Externe Schnittstellenbeschreibung JavaScript Client

Version 3.1

Stand 11.06.2015

Herausgeber:

alta4 AG

Fleischstraße 57
54290 Trier
Germany

Fon: +49.651.96626.0
Fax: +49.651.96626.26

www.alta4.com
info@alta4.com

Inhaltsverzeichnis

1 Allgemein

Die externe Programmierschnittstelle des atlasFX Javascript Webclients dient der programmatischen Steuerung durch externe Skripte. Sie stellt über das dojo utility atlasExternalInterface („com/alta4/atlas/extensions/AtlasExternalInterface“) Funktionen zur Steuerung des Verhaltens der Kartenanwendung bereit.

Anmerkung: Eine aktuellere Version des Handbuchs finden Sie zukünftig hier.

2 API Spezifikation

`setLayerFilter(layerId, whereExpression)`
Setz für einen Filter für einen durch layerId spezifizierten Feature-Layer. Die Syntax der Where-Klausel ist durch die jeweils angesprochene arcGIS-Server API gegeben.

`getExtentWebMercator()`
Liefert den Extent der Karte in WebMercator-Koordinaten.

`getExtentGeographic()`
Liefert den Extent der Karte in Längen- und Breitengrad.

`setCenter(x, y)`
Zentriert die Karte auf den übergebenen Punkt. Die Koordinaten müssen im gleichen Koordinatensystem wie die Map sein.

`setCenterWgs84(lat, long)`
Zentriert die Karte auf den übergebenen Punkt. Die Koordinaten sind als Längen- und Breitengrad anzugeben.

`setScale(scale)`
Setzt die Skalierung der Karte. Wenn die Karte im Maßstab 1:50.000 angezeigt werden soll, muss für den Parameter scale der Wert 50000 gesetzt sein.

`getScale()`
Liefert die aktuelle Skalierung der Karte.

`setCenterAndZoom(x, y, levelOrFactor)`
Kombination aus obigen Funktionen, wobei die Koordinaten aus dem Koordinatensystem der Karte angenommen werden und wahlweise ein konfigurierter Zoomlevel oder Faktor angegeben wird.

`setCenterAndScale(mapPoint, scale)`
Wie oben. Hier wird aber ein Skalierungsfaktor angegeben.

`setInfoWindowLinkClickCallback(callback)`
Wird in einem InfoBubble ein Link aufgerufen wird zuerst die Callback-Methode aufgerufen.
`callback(url)`: Callback gibt die angeklickte URL zurück. Die Callback-Methode muss false zurück geben, falls das Ausführen des Links unterdrückt werden soll.

`wgs84ToWebMercator(lat, lon)`
Rechnet Längen- und Breitengrad in WebMercator-Koordinaten um.

`refreshLayers()`
Veranlasst ein Neuzeichnen aller Layer.

`registerGraphicCallback(layerId, callback)`
Registriert auf dem angegebenen Layer einen Klick-Callback für darauf befindliche Grafiken.

`removeGraphicCallback(layerId)`
Entfernt die mit obiger Funktion registrierten Callback-Routinen.

`showFeature(layerId, featureId, showInfoBubble, scale)`
Zentriert die Karte auf das durch layerId und featureId identifizierte Feature. Mit dem booleschen Parameter showInfoBubble kann eingestellt werden, ob das zugehörige Info-Fenster angezeigt werden soll, sofern diese Eigenschaft des Features konfiguriert ist. Der scale-Parameter ist nur für Punktfeatures definiert und optional. Geometrien, die eine Ausdehnung besitzen (Flächen, Linien), werden immer in maximaler Zoom-Stufe dargestellt aber so, dass sie noch ganz auf der Karte liegen.

`setMouseFeatureInteraction(layerTypes, event, procedure, handler)`
Erweitert oder ersetzt die voreingestellte Maus-Interaktion mit Kartenfeatures. Zulässige Parameterwerte:

`layerTypes:` Einer der folgenden Strings oder eine Kombination daraus als Array.
POINT, LINE, POLYGON, CLUSTER, SINGLE_FEATURE_CLUSTER
Die letzten beiden Werte beziehen sich auf geclusterte Maßstabsbereiche. Es wird unterschieden zwischen „echten“ Clustern und solchen, die nur ein einzelnes Feature enthalten und wie ein normales Punktfeature dargestellt werden.

`event:` mouseOver, mouseOut, click
Die zu behandelnden Mausereignisse.

`procedure:` before, after, replace
Mit diesem Parameter wird festgelegt, wie die übergebene Funktion `handler` eingehängt werden soll. Für den Fall, dass der eingehängte Handler nach der default-Funktion ausgeführt wird oder diese ersetzt, wird das Maus-Event dem `handler` übergeben.

Anmerkung: Eine aktuellere Version des Handbuchs finden Sie zukünftig hier.

3 Optionale Werkzeuge

Im Folgenden werden Funktionen aufgelistet, die nur zur Verfügung stehen, wenn die entsprechenden Werkzeuge geladen und aktiviert sind. Nicht alle Werkzeuge gehören zur Standardfunktionsumfang und können über das CMS konfiguriert werden.

Anmerkung: Eine aktuellere Version des Handbuchs finden Sie zukünftig hier.

3.1 Routing Tool

Das Routing Tool erweitert, wenn es geladen und konfiguriert ist, die externe Schnittstelle. Folgende Funktionen stehen dann global unter `atlasExternalInterface.routingTool` zur Verfügung.

`activate()`
Aktiviert die Klick-Routing-Funktionalität, d.h. dann können Wegpunkte durch Klick auf die Karte gesetzt werden. Die Punkte werden durch drei konfigurierbare Symbole (Start, Stop, Zwischenstopp) markiert.

`deactivate()`
Deaktiviert das Klick-Routing auf der Karte. Die Anzeige der aktuellen Routenplanung, sowohl in der Listenansicht als auch gezeichnet in der Karte, sind davon nicht betroffen.

`highlightPosition(geometry)`
An der durch geometry definierten Koordinate wird ein konfigurierbares Hervorhebungssymbol gezeichnet. Dieses wird auf einen speziellen Markierungslayer, zwischen denen für die Route und Wegpunkte gelegt. Zur Verwendung kommt diese Funktion beim Hovern über die Tabellarische Routendarstellung, um eine schnelle Erfassung des Wegpunktes in der Karte zu ermöglichen.

`clearHighlighting()`
Löscht den gesamten Markierungslayer.

`addStop(geometry, title, force)`
Fügt dem Routenmodell einen neuen Stopp an der durch geometry definierten Ort hinzu. Hier wird eine Punktgeometrie erwartet. Wenn title undefiniert ist, wird automatisch die Adresse über einen Dienst für reverse Geokodierung ermittelt und diese als Titel für die tabellarische Darstellung herangezogen. Der Parameter force erzwingt eine unmittelbare Übernahme der Daten ins Modell. Im false-Fall wird eine Synchronisierung des Routenplaners mit dem Feature-Klickverhalten zwischengeschaltet. Falls nämlich ein Feature-Symbol geklickt wird, soll zunächst nur das Informationsfenster dazu angezeigt werden. Bei eingeschaltetem Klick-Routing wird auf diesem dann ein Knopf zur Verfügung stehen, der eine Aufnahme des Features in die Route veranlasst. In diesem Fall wird als Titel des Wegpunkts der konfigurierte Titel des Informationsfensters übernommen.

`addStopByAddress(address, plz, city)`
Über einen konfigurierten Geoservice wird die Adresse versucht geografisch zu codieren. Im Erfolgsfall wird das Ergebnis als Wegpunkt in die Route aufgenommen. Der Titel wird aus der Kandidatenadresse des Geocoding-Ergebnisses zusammengesetzt.
Das hinzufügen eines Wegpunktes veranlasst die Routenberechnung höchstens eines Routensegments zwischen dem hinzugefügten Ort und seinem Vorgänger. Bei Erfolg wird dieses Routensegment gezeichnet. Ist einer der beteiligten Punkte nicht traversierbar, wird das Segment nicht ins Modell übernommen.

`removeStop(routeSegment)`
Entfernt ein Routensegment, also einen Punkt und das letzte Routenteilstück, das zu ihm hinführt. Gegebenenfall wird das Segment zwischen den verbleibenden Teilrouten neu berechnet.

`clearRoute()`
Löscht die gesamte Route.

`ZoomToExtent()`
Zoomt und Verschiebt die Karte so, dass die Route in ihrer Gesamtheit zu sehen ist und annähernd die volle Darstellungsfläche ausfüllt.

`ExportRouteJSON()`
Exportiert die Route als Komplex von Segmenten in der Form, dass eine Rekonstruktion des internen Datenmodells möglich ist.

`ExportSimplifiedRouteJSON()`
Exportiert eine vereinfachte Form der Route, nur mit Start- und Zielpunkt als eine Geometrie.

`closeRoute()`
Schließt Route zu einem Rundweg.

Anmerkung: Eine aktuellere Version des Handbuchs finden Sie zukünftig hier.