Traffic API

Developer's Guide

Version 6.0.28.1

Contents

Legal Notices

Trademark Acknowledgements

HERE and Nokia are trademarks or registered trademarks of Nokia Corporation in the United States and other countries.

Other trade names are trademarks or registered trademarks of their owners.

Disclaimer

This content is provided "as-is" and without warranties of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability, fitness for a particular purpose, satisfactory quality and non-infringement. Nokia does not warrant that the content is error free and Nokia does not warrant or make any representations regarding the quality, correctness, accuracy, or reliability of the content. You should therefore verify any information contained in the content before acting on it.

To the furthest extent permitted by law, under no circumstances, including without limitation Nokia's negligence, shall Nokia be liable for any damages, including, without limitation, direct, special, indirect, punitive, consequential, exemplary and/ or incidental damages that result from the use or application of this content, even if Nokia or an authorized representative has been advised of the possibility of such damages.

Document Information

Product	
Name:	Traffic API
Version:	Version 6.0.28.1

Document	
Name:	Traffic API Developer's Guide
Id:	b9c9641-1391082634
Status:	FINAL
Date:	2014-Jan-30, 11:51 (GMT)

Chapter 1

Overview

Topics:

The articles that follow introduce the Traffic API, describe common use cases, and introduce essential concepts.

What is the Traffic API?

HERE Traffic API is a RESTful API that provides traffic flow information and traffic map tiles.

To obtain traffic data via the HERE Traffic API, it is necessary to formulate a request that combines the URL and a set of parameters to specify the required response.

The HERE Traffic API encompasses services outlined below.

Traffic Incident Data

The API provides aggregated information about traffic incidents in XML or JSON, including the type and location of each traffic incident, status (whether it is still active), start and end time, and other relevant data.

Traffic Map Tile Overlays (Traffic Tiles)

The Traffic API delivers pre-rendered map tile overlays with traffic information, ready to be displayed by mapping applications. You can request map tiles that show traffic trends (patterns) based on historical traffic data for a specific area.

Traffic Flow Data

The service offers access to real-time traffic flow data in XML or JSON, including information on speed and congestion for the region(s) defined in each request, and can deliver additional data such as the geometry of the road segments to which the flow data relate.

Traffic Flow Availability

The service allows client applications to access the general traffic information either specified by a map segment (defined as a map view), or in general for the whole world (without the map view).

Why use the Traffic API?

The Traffic API addresses the following high-level use cases:

- Get real time traffic flow data in XML or JSON
- Get a real-time traffic tile overlay for a map image

    - Filter traffic tile data by TMC table or profile
    - Specify the size of a traffic tile overlay in a mapping application
- Set traffic incident data in XML or JSON formats
    - Filter traffic incident data by TMC table or profile
    - Filter traffic incident data by attributes
- Get traffic flow availability data in XML or JSON

Chapter 2

Quick Start

Topics:

The section provides information to help you start using the HERE Traffic API. It outlines a simple scenario and shows the matching requests.

Note that each request must include the authentication parameters `app_id` and `app_code` to access the resources of the API. All examples in this guide use demo values of these parameters, but a real-life web site or application must substitute its own unique authentication credentials. For further information, please see *Acquiring Credentials* on page 13.

Requesting Traffic Incident Data

A basic scenario involving the use of the Traffic API is to obtain traffic incident data for a specific area. The two request examples below provide a working illustration.

Both requests ask for traffic incident data for an area around Frankfurt, Germany. To do this, they specify:

1. the base URL, and the URL path, consisting of the service name ("traffic") and service version
2. the targeted resource (`incidents`) – all the available resources and further examples are covered in the subsequent parts of this documentation
3. the area for which to retrieve traffic incident data – this is done by identifying a map tile in a grid of tiles covering the entire globe
4. the authentication credentials

The example requests differ in the way they identify the map tile (point 2 above) and in the response (output) format. The first request uses the [Z]/[X]/[Y] addressing scheme to indicate the map tile and thus provides the zoom level and the coordinates of the tile (see *Mercator Projection and Map Tiles* on page 17 for details). It indicates that XML is the required output format. The second example identifies the map tile via a quadkey (see also *Quadkeys* on page 19) and specifies JSON as the output format.

Incident Reports Example – XML Output and [Z]/[X]/[Y] Tile Addressing

A request for traffic incident data to be delivered in XML, with the [Z]/[X]/[Y] addressing scheme specifying the map tile, is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/incidents/xml/8/134/86
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
```

To specify the tile for which to obtain traffic incident data, the request indicates the zoom level as 8, the tile column as 134, and row as 86.

The complete response is too long to include here. It consists of the top-level element `<trafficcml_incidents timestamp="09/02/2013 13:37:19 GMT" VERSION="5.0">` followed by `<traffic_items>`. `<traffic_items>` includes any number of individual `<traffic_item>` elements that contain the incident location, time, duration, severity, description and other details.

Traffic Incident Data Example – JSON Output and Quadkey Tile Addressing

A request for traffic incident data for the same area around Frankfurt, Germany, to be delivered as a JSON object and using a *quadkey* to specify the map tile, is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/incidents.json
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=12020330
```

For additional use cases, see *Traffic Incident Data* on page 53.

Chapter 3

User Guide

Topics:

This section examines aspects of the HERE Traffic API in detail, covering:

- authentication credentials
- REST request formats and URL construction
- rendered traffic flow tiles
- incident data

Further articles in this section offer examples that match various use cases for the API. You can use and modify the examples to explore the API and build your own applications, provided that you replace the demo authentication parameter values with your own authentication credentials.

Deprecation Notice

HERE has changed the way traffic tiles are served. Live traffic tiles have been removed from the Traffic API and are now delivered by the Map Tile API infrastructure. Documentation relating to live traffic tiles is part of the Map Tile API documentation.

This does not affect traffic pattern tiles, which continue to be served by the Traffic API.

Acquiring Credentials

All users of HERE APIs must obtain authentication and authorization credentials and provide them as values for the parameters `app_id` and `app_code`. The credentials are assigned per application.

To obtain the credentials for an application, please visit *http://developer.here.com/get-started* for more details.

Constructing a Request

A request to the Traffic API includes the basic elements shown in the following table and, in addition, it may contain resource-specific parameters.

Table 1: Basic request elements

Element	Value/Example	Purpose	Description/comments
Base URL	`http://maps.nlp.nokia.com`	Tiles	Production environment.
	`http://traffic.api.here.com`	Incidents, Flow, or Flow Availability	
	`http://maps.st.nlp.nokia.com`	Tiles	CIT environment: see *Customer Integration Testing* on page 16.
	`http://traffic.cit.api.here.com`	Incidents, Flow, or Flow Availability	
Path	`/traffic/6.0/` or `/traffic/6.1/`	For `flow` resource	
Resource	`tiles`	To obtain traffic map tiles	Name of a resource. Only the GET method is supported. Parameters specify request details. See also *Examples* on page 27.
	`incidents`	To obtain traffic incident data	
	`flow`	To obtain traffic flow data (uses `/traffic/6.1/` as path)	
	`flowavailability`	To obtain information on traffic flow availability	
Addressing Scheme	Quadkey, [Z]/[X]/[Y], Bounding Box or Proximity	Traffic Flow Data	Geographic area for which to retrieve data; options depend on resource/function.
	Quadkey, [Z]/[X]/[Y]	Traffic Tiles	
	Quadkey, [Z]/[X]/[Y], Bounding Box or Proximity	Traffic Incidents	

Element	Value/Example	Purpose	Description/comments
	Customer Profile or Mapview	Traffic Flow Availability	
Application Code	`&app_code=AJKnXv84fjrb0KIHawS0Tg`	All requests	Substitute your own unique `app_code`.
Application Id	`&app_id=DemoAppId01082013GAL`	All requests	Substitute your own unique `app_id`.

The code excerpt that follows demonstrates a traffic incident search request. It uses the HTTP GET method and defines the area to search via the `bbox` parameter (bounding box). The value of the parameter is the latitude and longitude of the top left and bottom right corners of the area for which to retrieve traffic incident data. The last parameter, `criticality`, indicates the severity of the accidents about which to obtain information.

```
http://traffic.cit.api.here.com/traffic/6.0/incidents.json
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&bbox=52.516,13.355;52.511,13.400
&criticality=critical
```

Request Format

Each request must conform to the format appropriate to the *resource* (service function) and the addressing scheme (geospatial filter) the request uses to define the area for which to retrieve traffic information.

The supported addressing schemes are:

- Tile addressing schemes:
  - *[Z]/[X]/[Y]* – specifies the map zoom level and the coordinates of the map tile in the tile grid corresponding to that zoom level, see *Mercator Projection and Map Tiles* on page 17 for details
  - *quadkey* – an alternative to the *[Z]/[X]/[Y]* addressing scheme that uses a single value to identify the map tile, see also *Quadkeys* on page 19
- Area filters:
  - *bounding box* – defines the area for which to retrieve information as a bounding box (using the parameter `bbox`, whose value is specified as the latitude and longitude of the top left and bottom right corners of the area, for example, `bbox=52.516,13.355;52.511,13.400`)

- *proximity* – defines the area for which to retrieve information as a circle (using the parameter `prox`, whose value is the latitude and longitude of the search center, followed by the radius of the search area in meters)

Please see also *Table 1: Basic request elements* on page 13 for information about addressing schemes and available resources.

Once you have determined the resource and the addressing scheme, select the appropriate request format pattern, following the guidelines below (listed according to the API resource, whose name is shown in brackets).

Traffic Flow Data (flow)

Traffic flow data can be retrieved using one of the following request patterns, which reflect the addressing scheme:

Table 2: Traffic flow data request formats

Addressing scheme	URL pattern
[Z]/[X]/[Y]	`{base-url}/{path}/{zoom}/{column}/{row}/{format}?`
Quadkey	`{base-url}/{path}/{resource}.{format}?quadkey={quadkey}`
Bounding box	`{base-url}/{path}/{resource}.{format}?bbox={bounding box coordinates}`
Proximity	`{base-url}/{path}/{resource}.{format}?prox={proximity coordinates}`

Note `[path]` in the request patterns above must be `/traffic/6.1/`.

Traffic Tile (tiles)

Traffic tiles can be retrieved using one of the following request patterns, which reflect the addressing scheme:

Table 3: Traffic tile request patterns

Addressing scheme	URL pattern
[Z]/[X]/[Y]	`{tile-base-url}/{path}/{resource}/{zoom}/{column}/{row}/{size}/{format}?`
Quadkey	`{tile-base-url}/{path}/{resource}/quadkeytraffic?quadkey={quadkey}`

Note that when using the [Z]/[X]/[Y] tile addressing scheme, you can specify tile size and color depth as follows:

- `{size}`: tile size, can be one the following values 512, 256, or 128.

- `{format}`: color-depth, can be one of the following `png`, `png8`, or `png32`.

See also *Selecting Tile Size* on page 42 and *Selecting Tile Color Depth* on page 45.

Traffic Incidents (incidents)

Traffic incident reports can be retrieved using one of the following request patterns, which reflect the addressing scheme:

Table 4: Traffic incident request patterns

Addressing scheme	URL pattern
[Z]/[X]/[Y]	`{base-url}/{path}/{resource}/{format}/{zoom}/{column}/{row}?`
Quadkey	`{base-url}/{path}/{resource}.{format}?quadkey={quadkey}`
Bounding box	`{base-url}/{path}/{resource}.{format}?bbox={bounding box coordinates}`
Proximity	`{base-url}/{path}/{resource}.{format}?prox={proximity coordinates}`

Traffic Flow Availability (flowavailability)

Traffic flow availability information can be retrieved for your profile with an address following the pattern shown below:

Table 5: Traffic flow availability request pattern

Addressing scheme	URL pattern
Profile	`{base-url}/{path}/{resource}.{format}?profile={profile name}`

Customer Integration Testing

HERE is committed to maintain the best possible production service for all customers. Given that the production environment is live and common to all API users, we request that you use the alternative Customer Integration Testing (CIT) environment when evaluating our products, running tests, making changes in your code and altering the way you access our APIs.

The CIT environment also allows you to test your software against a newer version of the service before HERE brings that version into production. CIT offers a fully functional environment for customers to use for development and testing, but it does not support high loads or performance testing in general.

Note that the same application id can be used in both environments, but CIT may require a dedicated application code. If this is the case, please contact us as described under *Service Support* on page 64.

The CIT environment is not intended for general production use.

Key Concepts

This section defines key concepts that relate to the HERE Traffic API.

Mercator Projection and Map Tiles

The Traffic API serves map tiles obtained by mapping points on the surface of a sphere (the globe) to points on a plane, using the *normalized Mercator projection*.

The basic Mercator projection formula is this:

{λ, φ} -> x[-1, 1] y [-1, 1]

In this formula:

λ = longitude

φ = latitude

x = λ / π

y = ln(tan(π/4 + φ/2)) / π

The plane represents the globe as a square grid of map tiles. The size of the grid depends on the map zoom level. At the lowest zoom level, the world is contained in one map tile. At the next higher zoom level, the world is two tiles wide and two tiles high (2x2), at the next level above that, the grid is 4x4, then 8x8, 16x16, and so on up to the maximum zoom for a particular region. In other words, at each zoom level the tiles that make up the complete map of the world form a grid in which the number of tiles is equal to the zoom level raised to the power of two ($zoom^2$).

The relationship between tiles at two consecutive zoom levels can be expressed as follows:

$col_{1,z+1} = (2*col_z) + 1 row_{1,z+1} = (2*row_z) + 1$

In this formula:

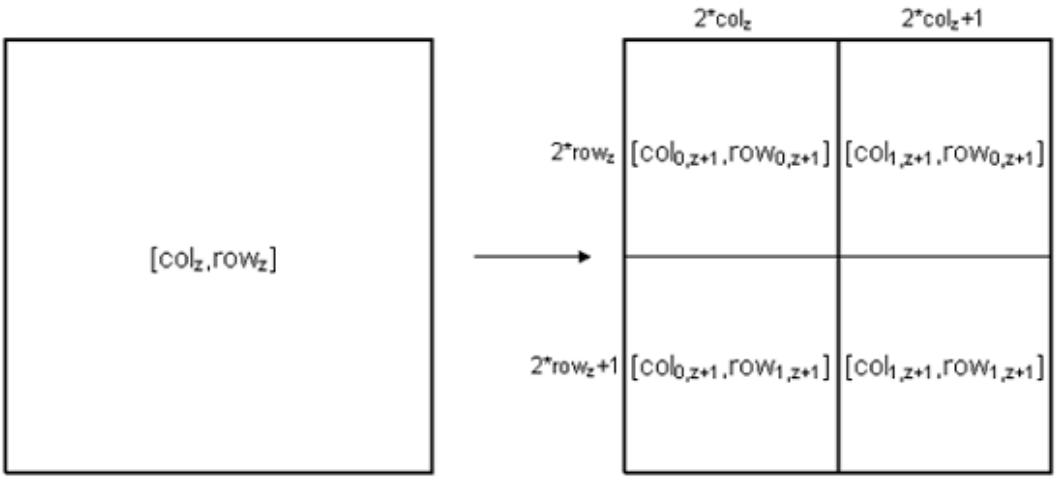
col = column number in the tile grid

row = row number in the tile grid

z = zoom level

The diagram below demonstrates this graphically:

Figure 1: Tiles at different zoom levels



You can use this information to obtain the grid coordinates (row and column) of the map tile for a particular geographic location in your application. The following pseudo code contains the complete algorithm:

```
--- javascript ---
var lat = 52.525439, // Latitude
lon = 13.38727,      // Longitude
z = 12,              // Zoom level
latRad,
n,
xTile,
yTile;

latRad = lat * Math.PI / 180;
n = Math.pow(2, z);
xTile = n * ((lon + 180) / 360);
yTile = n * (1-(Math.log(Math.tan(latRad) + 1/Math.cos(latRad)) /
Math.PI)) / 2;

--- output ---
lat_rad = 0.916
n = 4096
xTile = 2200.31 // Column
yTile = 1343.20 // Row
```

The zoom level and tile row and column can be used as URL variables separated by the '/' character in map tile requests. Note that they must be provided in this order: `zoom/column/row`. This is the *[Z]/[X]/[Y] addressing scheme*.

The map tile specification is typically preceded by other path variables and may be followed either by further path variables and/or query parameters.

Quadkeys

Specifying the grid location of a map tile in terms of zoom level, column and row as described under *The Mercator Projection* is easy to understand and intuitive in practise. However, the grid is a two-dimensional array and as such does not offer efficient storage and retrieval. A better solution is a one-dimensional array, where each item is uniquely addressable by a single value. This is made possible by quadkeys, which combine the zoom level, column and row information for a tile in a one value.

In fact, a quadkey is a string containing a numeric value. The value is obtained by interleaving the bits of the row and column coordinates of a tile in the grid at the given zoom level, then converting the result to a base-4 number (the leading zeros are retained). The length of a quadkey string (the number of digits/characters) equals the zoom level of the tile.

For example, we can obtain the quadkey for a map tile in column 3 and row 5 at zoom level 5 as follows:

```
// Convert the column (x) and row (y) values
// to their binary (b) equivalents:
x = 3 -> 011b

y = 5 -> 101b

// Interleave the binary values (b), convert the
// combined result to a base-4 (q) number and
// finally convert that to a string:
quadkey = 100111b -> 213q -> “213”
```

The code below shows a JavaScript implementation of the algorithm that calculates quadkeys. The inputs are the coordinates of the a map tile and the zoom level. The return value is a string containing the quadkey. The lower part of the code block shows the function called to calculate a quadkey for zoom level 16, which is also the length of the output string shown on the last line.

```
--- input ---
xTile: 35210  // Column
yTile: 21493  // Row
z: 16         // Zoom Level

--- JavaScript ---
function tileXYToQuadKey(xTile, yTile, z) {
    var quadKey = "";
    for (var i = z; i > 0; i--) {
        var digit = "0",
        mask = 1 << (i - 1);
```

```
        if ((xTile & mask) != 0) {
            digit++;
        }

        if ((yTile & mask) != 0) {
            digit = digit + 2;
        }
        quadKey += digit;
    } // for i return quadKey;

    return quadKey;
}
quadKey = tileXYToQuadKey(35210, 21493, 16);

--- output ---
quadKey = "1202102332221212"
```

Note that when using a quadkey address, you cannot specify the size or format of the map tile image. The response always includes an image that measures 256 x 256 pixels and the format is png32.

Traffic Coverage

The HERE Traffic API offers traffic data for the regions described in *Table 23: TMC Table IDs* on page 79. Note that coverage may vary, depending on the customer profile.

For details about traffic coverage available for your customer profile, please contact your HERE representative.

Traffic API Limitations

To reduce the size and increase the relevance of traffic data, the service limits the area and detail of traffic data that you can request for traffic flow data, traffic tiles and incident data. You can request tiles or incident reports only for zoom levels 8-21.

Traffic Flow Data Limitations

The HERE Traffic API restricts the maximum size of a geographical filter used to retrieve traffic flow data. The limitation applies to requests that specify a bounding box and proximity filters. The width and height of a bounding box must be no greater than 10 degrees of longitude and latitude, while the maximum permitted proximity radius is 100km.

Queries using larger spatial filters result in an error response.

The recommended maximum size for a bounding box and quadkey (tile) is 1 degree of longitude and latitude. The largest recommended radius in proximity-based queries is 100km. For these ranges, the SLAs apply, which are documented separately.

Traffic Tiles and Zoom Level

The traffic flow detail changes with the zoom level so as not to overcrowd the display. At low zoom levels, map tiles show traffic flow information only for the highest-level functional road classes. Traffic on minor roads becomes visible and then more finely grained as the zoom level goes up (greater map detail is apparent).

For example:

- At zoom level 14, traffic information is displayed on highways
- At zoom level 15, traffic information is shown on highways and state roads

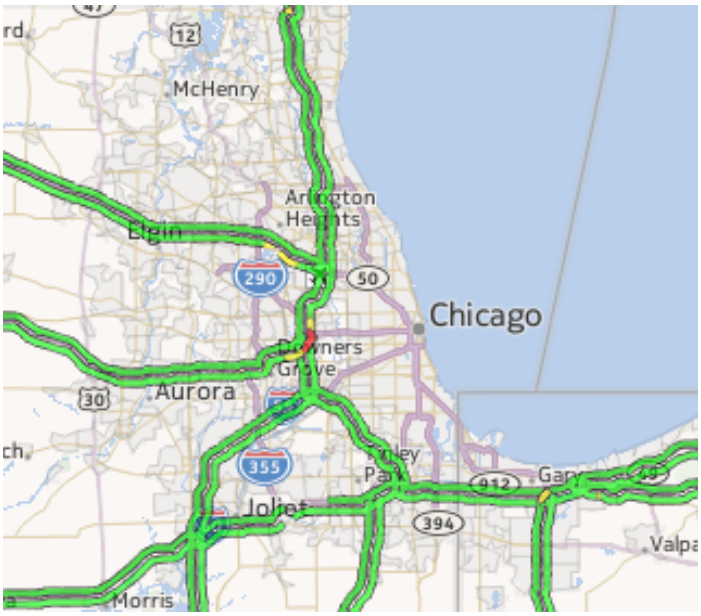
You may have access to a customized profile that displays different traffic flow data, depending on the zoom level.

The images below demonstrate traffic tile detail available at different zoom levels. The first two contain no traffic information, because the zoom level is too low at 5. As the zoom level increases in further tiles, traffic flow data on increasingly lower functional road classes become visible.

Figure 2: Zoom level too low to show traffic

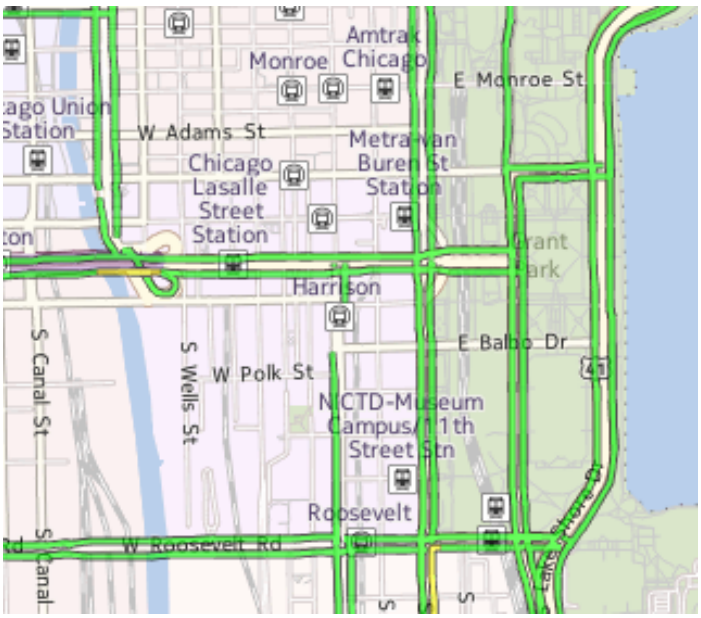
Some traffic flow information can be seen zoom level 8 in the following image.

Figure 3: Zoom level 8 begins to show traffic flow overlay



The higher the zoom level, the more detailed the traffic flow display becomes:

Figure 4: Zoom level 14 shows more detailed traffic flow overlay



Traffic Incident and Zoom Level

Mirroring its handling of traffic flow data, the HERE Traffic API does not return traffic incident information at zoom levels 0-7. A similar limitation applies to bounding box requests, with the maximum bounding box width and height restricted to 30 degrees of latitude and longitude. The largest permitted size of the proximity radius is 100km. The effect of these limits is to prevent unsustainably large response data transfers.

Queries that exceed the limits in terms of zoom level or proximity radius result in an error response.

Services

This section briefly defines the functional services offered by the HERE Traffic API. The services correspond to the resources listed under *Table 1: Basic request elements* on page 13.

Traffic Flow Data

You can use the `flow` resource of the HERE Traffic API to request traffic flow data containing speed and congestion ("jam factor") information for a geospatial area. The response data format is either XML or JSON. Optionally, your request can specify shape and functional class information for the roadways for which to obtain traffic flow information.

Geospatial Inclusion

The service supports the following geospatial filters (addressing schemes):

- quadkey
- [Z]/[X]/[Y]
- bounding box (`bbox`)
- proximity (`prox`)

For more information about these filters, please see *Request Format* on page 14.

Note that a response includes only those subcomponents of roadway items (`RW`) that fall within the geospatial address.

Content Filters

By default, a response contains all the available flow information for the specified spatial filter. It is, however, possible to add a further filter based on the functional class of the road through the

parameter *maxfunctionalclass*, and a filter for minimum and maximum jam factor values through the parameters *minjamfactor* and *maxjamfactor*.

Traffic Incident Data

The HERE Traffic API supports requests for traffic incident data from a specific geographic area (defined by a geospatial filter). The response includes data in XML or JSON format.

Geospatial Inclusion

Traffic incident items are included in the response based on their point of origin. This means that an incident in the response may potentially contain a geolocation outside the area indicated by the geospatial filter in the request. The figure below shows the `GEOLOC/ORIGIN` parameter used to determine if an item falls within the requested response boundaries.

Figure 5: Geospatial Inclusion

```
...
<GEOLOC>
        <ORIGIN>
                <LATITUDE>37.7894</LATITUDE>
                <LONGITUDE>-122.38723</LONGITUDE>
        </ORIGIN>
        <TO>
                <LATITUDE>37.8094693</LATITUDE>
                <LONGITUDE>-122.3657107</LONGITUDE>
        </TO>
</GEOLOC>
...
```

Incident Types and Filtering

Incident filters allow you to tailor incident requests to specific requirements. The filters support selection based on status, criticality, TMC table IDs, profiles, or start and end times.

In addition, traffic incidents are classified according to incident type:

- Accident
- Congestion
- Disabled vehicle
- Road hazard
- Construction
- Planned event
- Mass transit

- Other news
- Weather
- Miscellaneous

Localization Defaults

By default, incident reports returned in the response are localized to their geographic origin. That means, the time values are shown in local time and format, and message strings are in the default language for the country identified by the geospatial filter.

The table below shows the default languages for each supported country.

Table 6: Default Traffic Incident Reports Language

Country	Default Incident Reports Language
Austria	German (de)
Belgium	Dutch (nl)
Canada	English (en)
Denmark	Danish (da)
Finland	Finnish (fi)
France	French (fr)
Germany	German (de)
Greece	Greek (el)
Ireland	English (en)
Italy	Italian (it)
Luxembourg	French (fr)
Mexico	Spanish (es)
Netherlands	Dutch (nl)
Norway	Norwegian (no)
Poland	Polish (pl)
Portugal	Portuguese (pt)
South Africa	English (en)
Sweden	Swedish (sv)
Switzerland	German (de)
Turkey	Turkish (tr)
UK	English (en)

Country	Default Incident Reports Language
USA	English (en)

Supported Translation Languages

The supported languages for incident translation are:

- English (en)
- French (fr)
- Spanish (es)
- German (de)
- Italian (it)
- Swedish (sv)
- Danish (da)
- Norwegian (no)
- Finnish (fi)
- Dutch (nl)
- Slovak (sk)

Traffic Tiles

The Traffic API allows you to obtain a pre-rendered map tile reflecting traffic conditions. Each tile is a PNG overlay that uses line and color to indicate how freely traffic flows in the corresponding geographic area.

Deprecation Notice

HERE has changed the way traffic tiles are served. Live traffic tiles have been removed from the Traffic API and are now delivered by the Map Tile API infrastructure. Documentation relating to live traffic tiles is part of the Map Tile API documentation.

This does not affect traffic pattern tiles, which continue to be served by the Traffic API.

Color Mapping

By default, the service uses specific color mappings that match a traffic scale measured in Jam Factors, where 0 means "free flowing" and 10 means "completely obstructed". The following table describes the overlay colors, which follow a standard configuration to map Jam Factors to color.

Table 7: Traffic Flow to Color Mapping

Color	Description
Green	Free flow of traffic: 0 <= JAM_FACTOR < 4
Yellow	Sluggish flow of traffic: 4 <= JAM_FACTOR < 8
Red	Slow flow of traffic: 8 <= JAM_FACTOR < 10
Black	Traffic stopped flowing or road closed: JAM_FACTOR = 10

Note that customized profiles may be defined, where color mappings for traffic flow differ.

Caching

Traffic flow tiles can be cached for a limited period of time (up to three minutes), as the data must be fresh to represent the current traffic conditions reliably.

Examples

This section presents examples of common requests to the HERE Traffic API, as well as corresponding responses.

Note that all requests require the authentication credentials provided via the parameters `app_id` and `app_code` (see *Acquiring Credentials* on page 13). The example requests in the following articles show all the required parameters, but discuss only those specific to each request.

Traffic Flow Data

The examples listed below illustrate the common use cases for requesting traffic flow information from Traffic API in TrafficML Flow format version 3.1.

The following general request patterns are supported:

Table 8: Traffic flow data request formats

Addressing scheme	URL pattern
[Z]/[X]/[Y]	`{base-url}/{path}/{zoom}/{column}/{row}/{format}?`
Quadkey	`{base-url}/{path}/{resource}.{format}?quadkey={quadkey}`

Addressing scheme	URL pattern
Bounding box	`{base-url}/{path}/{resource}.{format}?bbox={bounding box coordinates}`
Proximity	`{base-url}/{path}/{resource}.{format}?prox={proximity coordinates}`

Requesting Traffic Flow Data

A basic scenario involving the use of the Traffic API is to obtain traffic flow data for a specific area. The two request examples below provide working illustrations.

The following general request patterns are supported:

All requests for traffic flow data must address the `flow` resource and indicate the area for which to obtain data. There are two ways to do the latter:

- [Z]/[X]/[Y] addressing scheme
- Quadkey

The example requests below reflect these two approaches.

Traffic Flow Data, [Z]/[X]/[Y] Addressing, XML Output

User Story

The user wants to obtain traffic flow data for an area around Frankfurt am Main, Germany. The area is to be identified using the [Z]/[X]/[Y] addressing scheme. The response must deliver data in XML.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*`flow`* [`flow`]
URL variables:	`{data_format}` [`xml`], response data format specification, here XML (to indicate that JSON is required, use `json`)
	`{zoom}/{column}/{row}` [`8/134/86`], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.1/flow/xml/8/134/86
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
```

To specify the tile for which to obtain traffic flow data, the request indicates the zoom level as 8, the tile column as 134, and row as 86.

Response

The response carries the following XML contents:

```
<?xml version="1.0" encoding="UTF-8"?>
    <TRAFFICML_REALTIME xmlns="http://traffic.nokia.com/trafficml-flow-3.1"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        MAP_VERSION="201301" TMC_TABLE_VERSION="11.0"
        CREATED_TIMESTAMP="2013-03-14T20:47:24" VERSION="y"
        UNITS="metric">
        <RWS TY="TMC" EBU_COUNTRY_CODE="D" EXTENDED_COUNTRY_CODE="E0" TABLE_ID="1"
 TMC_TABLE_VERSION="11.0">
            <RW LI="D01+07125" DE="A485" PBT="2013-03-14T20:47:10Z"
                mid="13ecd0f5f1f|23b6c310-0f15-4fd5-b76d-46382b027a6d">
                <FIS>
                    <FI>
                        <TMC PC="11530" DE="Gießener Südkreuz" QD="-" LE="2.35725"/>
                        <CF TY="TR" SP="65.79" FF="69.00" JF="0.58" CN="0.70"/>
                    </FI>
                    <!-- .... -->
                </FIS>
            </RW>
        </RWS>
        <diagnostic sfile="RealtimeFlowE0D01-1369239960627.xml.gz" pdd="c964101e-e61b-4a49-
be84-b954a00b2a4a"/>
    </TRAFFICML_REALTIME>
```

Traffic Flow Data, Quadkey Addressing, JSON Output

User Story

The user wants to obtain traffic flow data for an area around Frankfurt am Main, Germany. The area is to be identified using a quadkey. The response must deliver data in JSON.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*flow* [flow.json], the resource and resource extension indicating the required response data format, here JSON (to indicate XML, use flow.xml)
Parameters:	quadkey [quadkey=12020330], map tile location specified as a single value, see also *Quadkeys* on page 19

Request

A request matching the user story, is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.1/flow.json
?quadkey=12020330
&app_code=AJKnXv84fjrb0KIHawS0Tg
&app_id=DemoAppId01082013GAL
```

Response

The code block below shows the JSON response:


```
{
    "TRAFFICML_REALTIME": {
        "MAP_VERSION": "201301",
        "TMC_TABLE_VERSION": "11.0",
        "CREATED_TIMESTAMP": "2014-04-17T10:16:04",
        "VERSION": "y",
        "UNITS": "metric",
        "RWS": [
            {
                "RW": [
                    {
                        "FIS": [
                            {
                                "FI": [
                                    {
                                        "TMC": {
                                            "PC": 11530,
                                            "DE": "Gießener Südkreuz",
                                            "QD": "-",
                                            "LE": 2.35725
                                        },
                                        "CF": [
                                            {
                                                "TY": "TR",
                                                "SP": 65.79,
                                                "FF": 69,
                                                "JF": 0.58,
                                                "CN": 0.7
                                            }
                                        ]
                                    }
                                ]
                            }
                        ],
                        "LI": "D01+07125",
                        "DE": "A485",
                        "PBT": "2013-05-24T14:31:31Z",
                        "mid": "5ebf06d2-f671-47af-991b-33a57a76ac65|"
                    }
                ],
```

```
                "TY": "TMC",
                "EBUCOUNTRYCODE": "D",
                "EXTENDEDCOUNTRYCODE": "E0",
                "TABLEID": "1"
            }
        ]
    }
}
```

Requesting Traffic Flow Data with a Bounding Box Filter

The two examples below demonstrate requests for traffic flow data for an area defined by a bounding box. By default, all available traffic flow information is returned, but filters can be applied to the response, for example, to exclude flow information for roads matching a particular functional class.

Traffic flow is provided using TMC references.

Bounding Box Filter

User Story

The user wants to obtain traffic flow data for an area near Indianapolis, IN, USA. The area is a rectangle defined by the latitude and longitude of its top-left and bottom-right corners. The response must deliver data in XML.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*flow* [`flow.xml`], the resource and resource extension indicating the required response data format, here XML (to indicate JSON, use `flow.json`)
Parameters:	`bbox` [`bbox=39.8485715,-86.0969867;39.8358934,-86.0757964`], a rectangular area for which to obtain traffic data defined by the geographic coordinates of its top left and bottom-right corners

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.1/flow.xml
?app_id=DemoAppId01082013GAL
```

```
&app_code=AJKnXv84fjrb0KIHawS0Tg
&bbox=39.8485715,-86.0969867;39.8358934,-86.0757964
```

Response

Below is the response in XML format.

```
<?xml version="1.0" encoding="UTF-8"?>
<TRAFFICML_REALTIME xmlns="http://traffic.nokia.com/trafficml-flow-3.1"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" MAP_VERSION="lD.CkO7u"
    TMC_TABLE_VERSION="oiqDtKM4iE" CREATED_TIMESTAMP="2014-04-17T10:16:04" VERSION="y"
 UNITS="metric">
    <RWS TY="TMC" EBU_COUNTRY_CODE="1" EXTENDED_COUNTRY_CODE="A0" TABLE_ID="7"
 TMC_TABLE_VERSION="10.1">
        <RW LI="107-01073" DE="EMERSON AVE" PBT="2013-03-14T20:47:10Z"
            mid="0152c6dc-1359-4790-bbeb-41fed705275a|">
            <FIS>
                <FI>
                    <TMC PC="8354" DE="E 46TH ST" QD="+" LE="0.02346" FC="1"/>
                    <CF TY="TR" SP="21.70" FF="26.80" JF="2.82049" CN="0.70"/>
                </FI>
                <FI>
                    <TMC PC="8352" DE="E 38TH ST" QD="+" LE="0.52029"/>
                    <CF TY="TR" SP="31.52" FF="31.10" JF="0.0" CN="0.77"/>
                </FI>
                <FI>
                    <TMC PC="5655" DE="IL-122/EXIT 15" QD="-" LE="4.16074"/>
                    <CF TY="TR" SP="65.00" FF="65.00" JF="0.0" CN="0.85">
                        <SSS>
                            <SS LE="677.50603" SP="21.75" FF="12.02" JF="8.64613"/>
                            <SS LE="364.78040" SP="48.91" FF="78.88" JF="9.50912"/>
                            <SS LE="304.21572" SP="64.86" FF="60.19" JF="5.86003"/>
                        </SSS>
                    </CF>
                </FI>
            </FIS>
        </RW>
    </RWS>
    <diagnostic traceId="VAHZmq" sfile="MpL8Ncocv01J63huhmeMIUck82yWo"
 pdd="bZ6l5R677od6UXL5L5A-rd">
        <info>Yc.VdjLVkXPH.QXG</info>
    </diagnostic>
</TRAFFICML_REALTIME>
```

Bounding Box and Response Filters

User Story

The user wants to obtain traffic flow data for an area near Indianapolis, IN, USA. The area is a rectangle defined by the latitude and longitude of its top-left and bottom-right corners. The response must deliver data in XML, containing road shape and functional class information.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*flow* [flow.xml], the resource and resource extension indicating the required response data format, here XML (to indicate JSON, use flow.json)
Parameters:	bbox [bbox=39.8485715,-86.0969867;39.8358934,-86.0757964], a rectangular area for which to obtain traffic data defined by the geographic coordinates of its top-left and bottom-right corners responseattributes [responseattributes=sh,fc], a rectangular area for which to obtain traffic data defined by the geographic coordinates of its top-left and bottom-right corners

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.1/flow.xml
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&bbox=39.8485715,-86.0969867;39.8358934,-86.0757964
&responseattributes=sh,fc
```

Response

Shown below is an example response in XML format. Note the<SHP> element and the FC attribute which carry the shape and functional class information for the roadway corresponding to the flow items.

```
<![CDATA[<?xml version="1.0" encoding="UTF-8"?>
<TRAFFICML_REALTIME xmlns="http://traffic.nokia.com/trafficml-flow-3.1" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" MAP_VERSION="lD.CkO7u" TMC_TABLE_VERSION="oiqDtKM4iE"
 CREATED_TIMESTAMP="2014-04-17T10:16:04" VERSION="y" UNITS="metric">
    <RWS TY="TMC" EBU_COUNTRY_CODE="1" EXTENDED_COUNTRY_CODE="A0" TABLE_ID="7"
 TMC_TABLE_VERSION="10.1">
        <RW LI="107-01073" DE="EMERSON AVE" PBT="2013-03-14T20:47:10Z"
 mid="0152c6dc-1359-4790-bbeb-41fed705275a|">
            <FIS>
                <FI>
                    <TMC PC="8354" DE="E 46TH ST" QD="+" LE="0.02346" FC="1"/>
                    <SHP FC="3">39.8183517,-86.0836105 39.8184586,-86.0836105</SHP>
                    <SHP FC="3">39.8184586,-86.0836105 39.8188705,-86.0836105</SHP>
                    <CF TY="TR" SP="21.70" FF="26.80" JF="2.82049" CN="0.70"/>
                </FI>
                <FI>
                    <TMC PC="8352" DE="E 38TH ST" QD="+" LE="0.52029"/>
```

```
                    <SHP FC="4">39.8330688,-86.0837326 39.8342285,-86.0837173</SHP>
                    <SHP FC="4">39.8342285,-86.0837173 39.835289,-86.0837402
 39.8359108,-86.0837479 39.8363304,-86.0837402</SHP>
                    <CF TY="TR" SP="31.52" FF="31.10" JF="0.0" CN="0.77"/>
                </FI>
                <FI>
                    <TMC PC="5655" DE="IL-122/EXIT 15" QD="-" LE="4.16074"/>
                    <SHP FC="1">40.3239594,-89.4540329 40.3242989,-89.4540405
 40.3274689,-89.4541321 40.3372917,-89.454361 40.3400307,-89.454422 40.3411407,-89.4544601
 40.3460007,-89.454567 40.3483315,-89.454628 40.3495216,-89.4546509 40.3530693,-89.4547424
 40.3542404,-89.4547729 40.3577309,-89.4548492 40.3591003,-89.4548874 40.3633614,-89.4549866
 40.3664207,-89.4550705 40.3713608,-89.4551773</SHP>
                    <CF TY="TR" SP="65.00" FF="65.00" JF="0.0" CN="0.85">
                        <SSS>
                            <SS LE="677.50603" SP="21.75" FF="12.02" JF="8.64613"/>
                            <SS LE="364.78040" SP="48.91" FF="78.88" JF="9.50912"/>
                            <SS LE="304.21572" SP="64.86" FF="60.19" JF="5.86003"/>
                        </SSS>
                    </CF>
                </FI>
            </FIS>
        </RW>
    </RWS>
    <diagnostic traceId="VAHZmq" sfile="MpL8Ncocv01J63huhmeMIUck82yWo"
 pdd="bZ6l5R677od6UXL5L5A-rd">
        <info>Yc.VdjLVkXPH.QXG</info>
    </diagnostic>
</TRAFFICML_REALTIME>]]>
```

Requesting Traffic Flow Data with Max Functional Class Filter

This article illustrates use cases in which requests for traffic flow data use a filter to limit the amount of detail (and therefore the size of the response). The volume of traffic flow data can be very large, especially at high map zoom levels.

Max Functional Class, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain traffic flow data for an area around Frankfurt am Main, Germany. The area is to be identified using the [Z]/[X]/[Y] addressing scheme, with map zoom level 9. The response must deliver data in XML, filtering the results to include functional class road types no higher than 2.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*flow* [flow]

URL variables:	`{data_format}` [`xml`], response data format specification, here XML (to indicate that JSON is required, use `json`) `{zoom}/{column}/{row}` [`9/268/173`], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
Parameters:	`maxfunctionalclass` [`maxfunctionalclass=2`], maximum functional class for items included in the response (items with a higher functional class are filtered out) – see also *FunctionalClassType* on page 78

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.1/flow/xml/9/268/173
?app_code=AJKnXv84fjrb0KIHawS0Tg
&app_id=DemoAppId01082013GAL
&maxfunctionalclass=2
```

Response

The following code block shows the response the request for traffic flow data with a maximum functional code filter. Note that this response is abbreviated because of space limitations. In reality, the XML may contain a very large number of <RW> elements.

```
<![CDATA[<?xml version="1.0" encoding="UTF-8"?>
<TRAFFICML_REALTIME MAP_VERSION="" CREATED_TIMESTAMP="2013-10-14T16:19:08Z" VERSION="3.1"
 UNITS="imperial">
    <RWS TY="TMC" EBU_COUNTRY_CODE="D" EXTENDED_COUNTRY_CODE="E0" TABLE_ID="1">
        <RW LI="D01+07176" DE="A648" PBT="2013-10-14T16:19:08+0000" mid="141b7c32502|
5f144646-29dd-40ee-a626-94e1b9184b4b">
            <FIS>
                <FI>
                    <TMC PC="12103" DE="Eschborner Dreieck" QD="-" LE="0.75953"/>
                    <CF CN="0.89" FF="59.7" JF="0.0" SP="64.0" TY="TR"/>
                </FI>
                <FI>
                    <TMC PC="12104" DE="Frankfurt am Main-Rödelheim" QD="-" LE="0.46107"/>
                    <CF CN="0.89" FF="66.5" JF="0.59" SP="63.35" TY="TR"/>
                </FI>
                <FI>
                    <TMC PC="12105" DE="Westkreuz Frankfurt am Main" QD="-" LE="1.20252"/>
                    <CF CN="0.89" FF="63.4" JF="0.64" SP="60.16" TY="TR"/>
                </FI>
            </FIS>
        </RW>
        <RW LI="D01+07184" DE="A661" PBT="2013-10-14T16:19:08+0000" mid="141b7c32e76|
96d6710e-a38c-49d0-a0dd-ffc164ae9cf6"></RW>
        ...
    </RWS>
</TRAFFICML_REALTIME>]]>
```

Max Functional Class, Quadkey Addressing

A quadkey version of the previous request is shown below. Further alternatives are to use a proximity (`prox`) or a bounding box filter (`bbox`) to replace the `quadkey` string parameter / value.

User Story

The user wants to obtain traffic flow data for an area around Frankfurt am Main, Germany. The area is to be identified using a quadkey. The response must deliver data in XML, filtering the results to include functional class road types no higher than 2.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*`flow`* [`flow.xml`], the resource and resource extension indicating the required response data format, here XML (to indicate JSON, use `flow.json`)
Parameters:	`quadkey` [`quadkey=12020330`], map tile location specified as a single value, see also *Quadkeys* on page 19
	`maxfunctionalclass` [`maxfunctionalclass=2`], maximum functional class for items included in the response (items with a higher functional class are filtered out) – see also *FunctionalClassType* on page 78

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.1/flow.xml
?quadkey=12020330
&app_code=AJKnXv84fjrb0KIHawS0Tg
&app_id=DemoAppId01082013GAL
&maxfunctionalclass=2
```

Response

The result of this request is similar to that shown under *Response* on page 35.

Traffic Tiles

This section offers examples of requests for traffic tile overlays from the Traffic API.

Deprecation Notice

HERE has changed the way traffic tiles are served. Live traffic tiles have been removed from the Traffic API and are now delivered by the Map Tile API infrastructure. Documentation relating to live traffic tiles is part of the Map Tile API documentation.

This does not affect traffic pattern tiles, which continue to be served by the Traffic API.

The following general request patterns are supported:

Table 9: Traffic tile request patterns

Addressing scheme	URL pattern
[Z]/[X]/[Y]	`{tile-base-url}/{path}/{resource}/{zoom}/{column}/{row}/{size}/{format}?`
Quadkey	`{tile-base-url}/{path}/{resource}/quadkeytraffic?quadkey={quadkey}`

Requesting Traffic Flow Tile Overlays

One of the use cases for the HERE Traffic API is to obtain a traffic flow map tile overlay that can be used to provide a visual indication of the traffic conditions on the map.

Flow Overlays, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain a traffic flow tile overlay for an area near Frankfurt, Germany, using the [Z]/[X]/[Y] addressing scheme.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]

URL variables:	`{zoom}/{column}/{row}` [8/134/86], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
	`{size}` [256], the resolution of the response image
	`{format}` [png32], the format of the response image

Request

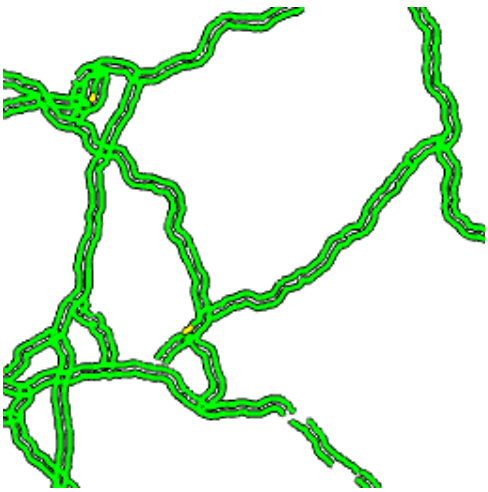
A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/8/134/86/256/png32
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
```

Response

The response to the requests for a tile overlay for an area near Frankfurt in Germany contains the following PNG traffic flow tile overlay image:

Figure 6: Traffic flow tile



This is a basic request to the API for a traffic flow tile and the response. For further examples, please see *Traffic Tiles* on page 37.

Flow Overlays, Quadkey Addressing

User Story

The user wants to obtain a traffic flow tile overlay for an area near Frankfurt, Germany, using the quadkey addressing scheme.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{quadkeytraffic} [quadkeytraffic], resource endpoint, indicating that a traffic tile overlay based identified by a quadkey is required
Parameters:	quadkey [quadkey=12020330],], map tile location specified as a single value, see also *Quadkeys* on page 19

Request

A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/quadkeytraffic
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=12020330
```

Response

The result is the same image as shown for the [Z]/[X]/[Y] version of the request under *Response* on page 38.

Requesting a Traffic Pattern Tile

The API supports requests for a tile overlay showing the typical traffic pattern for a specific time point during the week. The time point is specified as a number of seconds from the beginning of the week (00:00 hours, Sunday, in local time).

Traffic Pattern, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain a map tile showing a traffic pattern, 5:40 PM, using the [Z]/[X]/[Y] addressing scheme:

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{zoom}/{column}/{row} [8/134/86], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
	size [256], the resolution of the response image
	{format} [png32], the format of the response image
Parameters:	pattern_time [pattern_time=150000], a value indicating the time point for which to obtain the traffic pattern overlay as a number of seconds from 00:00 on Sunday, in local time)

Request

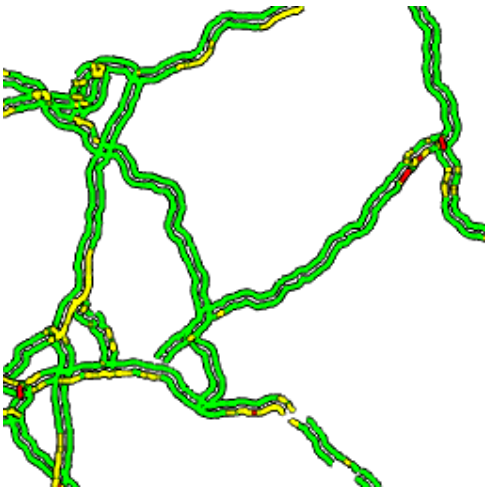
A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/8/134/86/256/png32
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&pattern_time=150000
```

Response

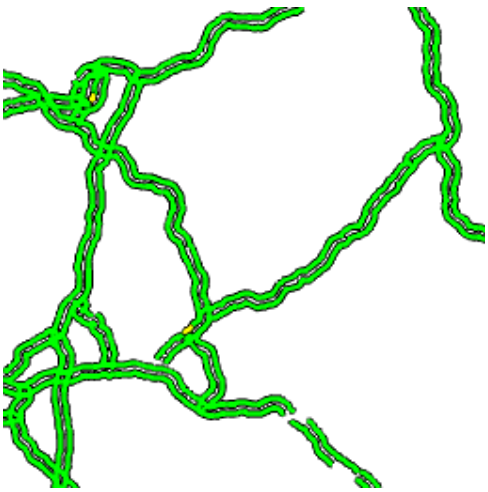
The response to the requests for a traffic pattern tile overlay delivers the following PNG image:

Figure 7: Traffic pattern tile



Note that if you remove the parameter `pattern_time`, the returned map tile is identical to the one in the flow tile example (see also *Figure 6: Traffic flow tile* on page 38):

Figure 8: Flow tile



Traffic Pattern, Quadkey Addressing

The same traffic tile request is shown below, but this time it uses the quadkey addressing scheme:

User Story

The user wants to obtain a map tile showing a traffic pattern, 5:40 PM, using the quadkey addressing scheme:

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{quadkeytraffic} [quadkeytraffic], resource endpoint, indicating that a traffic tile overlay based identified by a quadkey is required
Parameters:	quadkey [quadkey=12020330],], map tile location specified as a single value, see also *Quadkeys* on page 19 pattern_time [pattern_time=150000], a value indicating the time point for which to obtain the traffic pattern overlay as a number of seconds from 00:00 on Sunday, in local time)

Request

A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/quadkeytraffic
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=12020330
&pattern_time=150000
```

Response

The result is an image as shown for the quadkey version of the request under *Response* on page 41

Selecting Tile Size

The Traffic API allows requests to select the size of the size of the tile to be delivered in the response. Such requests follow the same pattern as requests for *traffic flow tile overlays*, but require the parameter res to select the tile size and/or depth to determine the required color depth.

Tile Size, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain a traffic tile for an area near Frankfurt, Germany, using the [Z]/[X]/[Y] addressing scheme. The response is to deliver a tile measuring 512x512 pixels (Other possibilities are: 256x256 pixels and 128x128 pixels).

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{zoom}/{column}/{row} [8/134/86], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
	{size} [512], the resolution of the response image
	{format} [png32], the format of the response image

Request

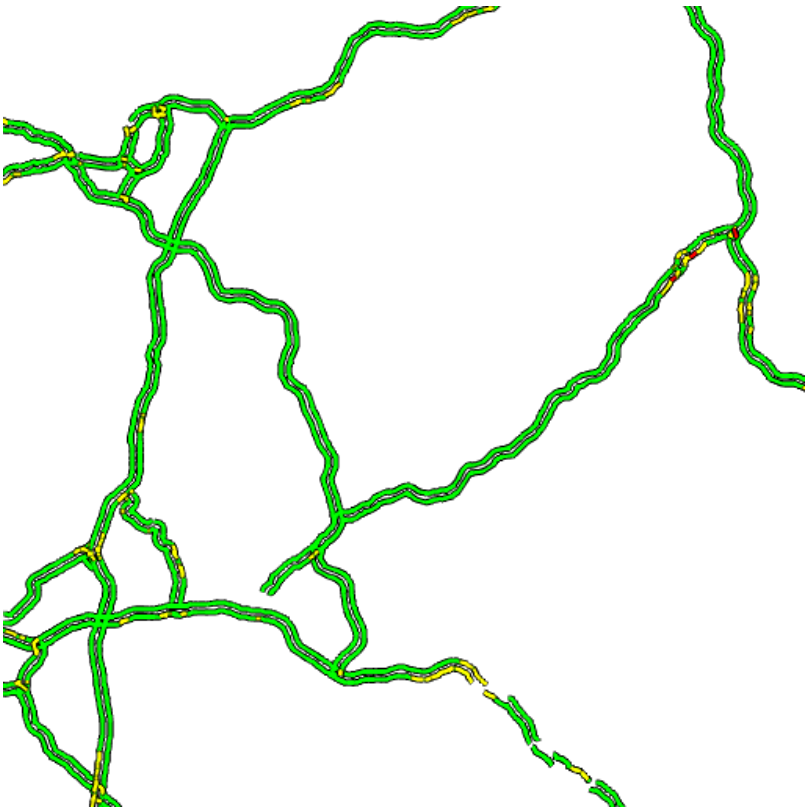
A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/8/134/86/512/png32
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
```

Response

The image below shows the tile delivered in response to the request.

Figure 9: Resized Large Flow Tile Request



Tile Size, Quadkey Addressing

User Story

The user wants to obtain a traffic tile for an area near Frankfurt, Germany, using a quadkey. The response is to deliver a tile measuring 128x128 pixels. (The default is 256x256 pixels and the third possibility is 256x256 pixels.)

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Note that quadkey addressing makes it necessary to use `res` as a query parameter.

Resource:	*`tiles`* [tiles]
URL variables:	`{quadkeytraffic}` [quadkeytraffic], resource endpoint, indicating that a traffic tile overlay based identified by a quadkey is required
Parameters:	`quadkey` [quadkey=1202030322223],], map tile location specified as a single value, see also *Quadkeys* on page 19
	`res` [res=bitmap128], an indication that a square bitmap with a side measuring 128 pixels is required

Request

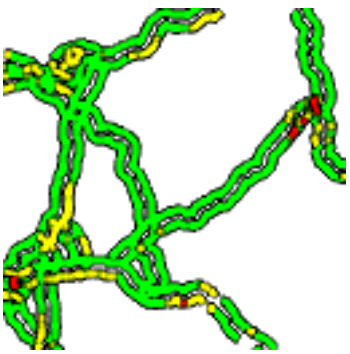
A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/quadkeytraffic
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=12020330
&res=bitmap128
```

Response

Here is the response image tile measuring 128x128 pixels:

Figure 10: Resized Small Flow Tile Request



Selecting Tile Color Depth

The Traffic API allows requests to select the size of the color depth of the tile to be delivered in the response. Such requests follow the same pattern as requests for *traffic flow tile overlays*, but require the parameter `res` to select the tile size and/or `depth` to determine the required color depth.

Color Depth, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain a traffic tile, using the [Z]/[X]/[Y] addressing scheme. The response is to deliver a tile with color depth of 8 bits

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{zoom}/{column}/{row} [8/134/86], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
	{size} [256], the resolution of the response image
	{format} [png8], the format of the response image, which specifies the required color depth

Request

A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/8/134/86/256/png8
?app_code=AJKnXv84fjrb0KIHawS0Tg&app_id=DemoAppId01082013GAL
```

Response

The response to both the requests that stipulate tile color depth delivers the tile shown below.

Figure 11: Flow Tile Request as 8-bit

Color Depth, Quadkey Addressing

User Story

The user wants to obtain a traffic tile, using a quadkey. The color depth of the tile delivered by the response is to be 8 bits.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Note that quadkey addressing makes it necessary to specify the color depth via the query parameter `depth`.

Resource:	`tiles` [`tiles`]
URL variables:	`{quadkeytraffic}` [quadkeytraffic], resource endpoint, indicating that a traffic tile overlay based identified by a quadkey is required
Parameters:	`quadkey` [`quadkey=12020330`],], map tile location specified as a single value, see also *Quadkeys* on page 19
	`depth` [`depth=8bit`], an indication color depth

Request

A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/quadkeytraffic
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=12020330
&depth=8bit
```

Response

The result is an image as shown for the [Z]/[X]/[Y] version of the request under *Response* on page 47.

Filtering by TMC Tables

A customer profile includes a set of TMC tables, which specify the traffic data that you can request from the service. Requests that use TMC tables follow the same pattern as requests for *traffic flow tile overlays*, but differ in terms of parameters.

For a list of sample TMC codes, see *Table 23: TMC Table IDs* on page 79.

You can include or exclude certain tables in your profile via the `tables` parameter in the request, paired with the TMC code (see also *TMCCodeType* on page 78). Positive values for the TMC code specify which tables are to be included, and negative values specify tables to exclude. You cannot combine a positive and negative value in a request: such a request is not support an results in an error.

TMC Table filtering

The examples in this article demonstrate how to filter by a TMC table in a request.

TMC Table filtering, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain a traffic tile for an area near Boston, Massachusetts, on the border of TMC table A0129. The request is to use the [Z]/[X]/[Y] addressing scheme and filter by the TMC table.:

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{zoom}/{column}/{row} [11/613/759/256], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
	{size} [256], the resolution of the response image
	{format} [png32], the format of the response image, which specifies the required color depth
Parameters:	tables [tables=A0129], specifies the TMC tables to use

Request

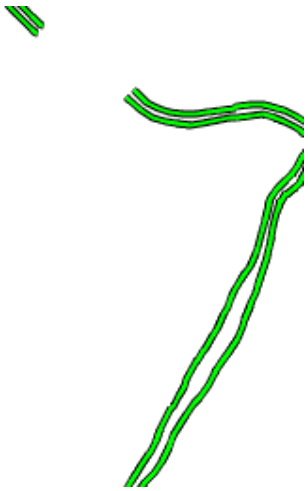
A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/11/613/759/256/png32
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&tables=A0129
```

Response

The response to both versions of the request above return the following PNG traffic flow tile overlay:

Figure 12: Request Including Table A0129



Including a TMC Table, Quadkey Addressing

User Story

The user wants to obtain a traffic tile for an area near Boston, Massachusetts, on the border of TMC table A0129. The request is to use quadkey addressing scheme and filter by the TMC table.:

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*tiles* [tiles]
URL variables:	{quadkeytraffic} [quadkeytraffic], resource endpoint, indicating that a traffic tile overlay based identified by a quadkey is required
Parameters:	quadkey [quadkey=03023320323], map tile location specified as a single value, see also *Quadkeys* on page 19
	tables [tables=A0129], a list of tables to use (in this case one table)

Request

A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/quadkeytraffic
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=03023320323
&tables=A0129
```

Response

The result is an image as shown for the [Z]/[X]/[Y] version of the request under *Response* on page 50.

Excluding a TMC Table

The examples in this article demonstrate how to exclude a TMC table in a request.

Excluding a TMC Table, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain a traffic tile for an area near Boston, Massachusetts, on the border of TMC table A0129. The request is to use the [Z]/[X]/[Y] addressing scheme and exclude the TMC table.:

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*`tiles`* [`tiles`]
URL variables:	`{zoom}/{column}/{row}` [`11/613/759/256`], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
	`{size}` [256], the resolution of the response image
	`{format}` [png32], the format of the response image, which specifies the required color depth
Parameters:	`tables` [tables=-A0129], specifies the TMC tables to exclude

Request

A request matching the user story is formulated as follows:

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/11/613/759/256/png32
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&tables=-A0129
```

Response

Both versions of the request above return this PNG traffic flow tile overlay:

Figure 13: Request Excluding Table A0129

Excluding a TMC Table, Quadkey Addressing

User Story

The user wants to obtain a traffic tile for an area near Boston, Massachusetts, on the border of TMC table A0129. The request is to use quadkey addressing scheme and exclude the TMC table.:

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	`tiles` [`tiles`]

URL variables:	`{quadkeytraffic}` [quadkeytraffic], resource endpoint, indicating that a traffic tile overlay based identified by a quadkey is required
Parameters:	`quadkey` [`quadkey=03023320323`], map tile location specified as a single value, see also *Quadkeys* on page 19
	`tables` [`tables=-A0129`], a list of tables to exclude (in this case one table)

Request

A request matching the user story is formulated as follows:

The code block below shows the same request, but with a quadkey instead of the [Z]/[X]/[Y] addressing scheme.

```
http://maps.st.nlp.nokia.com/traffic/6.0/tiles/quadkeytraffic
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=03023320323
&tables=-A0129
```

Response

The result is an image as shown for the [Z]/[X]/[Y] version of the request under *Response* on page 52.

Traffic Incident Data

The basic incidents data request is described under *Quick Start* on page 9. The following list contains links to examples that illustrate further common use cases for requesting traffic incident data from the Traffic API.

- *Requesting Localized Incident Data* on page 54
- *Request a Subset of Incident Data* on page 57

The following general request patterns are supported:

Table 10: Traffic tile request patterns

Addressing scheme	URL pattern
[Z]/[X]/[Y]	`{tile-base-url}/{path}/{resource}/{zoom}/{column}/{row}/{size}/{format}?`
Quadkey	`{tile-base-url}/{path}/{resource}/quadkeytraffic?quadkey={quadkey}`

Requesting Localized Incident Data

A request to the HERE Traffic API can localize the response to a specific time zone, format, or language, instead of relying on the defaults. Note, however, that localized incident descriptions (translations) are available only for the supported languages – for the full list, please see *Localization Defaults* on page 25.

Incident Data, [Z]/[X]/[Y] Addressing

User Story

The user wants to obtain traffic incident information for a tile at zoom level 8 representing an area near Frankfurt, Germany. The response is to be localized to the USA and the text it includes translated into English (en). Note that the default incident time zone, date format and language in the response are: timezone - UTC/GMT, date format - DIN ISO 8601:2006-09, and language - German (de). The request is to use the [Z]/[X]/[Y] addressing scheme.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*`incidents`* [`incidents`]
URL variables:	`{data_format}` [`xml`], response data format specification, here XML (to indicate that JSON is required, use `json`)
	`{zoom}/{column}/{row}` [`8/134/86`], the [Z]/[X]/[Y] (tile) addressing scheme, see also *Mercator Projection and Map Tiles* on page 17
Parameters:	`c` [`c=US`], country to which to localize
	`lg` [`lg=en`], localization language
	`i18n` [`i18n=true`], a flag indicating if incident messages are to be translated into the localization language
	`local_time` [`localtime=true`], a flag indicating whether to use local time

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/incidents/xml/8/134/86
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&c=US
&lg=en
&i18n=true
&localtime=true
```

Response

The following code block shows the response to the request for localized incident data. Note that this response is abbreviated because of space limitations. In reality, the XML may contain a very large number of <TRAFFIC_ITEM> elements.

```
<![CDATA[<?xml version="1.0" encoding="UTF-8"?>
<TRAFFICML_INCIDENTS TIMESTAMP="10/14/2013 14:37:06 GMT" VERSION="5.0">
    <TRAFFIC_ITEMS>
        <TRAFFIC_ITEM>
            <TRAFFIC_ITEM_ID>420872348</TRAFFIC_ITEM_ID>
            <ORIGINAL_TRAFFIC_ITEM_ID>420409397</ORIGINAL_TRAFFIC_ITEM_ID>
            <TRAFFIC_ITEM_STATUS_SHORT_DESC>ACTIVE</TRAFFIC_ITEM_STATUS_SHORT_DESC>
            <TRAFFIC_ITEM_TYPE_DESC>OTHER NEWS</TRAFFIC_ITEM_TYPE_DESC>
            <START_TIME>July 8, 2013 7:20:10 PM CEST</START_TIME>
            <END_TIME>October 31, 2013 1:00:00 PM CET</END_TIME>
            <ENTRY_TIME>July 10, 2013 7:23:21 AM CEST</ENTRY_TIME>
            <CRITICALITY>
                <ID>2</ID>
                <DESCRIPTION>minor</DESCRIPTION>
            </CRITICALITY>
            <VERIFIED>true</VERIFIED>
            <COMMENTS> </COMMENTS>
            <RDS-TMC_LOCATIONS>
                <RDS-TMC>
                    <ORIGIN>
                        <EBU_COUNTRY_CODE>D</EBU_COUNTRY_CODE>
                        <TABLE_ID>1</TABLE_ID>
                        <LOCATION_ID>23982</LOCATION_ID>
                        <LOCATION_DESC>Wetzlar-Ost</LOCATION_DESC>
                        <RDS_DIRECTION>+</RDS_DIRECTION>
                    </ORIGIN>
                    <DIRECTION>-</DIRECTION>
                    <ALERTC>
                        <TRAFFIC_CODE>485</TRAFFIC_CODE>
                        <DESCRIPTION>connecting carriageway blocked</DESCRIPTION>
                        <ALERTC_DURATION>D</ALERTC_DURATION>
                        <ALERTC_DIRECTION>1</ALERTC_DIRECTION>
                        <URGENCY>U</URGENCY>
                        <UPDATE_CLASS>7</UPDATE_CLASS>
                        <PHRASE_CODE>C24</PHRASE_CODE>
                        <EXTENT>0</EXTENT>
                        <DURATION>0</DURATION>
                    </ALERTC>
                </RDS-TMC>
            </RDS-TMC_LOCATIONS>
            <LOCATION>
                <DEFINED>
                    <ORIGIN>
```

```
                        <ROADWAY ID="50396"></ROADWAY>
                        <POINT ID="23982"></POINT>
                        <DIRECTION ID="0"></DIRECTION>
                        <PROXIMITY></PROXIMITY>
                    </ORIGIN>
                    <TO>
                        <ROADWAY ID="50396"></ROADWAY>
                        <POINT ID="23982"></POINT>
                        <DIRECTION ID="0"></DIRECTION>
                        <PROXIMITY></PROXIMITY>
                    </TO>
                </DEFINED>
                <GEOLOC></GEOLOC>
                <NAVTECH></NAVTECH>
            </LOCATION>
            <TRAFFIC_ITEM_DETAIL>
                <ROAD_CLOSED>false</ROAD_CLOSED>
                <NEWS></NEWS>
            </TRAFFIC_ITEM_DETAIL>
            <TRAFFIC_ITEM_DESCRIPTION TYPE="short_desc">
                between Wetzlar-Ost and Wetzlar-Ost, connecting carriageway blocked
            </TRAFFIC_ITEM_DESCRIPTION>
            <TRAFFIC_ITEM_DESCRIPTION TYPE="desc">
                between Wetzlar-Ost and Wetzlar-Ost, connecting carriageway blocked
            </TRAFFIC_ITEM_DESCRIPTION>
            <TRAFFIC_ITEM_DESCRIPTION TYPE="no_exit_description">
                connecting carriageway blocked
            </TRAFFIC_ITEM_DESCRIPTION>
        </TRAFFIC_ITEM>
    </TRAFFIC_ITEMS>
</TRAFFICML_INCIDENTS>]]>
```

Incident Data, Quadkey Addressing

A quadkey version of the same request for traffic incident data is shown below. Further possibilities are to use proximity (`prox`), bounding box (`bbox`), or corridor (`corridor`) in place of the `quadkey` string parameter-value pair.

User Story

The user wants to obtain traffic incident information for a tile representing an area near Frankfurt, Germany. The response is to be localized to the USA and the text it includes translated into English (en). Note that the default incident time zone, date format and language in the response are: timezone - UTC/GMT, date format - DIN ISO 8601:2006-09, and language - German (de). The request is to use quadkey addressing.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*incidents* [incidents.xml], the resource and resource extension indicating the required response data format, here XML (to indicate JSON, use incidents.json)
Parameters:	quadkey [quadkey=03023320323], map tile location specified as a single value, see also *Quadkeys* on page 19 c [c=US], country to which to localize lg [lg=en], localization language i18n [i18n=true], a flag indicating if incident messages are to be translated into the localization language local_time [localtime=true], a flag indicating whether to use local time

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/incidents.xml
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&quadkey=12020330
&c=US
&lg=en
&i18n=true
&localtime=true
```

Response

The result of this request is similar to that shown under *Response* on page 55.

Request a Subset of Incident Data

Requests to the Traffic API for incident data can limit the response to a data subset.

Common subsets include:

- limiting the total number of responses
- limiting responses to a certain type, status, criticality, TMC table, profile or timeframe
- excluding certain geospatial areas

Incident Data Subset, Bounding Box Area Filter

User Story

The user wants to obtain traffic incident information that uses the parameters to limit the response to 50 active incidents near Frankfurt am Main, Germany, but outside a defined bounding box area (negBBox). (Further possibilities are to use proximity (negProx) or corridor (negCorridor) instead of the bounding box (negBBox).) The request is to use the bounding box area filter. (Further possibilities are to use proximity (prox) or corridor (corridor) instead of the bounding box (bbox).)

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*incidents* [incidents]
URL variables:	{data_format} [xml], response data format specification, here XML (to indicate that JSON is required, use json)
	bbox[bbox=48.8379,8.4375;50.7364,9.84375], a rectangular area for which to obtain traffic data defined by the geographic coordinates of its top left and bottom-right corners
Parameters:	negBBox [negBBox=49.8379,8.4375;50.7364,9.84375], defines a bounding box for an area to exclude from the results
	status [status=active], indicates that results are to be filtered according to status, the example requests only "active" incidents
	maxresults [maxresults=50], the maximum number of incidents for which to retrieve information

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/incidents.xml?
bbox=48.8379,8.4375;50.7364,9.84375
&app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&negBBox=49.8379,8.4375;50.7364,9.84375
```

```
&status=active
&maxresults=50
```

Response

The following code block shows the response to the request for subset of incident data. Note that this response is abbreviated because of space limitations. In reality, the XML may contain a very large number of <TRAFFIC_ITEM> elements.

```
<![CDATA[<?xml version="1.0" encoding="UTF-8"?>
<TRAFFICML_INCIDENTS TIMESTAMP="10/14/2013 17:23:06 GMT" VERSION="5.0">
    <TRAFFIC_ITEMS>
        <TRAFFIC_ITEM>
            <TRAFFIC_ITEM_ID>445066184</TRAFFIC_ITEM_ID>
            <ORIGINAL_TRAFFIC_ITEM_ID>445066184</ORIGINAL_TRAFFIC_ITEM_ID>
            <TRAFFIC_ITEM_STATUS_SHORT_DESC>ACTIVE</TRAFFIC_ITEM_STATUS_SHORT_DESC>
            <TRAFFIC_ITEM_TYPE_DESC>CONSTRUCTION</TRAFFIC_ITEM_TYPE_DESC>
            <START_TIME>10/06/2013 22:00:00</START_TIME>
            <END_TIME>10/16/2013 12:00:00</END_TIME>
            <ENTRY_TIME>10/06/2013 20:25:34</ENTRY_TIME>
            <CRITICALITY>
                <ID>2</ID>
                <DESCRIPTION>minor</DESCRIPTION>
            </CRITICALITY>
            <VERIFIED>true</VERIFIED>
            <COMMENTS></COMMENTS>
            <RDS-TMC_LOCATIONS>
                <RDS-TMC>
                    <ORIGIN>
                        <EBU_COUNTRY_CODE>D</EBU_COUNTRY_CODE>
                        <TABLE_ID>1</TABLE_ID>
                        <LOCATION_ID>36612</LOCATION_ID>
                        <LOCATION_DESC>Feldatal-Ermenrod</LOCATION_DESC>
                        <RDS_DIRECTION>-</RDS_DIRECTION>
                    </ORIGIN>
                    <DIRECTION>+</DIRECTION>
                    <ALERTC>
                        <TRAFFIC_CODE>701</TRAFFIC_CODE>
                        <QUANTIFIERS>0</QUANTIFIERS>
                        <DESCRIPTION>(Q)(Baustelle//Baustellen)</DESCRIPTION>
                        <ALERTC_DURATION>L</ALERTC_DURATION>
                        <ALERTC_DIRECTION>1</ALERTC_DIRECTION>
                        <UPDATE_CLASS>11</UPDATE_CLASS>
                        <PHRASE_CODE>E1</PHRASE_CODE>
                        <EXTENT>0</EXTENT>
                        <DURATION>0</DURATION>
                    </ALERTC>
                </RDS-TMC>
            </RDS-TMC_LOCATIONS>
            <LOCATION>
                <DEFINED>
                    <ORIGIN>
                        <ROADWAY ID="50396">
                            <DESCRIPTION TYPE="NTCSA">B49</DESCRIPTION>
                            <DESCRIPTION TYPE="RDSGN">B49</DESCRIPTION>
                            <DESCRIPTION TYPE="LOCAL">B49</DESCRIPTION>
                        </ROADWAY>
                        <POINT ID="36612">
                            <DESCRIPTION TYPE="NTCSA">Feldatal-Ermenrod</DESCRIPTION>
                            <DESCRIPTION TYPE="RDSGN">Feldatal-Ermenrod</DESCRIPTION>
                            <DESCRIPTION TYPE="LOCAL">Feldatal-Ermenrod</DESCRIPTION>
                            <DESCRIPTION TYPE="BNAME">B49</DESCRIPTION>
                        </POINT>
                        <DIRECTION ID="0">
```

```
                        <DESCRIPTION TYPE="LOCAL">ALSFELD</DESCRIPTION>
                        <DESCRIPTION TYPE="NTCSA">ALSFELD</DESCRIPTION>
                        <DESCRIPTION TYPE="RDSGN">ALSFELD</DESCRIPTION>
                    </DIRECTION>
                    <PROXIMITY>
                        <ID>At</ID>
                        <DESCRIPTION>at</DESCRIPTION>
                    </PROXIMITY>
                </ORIGIN>
                <TO>
                    <ROADWAY ID="50396">
                        <DESCRIPTION TYPE="NTCSA">B49</DESCRIPTION>
                        <DESCRIPTION TYPE="RDSGN">B49</DESCRIPTION>
                        <DESCRIPTION TYPE="LOCAL">B49</DESCRIPTION>
                    </ROADWAY>
                    <POINT ID="36612">
                        <DESCRIPTION TYPE="NTCSA">Feldatal-Ermenrod</DESCRIPTION>
                        <DESCRIPTION TYPE="RDSGN">Feldatal-Ermenrod</DESCRIPTION>
                        <DESCRIPTION TYPE="LOCAL">Feldatal-Ermenrod</DESCRIPTION>
                        <DESCRIPTION TYPE="BNAME">B49</DESCRIPTION>
                    </POINT>
                    <DIRECTION ID="0">
                        <DESCRIPTION TYPE="LOCAL">ALSFELD</DESCRIPTION>
                        <DESCRIPTION TYPE="NTCSA">ALSFELD</DESCRIPTION>
                        <DESCRIPTION TYPE="RDSGN">ALSFELD</DESCRIPTION>
                    </DIRECTION>
                    <PROXIMITY>
                        <ID>At</ID>
                        <DESCRIPTION>at</DESCRIPTION>
                    </PROXIMITY>
                </TO>
            </DEFINED>
            <GEOLOC>
                <ORIGIN>
                    <LATITUDE>50.6528</LATITUDE>
                    <LONGITUDE>9.1382</LONGITUDE>
                </ORIGIN>
                <TO>
                    <LATITUDE>50.6528</LATITUDE>
                    <LONGITUDE>9.1382</LONGITUDE>
                </TO>
            </GEOLOC>
            <NAVTECH>
                <EDGE>
                    <EDGE_ID>61778142</EDGE_ID>
                    <EDGE_ID>61778159</EDGE_ID>
                    ...
                </EDGE>
                <VERSION_ID>201301</VERSION_ID>
            </NAVTECH>
        </LOCATION>
        <TRAFFIC_ITEM_DETAIL>
            <ROAD_CLOSED>false</ROAD_CLOSED>
                <EVENT>
                    <EVENT_ITEM_CANCELLED>false</EVENT_ITEM_CANCELLED>
                    <SCHEDULED_CONSTRUCTION_EVENT>
                        <SCHEDULED_CONSTRUCTION_TYPE_DESC>
                            CONSTRUCTION
                        </SCHEDULED_CONSTRUCTION_TYPE_DESC>
                        <SCHEDULED_CONSTRUCTION_DETAIL>
                            construction
                        </SCHEDULED_CONSTRUCTION_DETAIL>
                    </SCHEDULED_CONSTRUCTION_EVENT>
                </EVENT>
        </TRAFFIC_ITEM_DETAIL>
        <TRAFFIC_ITEM_DESCRIPTION TYPE="short_desc">
            bei Feldatal-Ermenrod - Baustelle.
        </TRAFFIC_ITEM_DESCRIPTION>
        ...
    </TRAFFIC_ITEM><TRAFFIC_ITEM>
```

```
    </TRAFFIC_ITEM>
</TRAFFIC_ITEMS></TRAFFICML_INCIDENTS>]]>
```

Traffic Flow Availability Data

This section offers examples of requests for traffic flow availability data.

Request Flow Availability

Flow availability requests allow you to see what traffic flow coverage exists in the current (version 6.0.28.1) Traffic API. You can construct a request to obtain the coverage for an entire customer profile or for the customer profile in a specific map view. This article demonstrates the both types of request.

Flow Availability by Profile

User Story

The user wants to obtain information about the traffic coverage available for the NTdefault profile at zoom level 10. The information is to be delivered in XML

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*`flowavailability`* [`flowavailability.xml`], the resource and resource extension indicating the required response data format, here XML (to indicate JSON, use `flowavailability.json`)
Parameters:	`profile` [`profile=NTdefault`], the identifier of the customer profile `zoom` [`zoom=10`], the map zoom level

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/flowavailability.xml
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&profile=NTdefault
```

```
&zoom=10
```

Response

Because the request is simply based on the profile, with no restrictions, the response is too large to display here, but it contains XML data similar to those shown in the next example (under *Response* on page 63).

Flow Availability by Map View

User Story

The wants to obtain information about traffic flow coverage for the default customer profile in a certain map view at zoom level 10.

Request Summary

The following list summarizes the elements required to create a request matching the user story and shows, in square brackets, how those elements are used in the request example below. Note that the request example also uses the authentication parameters.

Resource:	*`flowavailability`* [`flowavailability.xml`], the resource and resource extension indicating the required response data format, here JSON (to indicate XML, use `flowavailability.xml`), but see also the note below the table)
Parameters:	`mapview` [`mapview=50.73818,7.09992;50.73736,7.10123`], the geographic coordinates of the map view defined (the coordinates of the top-left and bottom-right corners of the map view) `profile` [`profile=NTdefault`], the identifier of the customer profile `zoom` [`zoom=10`], the map zoom level

Note: Note that the JSON output is not finalized and potentially subject to change. Also, please note that XML and JSON output are provided in a non-formatted way. Output has been formatted for this document for better readability only.

Request

A request matching the user story is formulated as follows:

```
http://traffic.cit.api.here.com/traffic/6.0/flowavailability.xml
```

```
?app_id=DemoAppId01082013GAL
&app_code=AJKnXv84fjrb0KIHawS0Tg
&mapview=50.73818,7.09992;50.73736,7.10123
&profile=NTdefault
&zoom=10
```

Response

The XML response contains the XML data similar to the excerpt shown below.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<traffic_flow_availability:GetTrafficCoverageType
    xmlns:traffic_flow_availability="http://www.navteq.com/lbsp/LBSP-
Traffic-GetCoverage/4">
    <Request>
        <MetaInfo>
            <VerboseMode>-1</VerboseMode>
        </MetaInfo>
        <MapView>
            <Bounds>
                <TopLeft>
                    <Latitude>50.73818</Latitude>
                    <Longitude>7.09992</Longitude>
                </TopLeft>
                <BottomRight>
                    <Latitude>50.73736</Latitude>
                    <Longitude>7.10123</Longitude>
                </BottomRight>
            </Bounds>
            <Zoom>10</Zoom>
        </MapView>
        <Profile>NTdefault</Profile>
    </Request>
    <Response>
        <MetaInfo>
            <MapVersion>LBSP_WORLD_2013Q1_ECC</MapVersion>
            <ModuleVersion>6.0.24.1</ModuleVersion>
            <InterfaceVersion>6.0</InterfaceVersion>
            <Timestamp>2013-08-15T14:56:41.829+00:00</Timestamp>
        </MetaInfo>
        <Profile>NTdefault</Profile>
        <MapView>
            <Bounds>
                <TopLeft>
                    <Latitude>50.73818</Latitude>
                    <Longitude>7.09992</Longitude>
                </TopLeft>
                <BottomRight>
                    <Latitude>50.73736</Latitude>
                    <Longitude>7.10123</Longitude>
                </BottomRight>
            </Bounds>
            <Zoom>10</Zoom>
        </MapView>
```

```
        <Region>
            <TmcTableId>E0D01</TmcTableId>
            <Country3>DEU</Country3>
            <Country2>DE</Country2>
            <Name>Germany</Name>
        </Region>
    </Response>
</traffic_flow_availability:GetTrafficCoverageType>
```

Service Support

If you need assistance with this or other HERE products, please contact your HERE representative or Technical Customer Support via email at *tcsplatform@here.com*.

Chapter 4

API Reference

Topics:

This section provides a comprehensive reference to the HERE Traffic API.

Resources

This section provides a comprehensive reference to the Traffic API resources and the parameters supported by them.

Flow

This resource is responsible for servicing requests for traffic flow data for a geospatial area. The response contains speed and congestion ("jam factor") information. The delivery formats are either XML or JSON. A request can optionally specify shape and functional class information for the roadways for which to obtain traffic flow information. .

Parameter Combinations for Common Use Cases

Table 11: Parameters for requesting Flow Data

Required	Optional
• `quadkey`, `prox`, `bbox` (except with Z/X/Y address)	• `maxfunctionalclass` • `minjamfactor` • `maxjamfactor`

Response Data Format

This resource can deliver response data either in XML of JSON, depending on the addressing scheme:

- [Z]/[X]/[Y] addressing – use the URL variables `xml` or `json`, immediately after the resource name, for example `.../flow/xml/...` or `.../incidents/json/...`
- otherwise – use resource extensions `.xml` or `.json`, for example `.../flow.json/...`, `.../incidents.xml/...`

See also *Examples* on page 27 for complete working URLs.

Resource Parameters

Table 12: Traffic Flow Request Parameters

Parameter	Description
`app_code`	`xs:string` A 20-byte Base64 URL-safe encoded string used for the authentication of the client application.

Parameter	Description
	You must include an `app_id` and `app_code` with every request. For further details, please see *Acquiring Credentials* on page 13.
`app_id`	`xs:string` A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an `app_id` and `app_code` with every request. For further details, please see *Acquiring Credentials* on page 13.
`bbox`	`GeoBoundingBoxType`: lat,lon; lat,lon A type of spatial filter. A spatial filter defines the area targeted by the request (the area for which to retrieve data or a map tile). A bounding box represents a rectangular area specified by two latitude/longitude pairs; the first pair defines the top-left corner of the bounding box, the second the bottom-right corner. **Note:** The maximum width and height of a bounding box is 10 degrees.
`prox`	`GeoProximityType`: lat,long,distance A type of spatial filter. A spatial filter defines the area targeted by the request (the area for which to retrieve data or a map tile). A response to a request using proximity includes any results that fall within a circle defined by the a latitude and a longitude of its center and radius (given in meters).
`maxfunctionalclass`	`xs:int` An indicator of the maximum functional class that must be matched for a traffic flow item to be included in the response. All items with functional class lower than or the same as the value of this parameter are included in the response. For more details about functional class, see *FunctionalClassType* on page 78.
`quadkey`	`QuadkeyType` An identifier of a map tile in a grid of tiles spanning the entire globe. A quadkey is a single number string from 0 to 21 digits long. The map zoom level for the grid to which a quadkey refers is equal to the number of digits in the quadkey string. Use only with a quadkey addressing scheme endpoint. See also *Quadkeys* on page 19.
`responseattributes`	`xs:list` A container for a list indicating optional information to be included in the traffic flow data response. By default, the response does not include shape or functional class information, but unless indicated by this parameter. The accepted value is a list of the following specifiers: • `sh` or `shape` • `fc` See *Requesting Traffic Flow Data with a Bounding Box Filter* on page 31 for an example request using the `responseattributes` parameter.
`minjamfactor`	`xs:double` An indicator of the minimum jam factor value for traffic flow items to be included in the response. All flow events with a *lower* jam factor than the value specified by the `minjamfactor` parameter are omitted. The parameter can be used by clients that require

Parameter	Description
	flow data for congested roadways only. The value must be in the range `0.0` to `10.0` (inclusive).
`maxjamfactor`	`xs:double` An indicator of the maximum jam factor value for traffic flow items to be included in the response. All flow events with a *higher*jam than the value specified by the `maxjamfactor` parameter are omitted. This parameter can be used by clients that require flow data only for congestion-free roads. The values must be in the range `0.0` to `10.0` (inclusive).
`jsoncallback`	`xs:string` A parameter providing the name of a user-defined function used to wrap the json response.

Tiles

This resource processes requests for map tiles reflecting traffic information for a particular geographic area. The response delivers the map tile as bitmap.

Deprecation Notice

HERE has changed the way traffic tiles are served. Live traffic tiles have been removed from the Traffic API and are now delivered by the Map Tile API infrastructure. Documentation relating to live traffic tiles is part of the Map Tile API documentation.

This does not affect traffic pattern tiles, which continue to be served by the Traffic API.

Parameter Combinations for Common Use Cases

Table 13: Request a Traffic Pattern Tile

Required	Optional
• `quadkey` (used only in a quadkey address) • `pattern_time`	• `noContent` • `profile` • `tables` • `res` (quadkey address only) • `depth` (quadkey address only)

Table 14: Set the Color Depth of a Traffic Tile

Required	Optional
• endpoint change (used only in a [Z]/[X]/[Y] address): `.../{zoom}/{column}/{row}/{size}/{format}?`	• `noContent` • `profile` • `tables`

Required	Optional
• quadkey (used only in a quadkey address) • depth (used only in a quadkey address)	• pattern_time • res (quadkey address only)

Table 15: Set the Size of a Traffic Tile

Required	Optional
• endpoint change (used only in a [Z]/[X]/[Y] address): `.../{zoom}/{column}/{row}/{size}/{format}?` • quadkey (used only in a quadkey address) • res (used only in a quadkey address)	• depth (quadkey address only) • noContent • profile • pattern_time • tables

Table 16: Filter Tile Display by TMC Table

Required	Optional
• quadkey (used only in a quadkey address) • tables	• depth (quadkey address only) • noContent • profile • pattern_time • res (quadkey address only)

Resource Parameters

Table 17: Tile Request Parameters

Parameter	Description
app_id	xs:string A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an app_id and app_code with every request. For further details, please see *Acquiring Credentials* on page 13.
app_code	xs:string A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an app_id and app_code with every request. For further details, please see *Acquiring Credentials* on page 13.
depth	xs:string An indicator of tile color depth. The permitted values are:

Parameter	Description
	• `8bit` • `32bit` Used only with quadkey tile addressing.
`nc`	`xs:int` A flag indicating whether the response is to return HTTP error code 204 (empty) for unpopulated traffic tiles (1). The permitted values are: • `1` • `0`
`pattern_time`	`xs:int` An indicator of the time (within the week) for which to obtain a traffic pattern tile. The value represents local time for the targeted area, in seconds beginning on Sunday at 0:00. The permitted values are numeric and must fall in the range from 0 to 604800.
`profile`	`xs:string` Profile defines such attributes as geographic coverage (TMC tables available for use) and custom color mapping for tile overlays.
`quadkey`	`QuadkeyType` An identifier of a map tile in a grid of tiles spanning the entire globe. A quadkey is a single number string from 0 to 21 digits long. The map zoom level for the grid to which a quadkey refers is equal to the number of digits in the quadkey string. Use only with a quadkey addressing scheme endpoint. See also *Quadkeys* on page 19.
`res`	`xs:string` An indicator of the map tile resolution to be used only with a quadkey tile address. It specifies the dimensions of a rendered square map tile in pixels. The permitted values are: • `bitmap128` • `bitmap256` • `bitmap512`
`tables`	`TMCCodeType` A parameter that specifies TMC tables in the profile to include or exclude from response. If the value specifies multiple tables, the table ids are separated by commas. Positive values are included, and negative values are excluded. **Note:** A combination of positive and negative values is not supported and results in an error response. By default (when the parameter is not used in a request) the response includes all tables in the profile. See *TMCCodeType* on page 78 for a full list of TMC tables or your customer profile for a list of tables available to you.

Incidents

This resource is responsible for handling requests for traffic incident information for or a geospatial area. The delivery formats are either XML or JSON.

Parameter Combinations for Common Use Cases

Table 18: Request Traffic Incident Data Using Localization

Required	Optional
• quadkey, prox, bbox (except with Z/X/Y address) • c and lg as a pair or with i18n	• criticality • endTime • inc22 • jsoncallback • localtime • maxresults • profile • sort • startTime • status • tables • type • verified

Table 19: Request a Subset of Traffic Incident Data

Required	Optional

• quadkey, prox, bbox (except with Z/X/Y address)
• One or more of these filter parameters:
  ◦ criticality
  ◦ endTime
  ◦ maxresults
  ◦ profile
  ◦ startTime
  ◦ status
  ◦ tables
  ◦ type
	• c and lg as a pair or with i18n
	• inc22
	• jsoncallback
	• localtime
	• sort

Required	Optional
◦ verified	

Response Data Format

This resource can deliver response data either in XML of JSON, depending on the addressing scheme:

- [Z]/[X]/[Y] addressing – use the URL variables `xml` or `json`, immediately after the resource name, for example `.../flow/xml/...` or `.../incidents/json/...`
- otherwise – use resource extensions `.xml` or `.json`, for example `.../flow.json/...`, `.../incidents.xml/...`

See also *Examples* on page 27 for complete working URLs.

Resource Parameters

Table 20: Incident Request Parameters

Parameter	Description
`app_code`	`xs:string` A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an `app_id` and `app_code` with every request. For further details, please see *Acquiring Credentials* on page 13.
`app_id`	`xs:string` A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an `app_id` and `app_code` with every request. For further details, please see *Acquiring Credentials* on page 13.
`bbox`	`GeoBoundingBoxType`: lat,lon; lat,lon A type of spatial filter. A spatial filter defines the area targeted by the request (the area for which to retrieve data or a map tile). A bounding box represents a rectangular area specified by two latitude/longitude pairs; the first pair defines the top-left corner of the bounding box, the second the bottom-right corner. **Note:** The maximum allowed size of the bounding box (width and height) is limited to a configurable value expressed in degrees. The default maximum values for the width and height of a bounding box is 30 degrees.
`c`	`xs:string` An indicator of the country (locale for the incident response). The accepted values include certain ISO country codes (two bytes, ISO 3166-1-alpha-2). The parameter must be used in conjunction with the `lg` parameter. By default (if the parameter is not specified), the time and format of the response are based on the location of the incident.
`criticality`	`xs:string`

Parameter	Description
	A filter that selects incident reports according to criticality, case insensitive. The parameter may contain multiple comma-separated selected from the following list: • `critical` or `0` • `major` or `1` • `minor` or `2` • `lowImpact` or `3` Note that if this parameter is not provided, all incident reports are included in the response, regardless of criticality (default behavior).

`endTime`	`xs:dateTime`
	The maximum end time that must be matched by an incident to be included in the results, in GMT format.
	`YYYY-MM-DDThh:mm:ss.s+zzzz`
	See *xs:dateTime* on page 83 for the format to use when filtering time values.

`i18n`	`xs:boolean` A flag indicating whether incident messages should be translated to the language specified in the `lg` parameter. The parameter must be used in conjunction with the `lg`. The permitted values for the parameter are: • `true` • `false`
`inc22`	`xs:boolean` A flag indicating whether to format the response to match version. 2.2 of the incidents XSD (`true`). Version 2.2 of the XSD contains fields and information not included in the default version 2.1 (5.0) The permitted values are: • `true` or `1` • `false` or `0`
`jsoncallback`	`xs:string` A parameter providing the name of a user-defined function used to wrap the json response.
`lg`	`xs:string` A specifier of the language of the locale for the incident response. The parameter must be used in conjunction with the `c` parameter. By default (if `lg` is not provided), the language of the response is based on the incident location. See *Supported Translation Languages* on page 26 for allowed values.
`localtime`	`xs:boolean` A flag indicating if all time values in the response to a request for traffic incident data should be in the local time of the incident (`true`), instead of GMT +h:mm (`false`). The permitted values are: • `true` or `1`

Parameter	Description
	• `false` or `0`
`maxresults`	`xs:int` An indicator of the maximum number of incidents to be included in the response. Only non-zero positive number values are allowed.
`profile`	`xs:string` Profile defines such attributes as geographic coverage (TMC tables available for use) and custom color mapping for tile overlays.
`prox`	`GeoProximityType`: lat,long,distance A type of spatial filter. A spatial filter defines the area targeted by the request (the area for which to retrieve data or a map tile). A response to a request using proximity includes any results that fall within a circle defined by the a latitude and a longitude of its center and radius (given in meters).
`quadkey`	`QuadkeyType` An identifier of a map tile in a grid of tiles spanning the entire globe. A quadkey is a single number string from 0 to 21 digits long. The map zoom level for the grid to which a quadkey refers is equal to the number of digits in the quadkey string. Use only with a quadkey addressing scheme endpoint. See also *Quadkeys* on page 19.
`sort`	`xs:string` An indicator, specifying how to sort the response data to a traffic incident request according to criticality. The permitted values are: • `criticalityDesc` • `criticalityAsc`

`startTime`	`xs:dateTime`
	The minimum start time to be matched by an incident to be included in the results, in GMT format.
	`YYYY-MM-DDThh:mm:ss.s+zzzzzz`
	See *xs:dateTime* on page 83 for the format to use when filtering time values.

`status`	`xs:string` A status indicator that triggers incident selection according to their status; case insensitive. The parameter value may consist of multiple values, separated by a comma. The accepted status values include: `active`, `inactive`, `deleted`, and `expired`. By default (when the parameter is not specified) the response includes all incident reports, regardless of status.
`tables`	`TMCCodeType` A parameter that specifies TMC tables in the profile to include or exclude from response. If the value specifies multiple tables, the table ids are separated by commas. Positive values are included, and negative values are excluded. **Note:** A combination of positive and negative values is not supported and results in an error response.

Parameter	Description
	By default (when the parameter is not used in a request) the response includes all tables in the profile. See *TMCCodeType* on page 78 for a full list of TMC tables or your customer profile for a list of tables available to you.
`type`	`xs:string` An indicator, specifying the type of items to include in the response, case insensitive. The value may contain multiple type identifiers separated by a comma. The permitted traffic item type identifiers that can be used in the value are: • `Accident` • `Congestion` • `DisabledVehicle` • `RoadHazard` • `Construction` • `PlannedEvent` • `MassTransit` • `OtherNews` • `Weather` • `Misc` By default (when the parameter is not specified in the request), the response includes all incident reports, regardless of type.
`verified`	`xs:boolean` A flag indicating if the response is to include or exclude incident reports that have been confirmed. The permitted values are: • `true` or `1` • `false` or `0` By default, when this parameter is not specified in the request, the response includes all incident reports, regardless of type.

Flowavailability

This resource is responsible for handling requests for information about the availability of traffic flow information.

Response Data Format

This resource can deliver response data either in XML or JSON. To select the format use resource extensions `.xml` or `.json`, for example `.../flow.json/...`, `.../incidents.xml/...`

See also *Examples* on page 27 for complete working URLs.

Request Paramters

Table 21: Availability Request Parameters

Parameter	Description
app_code	xs:string A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an app_id and app_code with every request. For further details, please see *Acquiring Credentials* on page 13.
app_id	xs:string A 20-byte Base64 URL-safe encoded string used for the authentication of the client application. You must include an app_id and app_code with every request. For further details, please see *Acquiring Credentials* on page 13.
id	xs:int A user-defined incident identifier. The response contains the specific incident requested. Any integer is allowed.
mapview	GeoBoundingBoxType: lat,lon; lat,lon A type of spatial filter. A spatial filter defines the area targeted by the request (the area for which to retrieve data or a map tile). mapview defines the area to check for traffic coverage in the same way as bbox, following the same pattern and rules. If this parameter is not specified, the response returns all countries supported by the specified profile.
profile	xs:string Profile defines such attributes as geographic coverage (TMC tables available for use) and custom color mapping for tile overlays.
zoom	xs:string An indicator of the map zoom level. It dictates the visibility rules to be taken in account when selecting traffic data items. By default, if this parameter is not specified in a traffic flow data or traffic incident request, the response includes all incident reports, regardless of type.

Meta Resources

The table below documents HERE Traffic API resources that provide API version information, trigger self-tests on the API or to obtain XSDs.

Table 22: Additional Request Parameters

Resource	URL variable	Description	Example	Response
version		Processes requests for version information	http://traffic.api.here.com/traffic/6.0/version	Service: Traffic

Resource	URL variable	Description	Example	Response
		for HERE Traffic API, including the components that make up the Traffic Service.		Service, Version: 6.0.28.1
`exodus.html`		Triggers a self-test on HERE Traffic API. If the service is not ready to serve tiles, a 404 error message results.	`http://traffic.cit.api.here.com/traffic/6.0/exodus.html?app_id=DemoAppId01082013GAL&app_code=AJKnXv84fjrb0KIHawS0Tg`	200 Ok
`xsd`	`incident.xsd`	Processes a request for the `incidents` XSD.	`http://traffic.cit.api.here.com/traffic/6.0/xsd/incident.xsd?app_id=DemoAppId01082013GAL&app_code=AJKnXv84fjrb0KIHawS0Tg`	XSD

Data Types

This section provides a comprehensive reference to types used throughout the Traffic API.

LatitudeType

This type corresponds to geographic coordinate latitude and represents the location of a place on the surface of the Earth north or south of the equator.

`LatitudeType` is a value of the type `xs:double` with the following restrictions:

- Must satisfy: -90 ≤ value ≤ 90
- Unit: decimal degrees
- Reference System: WGS 84
- Precision: 7 positions after decimal point, additional positions are ignored

Query Parameter Representation

Valid examples:

```
45.1234567
-12.3456789
45.123 -12.3
```

LongitudeType

This type corresponds to the geographic coordinate longitude and represents the location of a place on the surface of the Earth east or west of the prime meridian.

`LongitudeType` is a value of the type `xs:double` type with the following restrictions:

- Must satisfy: -180 ≤ value ≤ 180
- Unit decimal degrees
- Reference System: WGS 84
- Precision: 7 positions after decimal point, additional positions are ignored

Query Parameter Representation

Valid examples:

```
90.1234567
-120.3456789
90.123 -120.3
```

FunctionalClassType

Functional class is a road type indicator, reflecting traffic speed and volume, as well as the importance and connectivity of the road.

`FunctionalClassType` is a value of the type `xs:byte`. The following values are supported:

- 1: a road with high volume, maximum speed traffic
- 2: a road with high volume, high speed traffic
- 3: a road with high volume traffic
- 4: a road with high volume traffic at moderate speeds between neighborhoods
- 5: a road whose volume and traffic flow are below the level of any other functional class

TMCCodeType

`TMCCodeType` holds a value of the type`xs:string` with the following restrictions:

- Fixed length: 5 characters
- Format: `ECC+CCD+Table`, where the following rules apply:

ECC	is the Extended Country Code.
CCD	is the TMC country code.

Table is the defined TMC Table

Query Parameter Representation

A value for the query parameter `tables` to filter on TMC codes is constructed as shown below.

```
&tables=ECC+CCD+Table
```

For example, to filter for the New England region in the USA, concatenate the US extended country code (A0) with the TMC country code (1) and the New England region table ID (29), producing the following parameter/value pair:

```
&tables=A0129
```

TMC Table ID Examples

Below is a list of TMC table IDs supported by the Traffic API, along with the associated country/region. The table contains: the country name, three-letter ISO country code, TMC country ID ('CID'), extended country code ('ECC'), TMC country code ('CCD'), and TMC Table ('Table') as well as the associated region and table owner.

There may be different combinations of CCD and Table values from different owners pointing at the same country/region. The table below contains a superset of regions supported by HERE Traffic API.

Table 23: TMC Table IDs

Country	ISO Code	CID	ECC	CCD	Table	Region	Owner
Australia	AUS	62	F0	1	53	Canberra, CT	ATMCC
Australia	AUS	63	F0	2	54	Sydney, NSW	SENSIS
Australia	AUS	61	F0	3	01	Melbourne, V	SENSIS
Australia	AUS	64	F0	4	56	Brisbane, Q	SENSIS
Australia	AUS	60	F0	5	57	Adelaide, SA	ATMCC
Australia	AUS	65	F0	6	58	Perth, WA	ATMCC
Australia	AUS	66	F0	7	59	Hobart, TAS	ATMCC
Australia	AUS	67	F0	8	60	Darwin, NT	ATMCC
Austria	AUT	4	E0	A	01	Austria	TMC
Belgium	BEL	6	E0	6	01	Belgium	TMC
Brazil	BRA	397	A2	B	14	Sul, Sudeste	Nokia

Country	ISO Code	CID	ECC	CCD	Table	Region	Owner
Brazil	BRA	398	A2	B	15	Norte, Nordeste, Centro-Oeste	Nokia
Canada	CAN	996	A1	C	09	Ontario	Nokia
Canada	CAN	442	A1	C	23	British Columbia	Nokia
Canada	CAN	443	A1	C	24	Quebec	Nokia
Canada	CAN	446	A1	C	27	Alberta	Nokia
Denmark	DNK	12	E1	9	09	Denmark	TMC
Finland	FIN	15	E1	6	17	Finland	TMC
France	FRA	16	E1	F	32	France	TMC
Germany	DEU	58	E0	D	01	Germany	TMC
Greece	GRC	19	E1	1	18	Greece	BeMobile
India	IND	858	F2	5	02	Central	Nokia
India	IND	859	F2	5	03	South	Nokia
India	IND	860	F2	5	04	North	NATVEQ
Indonesia	IDN	917	F2	C	21	Jakarta	GEWI
Ireland	IRL	23	E3	2	42	Ireland	TrafficNav
Italy	ITA	25	E0	5	01	Italy	InfoBlu
Luxembourg	LUX	32	E1	7	01	Luxembourg	TMC
Mexico	MEX	995	A5	F	36	Mexico	Nokia
Netherlands	NLD	39	E3	8	17	Netherlands	TMC
Norway	NOR	40	E2	F	49	Norway	
Poland	POL	41	E2	3	05	Poland	
Portugal	PRT	42	E4	8	42	Portugal	
Russia	RUS	629	E0	7	23	Moscow	Nokia
Russia	RUS	630	E0	7	24	St. Petersburg	Nokia
Russia	RUS	631	E0	7	25	Yekaterinburg	Nokia
Russia	RUS	632	E0	7	26	Nizny Novgorod	Nokia
Russia	RUS	633	E0	7	27	Krasnodar, Rostov-on-Don	Nokia
Russia	RUS	634	E0	7	28	Novosibirsk	Nokia
Saudi Arabia	SAU	685	F0	9	26	Saudi Arabia	Nokia
South Africa	ZAF	712	D0	A	17	South Africa	Netstar

Country	ISO Code	CID	ECC	CCD	Table	Region	Owner
Spain	ESP	49	E2	E	17	Spain	
Sweden	SWE	50	E3	E	33	Sweden	
Switzerland	CHE	51	E1	4	09	Switzerland	
Thailand	THA	227	F3	2	19	Thailand	ITS Thailand
Turkey	TUR	243	E3	3	36	Turkey (former BeMobile table)	Nokia
UAE	ARE	1097	F2	D	19	United Arab Emirates (Dubai, Abu Dhabi, Sharjah)	Nokia
UK	GBR	55	E1	C	07	Great Britain	TrafficMaster
UK	GBR	55	E1	C	10	Great Britain	ITIS
Ukraine	UKR	56	E4	6	33	Ukraine	Nokia
USA	USA	999	A0	1	01	Alabama/Georgia	Nokia
USA	USA	266	A0	1	02	Florida	Nokia
USA	USA	267	A0	1	03	Eastern PA/Southern NJ/Delaware	Nokia
USA	USA	268	A0	1	04	Western PA/New York	Nokia
USA	USA	269	A0	1	05	Northern CA/Nevada	Nokia
USA	USA	270	A0	1	06	Southern CA & Las Vegas	Nokia
USA	USA	271	A0	1	07	Illinois/Indiana/Wisconsin	Nokia
USA	USA	272	A0	1	08	Michigan/Ohio	Nokia
USA	USA	274	A0	1	10	Virginia/Maryland/North Carolina	Nokia
USA	USA	275	A0	1	11	Northern Texas/Oklahoma	Nokia
USA	USA	276	A0	1	12	Southern Texas	Nokia
USA	USA	277	A0	1	13	AR/LA/MS/TN	Nokia
USA	USA	278	A0	1	14	Oregon/Washington	Nokia
USA	USA	279	A0	1	15	Arizona/New Mexico/Texas	Nokia
USA	USA	280	A0	1	16	Colorado/Utah	Nokia
USA	USA	281	A0	1	17	Idaho/Montana/Wyoming	Nokia
USA	USA	282	A0	1	18	North Central States	Nokia
USA	USA	283	A0	1	19	Kansas/Missouri/Southern Illinois	Nokia
USA	USA	284	A0	1	20	New York/Northern NJ/Connecticut	Nokia
USA	USA	285	A0	1	21	Kentucky/Tennessee/Indiana	Nokia

Country	ISO Code	CID	ECC	CCD	Table	Region	Owner
USA	USA	286	A0	1	22	Ohio/Maryland/West Virginia	Nokia
USA	USA	289	A0	1	25	North Carolina/South Carolina	Nokia
USA	USA	290	A0	1	26	Hawaii	Nokia
USA	USA	293	A0	1	29	New England	Nokia
USA	USA	297	A0	1	33	Alaska	Nokia

LanguageCodeType

This type is used whenever a language of a value needs to be referenced.

`LanguageCodeType` is of the type `xs:string` with the following restrictions:

- Format: Language code compliant with RFC5646

Query Parameter Representation

Valid examples:

```
de
en
```

LanguageCodeParameterType

This type is used whenever a language filter needs to be specified.

`LanguageCodeparameterType` is of the type `xs:string` with the following restrictions:

- Format: Language code following RFC4647

Query Parameter Representation

Valid examples:

```
de
en
```

CountryCodeType

This type is used whenever a country needs to be referenced for localization.

`CountryCodeParameterType` is of the type `xs:string` with the following restrictions:

- Format: An ISO 3166-1-alpha-2 country code

Query Parameter Representation

Valid examples:

```
FR
DE
```

Built-in Resources Types

The HERE Traffic API supports a number of built-in data types described on the following pages.

xs:dateTime

The lexical representation of xs:dateTime consists of finite-length sequences of characters of the following form:

```
YYYY + "-" + MM + "-" +
DD + "T" + hh + ":"+
mm + ":" + ss + ("." + s)? +
(zzzzzz)?
```

Syntax	Description
YYYY	is a four-or-more digit optionally negative-signed numeral that represents the year; if more than four digits, leading zeros are prohibited, and '0000' is prohibited
MM	is a two-digit numeral that represents the month
DD	is a two-digit numeral that represents the day
hh	is a two-digit numeral that represents the hour The value 24 is permitted if the minutes and seconds represented are zero, and the dateTime value so represented is the first instant of the following day (the hour property of a dateTime object in the value space cannot have a value greater than 23)
mm	is a two-digit numeral that represents the minute
ss	is a two-integer-digit numeral that represents the whole seconds
s	represents the fractional seconds (optional)
zzzzzz	represents the timezone

The timezone portion zzzzzz is represented as follows:

```
(("+" | "-") + hh + ":"+ mm) | "Z"
```

Syntax	Description
"+"|"-"	"+" indicates a non-negative duration, "-" indicates a non-positive duration.
hh	is a two-digit numeral (with leading zeros as required) that represents the hours
mm	is a two-digit numeral that represents the minutes
"Z"	"Z" represent the zero-length duration timezone, UTC "Z" is the canonical representation for UTC dateTime values.

DateTime Example

`2002-10-10T12:00:00-05:00` (noon on 10 October 2002, Central Daylight Savings Time as well as Eastern Standard Time in the U.S.) is `2002-10-10T17:00:00Z`, five hours later than `2002-10-10T12:00:00Z`.

xs:list

XML Schema provides a list type allowing more than one item within a single XML element. Items are separated by space.

The following example defines a list element named "Foo" which contains string elements.

```
<xs:simpleType name="MyListType">
    <xs:list itemType="xs:string"/>
</xs:simpleType>
<xs:element name="Foo" type="MyListType">
```

A valid instantiation of the "Foo" element would be

```
<Foo>ABC DEF GHI</Foo>
```

In this example, the "Foo" element contains the three items "ABC", "DEF", and "GHI".

In representation formats other than XML, the space as separator sometimes becomes unhandy. Therefore, a standard representation format for list types is defined for each representation format below.

Query Parameter Representation

In query strings, items of list types are separated by one of the following characters:

"," (comma) is used at top level

```
<Foo[0]> + "," + <Foo[1]> + "," + <Foo[2]> + "," + …
```

";" (semicolon) is used at second level, or if comma is already used in the representation of the list items

```
<Foo[0]> + ";" + <Foo[1]> + ";" + <Foo[2]> + ";" + …
```

"!" (exclamation point) is used at third level, or if comma and semicolon are already used in the representation of the list items

```
<Foo[0]> + "!" + <Foo[1]> + "!" + <Foo[2]> + "!" + …
```

The following syntax is introduced as shortcut to represent list types:

```
<Foo[]>
```

Note: List types and element sequences are represented in the same manner (see also *xs:sequence* on page 85).

JSON Representation

List types are represented as arrays:

```
[<list[0]>, <list[1]>, <list[2]>, …]
```

xs:sequence

A schema can define elements which can appear more than once. The following example defines an element named "Bar" which can be included *n* times.

```
<xs:element name="Bar" type="xs:string" maxOccurs="unbounded"/>
```

A valid instantiation of the "Bar" element would be

```
<Bar>ABC</Bar><Bar>DEF</Bar><Bar>GHI</Bar>
```

Element sequences are described in all API documentations with an "[]" as suffix like this:

```
Bar [] sequence of strings
```

In representation formats other than XML, listing the same element more than once is inconvenient, so a standard representation format for sequences is defined for each representation format below.

Query Parameter Representation

In query strings, items of list types are separated by one of the following characters:

"," (comma) is used at top level

```
<Bar[0]> + "," + <Bar[1]> + "," + <Bar[2]> + "," + …
```

";" (semicolon) is used at second level, or if comma is already used in the representation of the list items

```
<Bar[0]> + ";" + <Bar[1]> + ";" + <Bar[2]> + ";" + …
```

"!" (exclamation point) is used at third level, or if comma and semicolon are already used in the representation of the list items

```
<Bar[0]> + "!" + <Bar[1]> + "!" + <Bar[2]> + "!" + …
```

When applying this rule to the above example, the result would be: ABC,DEF,GHI

The following syntax is introduced as shortcut to represent element sequences:

```
<Bar[]>
```

Note: list types and element sequences are represented in the same manner. See also *xs:list* on page 84)

JSON Representation

Element sequences will be represented as arrays:

```
[<list[0]>, <list[1]>, <list[2]>,…]
```

HTTP Status Codes

Traffic API supports the standard *HTTP status codes*.

Table 24: HERE Error Codes

Error code	Description
200 OK	Indicates success, but may also be returned when an invalid resource name and/or an invalid parameter combination has been used in the request.
204 No Content	Indicates that the request was valid but there is no traffic data available.

Error code	Description
400 Bad request	Invalid parameter value in the request, for example `zoom` out of range. Examples include: • The quadkey parameter is invalid (length] 22 or any character not in [0,1,2,3]). • The required parameters are not provided. • The query combines both bbox and quadkey parameters, which is not permitted. You must use either the bbox or quadkey parameter. • An invalid bbox parameter format. The bbox parameter must be specified in the form [lat0],[lon0];[lat1],[lon1] • Incorrect values for the filtering parameters were used.
403 Forbidden	Incorrect `token` in the request. See *Acquiring Credentials* on page 13 for more information.
404 Not found	Unsupported parameter in the request. Examples include: • The domain-section of the URL does not point at a valid service environment. • The path-section of the URL does not point at a quadkey service instance.
500 Internal error	The service is not available or server configuration issue.
503 Service Unavailable	Indicates that the services is temporarily unavailable due to system overload or maintenance

The API supports the standard authorization mechanism common to all HERE APIs. This mechanism can return status codes detailed below.

Table 25: Authentication Responses

Scenario	Response Code	HTTP Response Message
`app_id` and `app_code` are both missing	400	The request is missing the `app_id` and `app_code` parameters. They must both be passed as query parameters. If you do not have `app_id` and `app_code`, please obtain them see *Acquiring Credentials* on page 13
`app_id` is there but the `app_code` is missing	400	The request is missing the `app_code`. It must be passed as a query parameter.
`app_code` is there but `app_id` is missing	400	The request is missing the `app_id`. It must be passed as a query parameter.
`app_id` and `app_code` are both present in the request but the validation fails.	401	This is not a valid `app_id` and `app_code` pair. Please verify that the values are not swapped between the `app_id` and `app_code`. See also *Acquiring Credentials* on page 13
No matching pattern in the authentication mechanism configuration for this requested URL	404	The URL resource cannot be found.

Scenario	Response Code	HTTP Response Message
If the requested URL does exist but the HTTP verb is not supported for that URL	405	The HTTP method is not supported for this URL.
If SSL is required, but the request is not sent over SSL.	400	SSL/TLS is required for this URL; please retry the request using SSL/TLS.
Unexpected authentication error	500	Server error - please retry again at a later time.