Dokumentation

td_geo.js

Kartendarstellungspaket von Telematik Design

Inhaltsverzeichnis

1. Einleitung

Die Javascript-Datei *td_geo.js* ist ein Kartendarstellungspaket von *Telematik Design*. Es nutzt *OpenLayers* mit Karten von *Openstreetmap* und *Google*.

2. Implementierung

Dieses Paket wird in das HTML-Dokument eingebunden, indem der folgende Code im `<head>`-Element eingefügt wird:

```
<script type="text/javascript"
 src="http://www.openlayers.org/api/OpenLayers.js"></script>

<script type="text/javascript"
 src="http://www.openstreetmap.org/openlayers/OpenStreetMap.js"></script>

<script type="text/javascript" src="http://maps.google.com/maps/api/js?
 v=3&amp;sensor=false"></script>

<script type="text/javascript" src="td_geo.js"></script>
```

An der gewünschten Stelle wird in einem Block-Element (z.B. direkt im `<body>`) ein leeres `<div>`-Element für die Karte erstellt. Es benötigt eine ID, um die Karte zu zeichnen. Ist die ID beispielsweise „`myMap`“, so sehe der Code folgendermaßen aus:

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

Nun kann die Karte in diese Region gezeichnet werden, indem – üblicherweise beim Laden des Dokuments – die `drawmap`-Funktion aufgerufen wird. Beispiel:

```
<body onload="drawmap('myMap', {path:'', relative:true},
 {hSpace:30, vSpace:50})">

   <div id="myMap"></div>

</body>
```

Näheres zu den einzelnen Funktionen erfahren Sie im nächsten Kapitel unter „Funktionen“.

3. Dokumentation

Grundlegende Eigenschaften sind in diesem Paket als Eigenschaften des Objekts `geo` festgelegt. Weiters sind hier Eigenschaften deklariert, die während der Kartendarstellung verfügbar sein müssen. Die Methoden des Objekts `geo` sind Kopien der Funktionen des Pakets.

3.1. Grundlegende Eigenschaften

Folgende Eigenschaften sind grundlegend für das Paket und können verändert werden; sie sollten es jedoch normalerweise nicht (Code zeigt Standardwert):

3.1.1. geo.langCode

Zeichenkette, die die Sprache von *OpenLayers* in der Funktion `drawmap` festlegt.

```
geo.langCode = "de";
```

3.1.2. geo.iconPath

Zeichenkette, die den Pfad des Marker-Ordners beschreibt.

```
geo.iconPath = "markers/";
```

3.1.3. geo.iconSize

Objekt, dessen Eigenschaften `w` die Breite und `h` die Höhe der Marker in Pixel angeben.

```
geo.iconSize = {w: 20, h: 34};
```

3.1.4. geo.popupClass

`OpenLayers.Class`-Objekt, das in der Funktion `setMarker` die `popupClass` des *Popup*-Fensters festlegt.

```
geo.popupClass = OpenLayers.Class(OpenLayers.Popup.FramedCloud, {autoSize:
  true});
```

3.2. Deklarierte Objekteigenschaften

Die folgenden Eingenschaften des Objekts `geo` sind im gesamten Paket gültige, wichtige Variablen; sie sollten nie verändert werden (Code zeigt Standardwert):

3.2.1. geo.map

Zu Beginn `undefined`, wird in der Funktion `drawmap` als Karte des Typs `OpenLayers.Map` definiert.

```
geo.map;
```

3.2.2. geo.path

Zu Beginn `undefined`, wird in der Funktion `drawmap` als Pfad definiert und sollte eine Zeichenkette sein.

```
geo.path;
```

3.2.3. geo.pathRelative

Zu Beginn `undefined`, wird in der Funktion `drawmap` definiert; legt fest, ob der Pfad relativ ist, und sollte ein Boolescher Wert sein.

```
geo.pathRelative;
```

3.2.4. geo.panel

`OpenLayers.Control.Panel`, das in den Funktionen `drawmap` und `initSingleMarker` Schaltflächen aufnimmt.

```
geo.panel = new OpenLayers.Control.Panel();
```

3.2.5. geo.markersName

Array, das in der Funktion `setMarker` die *Layer*-Namen der Marker aufnimmt. Das Array sollte dieselbe Länge wie `geo.markersLayer` haben und die Array-Objekte sollten Zeichenketten sein.

```
geo.markersName = new Array();
```

3.2.6. geo.markersLayer

Array, das in der Funktion `setMarker` die Marker aufnimmt. Das Array sollte dieselbe Länge wie `geo.markersName` haben und die Array-Objekte sollten vom Typ `OpenLayers.Layer.Markers` sein.

```
geo.markersLayer = new Array();
```

3.2.7. geo.singleLayer

Zu Beginn `undefined`, wird in der Funktion `initSingleMarker` als *SingleMarker***-Layer* definiert und sollte vom Typ `OpenLayers.Layer.Vector` sein.

```
geo.singleLayer;
```

* *SingleMarker* ist ein einzelner Marker, der nachträglich vom Benutzer verschoben oder neu gesetzt werden kann. Er kann nicht mit der Funktion `showOverlays` zentriert werden.

3.2.8. geo.singleLastPoint

Zu Beginn `undefined`, wird in den Funktionen `initSingleMarker` und `setSingleMarker` als *SingleMarker* definiert und sollte vom Typ `OpenLayers.Feature.Vector` sein.

```
geo.singleLastPoint;
```

3.2.9. geo.bounds

`OpenLayers.Bounds`-Objekt, das die Grenzen aller Marker und Linien beschreibt; wird in den Funktionen `setMarker` und `setLine` verändert und in der Funktion `showOverlays` ausgelesen.

```
geo.bounds = new OpenLayers.Bounds();
```

3.2.10. geo.boundsCheck

Boolescher Wert, der festlegt, ob `geo.bounds` verwendet wurde; wird in den Funktionen `setMarker` und `setLine` gesetzt.

```
geo.boundsCheck = false;
```

3.3. Funktionen

Die folgenden Funktionen können entweder direkt oder als Methode des Objekts `geo` verwendet werden:

3.3.1. drawmap

Der erste Parameter beschreibt die ID der Karte (→ 2. Implementierung) und sollte eine Zeichenkette sein. Der zweite Parameter beschreibt den Pfad und sollte ein Objekt mit den Eigenschaften `path` (Zeichenkette) und `relative` (Boolescher Wert) sein. Der dritte Parameter beschreibt die Kartengröße und sollte ein Objekt mit den Eigenschaften `vSpace` (Zahl) und `hSpace` (Zahl) sein.

In dieser Funktion werden einige Eigenschaften wie Pfad und Sprache festgelegt. Dann wird das mitgelieferte *Stylesheet* angewendet. Danach wird die Funktion `setSize` aufgerufen – erster und dritter `drawmap`-Parameter werden übergeben – und als `window.onresize`-Event festgelegt. Weiters wird die Karte erzeugt und Österreich zentriert.

3.3.2. setSize

Der erste Parameter beschreibt die ID der Karte und sollte eine Zeichenkette sein. Der zweite Parameter beschreibt die Kartengröße und sollte ein Objekt mit den Eigenschaften `vSpace` (Zahl) und `hSpace` (Zahl) sein.

In dieser Funktion werden die verfügbare Höhe und Breite ausgelesen und die Karte erhält dann diese Größe abzüglich der vertikalen und horizontalen Abstände.

3.3.3. setCenter

Die ersten beiden Parameter beschreiben *Longitude* und *Latitude*, also X- und Y-Koordinate. Der dritte Parameter beschreibt die Zoom-Stufe. Alle drei sollten Zahlen sein.

Die Position wird auf die Karte projiziert, zentriert und die Zoom-Stufe eingestellt.

3.3.4. getCenter

Diese Funktion benötigt keine Parameter. Sie liefert ein Objekt zurück, welches die Eigenschaften `lon` (*Longitude*), `lat` (*Latitude*) und `zoom` (Zoom-Stufe) besitzt.

3.3.5. setMarker

Der erste Parameter beschreibt den *Layer*-Namen des neuen Markers und sollte eine Zeichenkette sein. Der zweite Parameter legt fest, ob der neue *Layer* sichtbar sein soll, und sollte ein Boolescher Wert sein. Der dritte und vierte Parameter beschreibt *Longitude* bzw. *Latitude*; diese sollten Zahlen sein. Der fünfte Parameter beschreibt den *HTML*-Inhalt des *Popups*. Der sechste Parameter legt fest, welche Grafik für den Marker verwendet werden soll. Diese letzten beiden sollten Zeichenketten sein.

Wird ein neuer *Layer*-Name angegeben, so wird dieser *Layer* erstellt. Der Marker wird erstellt und dann dem gewählten *Layer* hinzugefügt.

3.3.6. initSingleMarker

Der Parameter beschreibt alle notwendigen Daten für diese Funktion und sollte ein Objekt sein, das folgende Eigenschaften besitzt: `layerName` (Zeichenkette) legt fest, wie der *SingleMarker-Layer* heißt; `icon` (Zeichenkette) legt fest, welche Grafik für den *SingleMarker* verwendet werden soll; `lon` und `lat` (Zahlen, optional) legen fest, ob – nicht wenn mindestens eine dieser Eigenschaften fehlen oder keine Zahl sein – und wo der *SingleMarker* platziert wird; `getLonLat` (Funktion, optional) beschreibt eine Funktion, die mit den Parametern Longitude und Latitude aufgerufen wird, wenn die Position des *SingleMarkers* verändert wurde.

3.3.7. setSingleMarker

Die beiden Parameter beschreiben *Longitude* und *Latitude*, also X- und Y-Koordinate.

Mit dieser Funktion wird der *SingleMarker* neu positioniert.

3.3.8. setLine

Der erste Parameter beschreibt den *Layer*-Namen der Linie und sollte eine Zeichenkette sein. Der zweite Parameter beschreibt die Koordinaten (abwechselnd *Longitude* und *Latitude*) der Eckpunkte der *PolyLine* und sollte ein Zahlen-Array sein. Der dritte Parameter ist optional und kann ein Objekt mit folgenden Eigenschaften sein: `color` (Zeichenkette, optional) legt die Linienfarbe fest (Standard: *blau* `"#0000FF"`); `thick` (Zahl, optional) legt die Linienbreite in Pixel fest (Standard: `3`); `label` (Zeichenkette, optional) legt einen Text fest (Standard: *nichts* `""`).

Diese Funktion erzeugt eine Linie (*PolyLine*).

3.3.9. osmLayer

Der erste Parameter beschreibt den *Layer*-Namen des *Opensteetmap-Layers*. Der zweite Parameter beschreibt den Pfad zur *XML*-Datei. Beide sollten Zeichenketten sein.

Diese Funktion zeichnet einen *Opensteetmap-Layer*.

3.3.10. showCountry

Der Parameter beschreibt das anzuzeigende Land und sollte eine Zeichenkette (Groß-/Kleinschreibung irrelevant) sein.

Diese Funktion zentriert das gewählte Land. Bisher sind nur die Zeichenketten `"Austria"` und `"AT"` für Österreich möglich. Bei einer anderen Eingabe wird die Meldung „Fehler: Land '…' nicht definiert!“ ausgegeben und die Zeichenkette `"Country not found!"` wird zurückgeliefert.

3.3.11. showOverlays

Diese Funktion benötigt keine Parameter. Sie zeigt alle mit `setMarker` und `setLine` erzeugten Objekte in der Karte an.

3.4. Interne Funktionen

Diese sind nur als Methoden von `geo` verfügbar:

3.4.1. `geo.getElementsByClassName`

Der erste Parameter beschreibt den *Tag*-Name der zu durchsuchenden Elemente. Der zweite Parameter legt fest, nach welcher *CSS*-Klasse gesucht werden soll.

Diese Funktion ist für *CSS*-Regeln, die zur Laufzeit festgelegt werden wichtig, da nicht jeder Webbrowser die Funktion `document.getElementsByClassName` zur Verfügung stellt.

4. Offene Fragen?

Nähere Informationen zu ...	finden Sie unter ...
OpenLayers	`http://www.openlayers.org/`
Openstreetmap	`http://www.openstreetmap.org/`
Google	`https://maps.google.at/`

Die *OpenLayers*-Funktionen finden Sie dokumentiert unter:
`http://dev.openlayers.org/docs/files/OpenLayers-js.html`

Bei weiteren Fragen wenden Sie sich an *Telematik Design*:
`http://www.telematik-design.at/`