

## Operation Monitoring

Operation Monitoring API refers to any operations dealing with real time information. In this release we publish information about vehicle positions and about waiting times.

### 1. Vehicle Position

The operation returns real time vehicle position for given line id's

<b>Endpoint</b>	<b>https://opendata-api.stib-mivb.be/OperationMonitoring/1.0/VehiclePositionByLine/{line id's}</b>
<b>Method</b>	GET
<b>MimeType</b>	application/json
<b>Arguments</b>	a comma separated list of line id's, maximum 10 id's are allowed in the list.
<b>Returned value</b>	an array of "lines" and for each requested line, an array of "vehiclePositions"

#### 1.1. Sample of a request using Curl

- `curl -k -X GET --header "Accept: application/json" --header "Authorization: Bearer b2ba6c7a35d667564ffa2765aec6ea07" "https://opendata-api.stib-mivb.be/OperationMonitoring/1.0/VehiclePositionByLine/1"`

#### 1.2. Sample of a request using JQuery

```
var me = this;

// get Vehicle
Positions $.ajax({
    url: me.openDataBaseUrl + '/OperationMonitoring/1.0/VehiclePositionByLine/' + me.lineId,
    type: 'GET',
    error: function (jqXHR, textStatus)
    { // process error
    },
    beforeSend: function setHeader(xhr) {
        xhr.setRequestHeader('Accept', 'application/json');
        xhr.setRequestHeader('Authorization', 'Bearer ' + me.apiToken);
    // add the api token here
    },
    success: function (data) {
        // process the result here
        console.log('vehiclePosition: ' + data);
    },
}); // end of $.ajax({
```

#### 1.3. Sample of the returned value

```
{
  "lines": [
    {
      "lineId": 1,
      "vehiclePositions": [
        {
          "directionId": 8161,
          "distanceFromPoint": 0,
          "pointId": 8012
        },
        {
          "directionId": 8731,
```

```

        "distanceFromPoint":0,
        "pointId": 8021
    }
}
}
}
}

```

#### 1.4. Remarks

- ☐ Note that the position of a vehicle is given in the following format:
  - ☐ "directionId": the direction of the vehicle as the terminal "pointId"
  - ☐ "distanceFromPoint": the distance (in meters) covered by a vehicle since the last point represented by the "pointId" attribute
  - ☐ "pointId": the last Stop crossed by a vehicle
- ☐ Note about the Id's usage
  - ☐ The Line Id used as argument of the operation refers to the field "route\_short\_name" of the GTFS file "routes.txt"
  - ☐ The returned "directionId" and "pointId" refer to the fields "stop\_id" of the GTFS file "stops.txt"
- ☐ Life time
  - ☐ A vehicle position has a lifetime of about 20 seconds, therefore it is not necessary to poll the service at a higher frequency.
- ☐ The array of vehiclePositions may include some technical stops. These kind of stops are not published in the GTFS stops files, which is why you should ignore them.

## 2. Waiting Time

The operation returns the waiting times for the next two vehicles of each line passing through the requested stop id's.

<b>Endpoint</b>	<b><a href="https://opendata-api.stib-mivb.be/OperationMonitoring/1.0/PassingTimeByPoint/{point id's}">https://opendata-api.stib-mivb.be/OperationMonitoring/1.0/PassingTimeByPoint/{point id's}</a></b>
<b>Method</b>	GET
<b>MimeType</b>	application/json
<b>Arguments</b>	a comma separated list of Point Id's (a.k.a. Stop Id's), maximum 10 id's are allowed in the list
<b>Returned value</b>	an array of "points" and for each requested point, an array of "passingTimes"

#### 2.1. Sample of a request using Curl

- `curl -k -X GET --header "Accept: application/json" --header "Authorization: Bearer 30ca85ad55a0e384772b653e149e39a5" https://opendata-api.stib-mivb.be/OperationMonitoring/1.0/PassingTimeByPoint/8031`

#### 2.2. Sample of a request using JQuery

```

var me = this;

$.ajax({

```

```

        url: me.openDataBaseUrl + '/OperationMonitoring/1.0/
        PassingTimeByPoint/' + item.join("%2C"), // item is an array of
        id's, joined to get a comma separated list
        type: 'GET',
        error: function (jqXHR, textStatus)
            { // process error
            },
        beforeSend: function setHeader(xhr) { xhr.setRequestHeader('Accept',
            'application/json'); xhr.setRequestHeader('Authorization', 'Bearer
            ' + me.apiToken);
        },
        success: function (data) {
            // process the result here
            console.log('PassingTimeByPoint: ' + data);
        },
    })// end of $.ajax({

```

### 2.3.Sample of the returned value

```

{
  "points": [
    {
      "pointId": 8031,
      "passingTimes": [
        {
          "expectedArrivalTime": "2016-07-
          11T07:50:00", "lineId": 1
        },
        {
          "expectedArrivalTime": "2016-07-
          11T07:46:00", "lineId": 5
        },
        {
          "expectedArrivalTime": "2016-07-
          11T07:59:00", "lineId": 1
        },
        {
          "expectedArrivalTime": "2016-07-
          11T07:55:00", "lineId": 5
        }
      ]
    }
  ]
}

```

### 2.4.Remarks

- ☐ Note about the Id's usage
  - ☐ The Point id used as argument of the operation refers to the fields "stop\_id" of the GTFS file "stops.txt"
  - ☐ The returned "lineId" refers to the field "route\_short\_name" of the GTFS file "routes.txt"
- ☐ Life time
  - ☐ A vehicle position has a lifetime of about 20 seconds, therefore it is not necessary to poll the service at a higher frequency.