

Webentwicklung – Teil 6



DHBW SS2018 – Elias Henrich

Überblick Heute

- Übung XMLHttpRequest
- JSON-Datenformat
- Abschluss Ergebnisliste
- JS-Libs: Leaflet
- Rückblick & Ausblick nächste VL



Schnittstellendoku

Flightsearch Web-API

Get all airports

Endpoint: `http://flights.eliashenrich.de/api.php`

Method: `GET`

Params: `action /airports/all`

Example:

```
http://flights.eliashenrich.de/api.php?  
    action=/airports/all
```

Schnittstellendoku

Flightsearch Web-API

Get routes between airports

Endpoint: `http://flights.eliashenrich.de/api.php`

Method: `GET`

Params: action `/route/find`
 from `<airport-id> (e.g. 25)`
 to `<airport-id> (e.g. 23)`

Example:

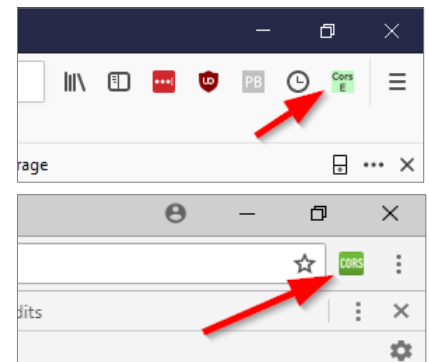
```
http://flights.eliashenrich.de/api.php?  
action=/route/find&from=25&to=23
```

Übung: XMLHttpRequest

Erstellen Sie eine Funktion, mit dieser Definition:
`requestAPI(action, data, callback)`

Funktion ruft den Endpunkt der Flug-Web-API mit der angegebenen `action` und allen Werten des `data`-Arrays auf. Anschließend wird die in `callback` übergebene Methode aufgerufen und an diese das Ergebnis der API-Abfrage übergeben.

```
var params = {  
  from: 25,  
  to: 23  
};  
  
requestAPI('/route/find', params, function(result) {  
  alert("Anfrage beendet");  
  console.log("Ergebnis", result);  
});
```



JavaScript Object Notation

- populäres Datenformat
zur Speicherung und Transport
- geringer Overhead
- ursprüngl. JS, jetzt auch andere Sprachen
- Syntaktisch wie JS-Objekt
- häufig Konkurrenz zu XML
(REST <> SOAP)

JSON

```
{  
  "airportFrom": {  
    "Name": "Honolulu International",  
    "CodeIATA": "HNL",  
    "PositionLat": 21.3186800000,  
    "PositionLon": -157.9219970000,  
    "Id": 25,  
    "IsInternational": true  
  }  
}
```

JSON

Key:Value-Pair

```
{  
  "airportFrom": {  
    "Name": "Honolulu International",  
    "CodeIATA": "HNL",  
    "PositionLat": 21.3186800000  
    "PositionLon": -157.92199700  
    "Id": 25,  
    "IsInternational": true  
  }  
}
```

Key:
Immer String

Value:
Unterschiedliche
Datentypen

JSON Datentypen

STRING

```
{  
  "airportFrom": {  
    "Name": "Honolulu International",  
    "CodeIATA": "HNL",  
    "PositionLat": 21.3186800000,  
    "PositionLon": -157.9219970000,  
    "Id": 25,  
    "IsInternational": true  
  }  
}
```

JSON Datentypen

BOOLEAN

```
{  
  "airportFrom": {  
    "Name": "Honolulu International",  
    "CodeIATA": "HNL",  
    "PositionLat": 21.3186800000,  
    "PositionLon": -157.9219970000,  
    "Id": 25,  
    "IsInternational": true  
  }  
}
```

JSON Datentypen

NUMBER

```
{  
  "airportFrom": {  
    "Name": "Honolulu International",  
    "CodeIATA": "HNL",  
    "PositionLat": 21.3186800000,  
    "PositionLon": -157.9219970000,  
    "Id": 25,  
    "IsInternational": true  
  }  
}
```

Alle Zahlen: Integer, Double, Long, Signed, Unsigned...

JSON Datentypen

OBJECT

Parent-Objekt

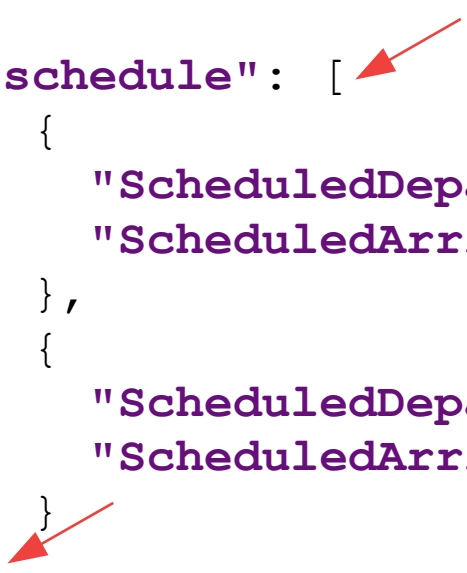
```
{  
  "airportFrom": {  
    "Name": "Honolulu International",  
    "CodeIATA": "HNL",  
    "PositionLat": 21.3186800000,  
    "PositionLon": -157.9219970000,  
    "Id": 25,  
    "IsInternational": true  
  }  
}
```

Eingebettetes
Objekt

JSON Datentypen

ARRAY (hier: von Objekten)

```
{  
  "schedule": [  
    {  
      "ScheduledDepartureTime": "12:05:00",  
      "ScheduledArrivalTime": "18:06:00"  
    },  
    {  
      "ScheduledDepartureTime": "17:00:00",  
      "ScheduledArrivalTime": "22:36:00"  
    }  
  ]  
}
```



Arrays können beliebige Datentypen enthalten

JSON parsen

JSON → JS-Objekt

```
var jsonObject = JSON.parse(jsonString);  
console.log(jsonObject.propertyA);
```

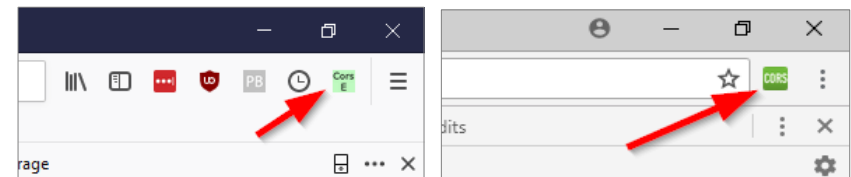
JS-Objekt → JSON

```
var jsonString = JSON.stringify(jsonObject);  
console.log(jsonString);
```

Übung: XMLHttpRequest

Fügen Sie alle Komponenten zusammen:

- Event onsubmit löst Abfrage der Route 25 nach 23 aus
- Während die Abfrage läuft, wird automatisch das Overlay ein- und ausgeblendet
- Die Ergebnisliste wird geleert
- Die empfangenen JSON-Daten werden in der Liste dargestellt



JS-Bibliotheken

...bezeichnet [...] eine Sammlung von Unterprogrammen, die Lösungswege für thematisch zusammengehörende Problemstellungen anbieten. Bibliotheken sind im Unterschied zu Programmen keine eigenständig lauffähigen Einheiten, sondern sie enthalten Hilfsmodule, die von Programmen angefordert werden.

Quelle: <https://de.wikipedia.org/wiki/Programmbibliothek>

JS-Bibliotheken

- Einbinden fertiggestellter Skripte
- Speicherort: Gleicher oder fremder Server
- Häufig bestehen Abhängigkeiten zu anderen Bibliotheken

daher: Package-Manager, z.B. YARN, NPM

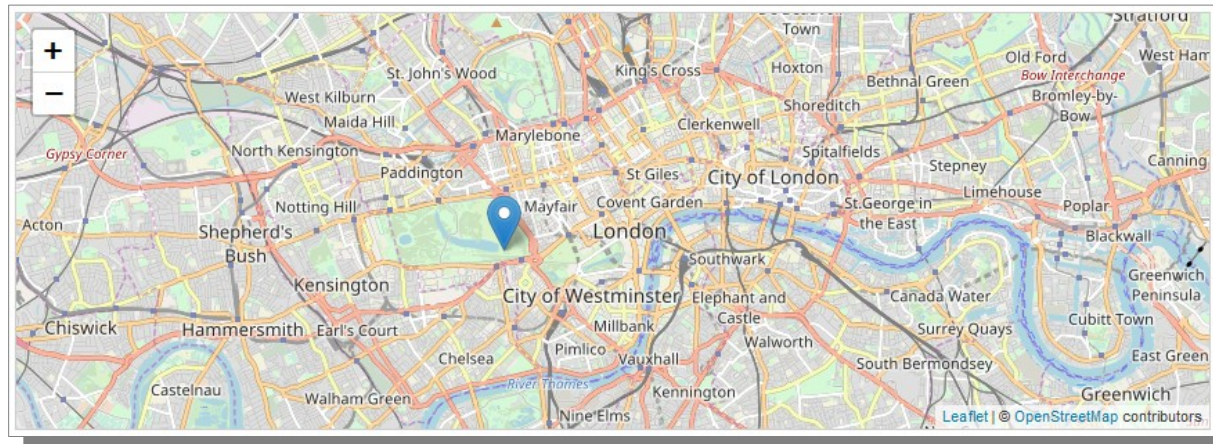
- Die meisten Libraries sind Lizenz-frei und auch für kommerzielle Zwecke verfügbar

```
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">

  <script src="[Externe Ressource]"></script>
</head>
```

JS-Library **Leaflet**

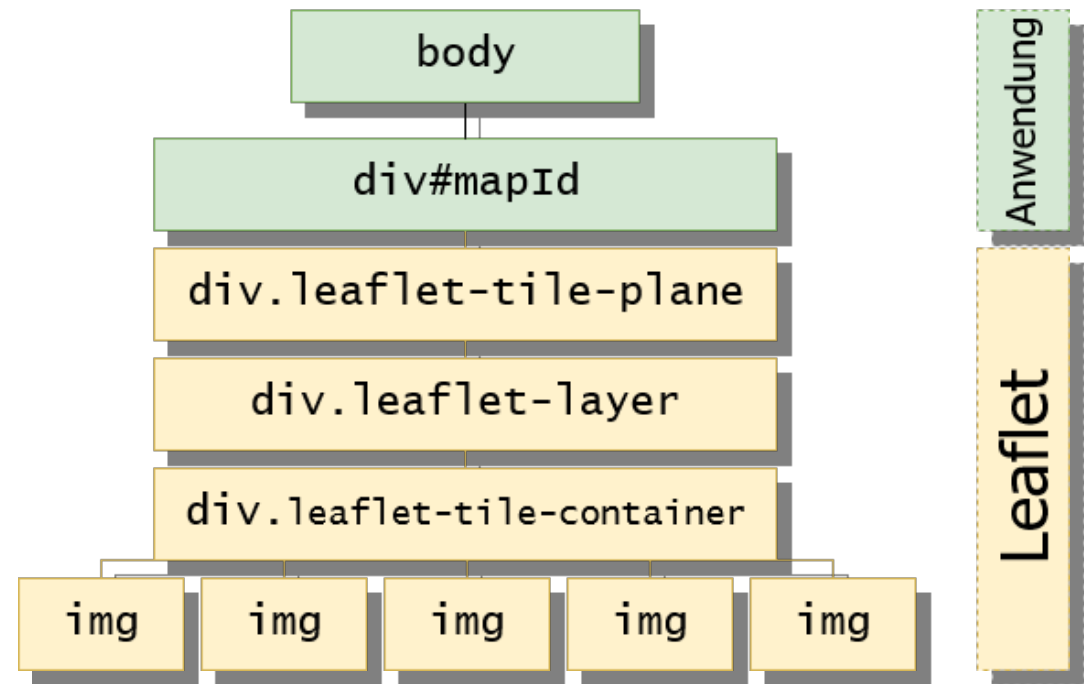
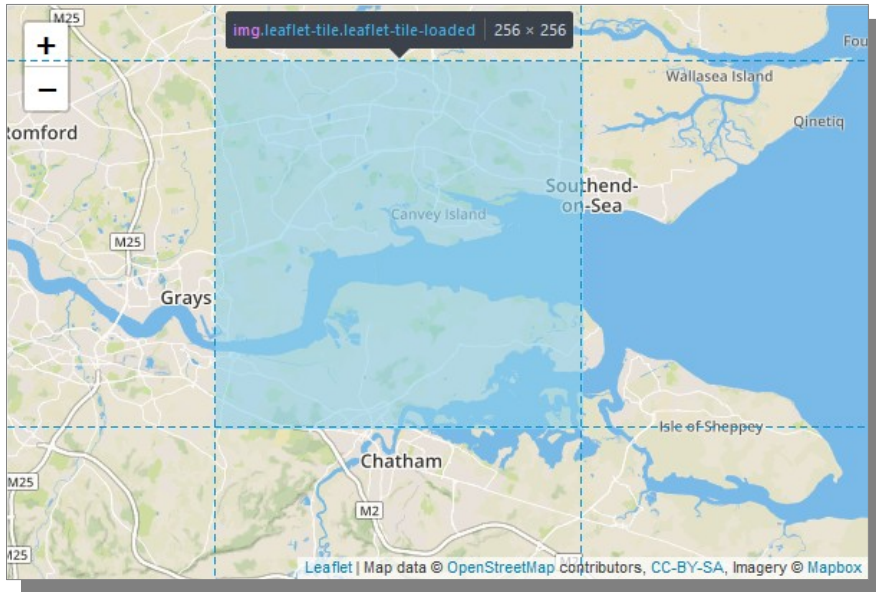
- populäre JS-Bibliothek zur Integration von (Straßen-) Karten in Webanwendungen
z.B. GitHub, Pinterest, Flickr
- Diverse Datenquellen: z.B. Google Maps, Open Street Map oder auch selbst-definierte Kartendaten



JS-Library Leaflet

Web Map Tile Service (WMTS)

Karte besteht aus vielen quadratischen Einzelgrafiken innerhalb eines div-Containers

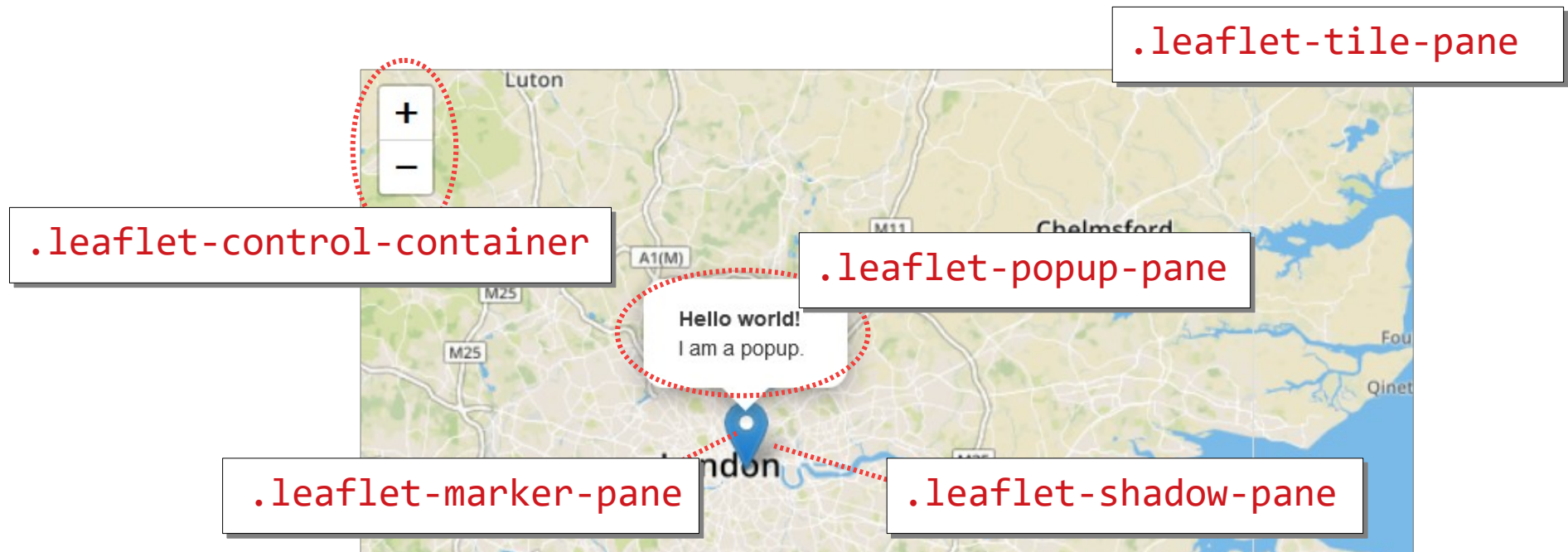


JS-Library Leaflet

Layer-basiert

mehrere Ebenen überlagern sich gegenseitig (z-Index)

Pane: „An individual sheet of glass in a window.“ - <https://en.wiktionary.org/wiki/pane>



1. Map-Container erstellen & stylen

index.html

```
<div id="map"></div>
```

style.css (oder entsprechende CSS-Datei)

```
#map {  
    height: 150px;  
    width: 100px;  
}
```

2. Leaflet Map-Objekt initialisieren

index.html

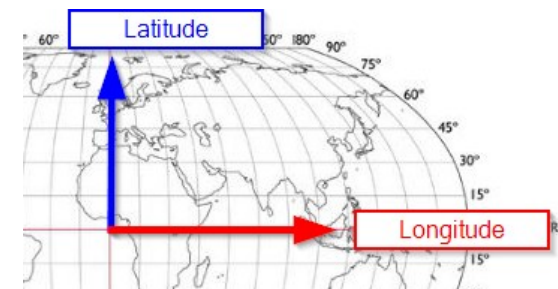
```
<script>
  var mapObject;

  function initMap(mapId) {
    mapObject = L.map(mapId, {
      center: [-10.0, -170.0],
      zoom: 4
    });
  }
</script>
```

`L.latlng(latitude, longitude)`

Latitude: Breitengrad

Longitude: Längengrad



3. TileLayer initialisieren und hinzufügen

index.html

```
var accessToken = '[ACCESS-TOKEN]';

function initMap(mapId) {
    [...]

    L.tileLayer('[URL]', {
        attribution: 'Copyright-Hinweise...',
        maxZoom: 18,
        id: 'mapbox.streets',
        accessToken: accessToken
    }).addTo(mapObject);
}
```

Access-Token (zur Verifizierung der Kartengrafiken beim Anbieter Mapbox)

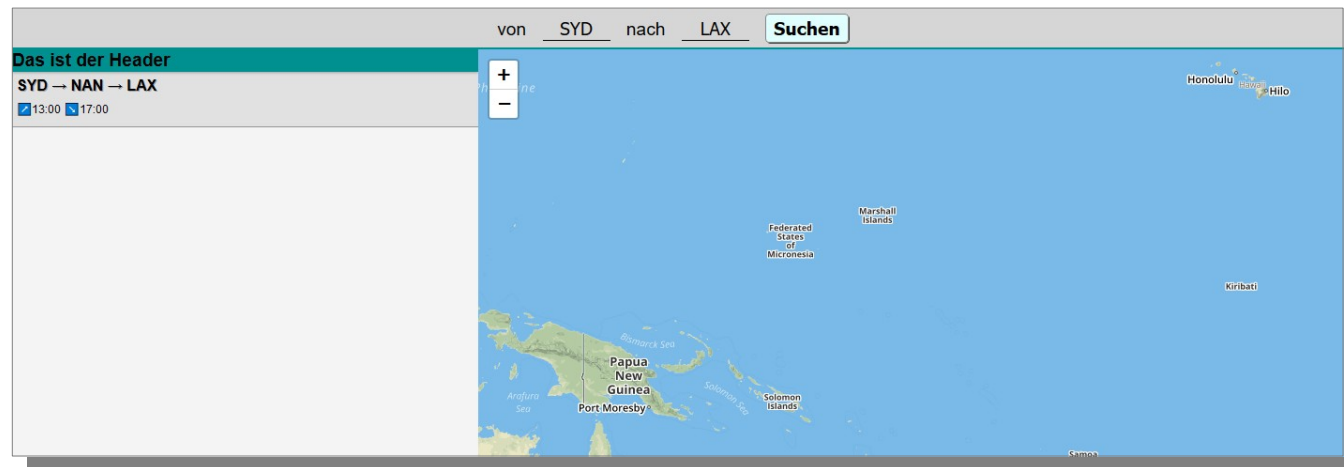
pk.eyJ1IjoiaWxrcm9rZXR0byIsImEiOiIjZjamp1Z2NqODQybG4wM3F0ZTU0N2s4azdxIn0.VL6YIZWFhnan5AWzxgIFpw

URL (Endpunkt der Mapbox-WTMS-API)

https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.png?access_token={accessToken}

Übung: Karte darstellen

Ziel: Karte in der Anwendung *Flugsuche* einbetten

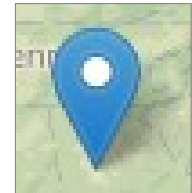


1. Führen Sie Schritt 1-3 in der bestehenden Anwendung aus
2. Passen Sie die CSS-Eigenschaften des Containers an
3. Rufen Sie die Methode `initMap()` auf und übergeben Sie die Id des Map-Containers (div)

Marker mit Popup darstellen

index.html

```
var marker = L.marker([-10.0, -170.0]);  
marker.bindPopup('<em>Popup-Text</em>');  
  
marker.addTo(mapObject);
```



Marker-Icon:

Frei definierbares Grafik-Objekt, das an einen Marker gebunden werden kann.

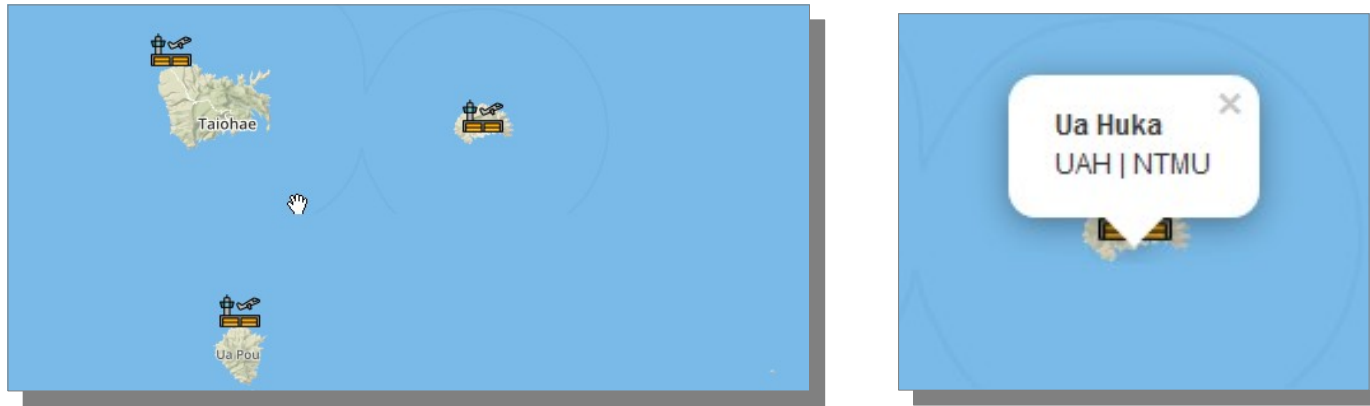
index.html

```
var myIcon = L.icon({  
  iconUrl: 'static/media/departures.png',  
  iconSize: [32, 32],  
  iconAnchor: [16, 32]  
});  
  
var marker = L.marker([-10.0, -170.0], {  
  icon: myIcon  
});  
  
marker.addTo(mapObject);
```



Übung: Flughäfen darstellen

Ziel: Alle Flughäfen in Karte darstellen



1. Sprechen Sie die Such-API mit `action=/airports/all` an und laden Sie eine Liste aller Flughäfen [`console.log()`]
2. Durchlaufen Sie die Liste* und erzeugen einen Marker je Flughafen. Bei Klick auf den Marker erscheint ein Popup
3. Ändern Sie das Icon des Markers

*Fehlerhafte Daten (Lat & Lon == 0) sollen übersprungen werden

Populäre JS-Bibliotheken



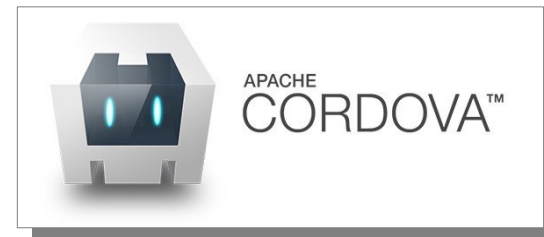
jQuery (Google)

DOM-Traversing & Manipulation, einfaches Event-Handling, simple AJAX-Calls



React.js (Facebook)

Deklarative Bibliothek für User-Interfaces von Web- und Mobile-Anwendungen



Cordova (Apache)

Nativer Container für HTML/JS-Anwendungen für mobile Plattformen (iOS, Android, Windows Phone etc.)



nodeJS

JS-Laufzeitumgebung, um Skripte server-seitig bzw. außerhalb des Browser laufen zu lassen.

Vorlesung: Donnerstag

- Kurze Einführung in jQuery
- Wiederholung und Praxistipps
 - Webseite mit .de-Domain online stellen
 - GitHub: Commit, Push und Merge
- Support beim Abschlussprojekt

Rückblick

- Abgrenzung Web- und Desktop-Anwendungen
- Das HTTP-Protokoll
- Wichtigste HTML-Elemente
 - Inhalte beschreiben
 - Formulare
 - HTML4 vs. HTML5
- Document-Object-Model

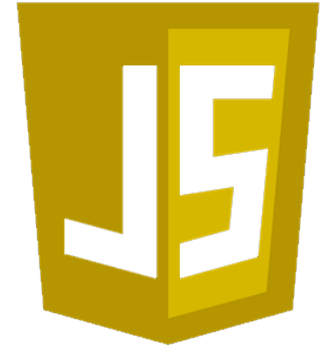


Rückblick

- Cascading Style Sheets
 - Farben
 - Größenangaben
 - Selektoren
 - Box-Model
 - Transitions und Animationen



Rückblick



- JavaScript
 - Interne vs. externe JS-Dateien
 - Einfache Objektorientierung
 - Datentypen
 - Kontrollstrukturen (Verzweigungen, Schleifen)
 - HTML- & CSS-Elemente manipulieren
 - AJAX
 - Bibliotheken

Feedback zur Vorlesung

Inhalte der Vorlesung

Vorlesungsmaterial

Allgemeines Niveau

Anzahl und Komplexität
der Übungen

Verbesserungen für
die Zukunft

