

# APP-Struktur und Libraries



# Der Plan

1. HTML5-Struktur unserer App
2. Zwei nützliche Libraries
3. **Wir basteln vier App-Module**

# Bitte einmal alle einen Arm heben!

# Vorlage herunterladen

[files.peterkroener.de/workshopvorlage.zip](http://files.peterkroener.de/workshopvorlage.zip) (<http://files.peterkroener.de/workshopvorlage.zip>)

## In ein Webserver-Verzeichnis extrahieren

Dann (und *erst* dann) den Arm wieder runter

- docs
- lib
  - vendor
  - canvas.js
  - drop.js
  - photo.js
  - read.js
- test
- app.css
- app.js
- index.html

# index.html

- Gesamtes UI unserer App
- Geschrieben in **sparsamer HTML5-Syntax**

```
<!doctype html>
<meta charset="utf-8">
<title>HTML5Stagram</title>
<link rel="stylesheet"
      href="lib/vendor/bootstrap/css/bootstrap.min.css">
<link rel="stylesheet" href="app.css">
<script src="lib/vendor/hacks.js"></script>
```

```
<div class="container">
```

```
...
```

# Die HTML(5)-Syntax ...

- Hat den Doctype `<!doctype html>`
- ... erlaubt fehlende Tags <sup>1</sup>
- ... erlaubt ungeschlossene leere Elemente <sup>1</sup>
- ... erlaubt diverse weitere Vereinfachungen <sup>1</sup>
- **... ist aber auch kompatibel zur XHTML-Syntax**

<sup>1</sup> Sieht komisch aus, [ist aber nichts neues](http://www.w3.org/History/19921103-hypertext/hypertext/WWW/TheProject.html) (<http://www.w3.org/History/19921103-hypertext/hypertext/WWW/TheProject.html>)



# Tag-Auslassung?

Es werden nur **die Tags im Dokument** ausgelassen, **die DOM-Elemente** werden trotzdem angelegt! Identische Dokumente:

```
<!DOCTYPE html>  
<meta charset="utf-8">  
<title>Hallo Welt!</title>  
<p class="welt">Hallo Welt
```

```
<!DOCTYPE html>  
<html xmlns="http://www.w3.org/1999/xhtml">  
  <head>  
    <meta charset="utf-8" />  
    <title>Hallo Welt!</title>  
  </head>  
  <body>  
    <p class="welt">Hallo Welt!</p>  
  </body>  
</html>
```

# Browsern sind manche Tags total egal

```
// Webkits HTMLParser.cpp
void HTMLParser::processCloseTag(Token* t)
{
    // we never close the body tag, since some stupid web
    // pages close it before the actual end of the doc.
    // let's rely on the end() call to close things.
    if (t->tagName == htmlTag || t->tagName == bodyTag ||
        t->tagName == commentAtom)
        return;
}
```

# Pro und Contra?

Letzlich eine Geschmacksfrage.

<header>

<h1>HTML5Stagram</h1>

</header>

<main>

<canvas id="Dropzone"></canvas>

# Neue semantische HTML5-Elemente

- `<section>` - Thematischer Block
- `<article>` - Thematischer, in sich geschlossener Block
- `<nav>` - (Haupt-) Navigationen
- `<main>` - Hauptinhalt des Dokuments
- `<aside>` - Periphärer Content
- `<header>` & `<footer>` - Kopf- und Fußbereiche

# Sinn und Zweck

- Bessere Code-Struktur
- Eingebaute [WAI-Aria-Properties](http://www.w3.org/WAI/intro/aria.php) (<http://www.w3.org/WAI/intro/aria.php>)
- SEO? Nö.

Warnung: Funktioniert nicht in IE < 9

```
<p>
  <label>
    Kontrast:<br>
    <input disabled id="Contrast" type="number" min="-100" max="100"
      data-default="0" value="0">
  </label>

  <label>
    Sättigung:<br>
    <input disabled id="Saturation" type="number" min="-100" max="100"
      data-default="0" value="0">
  </label>

  <label>
    Sepia:<br>
    <input disabled id="Sepia" type="number" min="-100" max="100"
      data-default="0" value="0">
  </label>
</p>
```

# Neue HTML5-Formularfeatures (Auswahl)

- `<type="email">`
- `<type="number">`
- `<type="range">`
- `<type="date">`
- `<type="color">`
- Automatische Validierung

The diagram illustrates various HTML5 form controls. At the top right is an empty email input field. Below it is a number input field with up and down arrow buttons. To the right of the number field is a date input field with the placeholder text 'tt.mm.jjjj', a dropdown arrow, and a calendar icon. To the left of the date field is a color input field with a black color swatch. At the bottom right is a button labeled 'Formular abschicken'.



# Data-Attribute

```
<p data-irgendwas="Wir können uns beliebige  
HTML-Attribute ausdenken!"></p>
```

```
<a id="track1" href="track1.mp3" data-length="4:56"  
data-band="Slayer">South Of Heaven</a>
```

## JavaScript-API

```
document.getElementById('track1').dataset.length;
```

Wird auch von `jQuery.data()` genutzt!

```
<p>  
  <input disabled id="Save" class="btn btn-primary"  
    type="button" value="Speichern">  
  <input disabled id="Delete" class="btn btn-danger"  
    type="button" value="Löschen">  
</p>
```

```
</section>
```

```
<u>footer</u>
```

```
  © 2012 HTML5Stagram
```

```
</u>
```

```
</div> <!-- Ende .container -->
```

# Alles klar bis hierhin?

```
<link rel="stylesheet"  
  href="lib/vendor/bootstrap/css/bootstrap.min.css">
```

...

```
<script data-main="app.js"  
  src="lib/vendor/require.js"></script>
```

## 2. Zwei nützliche Libraries

Bootstrap & RequireJS

# Library 1: Bootstrap

- **Ikea für Webentwickler**, made by Twitter
- Standardstyles, Responsive Grid-Layout, Komponenten, Icons
- Vieles mehr, all in one
- [twitter.github.io/bootstrap](http://twitter.github.io/bootstrap/) (<http://twitter.github.io/bootstrap/>)

```
<link rel="stylesheet"
      href="lib/vendor/bootstrap/css/bootstrap.min.css">
```

...

```
<div class="container">
```

...

```
<p>
  <input disabled id="Save" class="btn btn-primary"
    type="button" value="Speichern">
  <input disabled id="Delete" class="btn btn-danger"
    type="button" value="Löschen">
</p>
```

# Library 2: RequireJS

Scriptloader für Module nach der [Asynchronous Module Definition \(AMD\)](https://github.com/amdjs/amdjs-api/wiki/AMD)

<https://github.com/amdjs/amdjs-api/wiki/AMD>

```
<!-- Das ist kein Modulsystem,  
      das ist eine Zumutung -->  
<script src="jquery.js"></script>  
<script src="underscore.js"></script>  
<script src="backbone.js"></script>  
<script src="app.js"></script>
```



# Hallo AMD-Modul

```
define('optionalName', ['d1', 'd2'], callback);
```

1. Modulname ist optional (laden auch über Dateipfade möglich)
2. Abhängigkeiten im Array werden (parallel) geladen – können selbst AMD-Module sein, müssen aber nicht
3. Der Callback feuert, wenn alle Abhängigkeiten geladen wurden

# Modul-APIs

Der Rückgabewert des Callbacks in einem `define()` ist die Modul-API!

```
// antwort.js  
define(function(){  
    return 42;  
});
```

# Modul-APIs verwenden

Modul-APIs von Abhängigkeiten werden als Arguments von Modul-Callbacks übergeben

```
// frage.js  
define(['antwort'], function(antwort){  
    alert(antwort); // 42  
});
```

# require()

Lädt Module, ohne selbst eines zu definieren.

```
require(['antwort', 'jquery'], function(antwort, $){  
    $('#Foo').html(antwort);  
});
```

# Non-AMD-Scripts

Scripts, die kein AMD-Modul sind, geben nichts zurück, funktionieren aber weiterhin wie gewohnt

```
define(['keinAmd'], function(foo){  
    alert(typeof foo);           // "undefined"  
    alert(typeof window.foo);    // "object"  
});
```

```
// In der App modul42.js und jQuery laden
require(['modul42', 'jquery'], function(modul42){

  // In modul42 liegt die Modul-API
  var value = modul42.doSomething();
  alert(value);

});
```


```
// Moduldefinition von modul42.js
define(function(){

  // Return des Callbacks = Modul-API
  return {
    doSomething: function(){
      return 42;
    }
  };
});
```

```
// In der App modul42.js und jQuery laden
require(['modul42', 'jquery'], function(modul42){

  // In modul42 liegt die Modul-API
  var value = modul42.doSomething();
  alert(value);

});
```



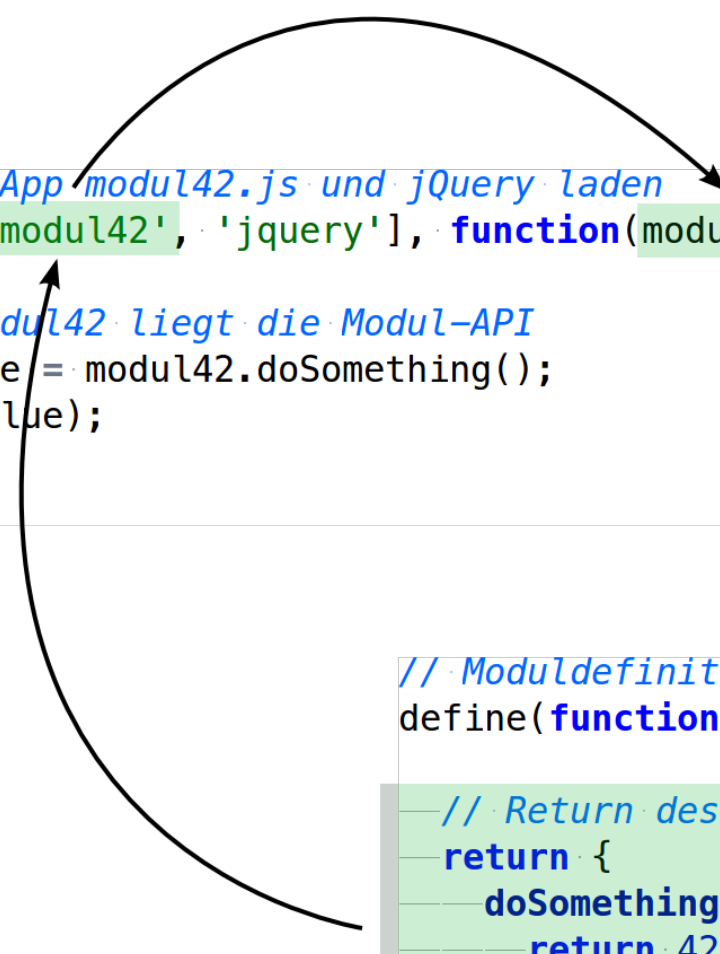
```
// Moduldefinition von modul42.js
define(function(){

  // Return des Callbacks = Modul-API
  return {
    doSomething: function(){
      return 42;
    }
  };
});
```

```
// In der App modul42.js und jQuery laden
require(['modul42', 'jquery'], function(modul42){

  // In modul42 liegt die Modul-API
  var value = modul42.doSomething();
  alert(value);

});
```



```
// Moduldefinition von modul42.js
define(function(){

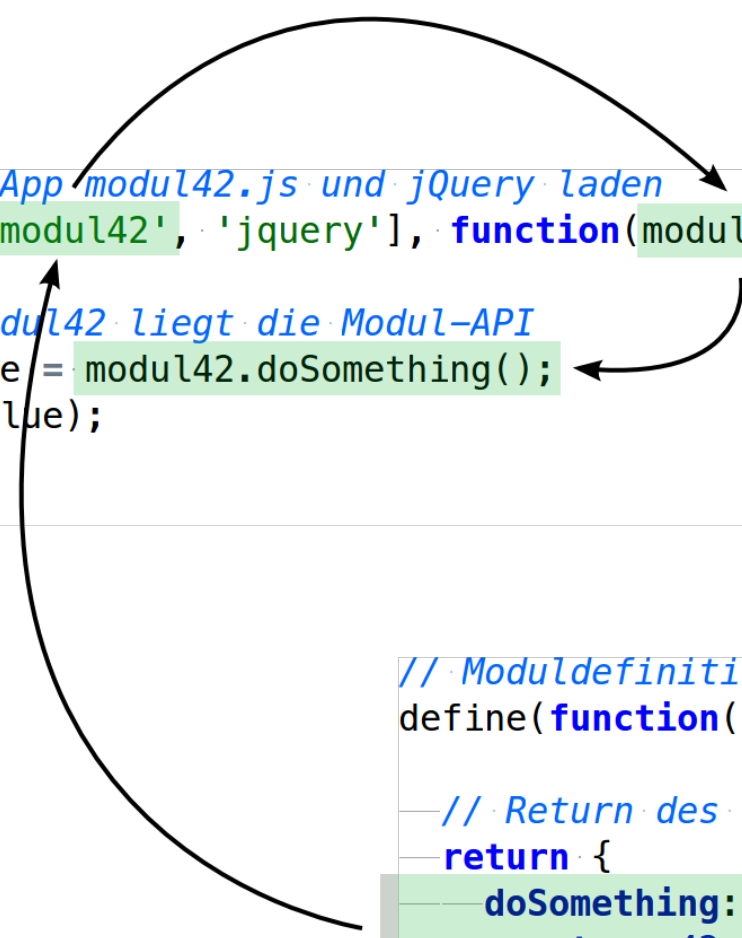
  // Return des Callbacks = Modul-API
  return {
    doSomething: function(){
      return 42;
    }
  };
});
```



```
// In der App modul42.js und jQuery laden
require(['modul42', 'jquery'], function(modul42){

  // In modul42 liegt die Modul-API
  var value = modul42.doSomething();
  alert(value);

});
```



```
// Moduldefinition von modul42.js
define(function(){

  // Return des Callbacks = Modul-API
  return {
    doSomething: function(){
      return 42;
    }
  };
});
```

# RequireJS (<http://requirejs.org/>)

- Scriptloader für AMD-Module
- Implementiert `define()` und `require`, konfiguriert und lädt Module, löst Abhängigkeiten auf
- Bonus: Optimizer, Node.js-Kompatibilität

# Verwendung von RequireJS

```
<script src="scripts/require.js"  
  data-main="scripts/main"></script>
```

# Einfach, oder?

- Modul definieren: `define(['A', 'B'], cb);`
- Modul laden: `require(['A', 'B'], cb);`
- Einbetten: `require.js` laden, App-Datei im Attribut `data-main` referenzieren

Alles klar?

# Wirklich alles klar?

Dann schreiten wir nämlich jetzt zur Tat ...

# 3. App-Module basteln

HTML5 in Aktion

# Letzte Chance!

1. Alles klar zu Drag & Drop, File API, Canvas und Webcam-API?
2. Alles klar zu RequireJS und AMD?

# Libraries bauen!

- Wir brauchen vier Libraries für unsere App:
  - `read.js` - Datei einlesen (einfach)
  - `drop.js` - Library für Drag & Drop (medium)
  - `canvas.js` - Bild-URL auf Canvas zeichnen (advanced)
  - `photo.js` - Webcam-Feed Canvas zeichnen (super-advanced)
- Vorschlag: **Thema aussuchen, Teams bilden**



# read.js

- Liest eine Datei als DataURL ein und ruft dann einen Callback mit dem Ergebnis auf
- Anforderungen:
  - AMD-Modul
  - API: `readAsDataURL(file, callback)`

# drop.js

- Macht ein Element zum Drop-Ziel, feuert bei Erfolg einen Callback mit dem Event-Objekt
- Anforderungen:
  - AMD-Modul
  - Das Drop-Ziel erhält die Klasse `active`, wenn etwas darüber gezogen wird
  - API: `drop( '#selektor', callback )`

# canvas.js

- Anforderungen:
  - AMD-Modul
  - API:
    - `canvas.init('#selektor')`
    - `canvas.el()`
    - `canvas.reset()`
    - `canvas.drawUrl('url', callback)` (mit Canvas-Resize auf die Bildmaße)

# photo.js

- Zeichnet einen Webcam-Stream auf eine Canvas
- Stoppt die Übertragung auf Befehl (Foto-Funktion)
- **Eine Aufgabe für die echten JS-Ninjas!**
- Anforderungen:
  - AMD-Modul
  - API: (nächste Folie, hier kein Platz)

# API für photo.js

- `photo.supportsRecording (bool)`
- `photo.isRecording (bool)`
- `photo.init(canvasSelector)`
- `photo.startRecording(callback)`
- `photo.stopRecording()`
- `photo.getVideo()`
- `photo.getCanvas()`

# Docs und Testcases

- Links, Docs und Spickzettel: /docs-Verzeichnis
- Tests: /test-Verzeichnis

# Also dann!

Wer möchte mit welchem Modul loslegen?

# Module basteln!

Links, Docs und Spickzettel: /docs

Tests: /test

# Fragen? **Rufen Sie mich!**