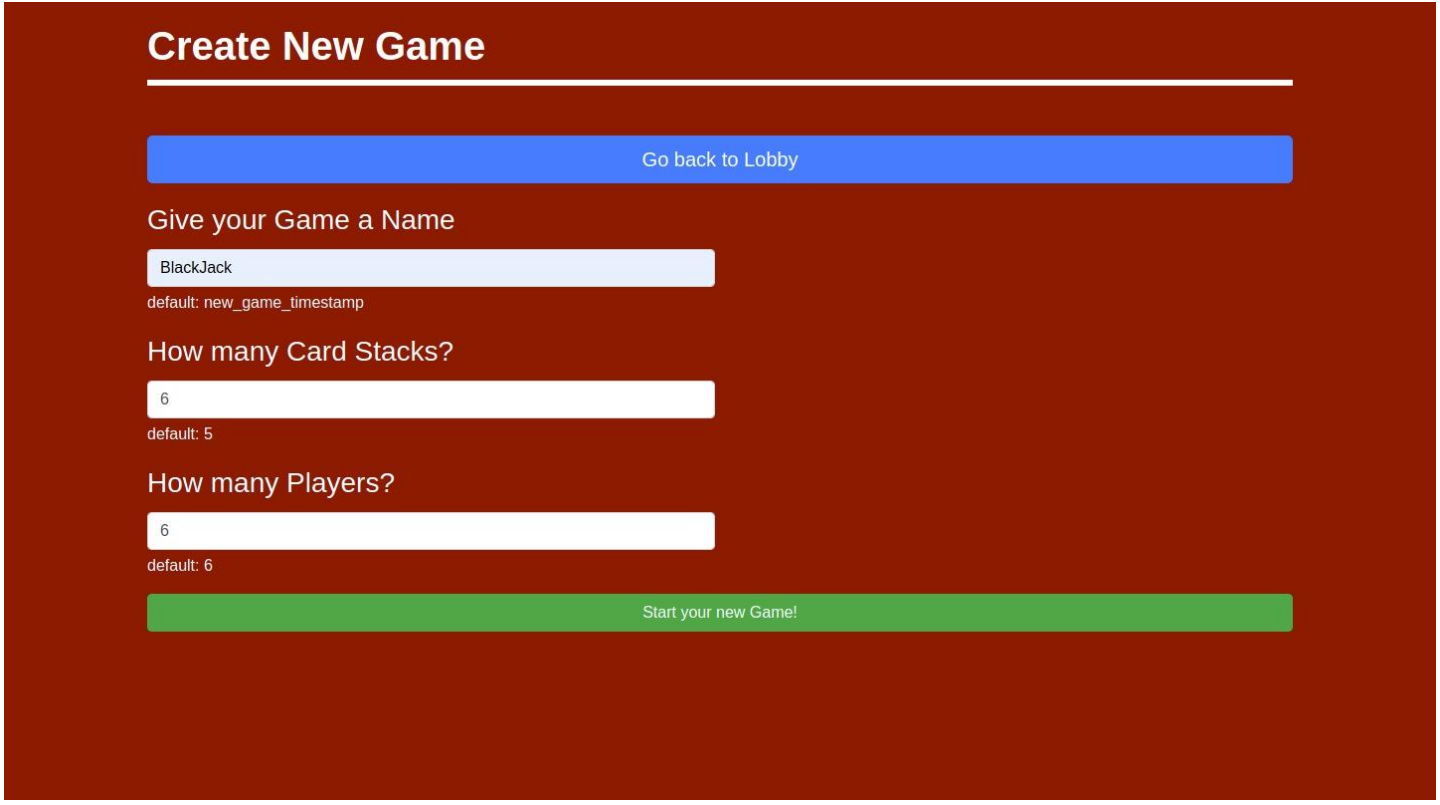
```
<div class='row text-white' style='margin-top: 10%'>
    <div class='col-md-6 offset-md-3 text-center'>
        <h1 style='border-bottom: solid black 2px'>Login</h1>
        <form role='form' method='post' action='/xlink/login' enctype="application/x-www-form-urlencoded">

            <div class="form-group">
                <label for='username'> Username </label>
                <input class="form-control" name='username' type='text' id='username'/>
            </div>

            <div class="form-group">
                <label for='password'> Password </label>
                <input class="form-control" name='password' type='text' id='password'/>
            </div>
            <button type='submit' class='btn btn-success btn-block'> Login </button>
        </form>
    </div>
</div>
```

Verwendung von Bootstrap für besseres UI (2)

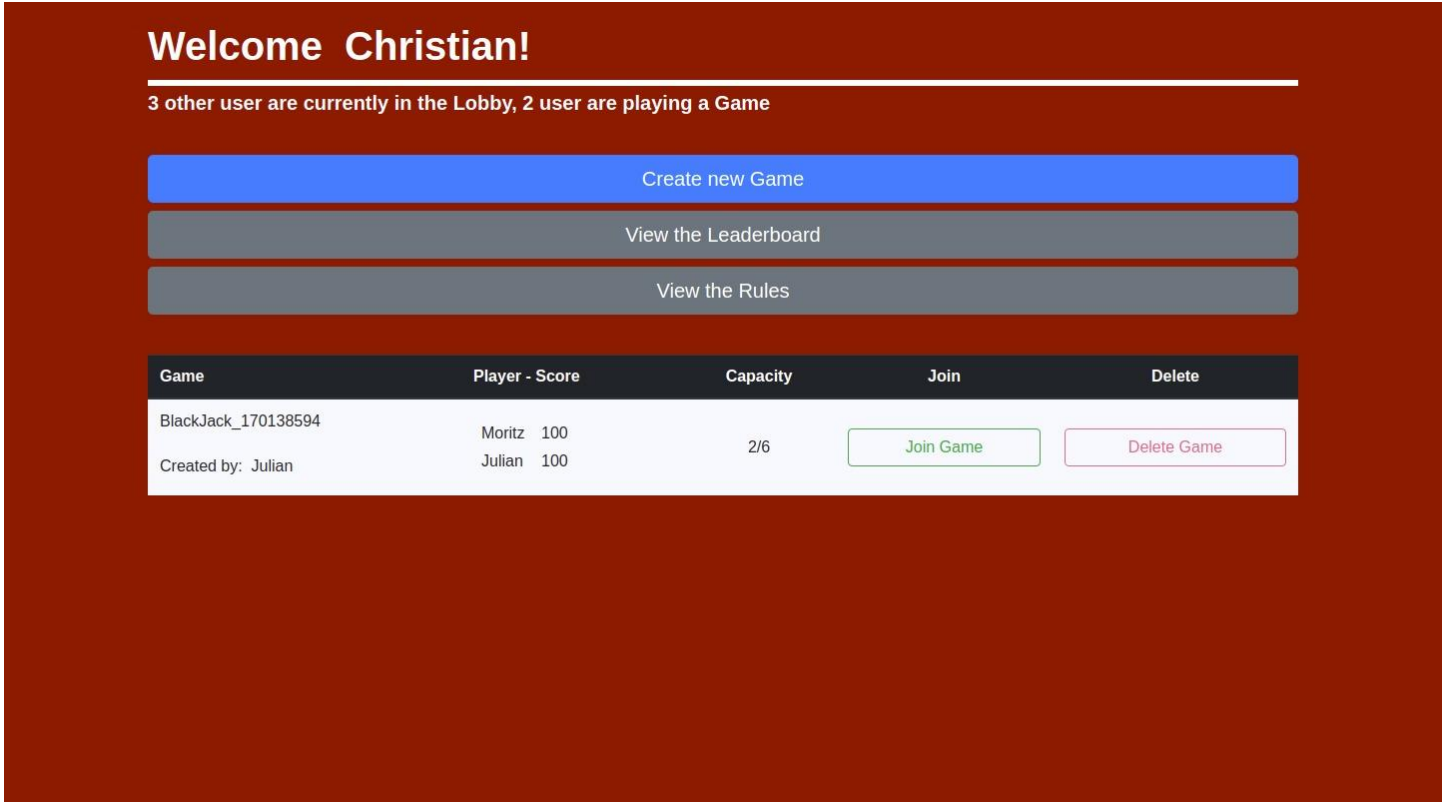
… als auch in der lobby

```
<table class='table table-hover'>
    <thead class='thead-dark'>
        <tr>
        <th >Game</th>
        <th >Player - Score</th>
        <th class='text-center'>Capacity</th>
        <th class='text-center'>Join</th>
        <th class='text-center'>Delete</th>
        </tr>
    </thead>
    <tbody>
        <xsl:for-each select='//game'>
            <xsl:variable name='gameID' select='@id' />
            <xsl:variable name='players' select='./gameInfo/numPlayers' />
            <xsl:variable name='activePlayers' select='./gameInfo/numActivePlayers' />
            <tr>
                <td style='vertical-align:middle;'>
                    <h6><xsl:value-of select='@id' /></h6>
                    <br/>
                    <h6>Created by:   <xsl:value-of select='./gameInfo/createdBy/username' /></h6>
                </td>
                <td  style='vertical-align:middle;'>

                    <table class='table-borderless table-no-hover'>
```

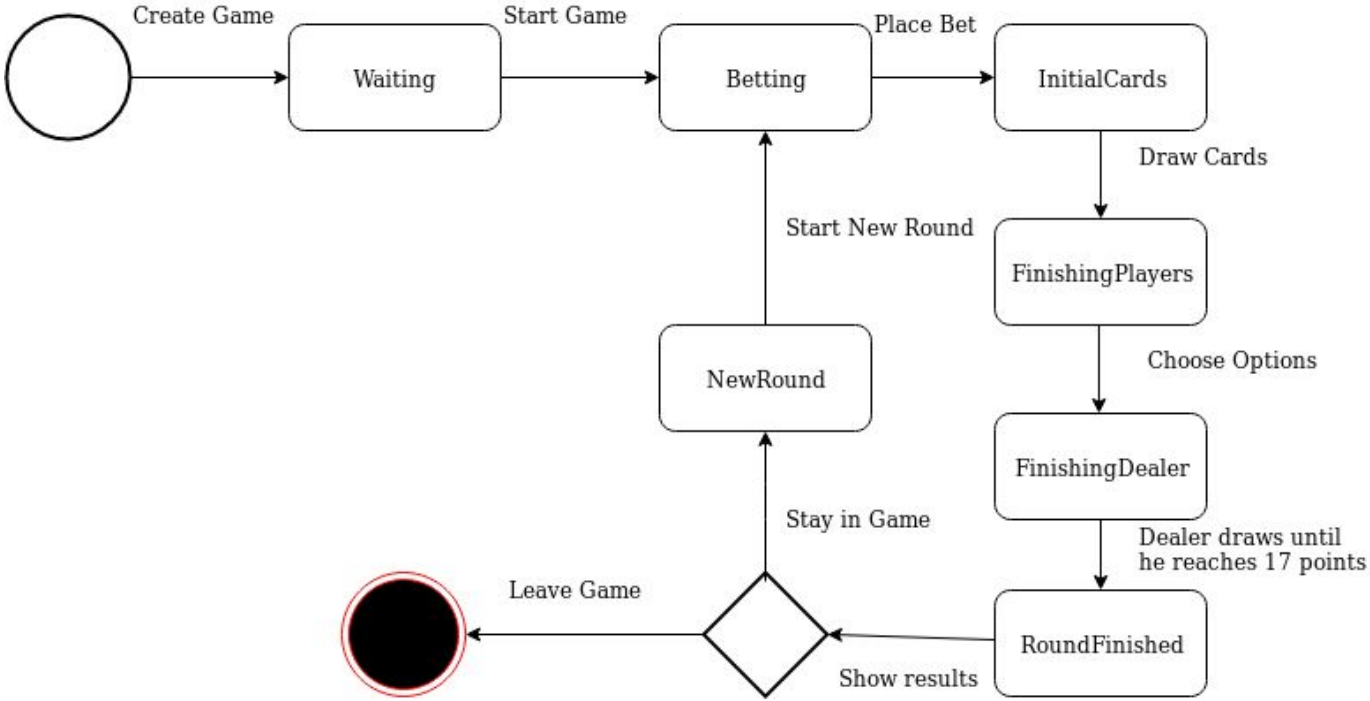
Mithilfe von newGame.xqm ein Spiel erstellen...



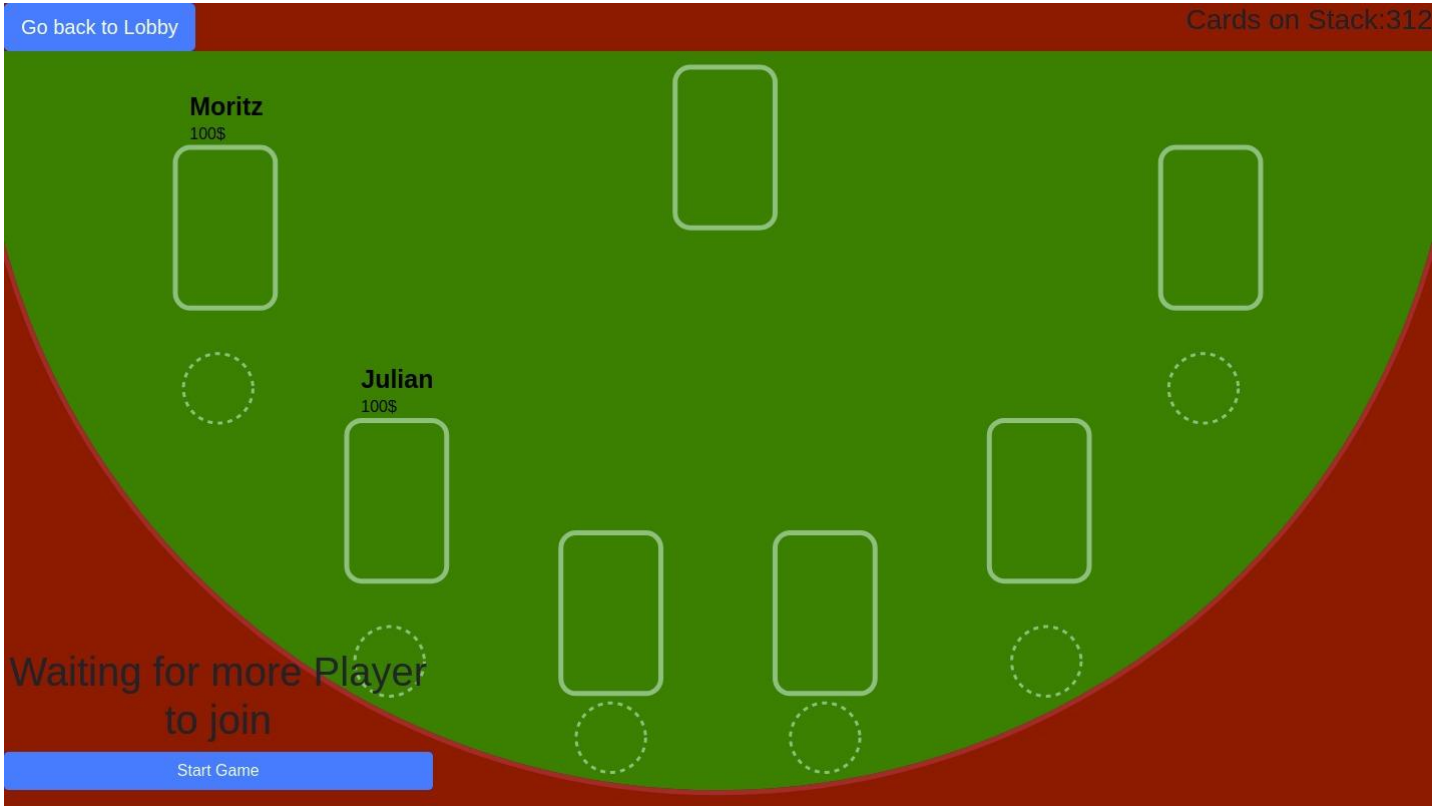
… dem andere Spieler beitreten können



State Chart Diagram

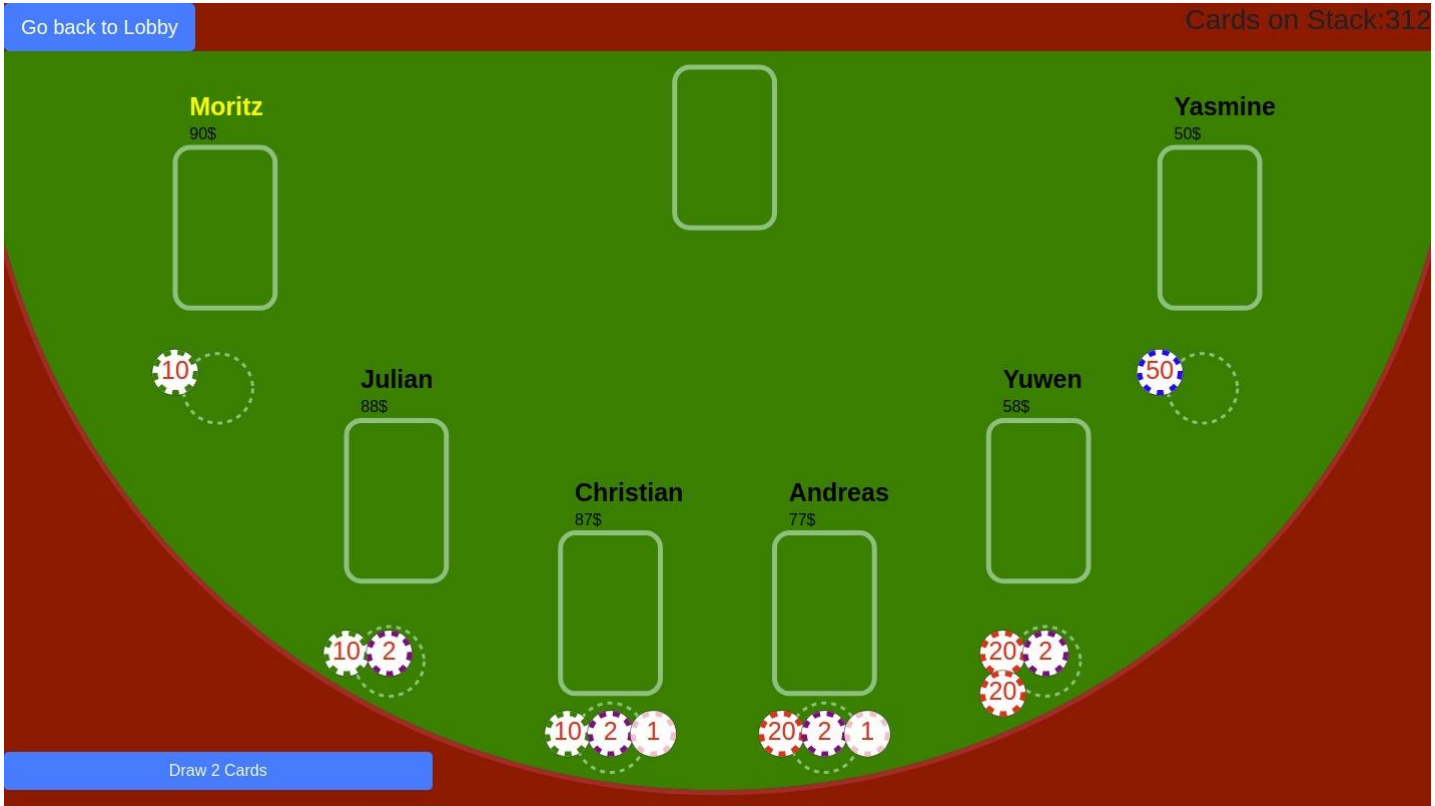


State: Waiting



State: Betting...

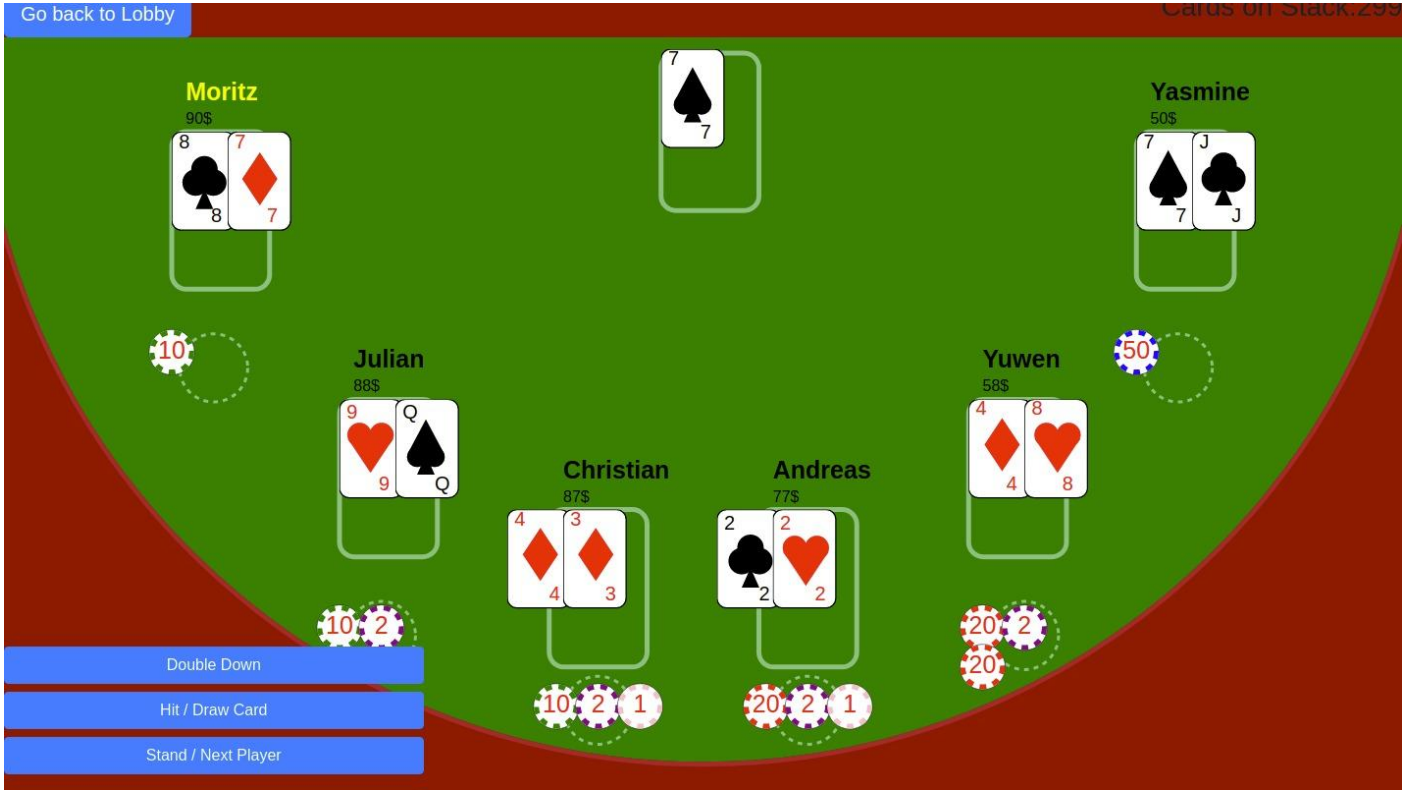
State: ...Betting



Ordnet Coins für jeden Spieler zu

```
<g id="PlayerCoins">
    <xsl:for-each select='game//player'>
        <xsl:variable name='player_id' select='@player_id' />
        <xsl:variable name='pos' select='$position//pos[@nr = $player_id]' />
        <xsl:for-each select=".//coin[@amount >= 1]">
            <text> <xsl:value-of select='$pos/firstCoin/@x' /> </text>
            <xsl:choose>
                <xsl:when test="@amount = 2">
                    <use href="{concat('#coin','_',@value)}" transform="scale(0.5)"
                        x="{($pos/firstCoin/@x + (3 * (position()- 1))) * 2}%"
                        y="{$pos/firstCoin/@y}%" />
                    <use href="{concat('#coin','_',@value)}" transform="scale(0.5)"
                        x="{($pos/firstCoin/@x + (3 * (position()- 1))) * 2}%"
                        y="{$pos/firstCoin/@y +10}%" />
                </xsl:when>
                <xsl:otherwise>
                    <use href="{concat('#coin','_',@value)}" transform="scale(0.5)"
                        x="{($pos/firstCoin/@x + (3 * (position()- 1))) * 2}%"
                        y="{$pos/firstCoin/@y}%" />
                </xsl:otherwise>
            </xsl:choose>
        </xsl:for-each>
    </xsl:for-each>
</g>
```

State: FinishingPlayers

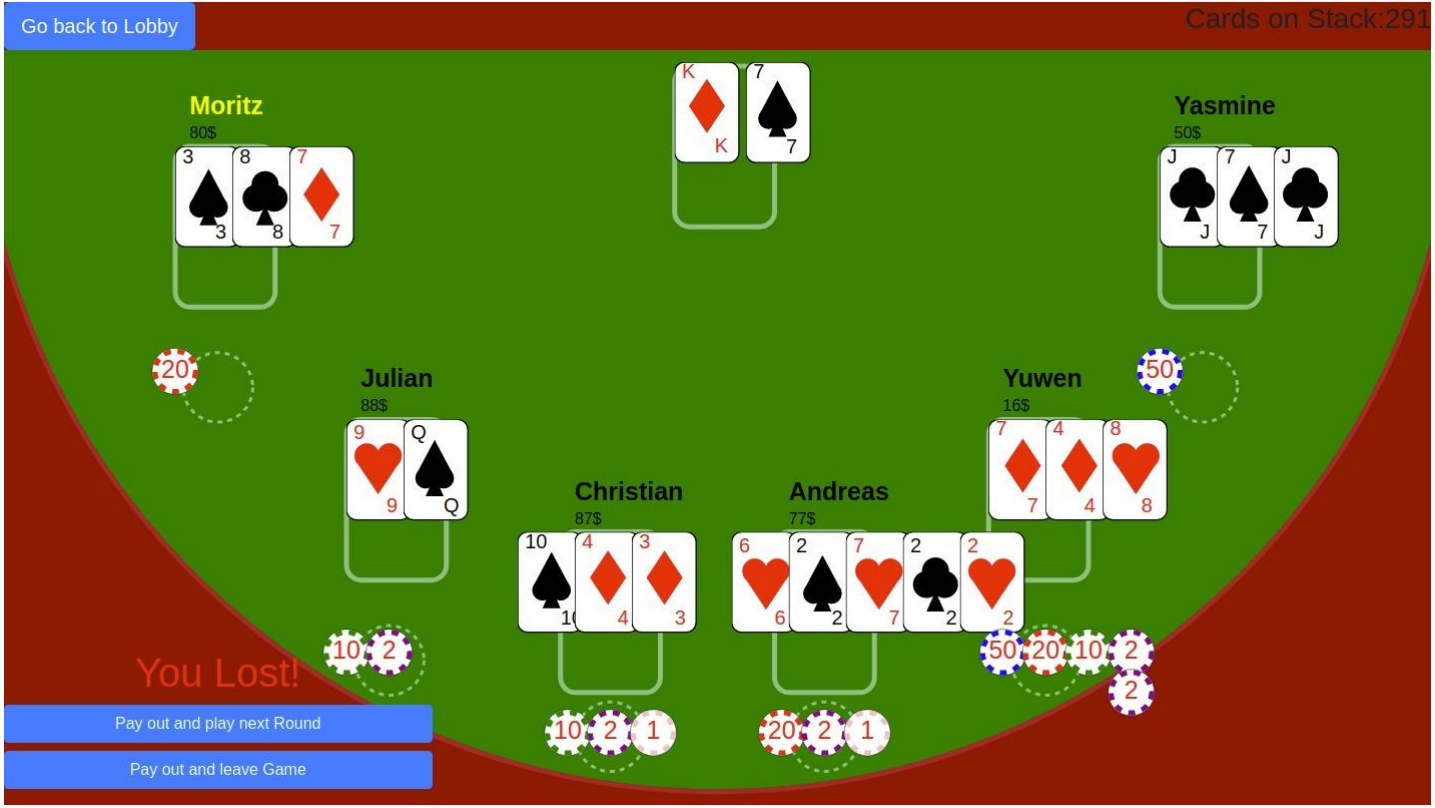


Karten : Position

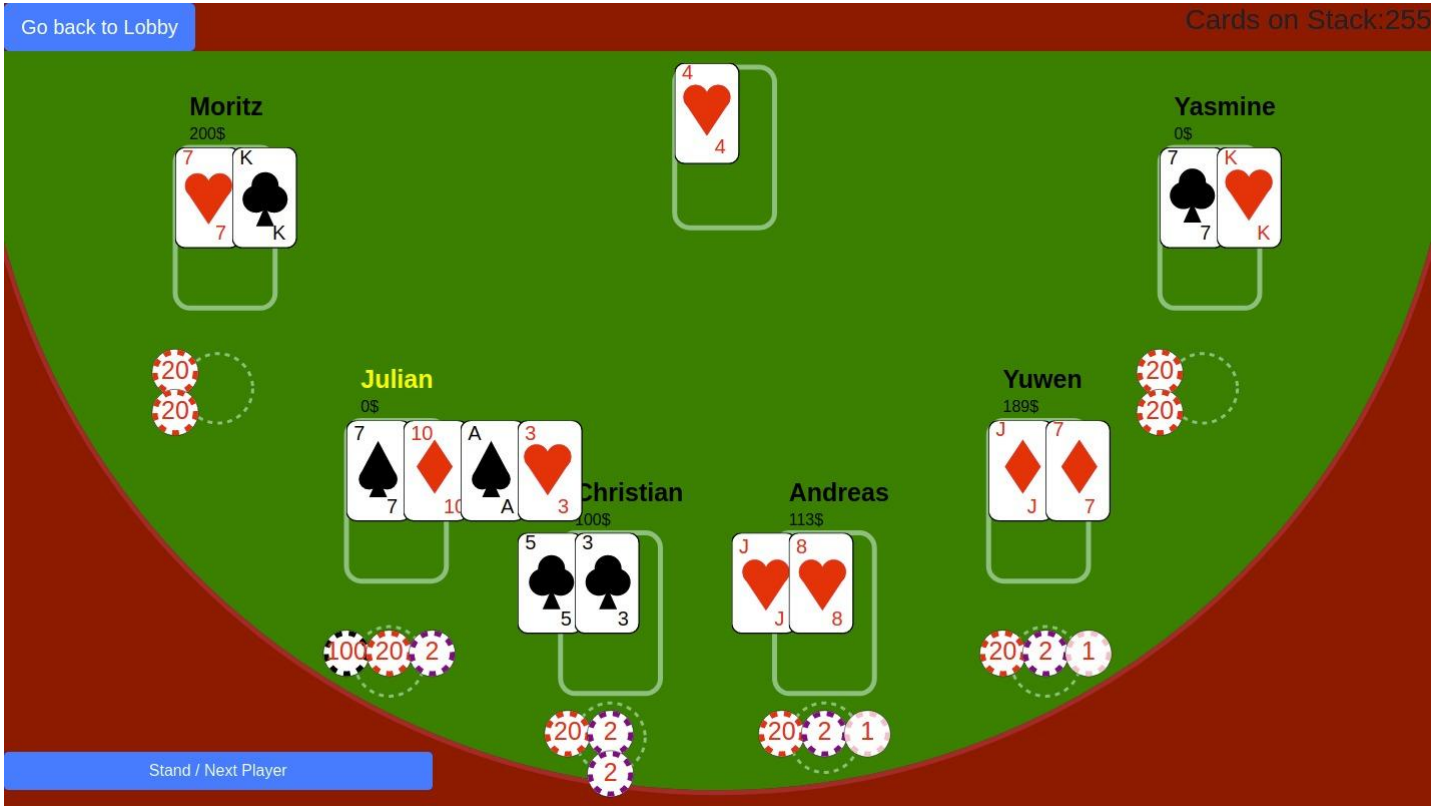
Dynamisches zusammensetzen und Platzierung über die Position der Spieler

```
<!-- show the cards -->

<g id="PlayerCards">
    <xsl:for-each select='game//player'>
        <xsl:variable name='player_id' select='@player_id' />
        <xsl:for-each select='.//card' >

            <use  href="#cardShape"
                x="{($position//pos[@nr = $player_id]/cardShape/@x + (4 * (position()- 1))) * 2}%"
                y="{$position//pos[@nr = $player_id]/cardShape/@y}%"
                transform="scale(0.5)">
            </use>

            <use href="#{@color}" transform="scale(0.45)" height="100%" width="100%"
                x="{($position//pos[@nr = $player_id]/cardColor/@x + (4.4 * (position()- 1))) * 2}%"
                y="{$position//pos[@nr = $player_id]/cardColor/@y}%"
                 />

            <xsl:if test="@color='diamond' or @color='heart' " >

                <text color="red" fill="red" font-size="250%" transform="scale(0.5)"
                    x="{($position//pos[@nr = $player_id]/cardValueTop/@x + (4 * (position()- 1))) * 2}%"
                    y="{$position//pos[@nr = $player_id]/cardValueTop/@y}%" >
                    <xsl:value-of select="@name"/>
                </text>
                <text color="red" fill="red" font-size="250%" transform="scale(0.5)"
                    x="{($position//pos[@nr = $player_id]/cardValueBottom/@x + (4 * (position()- 1))) * 2}%"
                    y="{$position//pos[@nr = $player_id]/cardValueBottom/@y}%" >
                    <xsl:value-of select="@name"/>
                </text>

            </xsl:if>

            <xsl:if test="@color='club' or @color='spade' " >
                <text color="black" fill="black" font-size="250%" transform="scale(0.5)"
                    x="{($position//pos[@nr = $player_id]/cardValueTop/@x + (4 * (position()- 1))) * 2}%"
                    y="{$position//pos[@nr = $player_id]/cardValueTop/@y}%" >
                    <xsl:value-of select="@name"/>
                </text>

                <text color="black" fill="black" font-size="250%" transform="scale(0.5)"
                    x="{($position//pos[@nr = $player_id]/cardValueBottom/@x + (4 * (position()- 1))) * 2}%"
                    y="{$position//pos[@nr = $player_id]/cardValueBottom/@y}%" >
                    <xsl:value-of select="@name"/>
                </text>
            </xsl:if>

        </xsl:for-each>
    </xsl:for-each>
</g>
```

State: RoundFinished



Nur erlaubte Spielzüge sind möglich…



State: NewRound

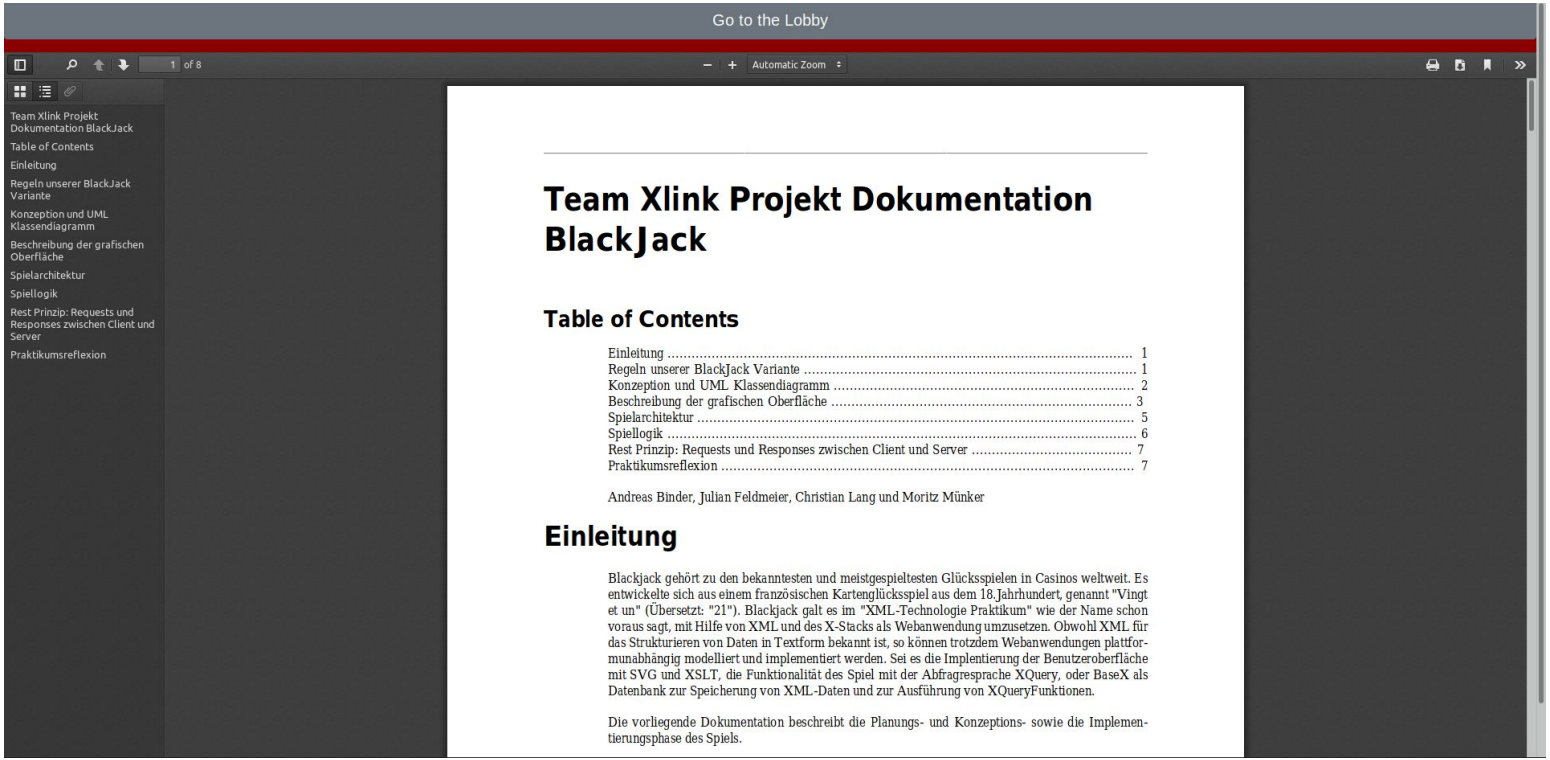
5. Show Case

Schritte zum Starten des Spiels

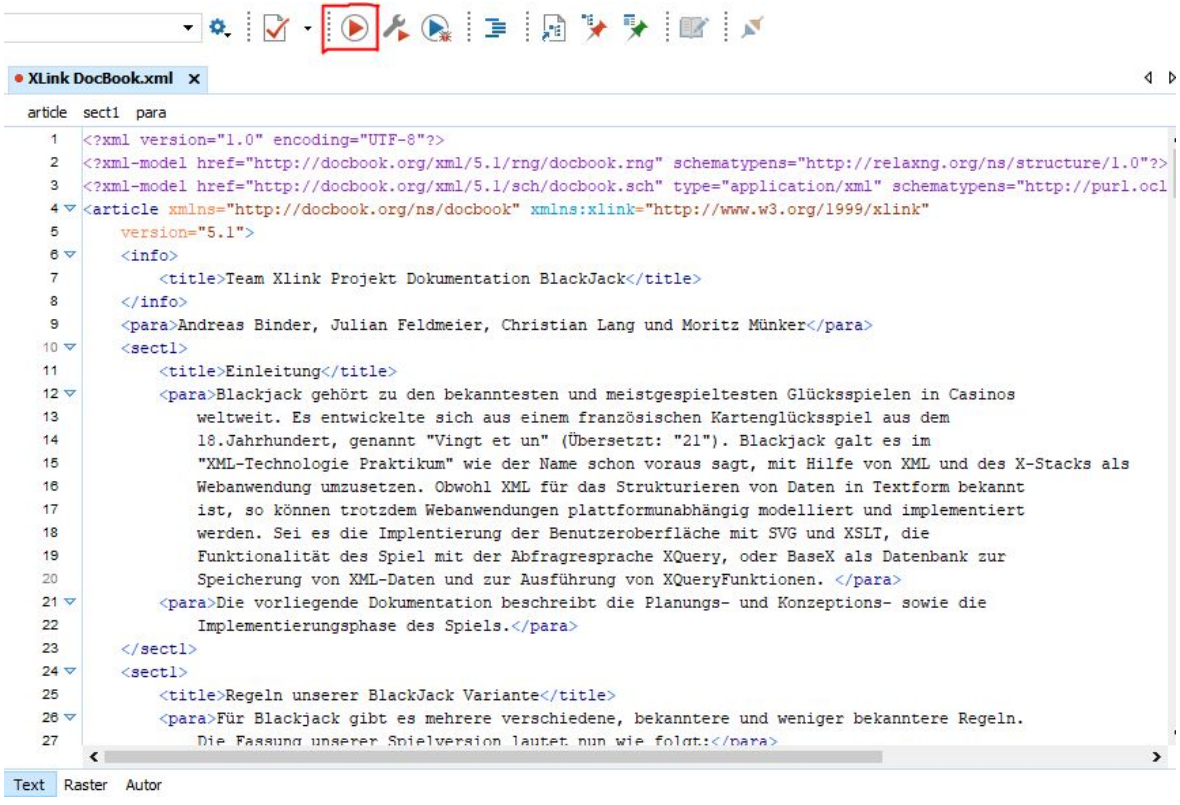
1. Ausführen des Befehls “mvn jetty:run” in basex-api Ordner
2. Aufrufen von “http://localhost:8984/xlink”
3. Sofern neu installiert, “http://localhost:8984/xlink/setup”
   a. zum resetten: “http://localhost:8984/xlink/reset”

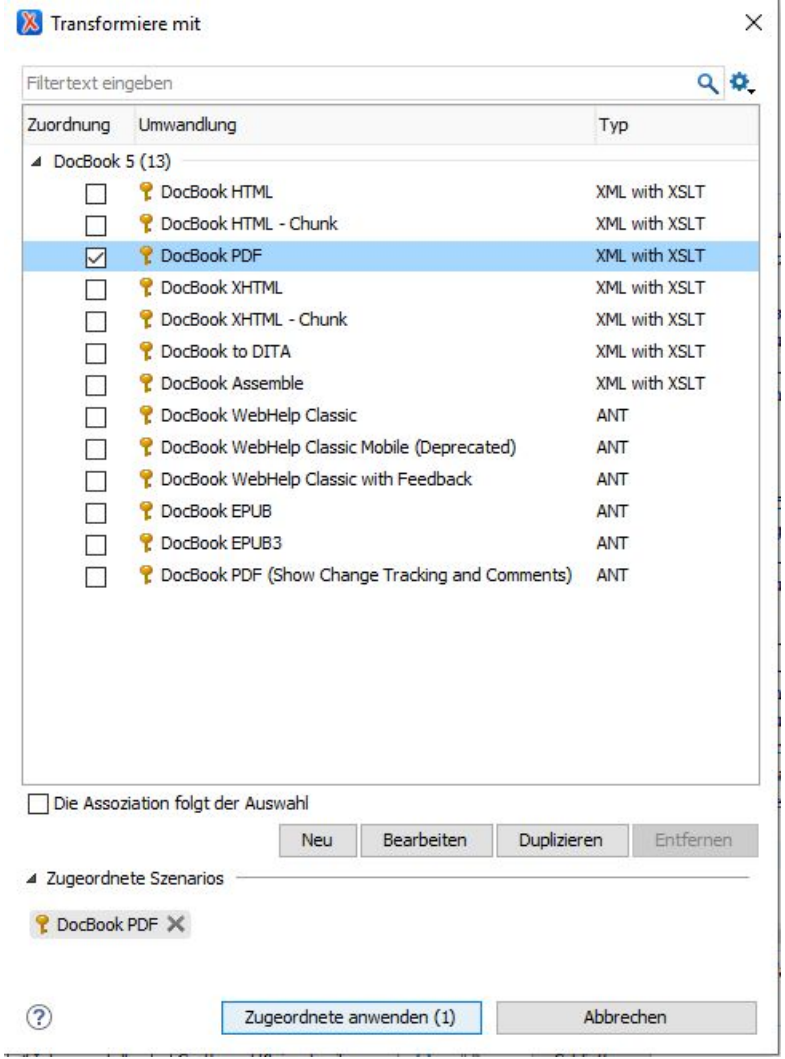
6. Vorstellen des DocBooks

Von der Lobby aus einsehbar...



Export als PDF mit Oxygen




Transformiere mit
Filtertext eingeben
Zuordnung
Umwandlung
Typ
DocBook 5 (13)
DocBook HTML
XML with XSLT
DocBook HTML - Chunk
XML with XSLT
DocBook PDF
XML with XSLT
DocBook XHTML
XML with XSLT
DocBook XHTML - Chunk
XML with XSLT
DocBook to DITA
XML with XSLT
DocBook Assemble
XML with XSLT
DocBook WebHelp Classic
ANT
DocBook WebHelp Classic Mobile (Deprecated)
ANT
DocBook WebHelp Classic with Feedback
ANT
DocBook EPUB
ANT
DocBook EPUB3
ANT
DocBook PDF (Show Change Tracking and Comments)
ANT
Die Assoziation folgt der Auswahl
Neu
Bearbeiten
Duplizieren
Entfernen
Zugeordnete Szenarios
DocBook PDF
Zugeordnete anwenden (1)
Abbrechen

7. Resume zum Praktikum

Vorgehen und Zusammenarbeit

- Arbeitsteilung
  - Je nach Kenntnisstand erfolgte eine Aufteilung
  - Trotzdem hat jeder an allem in irgendeiner Art und Weise mitgeholfen
- Meetings
  - Fanden ca alle zwei Wochen statt
  - benutzen dabei hauptsächlich die Studitum Gruppenräume zum Arbeiten
- Kommunikation
  - Neben den Meetings reger Kontakt über Whatsapp-Gruppe
- Versionsverwaltung
  - Das Projekt wurde jedem über gitlab zugängig gemacht

Lernergebnisse

- Software Entwicklungsprozess von Entwurf bis Implementierung, sowie mit Dokumentation
- Arbeiten mit gesamten XStack
- Darüber hinaus Restful Services und Websockets
- Schrittweises Heranführen über Übungsblätter erlauben Vertiefung