Praktischer Zwischentest – Gruppe rot

Bei diesem praktischen Zwischentest gibt es 26 Punkte zu erreichen. Der Prozentanteil der erreichten Punkte von 26 (4 Extrapunkte möglich) trägt zu 36% zur Note in Web Scripting bei.

Allgemeine Hinweise:

- Der praktische Zwischentest muss auf Ihrem eigenen Rechner durchgeführt werden.
- Ziel ist es Ihre Umsetzungskompetenz mit JavaScript zu evaluieren
- Lesen Sie die Angabe genau durch
- Halten Sie Ihren Studierenden-Ausweis bereit

Unterlagen:

- Sie dürfen Internet, Bücher und eigene Programmstücke verwenden.
- Definitiv **verboten** ist die Unterstützung durch **andere Personen** bzw. ein **Datenaustausch** in jeglicher Form (manuell, elektronisch, …).
- Ihre Abgaben werden einer Plagiatsprüfung unterzogen.

Abgabemodus:

- Packen Sie Ihr gesamtes Projekt (alle HTML/JS/etc. Files) in **eine** zip-**Datei** und laden Sie diese in Moodle hoch.
- Dabei soll die vorgegebene Ordnerstruktur erhalten bleiben. Kontrollieren Sie Ihren erfolgreichen Upload bzw. die Validität der hochgeladenen Datei!

Wenn ein oder mehrere Ergebnisse zur Gänze oder auch nur in Teilen gleich oder sehr ähnlich sind (sodass ersichtlich ist, dass Ergebnisse kopiert wurden), dann wird die Prüfung für alle Beteiligten mit 0 Punkten beurteilt.
Lesen Sie die Angaben genau und halten Sie sich an die vorgegebenen Nomenklaturen (Klassen, JS-Methoden, …)!

Aufgabenstellungen in grüner Schrift sind optional und können ggf. Punkte anderer fehlender bzw. unvollständiger Aufgabenstellungen kompensieren.

Viel Erfolg!

IMPLEMENTIERUNG WEBANWENDUNG

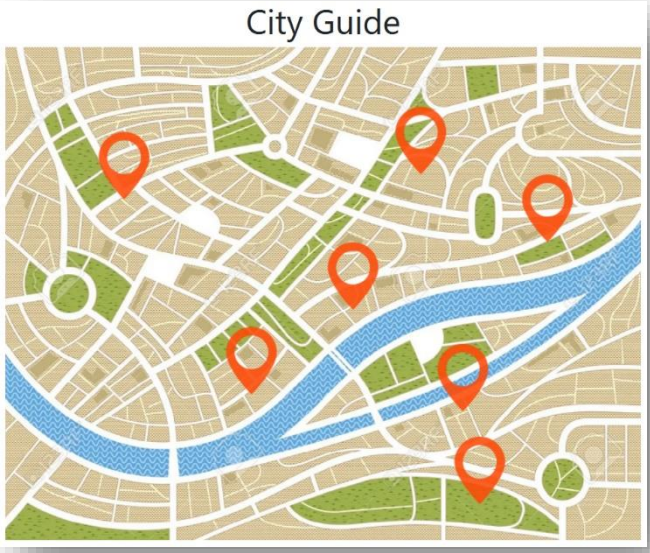
Vorbereitungen:

Nach dem Öffnen Ihrer Entwicklungsumgebung und dem Import der vorgegebenen Dateien erscheint die Projektstruktur, wie nachfolgend sichtbar. Dabei gilt, dass rote Teile erweitert werden müssen, schwarze Teile bleiben unverändert. Achtung! Sie müssen die Dateien über den Server ausführen, da sonst ein CORS Problem auftritt!

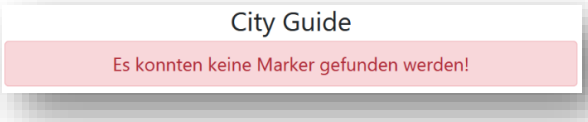
- Projektordner: **EXAM_WebScripting_CityGuide**
  - Datei: **index.html**
  - Datei: **getdata.js**
  - Datei: **myStyle.css**
  - Datei: **traveldata.json**
  - Folder: **[img]** (beinhaltet Map+Marker)

Überblick Beispiel-Lösung:

1) Beim Aufruf der Webseite wird eine City-Map mit Points of Interest (POIs) in Form von Markern automatisch geladen und wie folgt angezeigt:



2) Wenn der AJAX-Call nicht funktioniert, erscheint z.B.



3) Wenn einer der Marker aus **1)** mit der Maus „berührt“ wird, werden die Details zum Point of Interest direkt an Position des Markers in Form einer Infobox angezeigt:

Tipp:
Arbeiten Sie am Styling erst nachdem Sie die Funktionalitäten implementiert haben.

1 Grundidee

*Auf der Webseite (**index.html**) werden die Marker (POI) auf der Karte angezeigt, welche über die Datei **traveldata.json** via JavaScript geladen werden. Die Marker auf der Karte können mit der Maus berührt werden. Daraufhin werden Detailinformationen zum POI angezeigt. Wird ein neuer Marker berührt, werden nur diese Daten angezeigt und die anderen Infos verschwinden.*

2 Beschreibung von *traveldata.json*

*Bevor Sie mit der Entwicklung der Funktionalitäten beginnen, machen Sie sich bitte mit dem Inhalt der Datei **traveldata.json** vertraut. Jedes Objekt in diesem Array enthält Information zu „name“, „coordx“, “coordy“ und „detail“.*

3 Umsetzung der Funktionalitäten in getdata.js

Setzen Sie folgende Funktionalitäten mit JavaScript um*:*

3a Binden Sie jQuery und das getdata.js-File in Ihr Projekt ein. **(2P)**

3b Implementieren Sie einen AJAX-Call der ausgeführt wird, wenn die Seite aufgerufen wird und fertig geladen ist. **(2P)**

- i) Erstellen Sie eine beliebige Fehlermeldung, wenn die Daten nicht geladen werden können **(1P)**
- ii) Die Karte wird beim Laden der Site mit einem slideDown von 350ms animiert. **(2P)**

3c Laden Sie die Inhalte von traveldata.json via AJAX und nehmen Sie diese entgegen. **(2P)**

- i) Hinweis: via `console.log(response);` können Sie die Daten auch in der Konsole sichtbar machen.

3d Für die Oberfläche werden einige Daten aus dem JSON File benötigt. Zeigen Sie diese gemäß der Beispiel-Lösung auf der Webseite als Marker mit entsprechender Detailinformation an. Erstellen Sie via JavaScript die Bilder und Detailinformationen pro eingelesenes Item und konfigurieren Sie dieses folgendermaßen:

- i) Erzeugen Sie pro JSON-File-Eintrag einen Marker (siehe marker.png). **(2P)**
- ii) Positionieren Sie diesen via x/y-Koordinaten (css-Positionsangaben mittels top/left) auf der Karte (map). **(2P)**
- iii) Weisen Sie jedem Marker für die korrekte Optik die Klasse „poi“ zu. **(1P)**
- iv) Erzeugen Sie weiters die Zusatzinformationen (=Infobox) in einem Span mit Namen (fett) und Details der Sehenswürdigkeit. **(2P)**
- v) Positionieren Sie die Infobox 40px weiter oben und 85px weiter links des Markers. **(2P)**
- vi) Vergeben Sie jeder Infobox die Klasse „detail“. **(1P)**
- vii) Alle Infoboxen sollen beim Laden der Site versteckt sein. **(2P)**
- viii) Erstellen Sie einen Rahmen für die Karte via JavaScript. **(2P)**

3e Erstellen Sie eine neue Funktion mit dem Namen „*showDetail()*“ welche aufgerufen wird, sobald ein Marker auf der Karte mit der Maus berührt wird. **(3P)**

3f Blenden Sie die Infobox des aktuellen POIs mit einem FadeIn von 350 ms ein. Es soll immer nur 1 Infobox sichtbar sein, verstecken Sie die Detail-Infos aller anderen Marker wieder. **(2P)**

3g Blenden Sie auch die letzte Infobox aus, sobald sich die Maus vom Marker runterbewegt. **(2P)**