Aufgabe 2B: JavaScript/AJAX

Erstellung verschiedener Visualisierungen auf JavaScript-Basis

Das Fotoportal *PhotonPainter* bietet seinen Nutzern eine alternative Kartenansicht *PhotonMap*, auf der erkennbar ist, wo ein Foto aufgenommen wurde. Diese Ansicht stützt sich auf vorhandene Kartendienste im Web, wie etwa Google Maps oder Bing Maps ab. Auf einer solchen Karte sollen nun die Bilder aus dem Datenbestand von *PhotonPainter* erscheinen. Außerdem sollen die auf der Karte angedeuteten Bilder noch mit einer detaillierteren Listenansicht synchronisiert werden.

Ihre Aufgabe ist es, diese verschiedenen Ansichten umzusetzen. Die Fotos sollen dabei als Overlay auf der „Grundkarte“ sichtbar sein. Da das Darstellen aller Bilder gleichzeitig große Performanceverluste erzeugen würde, müssen die Bilder und ihre Zusatzinformationen je nach Kartenausschnitt vom Backend *PhotonEmitter* geladen werden. Zoomt der Nutzer zu weit heraus, sollen Cluster gebildet werden: Statt der Darstellung aller Bilder wird lediglich ein Stellvertreterbild zusammen mit der Anzahl der Bilder in diesem Cluster angezeigt. Die Listenansicht zeigt dagegen immer eine scrollbare Liste aller derzeit aktiven Bilder, auch wenn diese durch einen Cluster zusammengefasst werden. Zu jedem Bild sollen hier noch zusätzliche Details sichtbar sein. Schließlich können angemeldete Nutzer den Standort des Bildes durch Verschieben auf der Karte korrigieren.

Das Ergebnis muss folgende Anforderungen erfüllen:

- Durch das Ändern des Kartenausschnittes müssen die entsprechenden Bilddaten vom Backend geladen werden (Caching ist optional möglich). Die Listenansicht muss sich automatisch mit dem Kartenausschnitt synchronisieren.
- Beim Auswählen eines Eintrags in der Listenansicht soll dieser in der Kartenansicht sichtbar hervorgehoben werden.
- Auf der Karte sollen Bilder per Drag&Drop verschoben werden können, um Fehleintragungen zu korrigieren. Vor einer Aktualisierung muss der Benutzer jedoch gefragt werden.
- Verwenden Sie animierte Indikatoren, wenn ein Datenaustausch mit dem Server erfolgt, damit der Nutzer merkt, dass im Hintergrund gearbeitet wird.
- Die Anwendungslogik soll komplett clientseitig sein. Für die Daten dürfen Sie Zugriff auf das Backend nehmen. Der Webservice bietet hierfür allerdings nur eine XML-Schnittstelle.
- Die einzelnen Seiten müssen XHTML-konform und die eingesetzten JavaScripts müssen zumindest in den Browsern Firefox 3.6 und Internet Explorer 8 lauffähig sein.
- *Alle verwendeten Bestandteile müssen entweder Public Domain sein oder die erforderlichen Nutzungsrechte müssen schriftlich vorliegen.*

Abzugeben sind:

- Quellen (`index.html`, CSS-Dateien im Ordner `css`, JavaScript-Dateien im Ordner `js` sowie sonstige verwendete Medienobjekte im Ordner `assets`)

Tipp: Nutzen Sie JavaScript-Debugger wie Firebug oder die Chrome Developer Tools! Tasten Sie sich in kleinen Schritten vor, denn durch komplexe Änderungen entstehen häufig schwer nachvollziehbare Fehler. Nutzen Sie außerdem etablierte JavaScript-Frameworks wie jQuery, um Standardfunktionalität wie das Aufbauen eines XMLHTTPRequests abzudecken.