



FH Bielefeld
University of
Applied Sciences

Campus Minden

Webbasierte Anwendungen

SS 2018

Web-Bibliotheken und Frameworks

Dozent: B. Sc. Florian Fehring
mailto: florian.fehring@fh-bielefeld.de

Studiengang Informatik

Web-Bibliotheken und Frameworks

1. Kontext und Motivation

2. Bibliotheken und Frameworks

3. jQuery

4. Bootstrap

5. Knockout.js

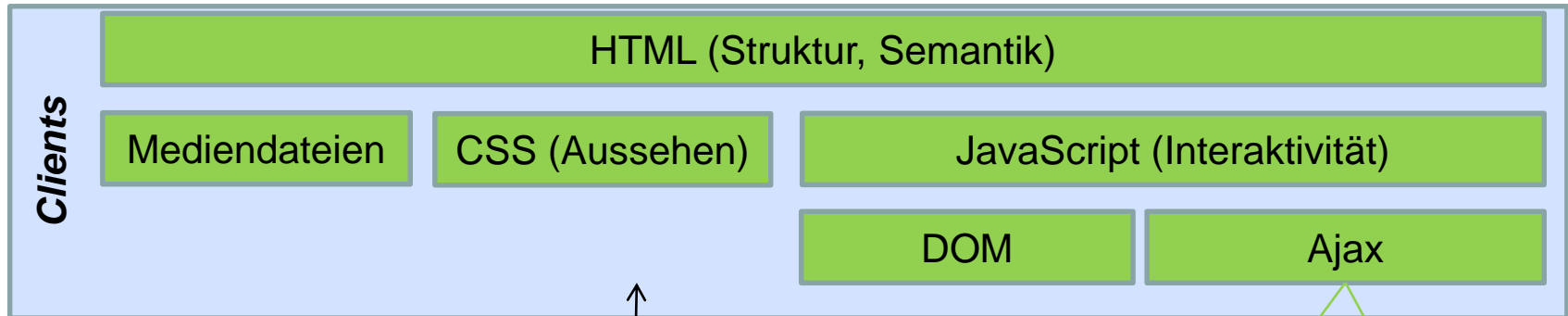
6. Angular

7. Darüber hinaus

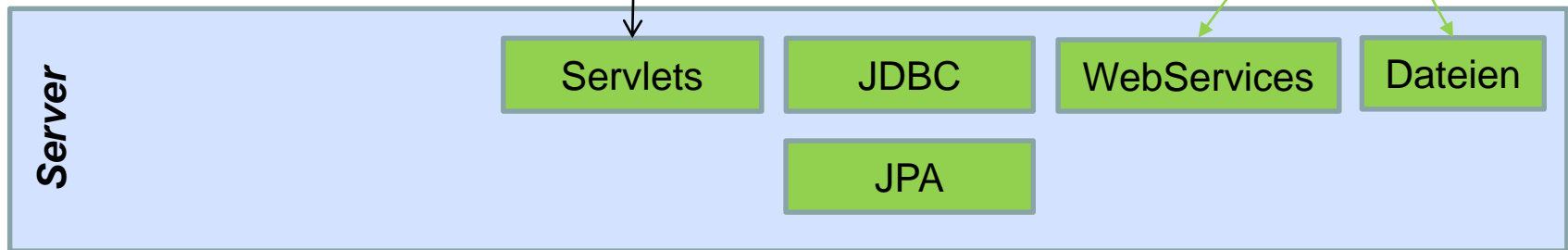
8. Projekt

Problemfelder

Mensch-Maschine-Kommunikation



Maschine-Maschine-Kommunikation



Web-Bibliotheken und Frameworks

1. Kontext und Motivation

2. Bibliotheken und Frameworks

3. jQuery

4. Bootstrap

5. Knockout.js

6. Angular

7. Darüber hinaus

8. Projekt

Bibliotheken

Definition: Bibliotheken für Webanwendungen stellen Funktionen zur Verfügung.

Bibliotheken sollen die Implementierung erleichtern, indem sie häufig verwendete Algorithmen fertig implementiert zur Verfügung stellen.

Bibliotheken werden häufig in Frameworks verwendet.

Eigenschaften:

- Anwendungsbezogen
 - Bibliotheken haben einen fest umrissenen Anwendungsbereich
- Programm-gesteuert
 - Funktionen der Bibliotheken werden vom Programm aufgerufen

Bibliotheken

Definition: Bibliotheken für Webanwendungen stellen Funktionen zur Verfügung.

Bibliotheken haben einen bestimmten Anwendungsbereich und sollen die Implementierung erleichtern, indem sie häufig verwendete Algorithmen fertig implementiert zur Verfügung stellen.

Bibliotheken werden häufig in Frameworks verwendet.

Bibliotheken nach Zielsetzung:

- UI Element-Bibliotheken
 - Stellen einzelne, komplexere UI Elemente zur Verfügung (z.B. Datenvisualisierungen)
- Vereinfachung
 - Zielen darauf ab die Implementierung zu beschleunigen
 - Stellen häufig verwendete Algorithmen zur Verfügung
 - Vereinfachen die Benutzung von Funktionen
- Kompatibilitätsbibliotheken
 - Zielen darauf ab Funktionen in älteren Browsern zur Verfügung zu stellen, die nur in neueren Browsern verfügbar sind

Frameworks

Definition: Frameworks für Webapplikationen stellen ein Rahmenwerk zur Umsetzung einer Architektur zur Verfügung. Sie beeinflussen den Entwicklungsprozess.

Eigenschaften:

- Allgemein verwendbar
 - Frameworks legen keine Art der Anwendung fest, sie geben nur einen Rahmen vor
- Inversion of Control
 - Das Framework ruft die Anwendungslogik auf, wenn sie benötigt wird
 - Das Framework gibt die Strukturen vor, in welche die Anwendungslogik implementiert wird

Frameworks

Definition: Frameworks für Webapplikationen stellen ein Rahmenwerk zur Umsetzung einer Architektur zur Verfügung. Sie beeinflussen den Entwicklungsprozess.

Frameworks nach Technologie:

- CSS Frameworks
 - Bieten Hilfsmittel und Gestaltungselemente für das Webseiten-Layout
 - Hauptbestandteil ist meist ein Grid-System für Responsive Webdesign
 - Beinhalten manchmal JavaScript für die Realisierung von Gestaltungselementen
- JavaScript Frameworks
 - Vereinfachen die Nutzung von JavaScript Funktionen
 - Vereinfachen den Zugriff und die Manipulation des DOM
 - Bieten umfangreiche Funktionsbibliotheken

Frameworks nach Zielsetzung:

- Design Frameworks
 - Zielen ab auf die Gestaltung der Benutzeroberfläche
 - Meistens Frameworks, die Webseiten auf Mobile Geräte und große gleichermaßen bringen
- Architekturframeworks
 - Zielen darauf ab ein bestimmtes Architekturmuster zu unterstützen

Web-Bibliotheken und Frameworks

1. Kontext und Motivation
2. Bibliotheken und Frameworks
- 3. jQuery**
4. Bootstrap
5. Knockout.js
6. Angular
7. Darüber hinaus
8. Projekt

jQuery

Definition: jQuery ist eine JavaScript Bibliothek, die im wesentlichen den Zugriff auf das DOM einfacher gestaltet



Eigenschaften

- Vereinfachungs und Kompatibilitäts-Bibliothek
- Meistverwendete JavaScript-Bibliothek
- Veröffentlicht 2006
- Konzepte von jQuery fanden Einzug in JavaScript6

Vorteile

- Verkürzte Schreibweise und vereinheitlichter Zugriff auf DOM-Elemente
- Kompensation browserabhängiger Besonderheiten
- Erweitertes Event-System
- Umfangreiche Bibliothek
- Erweiterbarkeit durch Plugins (z.B. jQueryUI für Oberflächen)

Nachteile

- Implementierungen neigen leicht zu Spaghetti-Code
- Teilweise überflüssig durch neue JavaScript-Funktionen

Weitere Informationen: www.jquery.com

jQuery II – DOM Zugriff

`$(CSS-Selector).jQueryFunktion()`

Eigenschaften

- Zugriff auf DOM-Elemente in Kurzschreibweise
- Zugriff über CSS3-Selektoren
- Zurückgelieferte Objekte sind jQuery-Objekte mit entsprechenden jQuery Funktionen

```
// Setzt den Inhalt eines Elements mit der id ausgabe  
$("#ausgabe").html("Du hast Button 1 geklickt");});
```

jQuery III – DOM Schutz

`$(document).ready()`

Eigenschaften

- Stellt sicher, dass das komplette Dokument geladen ist
- Code in der ready-Funktion wird danach ausgeführt

```
// Erst nach dem vollständigen Laden der Seite ausführen
$(document).ready(function () {
    // Setzt den Inhalt eines Elements mit der id Ausgabe
    $("#ausgabe").html("Das Dokument ist fertig geladen");
});
```

jQuery IV – Event-Handling

`$(CSS-Selektor).on("event",function)`

Eigenschaften

- Einfaches Binden einer Funktion an ein Event

```
// Erst nach dem vollständigen Laden der Seite ausführen
$(document).ready(function () {
    $(function() {
        $("div.test a").on('click', function() {
            alert("Hello world!");
        });
    });
});
```

jQuery V – Asynchrone Anfragen

```
$.ajax({RequestObject});
```

Eigenschaften

- Einfachere Verwendung als mit XMLHttpRequest-Objekt
- Angabe von Typ, URL und Daten als Attribute

```
$.ajax({  
  type: "POST",  
  url: "beispiel.php",  
  data: "name=Mustermann&location=Berlin",  
}).done(function(response) {  
  alert("Daten gespeichert: " + response);  
});
```

Web-Bibliotheken und Frameworks

1. Kontext und Motivation
2. Bibliotheken und Frameworks
3. jQuery
- 4. Bootstrap**
5. Knockout.js
6. Angular
7. Darüber hinaus
8. Projekt

Bootstrap

Definition: Bootstrap ist ein HTML-CSS Framework für die Gestaltung von Benutzeroberflächen



Eigenschaften

- Design- und CSS-Framework (mit JavaScript Erweiterungen)
- Entwickelt von Twitter
- Veröffentlicht 2011
- Optionale JavaScript-Erweiterungen
- Konzept der progressiven Verbesserung: Gestaltungselemente von neueren Browsern werden genutzt aber die Funktionalität bleibt auch in älteren Browsern erhalten
- Modularer Aufbau

Vorteile

- Unterstützt Responsive Design
- Verwendbar mit älteren Browsern
- Umfang der Datei für die eigenen Anwendung ist durch den modularen Aufbau und das Zusammenstellen in einer bootstrap-Datei bestimmbar.

Weitere Informationen: www.getbootstrap.com

Bootstrap II - Integration

Einbindung in Projekte

- Laden von:
- bootstrap.css (Hauptdatei des Frameworks)
- jQuery (in Bootstrap genutzt)
- popper.js (Bibliothek zum einfachen Platzieren von Elementen)
- bootstrap.js (Bootstraps JavaScript-Komponenten)

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">

<script
src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js
"></script>

<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></sc
ript>
```

Bootstrap III - Layout

Eigenschaften

- Container-basiert
- 12-spaltiges Grid System
- Automatische gleichmäßige Aufteilung
- Oder Aufteilung nach Anteilen von 12
- Responsive Layout

```
<div class="container-fluid">  
  Hallo Fluid-container!  
</div>  
<br />
```

```
<div class="container">  
  <div class="row">  
    <div class="col">  
      1 of 2  
    </div>  
    <div class="col">  
      2 of 2  
    </div>  
  </div>  
  <div class="row">  
    <div class="col">  
      1 of 3  
    </div>  
    <div class="col">  
      2 of 3  
    </div>  
    <div class="col">  
      3 of 3  
    </div>  
  </div>  
</div>
```

Hallo Fluid-container!

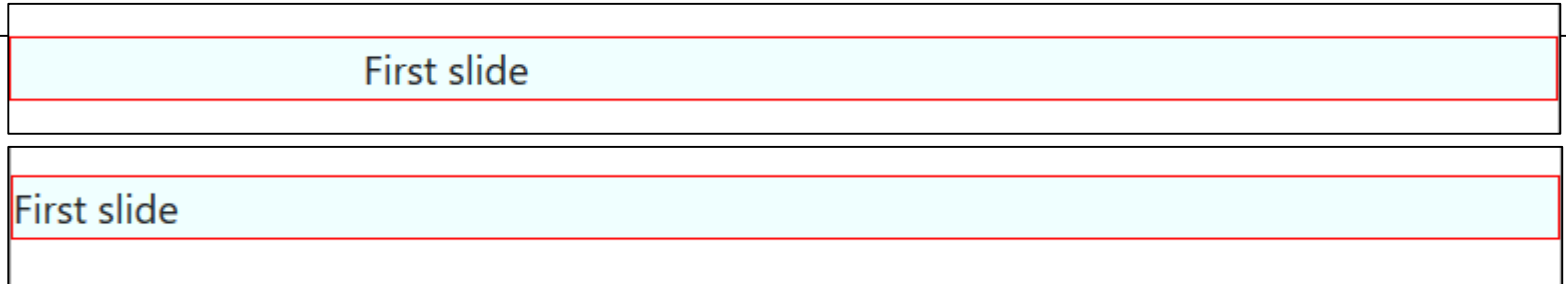
1 of 2		2 of 2	
1 of 3	2 of 3		3 of 3

Bootstrap IV - Komponenten

Eigenschaften

- Zahlreiche Komponenten
- Layout Komponenten (z.b. Card, Badage,...)
- Präsentations Komponenten (z.B. Carousel, ...)
- Dialog-Komponenten (z.B. Modal, Navbar,...)

```
<div id="carouselExampleSlidesOnly" class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      
    </div>
    <div class="carousel-item">
      
    </div>
    <div class="carousel-item">
      
    </div>
  </div>
</div>
```



Web-Bibliotheken und Frameworks

1. Kontext und Motivation
2. Bibliotheken und Frameworks
3. jQuery
4. Bootstrap
- 5. Knockout.js**
6. Angular
7. Darüber hinaus
8. Projekt

Knockout.js

Definition: Knockout.js ist ein JavaScript-Framework für die Umsetzung des MVVM-Entwicklungsmusters



Eigenschaften

- Umsetzung des Model-View-ViewModel-Musters auf Client-Seite
- Entwickelt von einem Microsoft Entwickler
- Verwendung von HTML5 und JavaScript
- Benötigt keine anderen Bibliotheken
- Deklarative Datenbindung
- WerteBindung in der View mit Beobachter Pattern
- Dependency Tracking
- HTML-Templates

Weitere Informationen: www.knockoutjs.com

Knockout.js II – Deklarative Datenbindung

`data-bind="binding: value"`

Eigenschaften

- Deklarative Bindung über data-bind-Attribut an beliebigen HTML-Elementen
- Werte werden aus normalem Java-Script Objekt ausgelesen und an deklarerter Stelle im View eingefügt

Bestandteile:

<code>data-bind</code>	Attribut eines HTML-Elements
<code>binding</code>	Name der Bindungs-Art
<code>value</code>	Name des Attributs, das gebunden werden soll

Bindungsarten:

- Text und Darstellungs-Bindungen
- Kontroll-Anweisungs-Bindungen
- Formular-Feld-Bindungen

Knockout.js III – Text-Bindung

`data-bind="text: attributename"`

Eigenschaften

- Fügt einen Text als Inhalt des Elements ein

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <script type="text/javascript" src="knockout-3.4.2.js"></script>
    <script src="knockoutjs-databinding.js"></script>
    <title>Knockout Beispiel</title>
  </head>
  <body>
    Hello <span data-bind="text: personName"></span>!
  </body>
</html>
```

```
var myViewModel = {
  personName: 'Bob',
  personAge: 123
};

window.onload = function() {
  ko.applyBindings(myViewModel);
}
```

Hello Bob!

Knockout.js IV – If-Bindung

`data-bind="if: javascript-expression"`

Eigenschaften

- Fügt den Inhalt des Elements in das Dokument ein, wenn der Javascript Ausdruck true ergibt
- Es sind beliebige Ausdrücke möglich, auch Funktionsaufrufe

```
<body>
  Hello <span data-bind="text: personName"></span>!
  <span data-bind="if: personAge > 100">Wow you are over 100 years!</span>
</body>
```

```
var myViewModel = {
  personName: 'Bob',
  personAge: 123
};

window.onload = function() {
  ko.applyBindings(myViewModel);
}
```

Hello Bob! Wow you are over 100 years!

Knockout.js IV – Value-Bindung

`data-bind="value: attributename"`

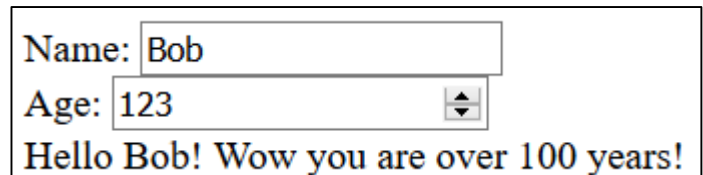
Eigenschaften

- Bindet den Wert eines Eingabefelds an ein Objekt-Attribut
- Die Bindung ist bidirektional

```
<body>
  Name: <input type="text" data-bind="value: personName"><br />
  Age: <input type="number" data-bind="value: personAge"><br />
  Hello <span data-bind="text: personName"></span>!
  <span data-bind="if: personAge > 100">Wow you are over 100 years!</span>
</body>
```

```
var myViewModel = {
  personName: 'Bob',
  personAge: 123
};

window.onload = function() {
  ko.applyBindings(myViewModel);
}
```



Name: Bob

Age: 123

Hello Bob! Wow you are over 100 years!

Knockout.js V – Observer-Pattern

```
Object {  
    attriute: ko.observeable(defaultvalue)  
}
```

Eigenschaften

- Können Standardwert besitzen oder nicht
- Werte werden zu beobachtbaren Objekten
- Werte auslesen über attribute()-Methoden
- Werte setzen über attribute(wert)-Methoden
- Beobachter werden bei Werteänderung informiert

Bestandteile:

`attribute`

Name des Objekt-Attributs

`Ko.observeable`

Erzeugt ein beobachtbares Objekt

`defaultvalue`

Standardwert des Attributs

Knockout.js IV – Observer-Pattern

```
<body>
  Name: <input type="text" data-bind="value: personName"><br />
  Age: <input data-bind="value: personAge"><br />
  Hello <span data-bind="text: personName"></span>!
  <span data-bind="if: personAge() > 100">Wow you are over 100 years!</span>
</body>
```

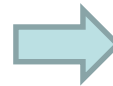
```
var myViewModel = {
  personName: ko.observable('Bob'),
  personAge: ko.observable(123)
};

window.onload = function() {
  ko.applyBindings(myViewModel);
}
```

Name:

Age:

Hello Bob! Wow you are over 100 years!



Name:

Age:

Hello Bobi!

Knockout.js IV – Computed Observeables

Eigenschaften

- Sind Funktionen
- Berechnen Werte aus anderen Attributen eines Objekts
- Benachrichtigen ihre Beobachter wie normale Attribute

```
<body>
  Name: <input type="text" data-bind="value: personName"><br />
  Age: <input data-bind="value: personAge"><br />
  Hello <span data-bind="text: personName"></span>!
  <span data-bind="if: personAge() > 100">Wow you are over 100
years!</span><br>
  Your dogage: <span data-bind="text: personDogAge"></span>
</body>
```

```
function myViewModel() {
  this.personName = ko.observable('Bob'),
  this.personAge = ko.observable(123),
  this.personDogAge = ko.computed(function() {
    return this.personAge() * 7;
  }, this)
};

window.onload = function() {
  ko.applyBindings(myViewModel);
}
```

Name: Bob
Age: 123
Hello Bob! Wow you are over 100 years!
Your dogage: 861

Knockout.js IV – Dependency Tracking

Definition: Dependency Tracking ist die Nachverfolgung abhängiger Attribute eines computed observable.

Algorithmus

1. Bei Deklaration des computed observable werden die abhängigen Attribute gesucht und der Initialwert berechnet
2. Dabei wird zu allen observables das computed observable als Listener hinzugefügt.
3. Es werden alle Beobachter des computed observable über den aktuellen Wert informiert.

Ändert sich der Wert eines abhängigen Attributs (das können auch computed observables sein), so wird der Algorithmus erneut ausgeführt. Dies ermöglicht auch die dynamische Anpassung von computed observables.

Web-Bibliotheken und Frameworks

1. Kontext und Motivation
2. Bibliotheken und Frameworks
3. jQuery
4. Bootstrap
5. Knockout.js
- 6. Angular**
7. Darüber hinaus
8. Projekt

Angular

Definition: Angular ist ein CSS-, JavaScript-, Architektur-Framework.



Eigenschaften

- Entwickelt unter Federführung von Google
- Erste Version von 2016
- Verwendet DataBinding
- Eigener Server, der Änderungen sofort sichtbar macht
- TypeScript basiert (Entwicklung von Microsoft, aufbauend auf JavaScript)
- HTML Templates werden mit Angular-Markup ergänzt
- Anwendungen müssen von TypeScript zu JavaScript kompiliert werden
- Implementiert wird mit einer eigenen (Konsolen-)Entwicklungsumgebung

Vorteile

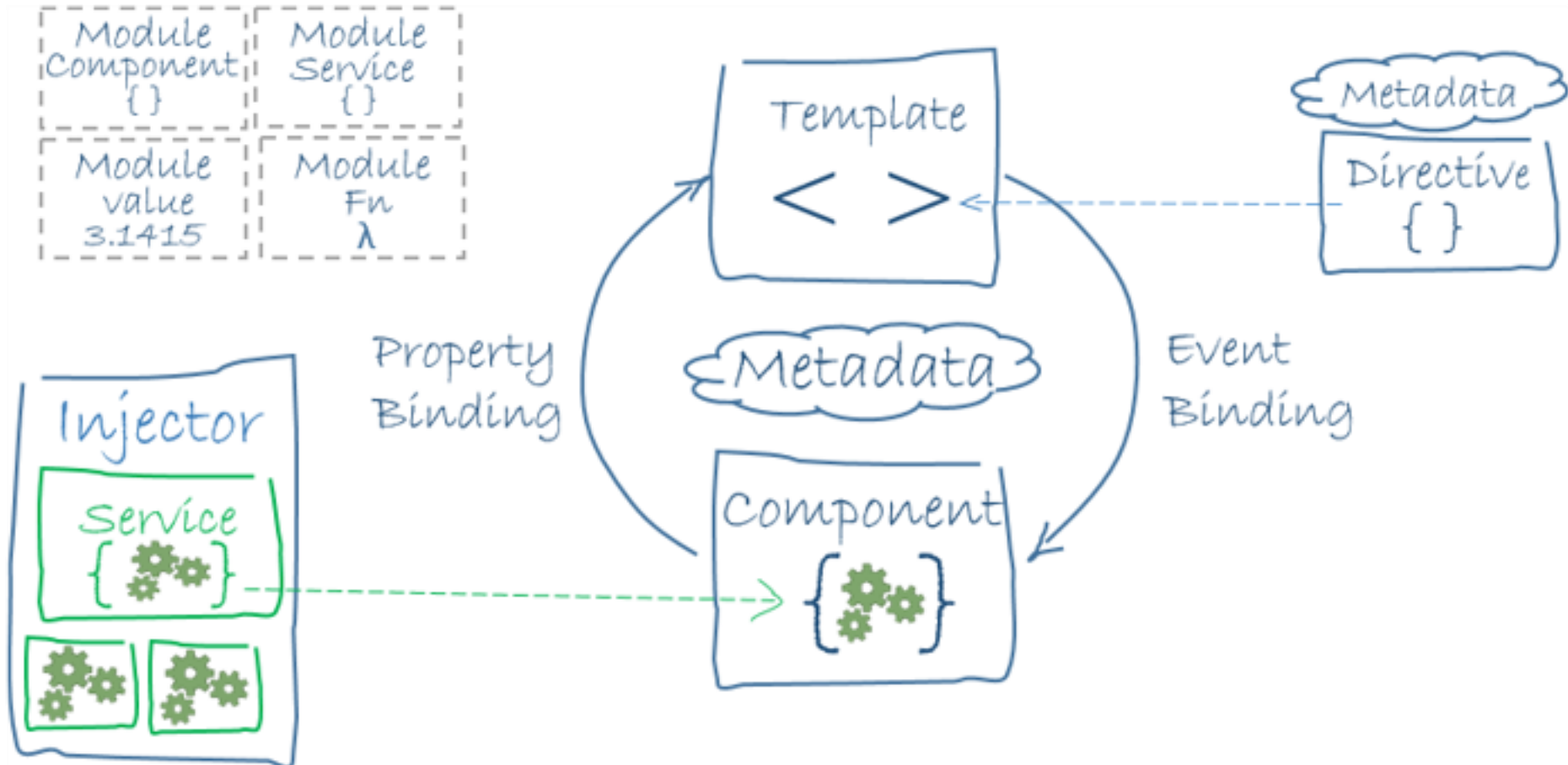
- Mächtige Bibliothek zur Umsetzung eines MVVM-ähnlichen Musters

Nachteile

- Aufwändige Einarbeitung

Weitere Informationen: www.angular.io

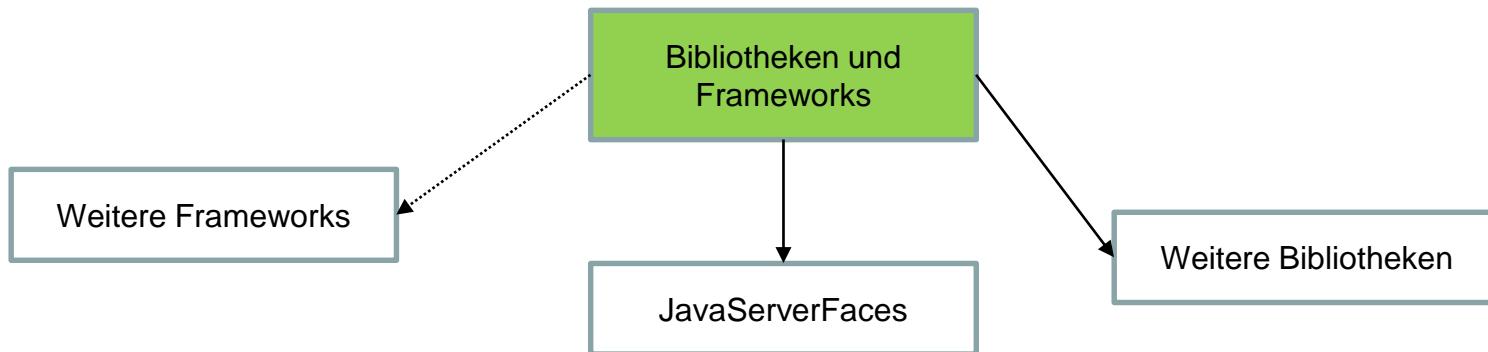
Angular



Web-Bibliotheken und Frameworks

1. Kontext und Motivation
2. Bibliotheken und Frameworks
3. jQuery
4. Bootstrap
5. Knockout.js
6. Angular
- 7. Darüber hinaus**
8. Projekt

Darüber hinaus



Vertiefung im Fach WebEngineering

Links:

Frameworks und Bibliotheken

- <https://hosting.1und1.de/digitalguide/websites/web-entwicklung/beliebte-javascript-frame>

Serverseitige Anwendungen

1. Kontext und Motivation
2. Webserver Interfaces
3. Servlets
4. JSP
5. Darüber hinaus
- 6. Projekt**

Projekt

Übungsaufgabe:

Analyse der bisher erstellten Webapplikation auf die Verwendung und Anwendbarkeit von Entwicklungsmustern.

Literatur: Internet und Netzwerke



Melzer, Ingo et al. „Service-orientierte Architekturen mit Web Services“ Konzepte – Standards – Praxis 4. Auflage 2010, 381 Seiten, ISBN 978-3-8274-2549-2, Spektrum Akademischer Verlag über Springer Link

Christian Ullenboom: „Java 7 – Mehr als eine Insel

Das Handbuch zu den Java SE-Bibliotheken“

ISBN 978-3-8362-1507-7,
Rheinwerk Verlag 2012



Online-Quellen:

Dokumentation zu JQuery:

<https://learn.jquery.com/ajax/working-with-jsonp/>

Kappel, Gerti & Pröll, Birgit & Reich, Siegfried & Retschitzegger, Werner. (2003). Web Engineering - Die Disziplin zur systematischen Entwicklung von Web-Anwendungen. .