

# Javascript Traffic

## Adding Traffic Flow & Incidents to a Map

Traffic can be added to the map by displaying markets, flow and incidents.

**To declare a new `MQA.Traffic` object**

```
myTraffic = new MQA.Traffic(myMap);
```

myMap is an `MQA.Tilemap` object

## Market Data

Traffic market data indicators are icons that will display on the map at zoom levels 2-6, showing that traffic is available for that area.

Valid JSON traffic market data elements - city, latitude, longitude, and state

**`MQA.Traffic.addMarkets()` ;**

This function will add the traffic market data to the map.

Example: `myTraffic.addMarkets()` ;

**`MQA.Traffic.removeMarkets()` ;**

This function will remove the traffic market data from the map.

Example: `myTraffic.removeMarkets()` ;

**`MQA.Traffic.setValue("marketTitleCallback", callback)`**

This function sets the title of the rollover window that displays when the mouse is moved over the market data icon.

Example:

```
myTraffic.setValue("marketTitleCallback",MarketCallbackTitle);

function MarketCallbackTitle(data)
{
    //sets the rollover window to display the city & //state
    for the market
    return data.city + ', ' + data.state;
}
```

**`MQA.Traffic.setValue("marketContentCallback", callback)`**

This function sets the content of the infowindow that displays when the market icon is clicked.

Example: `myTraffic.setValue("marketContentCallback",MarketCallbackContent)` ;

```
function MarketCallbackContent(data)
{
```

```

        //will set the contents of the market data indicator
        //infowindow with a link to zoom in to the street
        //level to view traffic
        var s;
        s = 'Click <a href="#" ';
        s += 'onclick="myMap.setCenter(new MQA.LatLng(';
        s += data.latitude + ',' + data.longitude;
        s += '), 7)">here</a> to zoom in.';
        return s;
    }

```

## **Flow Data**

Traffic flow is the image that is displayed on the map indicating whether or not the flow of a particular road is good (green), slow (yellow) or stopped (red).

### **MQA.Traffic.addFlow()**

This function will add the flow image to the map. This image will be displayed at zoom levels 6-16.

Example: `myTraffic.addFlow();`

### **MQA.Traffic.removeFlow()**

This function will remove the flow image from the map.

Example: `myTraffic.removeFlow();`

### **MQA.Traffic.setValue("flowOpacity",opacity)**

This function will set the opacity of the traffic flow image on the map.

opacity - float value from 0 to 1

Example: `myTraffic.setValue("flowOpacity",.7);`

### **MQA.Traffic.getValue("flowOpacity")**

This function will return the opacity of the traffic flow image.

Example: `var s = myTraffic.getValue("flowOpacity");`

## **Incident Data**

Incidents are displayed on the map for construction, traffic incidents, and events.

Valid JSON traffic incident data elements - description, endTime, incidentType, key, latitude, longitude, severity, startTime, and title

### **MQA.Traffic.addIncidents()**

This function will add all available incidents to the map.

Example: `myTraffic.addIncidents();`

### **MQA.Traffic.removeIncidents()**

This function will remove all incidents from the map.

Example: `myTraffic.removeIncidents();`

#### **MQA.Traffic.setValue("incidentTypeFilter",incidentTypes)**

incidentTypes - array of incident types. Valid types are:  
MQA.TRAFFIC\_INCIDENTS, MQA.TRAFFIC\_CONSTRUCTION,  
MQA.TRAFFIC\_EVENTS

Example: `var myIncidents = new Array();  
myIncidents.push(MQA.TRAFFIC_INCIDENTS);  
myTraffic.setValue("incidentTypeFilter",myIncidents);`

#### **MQA.Traffic.setValue("incidentTitleCallback",callback)**

This function sets the title of the rollover window that displays when the mouse is moved over the traffic incident icon.

Example:

```
myTraffic.setValue("incidentTitleCallback",IncidentCallbackTitle);  
  
function IncidentCallbackTitle(data)  
{  
    //sets the rollover window to display the incident type &  
    //title  
    return data.incidentType + ', ' + data.title;  
}
```

#### **MQA.Traffic.setValue("incidentContentCallback",callback)**

This function sets the contents of the infowindow that is displayed when a traffic icon is clicked.

Example:

```
myTraffic.setValue("incidentContentCallback",IncidentCallbackContent);  
  
function IncidentCallbackContent(data)  
{  
    //displays the description of the incident  
    var s;  
    s = data.description;  
    return s;  
}
```

## **Other Functions**

#### **MQA.Traffic.isAvailable()**

This function will return true/false, indicating if traffic is available.

Example: `myTraffic.isAvailable();`

#### **MQA.Traffic.setTimeoutDuration(m integer)**

This function sets the timeout duration in milliseconds before failing the request.

Default value is 10 seconds.

m - timeout in milliseconds

Example: `myTraffic.setTimeoutDuration(200);`

#### **MQA.Traffic.setTimeoutDuration()**

This function will return the timeout duration in milliseconds. Default value is 10 seconds.

Example: `var timeout = myTraffic.setTimeoutDuration();`

#### **MQA.Traffic.setMaxInfoWindowWidth(w integer)**

This function sets the width of the infowindow that displays for a traffic incident. Default value is 285 pixels.

w - maximum width in pixels

Example: `myTraffic.setMaxInfoWindowWidth(350);`

#### **MQA.Traffic.refresh()**

This function makes a new call to the traffic service thus 'updating' the map with the latest information.

Example: `myTraffic.refresh();`

## **Events**

Traffic events are handled the same as other events within the TileMap Toolkit. Utilize the `MQA.EventManager` to add/remove events.

flowadded  
flowremoved  
marketsadded  
marketsremoved  
markettimeout  
incidentsadded  
incidentsremoved  
incidenttimeout

The following are the event names that will be returned:

MQA.Traffic.flowadded  
MQA.Traffic.flowremoved  
MQA.Traffic.marketsadded  
MQA.Traffic.marketsremoved  
MQA.Traffic.markettimeout  
MQA.Traffic.incidentsadded  
MQA.Traffic.incidentsremoved  
MQA.Traffic.incidenttimeout

## **Traffic Control**

### **Constructor**

`MQA.TrafficControl(legendPosition)`

`legendPosition` is type `MQA.MapCornerPlacement` and specifies the initial position of the legend. It is optional. The default position is the bottom right corner.

The control's main button always goes in the upper left corner by default due to the construction of the graphic. It can be changed after construction by setting the "position" property.

### **Gettable & Settable Properties (using setValue/getValue)**

#### **position**

position: `MQA.MapCornerPlacement`      The position of the control's main activation button.

The line below will place the activation button in the upper most left corner of the map.

Example : `myTrafficCtrl.setValue('position', new MQA.MapCornerPlacement(MQA.MapCorner.TOP_LEFT, new MQA.Size(0,0)));`

#### **legendPosition**

legendPosition: `MQA.MapCornerPlacement`      The position of the legend. Since the legend is draggable, this is best used for setting up the starting position of the legend.

The line below will initially place the legend in the lower left corner of the map.

Example : `myTrafficCtrl.setValue('legendPosition', new MQA.MapCornerPlacement(MQA.MapCorner.BOTTOM_LEFT, new MQA.Size(0,0)));`

#### **legendIconImages**

legendIconImages: Array of `MQA.Icon`      Maximum 5-element array to specify what icons to use for the icon column of the legend.

Example:

```
var inclcons = new Array();
inclcons.push(new MQA.Icon("http://www.aolcdn.com/mqapi/mq00011",20,20));
inclcons.push(new MQA.Icon("http://www.aolcdn.com/mqapi/mq00012",20,20));
inclcons.push(new MQA.Icon("http://www.aolcdn.com/mqapi/mq00013",20,20));
inclcons.push(new MQA.Icon("http://www.aolcdn.com/mqapi/mq00014",20,20));
inclcons.push(new MQA.Icon("http://www.aolcdn.com/mqapi/mq00015",20,20));
myTrafficCtrl.setValue('legendIconImages',inclcons);
```

#### **legendIconText**

legendIconText: Array of Strings      Maximum 5-element array to specify what text to use to label the icons set by "legendIconImages." The array size must match the array set for legendIconImages."

Example:

```
var stext = new Array();
stext.push("Minimal0","Minimal1","Moderate2","Moderate3","Severe4");
myTrafficCtrl.setValue('legendIconText',stext);
```

#### **legendIconsVisible**

legendIconsVisible: Boolean      Specifies whether the icon part of the legend is visible.

Example: `myTrafficCtrl.setValue("legendIconsVisible",true);`

**legendFlowVisible**

legendFlowVisible: Boolean

Specifies whether the flow color part of the legend is visible

Example: `myTrafficCtrl.setValue("legendFlowVisible",true);`**legendCheckboxes**

legendCheckboxes: Array of Boolean

3-element array to specify which checkboxes to show for the control. The order is [Incident, Construction, Event]. True or False to specify which to show.

The following example will enable all three checkboxes for the control

Example: `myTrafficCtrl.setValue('legendCheckboxes',[true,true,true]);`**legendSliderVisible**

legendSliderVisible: Boolean

Specifies whether the flow opacity slider is visible.

Example: `myTrafficCtrl.setValue("legendSliderVisible",true);`**traffic**

traffic:Traffic Object

Used to access the Traffic object on the map that is associated with the Traffic Control.

Example: 

```
var myTraffic = new MQA.Traffic();
myTraffic = myTrafficCtrl.getValue('traffic');
```

**Events**

The Traffic Control events are handled the same as other events within the TileMap Toolkit. Utilize the MQA.EventManager to add/remove events.

<u>Listener</u>	<u>Event Name</u>
activate	MQA.TrafficControl.activate
deactivate	MQA.TrafficControl.deactivate
legendclose	MQA.TrafficControl.legendClose
legendopen	MQA.TrafficControl.legendOpen
refreshclick	MQA.TrafficControl.refreshClick
incidentclick	MQA.TrafficControl.incidentClick
constructionclick	MQA.TrafficControl.constructionClick
eventclick	MQA.TrafficControl.eventClick
helpclick	MQA.TrafficControl.helpClick