Invoking Navigation Global From Other Applications

Last updated on February 23, 2022

Overview

Verizon Connect Navigation Global Edition (NavGE) can be called by other applications to provide route guidance to destinations provided by those applications. Since NavGE requires authentication, the calling app must provide authentication information before it can be used. Authentication is in the form of a TDE token and a username. All data is passed as key-value pairs (KVP), but the mechanism for doing this depends on the OS.

For Android, the call is made using an explicit intent, and all the data is passed as KVPs (called “extra” data in an Android operating system) in the intent. For iOS, most of the data is passed on the URL of a URL scheme, with extra data being passed on the iOS pasteboard.

Invocation Data

Command type

`command=[login|navigate]`

The `login` command launches Navigation or brings it to the foreground, and validates authentication KVP data.

The `navigate` command launches Navigation or brings it to the foreground, and uses the destination and optional route settings KVP data to calculate a route. If provided, it also validates authentication KVP data.

Banner Specification

The banner parameters let you reserve the top or bottom (or both) parts of the screen for customization (for example, icons for a specific customer's system). By default, the Navigation app populates the full screen.

- `TopPadding` = the height of the top banner in DIP (Device-Independent-Unit).
- `BottomPadding` = the height of the bottom banner in DIP.

Authentication

TDE Authentication:

- `tdetoken` = token as received from an authentication request from TDE.
- `tokenexpiry` = the `tdetoken` expiration date (in ISO 8601 format, UTC time zone) as received from the authentication request from TDE.
- `user` = username in Fleet user format (`company:driver`) used to acquire the tdetoken.

Authentication with Verizon Connect Reveal STSv4 Token

Instead of a TDE token, Navigation can accept a Reveal STSv4 token.

- `externaltoken` = string value containing the token value, optionally prefixed with a region identifier and the "-" character; for example, "EU-" for Europe. If the prefix is omitted, the US region is assumed.
- `externaltokentype` = "VZCSMB" for Reveal/STSv4 token.

Destination

NavGE checks the destination parameters in the following order: `site`, `locationid`, `latitude` and `longitude`, `address`. As soon as a parameter is found, the remainder are ignored. If no parameters are found, NavGE reports an error.

Invoking Navigation Global From Other Applications

Option 1: The site data of the location in JSON format.

- `site` = For Android, this is a JSON string in the format of a TDE Site. For iOS, this is a key pointing to a JSON string of a TDE Site on the iOS pasteboard. `site` is used in preference to `latitude`/`longitude`. It is generally preferred for two main reasons:
    - Routing to geodetic points (`latitude`/`longitude`) can produce unexpected results as the point can snap to the wrong road near the point.
    - With only a geodetic point, the destination information must be deduced via reverse geocoding. This makes it possible to get the road name wrong, but also makes it impossible to determine building numbers reliably.

  The full specification for a TDE site is outside the scope of this document. However, the minimum required site elements for use with NavGE are as follows:

    - An ID value (`0` unless the site data corresponds to an actual Fleet Site record).
    - Location point.
    - One member of the Addresses array with a street number (if known), and a street name.

Option 2: The ID of the customer's marker.

- `locationid` = The ID of a customer's marker. The `locationid` is used to find the marker and download the marker information, which is then used to generate a route to the location.

Option 3: The latitude and longitude of the location.

- `latitude` = Must be provided along with longitude.

- `longitude` = Must be provided along with latitude.

Option 4: The textual address of the location.

- `address` = Textual address.

Region Restriction

- `restrictedregions` = the codes of the restricted regions, each separated by a semicolon. A region code is the code of the subdivision defined in ISO 3166-2; for example, US-CA is the code for California. US-CA is the only region code currently supported. Other regions might be added in the future.

The California Air Resources Board has implemented regulations to control the amount of greenhouse gas emissions from motor vehicles in California. As a result, most trucks made before 2010 cannot legally operate in California.

The `restrictedregions` parameter can be used to support the California region restriction.

A region restriction parameter needs to be specified in the route request if the regional restriction is applicable to the vehicle. If a region restriction is specified, the following results:

- The vehicle is not permitted to enter the restricted region, so Navigation does not generate a route from outside of the restricted region into the restricted region.
- Navigation generates a route that avoids passing through the restricted region.
- Vehicles may navigate from a restricted region to a non-restricted region. In this case, a warning is provided to the driver.

Route Settings

- `hazmat` = (Optional) Has a list of hazmat codes, each corresponding to a hazmat setting, separated by a comma to be enabled for the route. For more information see Nine Classes of Hazardous Materials.

Vehicle Dimensions

- `dimensionunits` = (Optional) Preferred unit of measure for vehicle's dimension. This is case-insensitive. Valid values are 'm' and 'ft', for meters and feet, respectively.
- `length` = (Optional) Length of vehicle. For decimal numbers, enter a dot for a decimal point (a comma is unavailable). This value will be ignored if 'dimensionunits' is missing or invalid. Valid value range is 8 ft - 10000 ft.
- `width` = (Optional) Width of vehicle. For decimal numbers, enter a dot for a decimal point (a comma is unavailable). This value will be ignored if 'dimensionunits' is missing or invalid. Valid value range is 2 ft - 500 ft.
- `height` = (Optional) Height of vehicle. For decimal numbers, enter a dot for a decimal point (a comma is unavailable). This value will be ignored if 'dimensionunits' is missing or invalid. Valid value range is 2 ft -100 ft.
- `weightunit` = Preferred unit of measure for vehicle's weight. Valid values are 'lb' and 'kg', for pounds and kilogram, respectively.
- `grossweight` = Gross Weight of the vehicle, which includes weight of the vehicle and cargo. For decimal numbers, enter a dot for a decimal point (a comma is unavailable). This value will be ignored if 'weightunit' is missing or invalid. Valid value range is 10000 lb - 500000 lb.

Back-to-app behavior

Navigation goes back to the calling app in the following cases:

- The route was completed by arrival at the destination.

- The user taps the back arrow on the **Route Overview** screen.

To enable this functionality in iOS, an `appId` property must be set in the navigation request. When Nav is to go back to the calling app, it opens a URL with the value of `appId` if the value is in the list supported by Nav (supported values: TelogisWorkplan). For example, if `appId=MyApp`, Nav opens the URL `MyApp://`. The calling app should register itself with iOS to open URLs with that pattern.

For Android, no measures are needed to enable this functionality. In the cases above, Nav sends itself to the background.

API Authentication Behavior

Navigation checks for the presence of authentication data when it receives an intent. For intent requests with `command=login`, authentication data is required. For requests with `command=navigate`, Navigation's behavior depends on the intent data and the application's current state. The following table describes the possible outcomes:

	Navigation is already authenticated	Navigation is not authenticated
Intent **contains** authentication data	Verify and use new intent authentication data	Verify and use intent data
Intent **omits** authentication data	Keep Navigation's existing authentication	Show login page

If Navigation is already authenticated and it receives an intent with valid authentication data, Navigation will change to the user specified in the intent. If the user change cannot be completed, Navigation returns to the login page momentarily before reverting back to the original user.

Code Examples

Android

The request should be sent to NavGE through Android intent. NavGE is brought to the foreground when the intent is received.

The package name of NavGE is “com.telogis.nav6”. It can be used for launching NavGE and passing request parameters to NavGE.

Code examples

The following are Java examples. Another common way to test intents on Android is to invoke the device Activity Manager using adb.

1. Login Request:

```
Intent intent =
getPackageManager().getLaunchIntentForPackage("com.telogis
.nav6");
intent.PutExtra("command", "login");
intent.PutExtra("user", "FleetName:Username");
intent.PutExtra("tokenexpiry", "2020-09-29T18:00:00Z");
intent.PutExtra("tdetoken", "Token");
// Launch NavGE and pass request parameters
startActivity(launchIntent);
```

Via ADB activity manager (Linux/MacOS command line):

```
adb shell 'am start -a android.intent.action.MAIN -n
"com.telogis.nav6/telogis.navigation.MainActivity" -e
command login -e user "examlcustmr:exampldrvr" -e tdetoken
"tokendata" -e tokenexpiry "YYYY-MM-DDThh:mm:ssZ"'
```

Invoking Navigation Global From Other Applications

Via ADB activity manager (Windows command line):

```
adb shell ""am start -a android.intent.action.MAIN -n
"com.telogis.nav6/telogis.navigation.MainActivity" -e
command login -e user "examplecustomer:exampledriver" -e
tdetoken "tokendata" -e tokenexpiry "YYYY-MM-DDThh:mm:ssZ"
""
```

Navigation request:

```
Intent intent =
getPackageManager().getLaunchIntentForPackage("com.telogis
.nav6");
double latitude = 43.645238
double longitude =  -79.364566
string hazmatSettings = "1,2,3"
intent.PutExtra("command", "navigate");

// Optional banner specification for a customized top or
// bottom (or both) banner; for example:
int topBannerHeight = 50;
int bottomBannerHeight = 50;
intent.PutExtra("TopPadding", topBannerHeight);
intent.PutExtra("BottomPadding", bottomBannerHeight);

// Optional authentication data
intent.PutExtra("user", "FleetName:Username");
intent.PutExtra("tdetoken", "Token");
intent.PutExtra("tokenexpiry", "YYYY-MM-DDThh:mm:ssZ");
// End of optional authentication data
intent.PutExtra("latitude", latitude);
intent.PutExtra("longitude", longitude);
intent.PutExtra("address", "Address");
intent.PutExtra("hazmat", hazmatSettings);
intent.PutExtra("dimensionunits", "m");
intent.PutExtra("length", "27.4321");
intent.PutExtra("width", "2.5908");
intent.PutExtra("height", "4.1148");
intent.PutExtra("weightunit", "lb");
```

```
intent.PutExtra("grossweight", "79365");

// Launch NavGE and pass request parameters
startActivity(launchIntent);
```

Via ADB activity manager (Linux/MacOS command line):

```
adb shell am start -a android.intent.action.VIEW -n
com.telogis.nav6/telogis.navigation.MainActivity --es
command navigate --es latitude 43.651511 --es longitude
-79.372205 --es hazmat 1,2,3 --es dimensionunits m --es
length 27.4321 --es width 2.59082 --es height 4.11480 --es
weightunit lb --es grossweight 79367
```

Via ADB activity manager (Windows command line):

```
adb shell am start -a android.intent.action.VIEW -n
com.telogis.nav6/telogis.navigation.MainActivity --es
command navigate --es latitude 43.651511 --es longitude
-79.372205 --es hazmat 1,2,3 --es dimensionunits m --es
length 27.4321 --es width 2.59082 --es height 4.11480 --es
weightunit lb --es grossweight 79367
```

iOS

1. Register NavGE URL scheme in iOS app:

   NavGE provides a URL scheme `TelogisNavigationGlobal://`. An iOS app can pass a query string to NavGE when launching NavGE.

   To check if NavGE can be opened with the canOpenURL method, register the URL scheme in `info.plist` of the iOS app.

```
<key>LSApplicationQueriesSchemes</key>
<array>
     <string>TelogisNavigationGlobal</string>
</array>
```

2. Launch NavGE and pass request to NavGE with OpenUrl:

   Login request

```
string url = "TelogisNavigationGlobal://?" +
 "command=login" +
 "&user=FleetName:UserName" +
 "&tdetoken=Token" +
 "&tokenexpiry=TokenExpiry";

 // Launch NavGE and pass request parameters
 UIApplication.SharedApplication.OpenUrl(url);
```

   Navigation request

```
// Write the site data to the pasteboard if there is a
site associated with the location.
String SiteData = …;
String siteKeyInPasteBoard = "site";
UIPasteboard.FromName pasteBoard =
UIPasteboard.FromName(siteKey, true);
pasteBoard.String = SiteData;

string url = "TelogisNavigationGlobal://?" +
        "command=navigate" +
     "&latitude=Latitude" +
     "&longitude=Longitude" +
     "&address= Address" +
     "&site=" + siteKeyInPasteBoard // the name of the
pasteboard of the site data.
     "&user=FleetName:UserName" + // optional, do not have
to pass user credentials if the user has logged in
previously.
```

```
        "&tdetoken=Token" +
        "&tokenexpiry=TokenExpiry" +
        "&hazmat=HazmatSettings" +
        "&dimensionunits=DimensionUnits" +
        "&length=Length" +
        "&width=Width" +
        "&height=Height" +
        "&weightunit=WeightUnit" +
        "&grossweight=GrossWeight" +

// // Launch NavGE and pass request parameters
UIApplication.SharedApplication.OpenUrl(url);
```

Android General Navigation Intents

On the Android platform, Navigation also listens for implicit intents from other applications. The supported intent schemes, and Navigation's behavior for each, is described below.

NavGE supports the Geo scheme and Navigation scheme.

1. **Geo scheme**

   `geo:latitude,longitude`

   Navigate to location at the given longitude and latitude.

   Example: `geo:43.780252,-79.411594`

   `geo:latitude,longitude?z=zoom`

   Navigate to location at the given longitude and latitude at a certain zoom level.

   Example: `geo:43.780252,-79.411594?z=11`

   NavGE ignores the z=zoom value.

   `geo:0,0?q=latitude,longitude(label)`

   Navigate to location at the given longitude and latitude with a string label.

Example: `geo:0,0?q=43.780252,-79.411594(Treasure)`
NavGE attempts navigation to the given coordinates, but the label string is ignored.

`geo:0,0?q=my+street+address`
Navigate to location for "my street address" (may be a specific address or location query).
Example: `geo:0,0?q=20+Enterprise%2C+CA`
Note: All strings passed in the geo URI must be encoded. For example, the string "1st & Pike, Seattle" should become `1st%20%26%20Pike%2C%20Seattle`. Spaces in the string can be encoded with %20 or replaced with the plus sign (+).

2. **Google Navigation scheme**
   Use this intent to navigate to the address or specified coordinate.
   - `google.navigation:q=a+street+address`
   - `google.navigation:q=latitude,longitude`
   - `google.navigation:ll=latitude,longitude`

Specific Use Cases

Addresses commonly unknown to NavGE

Users who navigate to newly developed areas that have addresses that cannot be geocoded should use a site definition. A site definition should include the following:

- Longitude and latitude coordinates, entered in the "lat" and "lon" fields.
- Address information, entered in the "name" field.

The following example shows a navigate command and a site definition:

```
command = navigate
```

```
site = "{\"id\": 0,\"position\":
{\"lat\": 54.980771,\"lon\": -1.646463}
,\"name\":\"Address or name they want displayed to the
driver\",\"addresses\": [
{\"city\": \"City name to display, if desired\"}
]}";
```

This command navigates to the provided lat/lon. The value of the "name" key is displayed at the top of the **Location Overview** screen on the NavGE app.