# BusTime® Developer API Version 3 Guide

**Revision 3.10**
**September 4, 2019**


**300 Crossways Park Drive**
**Woodbury, New York 11797**
**(516)433-6100 Phone**
**(516)433-5088 Fax**
**www.cleverdevices.com**

Clever Devices Ltd.
300 Crossways Park Drive
Woodbury, NY, USA 11797
Phone – (516) 433-6100
Fax – (516) 433-5088
www.cleverdevices.com

BusTime® Developer API Guide
Revision 3.10: 09/04/2019

# Contents

# 1 Overview

## 1.1 *What is the BusTime® Developer API?*

The BusTime® Developer API allows you to request and retrieve real-time data directly from BusTime®. Registered third-party developers can make HTTP requests for data and receive XML or JSON responses from the BusTime® web server.

## 1.2 *What data is available through the API?*

Data available through the API includes:

- Vehicle locations
- Route data (route lists, stop lists geo-positional route definitions, stop lists, etc.)
- Prediction Data
- Service Bulletins

## 1.3 *Will my application break if changes are made to the API?*

No. The versioning of the API allows time for developers to upgrade their applications to make use of new API features. Note that occasionally new parameters may be added to an existing request or its response. However, existing parameters will never be removed or stop accepting previously legal values.

Continuing to work with a particular version of the API guarantees that an application will not break. When a new version is released, it will offer new features and fixes that would break compatibility if added to the previous version. Using this method allows developers to continue using the same version in their current applications while working to make use of the new features of the next version.

## 1.4 *How does the Developer API work?*

In order to use the API, you must sign in to your BusTime® account and request an API key using the following steps.

- Create an account on the website.
- Sign into your account
- Select “My Account” from the top menu.
- Click on the “Developer API” link and fill out the form.

Only one key will be available per account. Once your request has been approved, an e-mail will be sent to you, containing the API key.

After receiving the key, you will be able to make calls to the API, entering the key as part of the data request.

## 1.5 *Is there a limit to the number of requests I can make to the Developer API?*

Yes. By default, one API key can make a maximum of 10,000 requests per day. If you believe that you will require more than 10,000 daily requests, you must request that the cap on your key be raised to handle the additional traffic.

## 1.6 *Is there support for different languages?*

Yes. A list of supported languages can be requested over the API, and each request can include the language to be used.

## 1.7 *How are external and multiple data feeds handled?*

If BusTime® is set up to support multiple prediction feeds, the developer API can be used to access to those feeds.

A list of supported feeds can be requested using the **`getrtpidatafeeds`** request. The name of the desired datafeed can be included as **`rtpidatafeed`** in Vehicles, Routes, Route Directions, Patterns, Stops, Predictions, and Service Bulletins requests. Note that some of these requests **require** an **`rtpidatafeed`** parameter when working within a system with multiple configured feeds (even if only one of those feeds is enabled). Other requests can be called without this parameter and doing so will expand the query across all feeds. See the reference for each specific call for information about how that call handles this parameter.

A system with multiple-configured feeds will also return an **`rtpidatafeed`** element in the response of some calls. See the reference for each specific call for information about this element.

A single-feed system will never show the **`rtpidatafeed`** element and will never require the **`rtpidatafeed`** parameter, so developers making use of a single-feed system's API do not have to concern themselves with data feeds.

Sample request of all feeds:

```
http://localhost:8080/bustime/api/v3/getrtpidatafeeds?key=89dj2he89d8j3j3ksjhdue93j
```

Sample response:

```
<bustime-response>
  <rtpidatafeed>
    <name>bustime</name>
    <source>Bus Tracker</source>
    <displayname>TA</displayname>
    <enabled>true</enabled>
    <visible>true</visible>
  </rtpidatafeed>
  <rtpidatafeed>
    <name>ac transit</name>
    <source>NEXTBUS</source>
    <displayname>actransit</displayname>
    <enabled>true</enabled>
    <visible>true</visible>
  </rtpidatafeed>
```

```
</bustime-response>
```

Sample request using external feed:

```
http://localhost:8080/bustime/api/v3/getroutes?key=89dj2he89d8j3j3ksjhdue93j
&rtpidatafeed=ac%20transit
```

Sample response:

```
<bustime-response>
  <route>
    <rt>1</rt>
    <rtnm>MONUMENT / CHURCH HILL</rtnm>
    <rtclr>#000000</rtclr>
    <rtdd>1</rtdd>
    <rtpidatafeed>ac transit</rtpidatafeed>
  </route>
  <route>
    <rt>2</rt>
    <rtnm>MONUMENT / CHURCH HILL</rtnm>
    <rtclr>#ff0000</rtclr>
    <rtdd>2</rtdd>
    <rtpidatafeed>ac transit</rtpidatafeed>
  </route>
 </bustime-response>
```

## 1.8 *How are dynamic changes to schedule data handled?*

Version 3 introduces some dynamic data which fundamentally changes the proper use of the API. Dynamic changes can be split into two categories: Detours and Disruption Management. Before these changes, it may have been sufficient for an application to request route data once during startup. If the API user wants to support dynamic changes, it is likely that the client will need to make repeated requests for route data such as stops and patterns.

Detours are temporary changes in pattern data. These new temporary patterns may add or remove stops from the original pattern that is being detoured. The client application should rely on the new **getdetours** call to retrieve detour data and present this data to the end user when detour changes are encountered throughout the API. Scheduled arrival times may also be affected by detours.

Disruption Management is a suite of actions which can change the route data of the schedule. Some examples are canceling arrivals and canceling, shifting, or expressing trips. The API represents these changes in existing calls (e.g. dynamic changes to predictions are shown in the predictions call) while some will be presented in a new API call (**getdynamicchanges**, not yet implemented). At the moment, only canceled arrivals are supported by the API.

# 2 Web Service

The BusTime® Developer API is a web service that uses HTTP/1.1 as its application protocol. Each type of call or request that can be made to the API is represented by a unique URL. Requests are made to the API using HTTP GET calls to the appropriate URL. Parameters are encoded in the HTTP GET request by following the URL with a "?" and "argument=value" pairs separated by "&".

A response is returned as a well-formed XML document with a Content-Type of "text/xml", or as a JSON document with a Content-Type of "application/json".

For example, to request the current system time through the developer API, a program or script will make a HTTP/1.1 GET request to the following URL with parameters:

**http://[host:port]/bustime/api/v3/gettime?key=89dj2he89d8j3j3ksjhdue93j**

The **[host:port]** is the host and port on which the Developer API is servicing HTTP requests. The port is not required if requests are being serviced on port 80.

The version of the API that is being accessed is built into the URL. In the above example, "**v3**" represents version 3.0 of the API.

The "**key**" parameter represents the API key assigned to the developer making the request. All requests to the API must be accompanied by a valid API key.

In Versions 2 and later, an optional "**format**" parameter can be included to specify the response type. XML is the default response format, and is used as the default if the "**format**" parameter is not included. JSON can be chosen by including "**format=json**".

**This document's reference only details information about Version 3.** For information about other versions of the API, review that version's document instead, as the request and response formats of different versions may not be compatible with one another.

# 3 Reference

This section describes all possible requests that can be made to the BusTime® Developer API. For every request, a complete set of possible arguments is specified, along with the response. For XML responses, the schema is specified.

***Definitions***

- **Delayed Vehicle** – The state entered by a vehicle when it has been determined to be stationary for more than a pre-defined time period.
- **Direction** – Common direction of travel of a route.
- **Format** – The document type of the response. Currently XML and JSON are supported.
- **Locale** – A string that represents the language to be used for the request. A list of valid locales can be retrieved using getlocalelist. They are in ISO form, such as "en", which would be English.
- **Off-route Vehicle** – State entered by a transit vehicle when it has strayed from its scheduled pattern.
- **Pattern** – A unique sequence of geo-positional points (waypoints and stops) that combine to form the path that a transit vehicle will repetitively travel. A route often has more than one possible pattern.
- **Route** – One or more set of patterns that together form a single service.
- **Service Bulletin** – Text-based announcements affecting a set of one or more services (route, stops, etc.).
- **Stop** – Location where a transit vehicle can pick-up or drop-off passengers. Predictions are only generated at stops.
- **Waypoint** – A geo-positional point in a pattern used to define the travel path of a transit vehicle.

## 3.1 *Common Parameters*

All request URLs have these parameters in common:

| Name | Supported Versions | Required? | Example | Description |
|---|---|---|---|---|
| **version** | All | Yes | /v3/ | The version of the API being used. Legal values are v1, v2, and v3. |
| **locale** | All | No | locale=en | The language that the response should be in. See the reference for "Locale" for more details on how to use this field. |
| **format** | v2+ | No | format=json | The format of the response. Legal values are "xml" and "json". XML is the default if no format is requested. |

## 3.2 *Time*

**Base URL: http://[host:port]/bustime/api/v3/gettime**

**Parameters**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **unixTime** | boolean (optional) | If true, returns the number of milliseconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970. |

**Response:**
A well-formed XML or JSON document, containing the current system time, will be returned as a response to **gettime**.

**Response Fields**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Contains a message if the processing of the request resulted in an error. |
| **tm** | Child element of the root element containing the current system date and time (local). Date and time are represented in the following format: YYYYMMDD HH:MM:SS. Month is represented as two digits where January is "01" and December is "12". Time is represented using a 24-hour clock.<br>If the param unixTime=true, returns the number of milliseconds that have elapsed since 00:00:00 Coordinated Universal Time (UTC), Thursday, 1 January 1970. |

**Remarks:**
Use the **gettime** request to retrieve the current system date and time. Since BusTime® is a time-

dependent system, it is important to synchronize your application with BusTime's system date and time.

This call is unchanged from v1. A JSON response requires v2 or higher.

The time given in the schema below is the local time.

**XML Schema:**

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="bustime-response" type=" bustime-response" />
	<xs:complexType name="bustime-response">
		<xs:sequence>
			<xs:element name="error" type="error" minOccurs="0"
			maxOccurs="unbounded"/>
			<xs:element name="tm" type="xs:string" minOccurs="0" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="error">
		<xs:sequence>
			<xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
</xs:schema>
```

**Example:**
The XML document below is a response to the following request:

**Request:**
http://localhost:8080/bustime/api/v3/gettime?key=89dj2he89d8j3j3ksjhdue93j

**Response:**

```
<?xml version="1.0"?>
<bustime-response>
	<tm>20160308 14:42:32</tm>
</bustime-response>
```

**Request:**

http://localhost:8080/bustime/api/v3/gettime?key=89dj2he89d8j3j3ksjhdue93j&format=json

**Response:**

```
{
	"bustime-response": {
		"tm": "20160308 14:51:54"
	}
}
```

**Request:**
http://localhost:8080/bustime/api/v3/gettime?key=Qskvu4Z5JDwGEVswqdAVkiA5B&unixTime=true

**Response:**

```
<?xml version="1.0"?>
<bustime-response>
        <tm>1531859957528</tm>
</bustime-response>
```

### Request:

http://localhost:8080/bustime/api/v3/gettime?key=Qskvu4Z5JDwGEVswqdAVkiA5B&unixTime=true&format=json

### Response:

```
{
        "bustime-response": {
                "tm": "1531860021189"
        }
}
```

## 3.3 *Vehicles*

**Base URL: http://[host:port]/bustime/api/v3/getvehicles**

***Parameters***

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **vid** | Comma-delimited list of vehicle IDs (not available with **rt** parameter) | Set of one or more vehicle IDs whose location should be returned. For example: 509,392,201,4367 will return information for four vehicles (if available). A maximum of 10 identifiers can be specified. |
| **rt** | Comma-delimited list of route designators (not available with the **vid** parameter) | A set of one or more route designators for which matching vehicles should be returned. For example: **X3,4,20** will return information for all vehicles currently running on those three routes (if available). A maximum of 10 identifiers can be specified. |
| **tmres** | string (optional) | Resolution of time stamps. Set to "s" to get time resolution to the second. Set to "m" to get time resolution to the minute. If omitted, defaults to "m".<br>Date and time is represented in the following format:<br>If specified as "s"<br>YYYYMMDD HH:MM:SS<br>If specified as "m"<br>YYYYMMDD HH:MM<br>Month is represented as two digits where January is equal to "01" and December is equal to "12". Time is represented using a 24-hour clock. |
| **rtpidatafeed** | (multi-feed only) string (optional) | Specify the name of the Real-Time Passenger Information data feed to retrieve vehicles for. If not given, results will span across all feeds. |

**Response:**
A well-formed XML or JSON document will be returned as a response to **getvehicles**. The response will include the most-recent status for each vehicle.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **vehicle** | Child element of the root element. Encapsulates all information available for a single vehicle in the response. |
| **vid** | Child element of the **vehicle** element. Alphanumeric string representing the vehicle ID (ie. bus number) |
| **rtpidatafeed** | (Multi-feed only) Child element of the **vehicle** element. The name of the data feed that the vehicle was retrieved from. |
| **tmstmp** | Child element of the **vehicle** element. Date and local time of the last positional update of the vehicle. Date and time is represented in the following format: YYYYMMDD HH:MM. Month is represented as two digits where January is equal to "01" and December is equal to "12". Time is represented using a 24-hour clock. |
| **lat** | Child element of the **vehicle** element. Latitude position of the vehicle in decimal degrees (WGS 84). |
| **lon** | Child element of the **vehicle** element. Longitude position of the vehicle in decimal degrees (WGS 84). |
| **hdg** | Child element of the **vehicle** element. Heading of vehicle as a 360º value, where 0º is North, 90º is East, 180º is South and 270º is West. |
| **pid** | Child element of the **vehicle** element. Pattern ID of trip currently being executed. |
| **pdist** | Child element of the **vehicle** element. Linear distance in feet that the vehicle has traveled into the pattern currently being executed. |
| **rt** | Child element of the **vehicle** element. Route that is currently being executed by the vehicle (ex. "20"). |
| **des** | Child element of the **vehicle** element. Destination of the trip being executed by the vehicle (ex. "Austin"). |
| **dly** | Child element of the **vehicle** element. The value is "true" if the vehicle is delayed. The **dly** element is only present if the vehicle is delayed. |
| **spd** | Child element of the **vehicle** element. Speed as reported from the vehicle expressed in miles per hour (MPH). |
| **tablockid** | Child element of the **vehicle** element. TA's version of the scheduled block identifier for the work currently being performed by the vehicle. |
| **tatripid** | Child element of the **vehicle** element. TA's version of the scheduled trip identifier for the vehicle's current trip. |
| **zone** | Child element of the **prd** element. The zone name if the vehicle has entered a defined zone, otherwise blank. |

| | |
|---|---|
| **mode** | Child element of the **vehicle** element. Mode of transportation for the vehicle as a byte with range 0-4. 0 is None, 1 is Bus, 2 is Ferry, 3 is Rail, and 4 is People_Mover. |
| **psgld** | Child element of the **vehicle** element. String representing the ratio of the current passenger count to the vehicle's total capacity. "FULL" indicates greater than or equal to 90% capacity, "EMPTY" indicates less than 10% capacity, and "HALF_EMPTY" indicates any capacity in between. It can also be "N/A" if the vehicle's passenger load is unknown. |
| **timepointid** | Child element of the **vehicle** element. Contains the timepoint id for the current stop for this vehicle. Only included if the Bustime property "gtfs.vehicle.stop.status.enabled" is true. |
| **sequence** | Child element of the **vehicle** element. Contains the sequence number of the current stop for this vehicle. Only included if the Bustime property "gtfs.vehicle.stop.status.enabled" is true. |
| **stopstatus** | Child element of the **vehicle** element. Integer representing the current stop status of this vehicle per GTFS Realtime's VehicleStopStatus: STOPPED_AT (0), INCOMING_AT (1), IN_TRANSIT_TO (2). Only included if the Bustime property "gtfs.vehicle.stop.status.enabled" is true. |
| **stopid** | Child element of the **vehicle** element. Contains the stop id for the current stop for this vehicle. Only included if the Bustime property "gtfs.vehicle.stop.status.enabled" is true. |

**Remarks:**
Use the **getvehicles** request to retrieve vehicle information (i.e., locations) of all or a subset of vehicles currently being tracked by BusTime.

Use the **vid** parameter to retrieve information for one or more vehicles currently being tracked.

Use the **rt** parameter to retrieve information for vehicles currently running one or more of the specified routes.

**Note:** The **vid** and **rt** parameters cannot be combined in one request. If both parameters are specified on a request to **getvehicles,** only the first parameter specified on the request will be processed.

**Note:** Data feeds with a source of "NEXTBUS" do not support this call. Feeds with the "SYNCROMATICS" source are configurable to support this call with the property 'rtpi.syncromatics.getvehicles.enabled' which by default is disabled.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="bustime-response" type="bustime-response"/>
        <xs:complexType name="bustime-response">
                <xs:sequence>
```

```
                <xs:element name="error" type="error" minOccurs="0"
                maxOccurs="unbounded"/>
                <xs:element name="vehicle" type="vehicle" minOccurs="0"
                maxOccurs="unbounded"/>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="error">
        <xs:sequence>
                <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                maxOccurs="1"/>
                <xs:element name="vid" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
                <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="vehicle">
        <xs:sequence>
                <xs:element name="vid" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                maxOccurs="1"/>
                <xs:element name="tmpstmp" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="lat" type="xs:double" minOccurs="1" maxOccurs="1"/>
                <xs:element name="lon" type="xs:double" minOccurs="1" maxOccurs="1"/>
                <xs:element name="hdg" type="xs:int" minOccurs="1" maxOccurs="1"/>
                <xs:element name="pid" type="xs:int" minOccurs="1" maxOccurs="1"/>
                <xs:element name="pdist" type="xs:int" minOccurs="1" maxOccurs="1"/>
                <xs:element name="rt" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="des" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="dly" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
                <xs:element name="srvtmstmp" type="xs:string" minOccurs="0"
                maxOccurs="1"/>
                <xs:element name="spd" type="xs:int" minOccurs="1" maxOccurs="1"/>
                <xs:element name="blk" type="xs:int" minOccurs="0" maxOccurs="1"/>
                <xs:element name="tablockid" type="xs:string" minOccurs="1"
                maxOccurs="1"/>
                <xs:element name="tatripid" type="xs:string" minOccurs="1"
                maxOccurs="1"/>
                <xs:element name="zone" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="mode" type="xs:byte" minOccurs="1" maxOccurs="1"/>
                <xs:element name="psgld" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
```

**Example:**
The XML document below is a response to the following request:

**Request:**
http://localhost:8080/bustime/api/v3/getvehicles?key=89dj2he89d8j3j3ksjhdue93j&vid=509,392

**Response:**

```
<?xml version="1.0"?>
<bustime-response>
        <vehicle>
                <vid>509</vid>
                <tmstmp>20160308 10:28</tmstmp>
                <lat>41.92124938964844</lat>
                <lon>-87.64849853515625</lon>
                <hdg>358</hdg>
                <pid>3630</pid>
                <pdist>5678</pdist>
                <rt>8</rt>
                <des>Waveland/Broadway</des>
                <spd>27</spd>
                <tablockid>2 -701</tablockid>
                <tatripid>108</tatripid>
                <zone>Bay 1</zone>
                <mode>1</mode>
```

```
                <psgld>EMPTY</psgld>
        </vehicle>
        <vehicle>
                <vid>392</vid>
                <tmstmp>20160308 10:28</tmstmp>
                <lat>41.91095733642578</lat>
                <lon-87.64120713719782</lon>
                <hdg>88</hdg>
                <pid>1519</pid>
                <pdist>11203</pdist>
                <rt>72</rt>
                <des>Clark</des>
                <spd>36</spd>
                <tablockid>3 -703</tablockid>
                <tatripid>108156</tatripid>
                <zone>Bay 1</zone>
                <mode>1</mode>
                <psgld>FULL</psgld>
        </vehicle>
</bustime-response>
```

**Example:**
The JSON document below is a response to the following request in a multi-feed system:

**Request:**
http://localhost:8080/bustime/api/v3/getvehicles?key=89dj2he89d8j3j3ksjhdue93j&vid=6438,1295&tmres=s&rtpidatafeed=bustime&format=json

**Response:**

```
{
        "bustime-response": {
                "vehicle": [
                        {
                                "vid": "1",
                                "rtpidatafeed": "bustime",
                                "tmstmp": "20160307 13:14",
                                "lat": "37.54381",
                                "lon": "-77.43878166666667",
                                "hdg": "308",
                                "pid": 1689,
                                "rt": "6",
                                "des": "BROAD WILLOW LAWN",
                                "pdist": 3481,
                                "dly": false,
                                "spd": 3,
                                "tatripid": "12",
                                "tablockid": "6-05",
                                "zone": "",
                                "mode": 1,
                                "psgld": "EMPTY"
                        },
                        {
                                "vid": "2",
                                "rtpidatafeed": "bustime",
                                "tmstmp": "20160307 13:14",
                                "lat": "37.55896532837857",
                                "lon": "-77.48567781754004",
                                "hdg": "294",
                                "pid": 1559,
                                "rt": "16",
                                "des": "GROVE BF",
                                "pdist": 20156,
                                "dly": false,
                                "spd": 5,
                                "tatripid": "12",
                                "tablockid": "16-02",
                                "zone": "",
                                "mode": 1,
```

```
                        "psgId": "FULL"
                    }
                ]
        }
}
```

## 3.4 *Routes*

**Base URL: http://[host:port]/bustime/api/v3/getroutes**

**Parameters**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **rtpidatafeed** | (multi-feed only) string (optional) | Specify the name of the Real-Time Passenger Information data feed to retrieve routes for. If not given, results will span across all feeds. |

**Response:**
A well-formed XML or JSON document will be returned as a response to **getroutes**.

**Response Fields**:

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **route**<br>JSON Array: **routes** | Child element of the root element. Encapsulates a route serviced by the system. |
| **rt** | Child element of the **route** element. Alphanumeric designator of a route (ex. “20” or “X20”). |
| **rtnm** | Child element of the **route** element. Common name of the route (ex. “Madison” for the 20 route). |
| **rtclr** | Child element of the **route** element. Color of the route line used in map (ex. "#ffffff") |
| **rtdd** | Child element of the **route** element. Language-specific route designator meant for display. |
| **rtpidatafeed** | (Multi-feed only) Child element of the **route** element. The name of the data feed that the route was retrieved from. |

**Remarks:**
Use the **getroutes** request to retrieve the set of routes serviced by the system.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="bustime-response" type="bustime-response"/>
        <xs:complexType name="bustime-response">
                <xs:sequence>
                        <xs:element name="error" type="error" minOccurs="0"
                        maxOccurs="unbounded"/>
                        <xs:element name="route" type="route" minOccurs="0"
                        maxOccurs="unbounded"/>
                </xs:sequence>
        </xs:complexType>
```

```
        <xs:complexType name="error">
                <xs:sequence>
                        <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                        maxOccurs="1"/>
                        <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="route">
                <xs:sequence>
                        <xs:element name="rt" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="rtnm" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="rtclr" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="rtdd" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                        maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
</xs:schema>
```

**Example:**
The XML document below is a response to the following request:

**Request**
http://localhost:8080/bustime/api/v3/getroutes?key=89dj2he89d8j3j3ksjhdue93j

**Response**

```
<?xml version="1.0"?>
<bustime-response>
        <route>
                <rt>1</rt>
                <rtnm>Indiana/Hyde Park</rtnm>
                <rtclr>#000000</rtclr>
                <rtdd>1</rtdd>
        </route>
        <route>
                <rt>2</rt>
                <rtnm>Hyde Park Express</rtnm>
                <rtclr>#dc78af</rtclr>
                <rtdd>2</rtdd>
        </route>
        <route>
                <rt>3</rt>
                <rtnm>King Drive</rtnm>
                <rtclr>#ff0000</rtclr>
                <rtdd>3</rtdd>
        </route>
        <route>
                <rt>X3</rt>
                <rtnm>King Drive Express</rtnm>
                <rtclr>#ffffff</rtclr>
                <rtdd>X3</rtdd>
        </route>
        ...
</bustime-response>
```

**Request**
http://localhost:8080/bustime/api/v3/getroutes?key=89dj2he89d8j3j3ksjhdue93j&rtpidatafeed=ExternalFeedName&format=json

**Response**

```
{
        "bustime-response": {
                "routes": [
                        {
                                "rt": "1",
                                "rtnm": "Pontiac - Dhu Varren
                                "rtdd": "1",
```

```
                        "rtclr": "#ffffff"
                        "rtpidatafeed": "ExternalFeedName"
                },
                {
                        "rt": "2",
                        "rtnm": "Pontiac - University",
                        "rtdd": "1",
                        "rtclr": "#dc78af"
                        "rtpidatafeed": "ExternalFeedName"
                },
                ...
        ]
    }
}
```


## 3.5 *Route Directions*

**Base URL: http://[host:port]/bustime/api/v3/getdirections**

***Parameters***

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **rt** | single route designator (required) | Alphanumeric designator of a route (ex. "20" or "X20") for which a list of available directions is to be returned. |
| **rtpidatafeed** | (multi-feed only) string (required) | Specify the name of the Real-Time Passenger Information data feed to retrieve route directions for. |

**Response:**
A well-formed XML or JSON document will be returned as a response to **getdirections**.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **dir**<br>Json Array:<br>**directions** | Child element of the root element. Encapsulates a route's direction serviced by the system. |
| **id** | Child element of the **dir** element. This is the direction designator that should be used in other requests such as getpredictions. |
| **name** | Child element of the **dir** element. This is the human-readable, locale-dependent name of the direction. |

**Remarks:**
Use the **getdirections** request to retrieve the set of directions serviced by the specified route.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="bustime-response" type="bustime-response"/>
        <xs:complexType name="bustime-response">
```

```
                <xs:sequence>
                        <xs:element name="error" type="error" minOccurs="0"
                        maxOccurs="unbounded"/>
                        <xs:element name="dir" type="dir" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="error">
                <xs:sequence>
                        <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                        maxOccurs="1"/>
                        <xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="dir">
                <xs:sequence>
                        <xs:element name="id" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
</xs:schema>
```

**Example:**
The XML document below is a response to the following request:

## Request

http://localhost:8080/bustime/api/v3/getdirections?key=89dj2he89d8j3j3ksjhdue93j&rt=20&rtpidatafeed=acmeta

## Response

```
<?xml version="1.0"?>
<bustime-response>
        <dir>
                <id>FLEX_0_1</id>
                <name>West toward Town Square</title>
        </dir>
        <dir>
                <id>FLEX_0_2</id>
                <name>East toward Downtown</title>
        </dir>
</bustime-response>
```

## Request

http://localhost:8080/bustime/api/v3/getdirections?key=89dj2he89d8j3j3ksjhdue93j&rt=20&format=json&rtpidatafeed=acmeta

## Response

```
{
        "bustime-response": {
                "directions": [
                        {
                                "id": "FLEX_0_1",
                                "name": "West toward Town Square"
                        },
                        {
                                "id": "FLEX_0_2",
                                "name": "East toward Downtown"
                        }
                ]
        }
}
```

## 3.6 *Stops*

**Base URL: http://[host:port]/bustime/api/v3/getstops**

**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **rt** | single route designator (required if stpid is not provided) | Alphanumeric designator of the route (ex. “20” or “X20”) for which a list of available stops is to be returned. |
| **dir** | single route direction (required if stpid is not provided) | Direction of the route (ex. “East Bound”) for which a list of available stops is to be returned.<br>This needs to match the direction’s id in the **getdirections** call. |
| **stpid** | comma-delimited list of stop ids (required if rt and dir are not provided) | Numeric ID number for a specific stop (ex. "305") for which a single stop is to be returned. Can send up to 10 stop parameters. |
| **rtpidatafeed** | (multi-feed only) string (required) | Specify the name of the Real-Time Passenger Information data feed to retrieve stops for. |

**Response:**
A well-formed XML or JSON document will be returned as a response to **getstops**.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **stop**<br>JSON Array: **stops** | Child element of the root element. Encapsulates all descriptive information about a particular stop. |
| **stpid** | Child element of the **stop** element. Unique identifier representing this stop. |
| **stpnm** | Child element of the **stop** element. Display name of this stop (ex. “Madison and Clark”) |
| **lat** | Child element of the **stop** element. Latitude position of the stop in decimal degrees (WGS 84). |
| **lon** | Child element of the **stop** element. Longitude position of the stop in decimal degrees (WGS 84). |
| **dtradd** | Child element of the **stop** element. A list of detour ids which add (temporarily service) this stop. |
| **dtrrem** | Child element of the **stop** element. A list of detour ids which remove (detour around) this stop. |

**Remarks:**
Use the **getstops** request to retrieve the set of stops for the specified route and direction. A request must provide either a **rt & dir** or up to 10 **stpid**s, but not both.

Stop lists are only available for a valid route/direction pair. In other words, a list of all stops that service a particular route (regardless of direction) cannot be requested.

**XML Schema**:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="bustime-response" type="bustime-response"/>
  <xs:complexType name="bustime-response">
    <xs:sequence>
      <xs:element name="error" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="stop" type="stop" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="error">
    <xs:sequence>
      <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="dir" type="xs:string" minOccurs="0" maxOccurs="1"/>
      <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="stop">
    <xs:sequence>
      <xs:element name="stpid" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="stpnm" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="lat" type="xs:double" minOccurs="1" maxOccurs="1"/>
      <xs:element name="lon" type="xs:double" minOccurs="1" maxOccurs="1"/>
      <xs:element name="dtradd" type="xs:int" minOccurs="0" maxOccurs="unbounded"/>
      <xs:element name="dtrrem" type="xs:int" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

**Example:**
The XML document below is a response to the following request:

**Request**
http://localhost:8080/bustime/api/v3/getstops?key=89dj2he89d8j3j3ksjhdue93j&rt=20&dir=East%20Bound

**Response**

```
<?xml version=”1.0”?>
<bustime-response>
        <stop>
                <stpid>4727</stpid>
                <stpnm>1633 W Madison</stpnm>
                <lat>41.881265</lat>
                <lon>-87.66849</lon>
        </stop>
        <stop>
                <stpid>100123</stpid>
                <stpnm>Temporary stop on Austin</stpnm>
                <lat>41.885206667</lat>
                <lon>-87.7748733333333</lon>
                <dtradd>0F0119D3-9E18-4B72-9532-CA00C3C68022</dtradd>
        </stop>
        <stop>
                <stpid>9605</stpid>
                <stpnm>Austin & Randolph/West End</stpnm>
                <lon>41.8838633333333</lon>
```

```
                <lat>-87.7748566666667</lat>
        </stop>
        <stop>
                <stpid>9603</stpid>
                <stpnm>Austin & South Blvd/Corcoran</stpnm>
                <lat>41.886908333</lat>
                <lon>-87.77493667</lon>
        </stop>
        ...
</bustime-response>
```

## Request

http://localhost:8080/bustime/api/v3/getstops?key=89dj2he89d8j3j3ksjhdue93j&rt=20&dir=East%20Bound&format=json

## Response

```
{
        "bustime-response": {
                "stops": [
                        {
                                "stpid": "1577",
                                "stpnm": "1509 S Michigan",
                                "lat": 41.861706666665,
                                "lon": -87.623969999999
                        },
                        {
                                "stpid": "1564",
                                "stpnm": "3000 S Michigan",
                                "lat": 41.840606666667,
                                "lon": -87.623206666667
                                "dtrrem": [
                                                "BFC46F62-990F-4AB4-A85C-3AF84574EC99",
                                                "50C633C7-0891-4E5A-83A8-FF0C6214BF69"
                                        ]

                        },
                        ...
                ]
        }
}
```

## 3.7 *Patterns*

**Base URL: http://[host:port]/bustime/api/v3/getpatterns**

**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **pid** | comma-delimited list of pattern IDs (not available with **rt** parameter) | Set of one or more pattern IDs whose points should be returned. For example: 56,436,122 will return points from three (3) patterns. A maximum of 10 identifiers can be specified. |
| **rt** | single route designator (not available with **pid** parameter) | Route designator for which all active patterns should be returned. |
| **rtpidatafeed** | (multi-feed only) string (required) | Specify the name of the Real-Time Passenger Information data feed to retrieve patterns for. |

**Response:**

A well-formed XML or JSON document will be returned as a response to **getpatterns**.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **ptr** | Child element of the root element. Encapsulates a set of points which define a pattern. |
| **pid** | Child element of the **ptr** element. ID of pattern. |
| **ln** | Child element of the **ptr** element. Length of the pattern in feet. |
| **rtdir** | Child element of the **ptr** element. Direction that is valid for the specified route designator. For example, “INBOUND”. This needs to match the direction id seen in the getdirections call. |
| **pt** | Child element of the **ptr** element. Child element of the root element. Encapsulates one a set of geo-positional points (including stops) that when connected define a pattern. |
| **seq** | Child element of the **pt** element. Position of this point in the overall sequence of points. |
| **typ** | Child element of the **pt** element. ‘S’ if the point represents a Stop, ‘W’ if the point represents a waypoint along the route. |
| **stpid** | Child element of the **pt** element. If the point represents a stop, the unique identifier of the stop. |
| **stpnm** | Child element of the **pt** element. If the point represents a stop, the display name of the stop. |
| **pdist** | Child element of the **pt** element. If the point represents a stop, the linear distance of this point (feet) into the requested pattern. |

| | |
|---|---|
| **lat** | Child element of the **pt** element. Latitude position of the point in decimal degrees (WGS 84). |
| **lon** | Child element of the **pt** element. Longitude position of the point in decimal degrees (WGS 84). |
| **dtrid** | Child element of the **ptr** element. If this pattern was created by a detour, contains the id of the detour. Does not appear for normal patterns. |
| **dtrpt** | Child element of the **ptr** element. If this pattern was created by a detour, encapsulates a set of geo-positional points that represent the *original* pattern. Useful for drawing dashed lines on a map. |

**Remarks:**

Use the **getpatterns** request to retrieve the set of geo-positional points and stops that when connected can be used to construct the geo-positional layout of a pattern (i.e., route variation).

Use **pid** to specify one or more identifiers of patterns whose points are to be returned. A maximum of 10 patterns can be specified.

Use **rt** to specify a route identifier where all active patterns are returned. The set of active patterns returned includes: one or more patterns marked as "default" patterns for the specified route and all patterns that are currently being executed by at least one vehicle on the specified route.

> **Note:** The **pid** and **rt** parameters cannot be combined in one request. If both parameters are specified on a request to **getpatterns**, only the first parameter specified on the request will be processed.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="bustime-response" type="bustime-response"/>
	<xs:complexType name="bustime-response">
		<xs:sequence>
			<xs:element name="error" type="error" minOccurs="0"
			maxOccurs="unbounded"/>
			<xs:element name="ptr" type="ptr" minOccurs="0" maxOccurs="10"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="error">
		<xs:sequence>
			<xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
			maxOccurs="1"/>
			<xs:element name="pid" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="ptr">
		<xs:element name="pid" type="xs:int" minOccurs="1" maxOccurs="1"/>
		<xs:element name="ln" type="xs:int" minOccurs="1" maxOccurs="1"/>
		<xs:element name="rtdir" type="xs:string" minOccurs="1" maxOccurs="1"/>
		<xs:element name="pt" type="pt" minOccurs="1" maxOccurs="unbounded"/>
		<xs:element name="dtrid" type="xs:string" minOccurs="0" maxOccurs="1"/>
		<xs:element name="dtrpt" type="pt" minOccurs="0" maxOccurs="unbounded"/>
	</xs:complexType>
	<xs:complexType name="pt">
```

```
                <xs:sequence>
                        <xs:element name="seq" type="xs:int" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="typ" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="stpid" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="stpnm" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="pdist" type="xs:float" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="lat" type="xs:double" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="lon" type="xs:double" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
</xs:schema>
```

**Example:**
The XML document below is a response to the following request:

**Request**
http://localhost:8080/bustime/api/v3/getpatterns?key=89dj2he89d8j3j3ksjhdue93j&rt=20&pid=954

**Response**

```
<?xml version="1.0"?>
<bustime-response>
        <ptr>
                <pid>954</pid>
                <ln>35569</ln>
                <rtdir>INBOUND</rtdir>
                <pt>
                        <seq>1</seq>
                        <typ>S</typ>
                        <stpid>409</stpid>
                        <stpnm>Madison & Pulaski</stpnm>
                        <lat>41.880641167057</lat>
                        <lon>-87.725835442543</lon>
                        <pdist>0.0</pdist>
                </pt>
                <pt>
                        <seq>2</seq>
                        <typ>W</typ>
                        <lat>41.880693089146</lat>
                        <lon>-87.725765705109</lon>
                </pt>
                <pt>
                        <seq>3</seq>
                        <typ>W</typ>
                        <lat>41.880693089146</lat>
                        <lon>-87.725674510002</lon>
                        <pdist>97.0</pdist>
                </pt>
                ...
        </ptr>
</bustime-response>
```

**Request**
http://localhost:8080/bustime/api/v3/getpatterns?key=89dj2he89d8j3j3ksjhdue93j&rtpidatafeed=bustime&rt=20&pid=954&format=json

**Response**

```
{
        "bustime-response": {
                "ptr": [
                        {
                                "pid": 1146,
                                "ln": 42608.0,
                                "rtdir": "EAST",
                                "pt": [
                                        {
                                                "seq": 1,
```

```
                                        "lat": 37.537591575456,
                                        "lon": -77.472311666667,
                                        "typ": "S",
                                        "stpid": "1697",
                                        "stpnm": "Meadow + Colorado",
                                        "pdist": 0.0
                                    },
                                    {
                                        "seq": 2,
                                        "lat": 37.536418242205,
                                        "lon": -77.472629999998,
                                        "typ": "S",
                                        "stpid": "1699",
                                        "stpnm": "Meadow + Dakota",
                                        "pdist": 440.0
                                    }
                                ]
                            }
                        ]
                    }
}
```


## 3.8 *Predictions*

**Base URL: http://[host:port]/bustime/api/v3/getpredictions**

**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **stpid** | comma-delimited list of stop IDs (not available with **vid** parameter) | Set of one or more stop IDs whose predictions are to be returned. For example: 5029,1392,2019,4367 will return predictions for the four stops. A maximum of 10 identifiers can be specified. |
| **rt** | comma-delimited list of route designators (optional, available with **stpid** parameter) | Set of one or more route designators for which matching predictions are to be returned. |
| **vid** | comma-delimited list of vehicle IDs (not available with **stpid** parameter) | Set of one or more vehicle IDs whose predictions should be returned. For example: 509,392,201,4367 will return predictions for four vehicles. A maximum of 10 identifiers can be specified. |
| **top** | number (optional) | Maximum number of predictions to be returned. |
| **tmres** | string(optional) | Resolution of time stamps. Set to “s” to get time resolution to the second. Set to “m” to get time resolution to the minute. If omitted, defaults to “m”.<br>Date and time is represented in the following format:<br>If specified as “s”<br>YYYYMMDD HH:MM:SS<br>If specified as “m” |

| | | |
|---|---|---|
| | | YYYYMMDD HH:MM<br>Month is represented as two digits where January is equal to "01" and December is equal to "12". Time is represented using a 24-hour clock. |
| **rtpidatafeed** | (multi-feed only)<br>string (required) | Specify the name of the Real-Time Passenger Information data feed to retrieve predictions for. |

**Response:**
A well-formed XML or JSON document will be returned as a response to **getpredictions**.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **prd** | Child element of the root element. Encapsulates a predicted arrival or departure time for the specified set of stops or vehicles. |
| **tmstmp** | Child element of the **prd** element. Date and time (local) the prediction was generated. Date and time is represented based on the tmres parameter. |
| **typ** | Child element of the **prd** element. Type of prediction. 'A' for an arrival prediction (prediction of when the vehicle will arrive at this stop). 'D' for a departure prediction (prediction of when the vehicle will depart this stop, if applicable). Predictions made for first stops of a route or layovers are examples of departure predictions. |
| **stpid** | Child element of the **prd** element. Unique identifier representing the stop for which this prediction was generated. |
| **stpnm** | Child element of the **prd** element. Display name of the stop for which this prediction was generated. |
| **vid** | Child element of the **prd** element. Unique ID of the vehicle for which this prediction was generated. |
| **dstp** | Child element of the **prd** element. Linear distance (feet) left to be traveled by the vehicle before it reaches the stop associated with this prediction. |
| **rt** | Child element of the **prd** element. Alphanumeric designator of the route (ex. "20" or "X20") for which this prediction was generated. |
| **rtdd** | Child element of the **prd** element. Language-specific route designator meant for display. |
| **rtdir** | Child element of the **prd** element. Direction of travel of the route associated with this prediction (ex. "INBOUND"). This matches the direction id seen in the getdirections call. |
| **des** | Child element of the **prd** element. Final destination of the vehicle associated with this prediction. |
| **prdtm** | Child element of the **prd** element. Predicted date and time (local) of |

| | |
|---|---|
| | a vehicle's arrival or departure to the stop associated with this prediction. Date and time is represented based on the tmres parameter. |
| **Name** | **Description** |
| **dly** | Child element of the **prd** element. "true" if the vehicle is delayed. In version 3 this element is always present. This is not used by RTPI feeds. |
| **dyn** | Child element of the **prd** element. The dynamic action type affecting this prediction:<br>0 = None. No dynamic change.<br>1 = Canceled. The vehicle will not make a stop for this prediction.<br>3 = Shifted. The scheduled arrival time has changed.<br>4 = Expressed. The vehicle will only stop here at a rider's request. (Drop-off only/No pickup) |
| **tablockid** | Child element of the **prd** element. TA's version of the scheduled block identifier for the work currently being performed by the vehicle. |
| **tatripid** | Child element of the **prd** element. TA's version of the scheduled trip identifier for the vehicle's current trip. |
| **prdctdn** | Child element of the **prd** element. This is the time left, in minutes, until the bus arrives at this stop. |
| **zone** | Child element of the **prd** element. The zone name if the vehicle has entered a defined zones, otherwise blank.<br>This is not used by RTPI feeds. |
| **nbus** | Child element of the **prd** element. If this prediction is the last arrival (for this route) before a service gap, this represents the number of minutes until the next scheduled bus arrival (from the prediction time). |

**Remarks:**

Use the **getpredictions** request to retrieve predictions for one or more stops or one or more vehicles. Predictions are always returned in ascending order according to **prdtm**.

Use the **vid** parameter to retrieve predictions for one or more vehicles currently being tracked. A maximum of 10 vehicles can be specified.

Use the **stpid** parameter to retrieve predictions for one or more stops. A maximum of 10 stops can be specified.

> **Note:** The **vid** and **stpid** parameters cannot be combined in one request. If both parameters are specified on a request to **getpredictions,** only the first parameter specified on the request will be processed.

Calls to **getpredictions** without specifying the **vid** or **stpid** parameters are not allowed.

Use the **top** parameter to specify the maximum number of predictions to return. If **top** is not specified, then all predictions matching the specified parameters will be returned.

**nBus** only appears if the Transit Authority has the service gap feature enabled. If **nBus** would have a value less than the configured minimum gap of time (default 120 minutes), the element is empty. If **nBus** is "-1", then this prediction is the last bus of the day for this route.

### XML Schema

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="bustime-response" type="bustime-response"/>
	<xs:complexType name="bustime-response">
		<xs:sequence>
			<xs:element name="error" type="error" minOccurs="0"
			maxOccurs="unbounded"/>
			<xs:element name="prd" type="prediction" minOccurs="0"
			maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="error">
		<xs:sequence>
			<xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
			maxOccurs="1"/>
			<xs:element name="stpid" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="vid" type="xs:string" minOccurs="0" maxOccurs="1"/>
			<xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="prediction">
		<xs:all>
			<xs:element name="tmstmp" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="typ" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="stpid" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="stpnm" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="vid" type="xs:int" minOccurs="1" maxOccurs="1"/>
			<xs:element name="dstp" type="xs:int" minOccurs="1" maxOccurs="1"/>
			<xs:element name="rt" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="rtdd" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="rtdir" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="des" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="prdtm" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="dly" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
			<xs:element name="dyn" type="xs:byte" minOccurs="1" maxOccurs="1"/>
			<xs:element name="tablockid" type="xs:string" minOccurs="1"
			maxOccurs="1"/>
			<xs:element name="tatripid" type="xs:string" minOccurs="1"
			maxOccurs="1"/>
			<xs:element name="zone" type="xs:string" minOccurs="1" maxOccurs="1"/>
			<xs:element name="nbus" type="xs:string" minOccurs="0" maxOccurs="1"/>
		</xs:all>
	</xs:complexType>
</xs:schema>
```

### Example:

The XML document below is a response to the following request:

**Request**

http://localhost:8080/bustime/api/v3/getpredictions?key=89dj2he89d8j3j3ksjhdue93j&rt=20&stpid=456

**Response**

```
<?xml version="1.0"?>
<bustime-response>
```

```
        <tm></tm>
        <prd>
                <tmstmp>20090611 14:34</tmstmp>
                <typ>A</typ>
                <stpid>456</stpid>
                <stpnm>Madison & Jefferson</stpnm>
                <vid>2013</vid>
                <dstp>891</dstp>
                <rt>20</rt>
                <rtdd>20</rtdd>
                <rtdir>West Bound</rtdir>
                <des>Austin</des>
                <prdtm>20090611 14:40</prdtm>
                <tablockid>3 -701</tablockid>
                <tatripid>106</tatripid>
                <zone></zone>

        </prd>
        <prd>
                <tmstmp>20090611 14:34</tmstmp>
                <typ>A</typ>
                <stpid>456</stpid>
                <stpnm>Madison & Jefferson</stpnm>
                <vid>6435</vid>
                <dstp>1587</dstp>
                <rt>20</rt>
                <rtdd>20</rtdd>
                <rtdir>West Bound</rtdir>
                <des>Austin</des>
                <prdtm>20090611 14:48</prdtm>
                <tablockid>3 -706</tablockid>
                <tatripid>108</tatripid>
                <zone>Bay 1</zone>
        </prd>
</bustime-response>
```

## Request

http://localhost:8080/bustime/api/v3/getpredictions?key=89dj2he89d8j3j3ksjhdue93j&rt=20&stpid=456&format=json

## Response

```
{
        "bustime-response": {
                "prd": [
                        {
                                "tmstmp": "20130104 15:00",
                                "typ": "A",
                                "stpnm": "87th Street \u0026 Wentworth",
                                "stpid": "9405",
                                "vid": "",
                                "dstp": 0,
                                "rt": "87",
                                "rtdd": "87",
                                "rtdir": "INBOUND",
                                "des": "91st/Commercial",
                                "prdtm": "20130104 15:08",
                                "tablockid": "87 -706",
                                "tatripid": "1007569",
                                "dly": false,
                                "prdctdn": "8",
                                "zone": ""
                        },
                        ...
                ]
        }
}
```

## 3.9 *Service Bulletins*

**Base URL: http://[host:port]/bustime/api/v3/getservicebulletins**

**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **rt** | comma-delimited list of route designators (required if **stpid** not specified) | Alphanumeric designator of the route(s) (ex. "20" or "X20") for which a list of service bulletins is to be returned. If combined with **rtdir**, only one route can be specified. |
| **rtdir** | single route direction (optional) | Direction of travel of the route specified in the **rt** parameter. The **rt** parameter is required when using the **rtdir** parameter. This needs to match the direction id seen in the getdirections call. |
| **stpid** | comma-delimited list of stop IDs (required if **rt** not specified) | Set of one or more stop IDs for which service bulletins are to be returned. For example: 5029,1392,2019,4367 will return predictions for the four stops (if available). If combined with **rt** and **rtdir**, only one stop can be specified. |
| **rtpidatafeed** | (multi-feed only) string (required) | Specify the name of the Real-Time Passenger Information data feed to retrieve service bulletins for. |

**Response:**

A well-formed XML or JSON document will be returned as a response to **getservicebulletins**.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **sb** | Child element of the root element. Encapsulates all data about a service bulletin. |
| **nm** | Child element of the **sb** element. Unique name/identifier of the service bulletin. |
| **sbj** | Child element of the **sb** element. Service bulletin subject. A short title for this service bulletin. |
| **dtl** | Child element of the **sb** element. Service bulletin detail. Full text of the service bulletin. |
| **brf** | Child element of the **sb** element. Service bulletin brief. A short text alternative to the service bulletin detail. |
| **cse** | Child element of the **sb** element. Cause for service bulletin. |
| **efct** | Child element of the **sb** element. Effect for service bulletin. |

| | |
|---|---|
| **prty** | Child element of the **sb** element. Service bulletin priority. The possible values are "High," "Medium," and "Low". |
| **Name** | **Description** |
| **rtpidatafeed** | (multi-feed only) Child element of the **sb** element. The name of the data feed that the service bulletin affects. If the **rtpidatafeed** element is empty, the service bulletin affects the entire system. |
| **srvc** | Child element of the **sb** element. Each **srvc** element represents one or a combination of route, direction and stop for which this service bulletin is valid. If the **srvc** element is empty, the service bulletin affects all routes and stops of its feed. |
| **rt** | Child element of **srvc**. Alphanumeric designator of the route (ex. "20" or "X20") for which this service bulletin is in effect. |
| **rtdir** | Child element of **srvc**. Direction of travel of the route for which this service bulletin is in effect. This matches the direction id seen in the getdirections call. |
| **stpid** | Child element of **srvc**. ID of the stop for which this service bulletin is in effect. |
| **stpnm** | Child element of **srvc**. Name of the stop for which this service bulletin is in effect. |
| **mod** | The date/time of the last service bulletin modification in local time zone in YYYYMMDD HH:MM:SS format |
| **url** | Child element of the **sb** element. Contains URL to site with additional information about this service bulletin. |

**Remarks:**

Use the **getservicebulletins** for a list of service bulletins that are in effect for a route(s) (**rt**), route & direction (**rt & rtdir**), route & direction & stop (**rt & rtdir & stpid**), or stop(s) (**stpid**).

**Note:** At a minimum, the **rt** or **stpid** parameter must be specified.

A service bulletin (**sb**) definition without a **srvc** element indicates a "feed-wide" service bulletin. A service bulletin (**sb**) definition without a **srvc** *and* without a **rtpidatafeed** element indicates a "system-wide" service bulletin. System-wide service bulletins are valid for all routes/stops in the system, while feed-wide bulletins only affects routes/stops of that feed.

**Note:** Data feeds with a source of "NEXTBUS" do not support this call.

The service bulletin detail field (**dtl**) may contain html tags such as `<b>` or `<a href...>` which should be supported by the developer.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <xs:element name="bustime-response" type="bustime-response"/>
        <xs:complexType name="bustime-response">
                <xs:sequence>
                        <xs:element name="error" type="error" minOccurs="0"
                        maxOccurs="unbounded"/>
                        <xs:element name="sb" type="servicebulletin" minOccurs="1"
                        maxOccurs="unbounded"/>
                </xs:sequence>
```

```
        </xs:complexType>
        <xs:complexType name="error">
                <xs:sequence>
                        <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                        maxOccurs="1"/>
                        <xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="rtdir" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="stpid" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="servicebulletin">
                <xs:sequence>
                        <xs:element name="nm " type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="sbj" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="dtl" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="brf" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="cse" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="efct" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="prty" type="xs:string" minOccurs="1" maxOccurs="1"/>
                        <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0"
                        maxOccurs="1"/>
                        <xs:element name="srvc" type="affectedservice" minOccurs="0"
                        maxOccurs="unbounded"/>
                        <xs:element name="lastModified" type="xs:string" minOccurs="1"
                        maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="affectedservice">
                <xs:sequence>
                        <xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="rtdir" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="stpid" type="xs:string" minOccurs="0" maxOccurs="1"/>
                        <xs:element name="stpnm" type="xs:string" minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
</xs:schema>
```

### Example:

The XML document below is a response to the following request:

### Request:

http://localhost:8080/bustime/api/v3/getservicebulletins?key=89dj2he89d8j3j3ksjhdue93j&stpid=456

### Response:

```
<?xml version=”1.0”?>
<bustime-response>
        <sb>
                <sbj>Stop Relocation</sbj>
                <dtl>The westbound stop located at Madison/Lavergne has been moved to the
northeast corner at Madison/Lavergne.</dtl>
                <brf> The westbound stop located at Madison/Lavergne has been moved to the
northeast corner at Madison/Lavergne.</brf>
                <prty>low</prty>
                <srvc>
                        <rt>20</rt>
                        <rtdir/>
                        <stpid/>
                        <stpnm/>
                </srvc>
                <mod>20171218 15:22:29</mod>
        </sb>
        <sb>
                <sbj>Stop Relocations/Eliminations</sbj>
                <dtl>Bus stops are being changed to provide faster travel time.</dtl>
                <brf>Bus stops are being changed to provide faster travel time.</brf>
                <prty>low</prty>
```

```
                <srvc>
                        <rt/>
                        <rtdir/>
                        <stpid>456</stpid>
                        <stpnm>1ST & 5TH</stpnm>
                </srvc>
                <mod>20171218 15:19:17</mod>
        </sb>
</bustime-response>
```

**Request:**
http://localhost:8080/bustime/api/v3/getservicebulletins?key=89dj2he89d8j3j3ksjhdue93j&rtpidatafeed=ExternalFeedName&stpid=456&format=json

**Response:**

```
{
        "bustime-response": {
                "sb": [
                        {
                                "nm": "System Wide",
                                "sbj": "Sys Wide English",
                                "dtl": "Sys Wide English",
                                "brf": "Sys Wide English",
                                "prty": "Low",
                                "rtpidatafeed": "",
                                "srvc": [],
                                "mod": "20171218 15:22:29"
                        },
                        {
                                "nm": "Route 1 East",
                                "sbj": "Route 1 East Delays",
                                "dtl": "Route 1 has service delays on the East branches",
                                "brf": "R1 East DELAYED",
                                "prty": "Low",
                                "rtpidatafeed": "ExternalFeedName",
                                "srvc": [
                                        {
                                                "rt": "1",
                                                "rtdir": "EAST",
                                                "stpid": "",
                                                "stpnm": ""
                                        }
                                ],
                                "mod": "20171218 15:19:17"
                        }
                ]
        }
}
```

## 3.10 *Locales*

**Base URL: http://[host:port]/bustime/api/v3/getlocalelist**

**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **locale** | string(optional) | The language to use for the response. Must match a supported locale id – See **localestring** below |
| **inLocaleLanguage** | boolean (optional) | Gets each locale with their display names in the native language of the locale when true. If omitted, defaults to false. |

| | | |
|---|---|---|
| | | When true, this parameter takes precedence over the setting of the 'locale' parameter. |

**Response:**
A well-formed XML or JSON document will be returned as a response to **getlocalelist**.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **locale** | Child element of the root element. Encapsulates all data about a locale (language). |
| **localestring** | Child element of the **locale** element. Unique name/identifier of the locale. This is what is passed as the locale parameter in all API calls.<br>The **localestring** contains an ISO 639 language code. Examples are "es". |
| **displayname** | Child element of the **locale** element. The name of the language. If the locale parameter was included, then this will be in that language. For human-readable use only. If the inLocaleLanguage parameter was true, then this will be in the language of the locale that it represents. |

**Remarks:**

Use the **getlocalelist** to get a list of what languages can be used as the locale parameter. It can be called a second time with a locale parameter that matches one of the previously returned localestrings to see the human-readable language names in that given language.

> **Note:** The locale parameter in all requests is meant to match values in this list, but it does support the inheritance model of Java Locale. If the given language is not supported then the default language of the Transit Authority is used. No indication of which language used is given in the response, so it is best to use a locale string out of the list returned by **getlocalelist**.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="bustime-response" type="bustime-response"/>
	<xs:complexType name="bustime-response">
		<xs:sequence>
			<xs:element name="error" type="error" minOccurs="0"
			maxOccurs="unbounded"/>
			<xs:element name="locale" type="locale" minOccurs="1"
			maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
	<xs:complexType name="error">
		<xs:sequence>
```

```
                        <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
        <xs:complexType name="locale">
                <xs:sequence>
                        <xs:element name="localestring" type="xs:string" minOccurs="1"
                        maxOccurs="1"/>
                        <xs:element name="displayname" type="xs:string" minOccurs="1"
                        maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
</xs:schema>
```

## Examples:

### Request

http://localhost:8080/bustime/api/v3/getlocalelist?key=89dj2he89d8j3j3ksjhdue93j

### Response

```
<?xml version="1.0"?>
<bustime-response>
   <locale>
      <localestring>en</localestring>
      <displayname>English</displayname>
   </locale>
   <locale>
      <localestring>es</localestring>
      <displayname>Spanish</displayname>
   </locale>
</bustime-response>
```

### Request

http://localhost:8080/bustime/api/v3/getlocalelist?key=89dj2he89d8j3j3ksjhdue93j&locale=es

### Response

```
<?xml version="1.0"?>
<bustime-response>
   <locale>
      <localestring>en</localestring>
      <displayname>inglés</displayname>
   </locale>
   <locale>
      <localestring>es</localestring>
      <displayname>español</displayname>
   </locale>
</bustime-response>
```

### Request

http://localhost:8080/bustime/api/v3/getlocalelist?key=89dj2he89d8j3j3ksjhdue93j&inLocaleLanguage=true

### Response

```
<?xml version="1.0"?>
<bustime-response>
   <locale>
      <localestring>en</localestring>
      <displayname>English</displayname>
   </locale>
   <locale>
      <localestring>es</localestring>
      <displayname>español</displayname>
   </locale>
</bustime-response>
```

## 3.11 *Real-Time Passenger Information*

**Base URL: http://[host:port]/bustime/api/v3/getrtpidatafeeds**

**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |

**Response:**
A well-formed XML or JSON document will be returned as a response to getrtpidatafeeds.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **rtpidatafeed**<br>JSON Array:<br>**rtpidatafeeds** | Child element of the root element. Encapsulates an external or internal data feed serviced by the system. |
| **name** | Child element of the **rtpidatafeed** element. Alphanumeric designator of rtpi datafeed (ex. “Nextbus feed”). This is the value that should be used in the rtpidatafeed parameter in other requests. |
| **source** | Child element of the **rtpidatafeed** element. Origin of RTPI information. (ex. “NEXTBUS” for the nextbus TA information). |
| **displayname** | Child element of the **rtpidatafeed** element. TA for which this data feed returns information (ex. “MBTA”). |
| **enabled** | Child element of the **rtpidatafeed** element. True if the feed is enabled; false otherwise. |
| **visible** | Child element of the **rtpidatafeed** element. True if this feed may be displayed to the public; false if the feed is for internal use only. |

**Remarks:**
Use the **getrtpidatafeeds** request to retrieve the set of external and internal data feeds serviced by the system.

**XML Schema:**

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <xs:element name=”bustime-response” type=”bustime-response”/>
        <xs:complexType name="bustime-response">
           <xs:sequence>
                <xs:element name="rtpidatafeed" maxOccurs="unbounded" minOccurs="0">
                  <xs:complexType>
                    <xs:sequence>
                        <xs:element type="xs:string" name="name" minOccurs="1" maxOccurs="1"/>
                        <xs:element type="xs:string" name="source" minOccurs="1" maxOccurs="1"/>
                        <xs:element type="xs:string" name="displayname" minOccurs="1"
                        maxOccurs="1"/>
```

```
                    <xs:element type="xs:boolean" name="enabled" minOccurs="1"
                    maxOccurs="1"/>
                    <xs:element type="xs:boolean" name="visible" minOccurs="1"
                    maxOccurs="1"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
           </xs:sequence>
        </xs:complexType>
        <xs:complexType name="error">
            <xs:sequence>
                 <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
           </xs:sequence>
        </xs:complexType>
</xs:schema>
```

**Example:**
The XML document below is a response to the following request:

**Request**
http://localhost:8080/bustime/api/v3/getrtpidatafeeds?key=id2YzEgRZ

**Response**

```
<bustime-response>
        <rtpidatafeed>
                <name>bustime</name>
                <source>Bus Tracker</source>
                <displayname>CTA</displayname>
                <enabled>true</enabled>
                <visible>true</visible>
        </rtpidatafeed>
        <rtpidatafeed>
                <name>External Feed</name>
                <source>NEXTBUS</source>
                <displayname>actransit</displayname>
                <enabled>true</enabled>
                <visible>true</visible>
        </rtpidatafeed>
</bustime-response>
```

## 3.12 *Detours*

**Base URL: http://[host:port]/bustime/api/v3/getdetours**
**Parameters:**

| Name | Value | Description |
|---|---|---|
| **key** | string (required) | 25-digit BusTime Developer API access key. |
| **rt** | route designator (optional) | Alphanumeric designator of the route (ex. "20" or "X20") for which a list of detours is to be returned. |
| **rtdir** | route direction (optional) | Direction of travel of the route specified in the **rt** parameter. The **rt** parameter is required when using the **rtdir** parameter. This needs to match the direction id seen in the getdirections call. |
| **rtpidatafeed** | (multi-feed only) string (optional) | Specify the name of the Real-Time Passenger Information data feed to retrieve detours for. Required in multi-feed systems if the rt parameter is provided. |

**Response:**
A well-formed XML or JSON document will be returned as a response to getdetours.

**Response Fields:**

| Name | Description |
|---|---|
| **bustime-response** | Root element of the response document. |
| **error** | Child element of the root element. Message if the processing of the request resulted in an error. |
| **dtr**<br>JSON Array: **dtrs** | Child element of the root element. Encapsulates data about a detour. |
| **id** | Child element of the **dtr** element. The unique id of the detour. Other API calls reference these identifiers. |
| **ver** | Child element of the **dtr** element. The version of this detour. Only the newest version of each detour is returned. |
| **st** | Child element of the **dtr** element. The state of the detour. A value of 1 indicates the detour is active; 0 indicates a canceled detour. |
| **desc** | Child element of the **dtr** element. Description of the detour. |
| **rtdirs** | Child element of the **dtr** element. Contains a series of **rtdir** elements, each with **rt**, the designator of the route this detour is affecting, and **dir**, the id of the direction this detour is affecting. |
| **startdt** | Child element of the **dtr** element. The start date and time of this detour. |
| **enddt** | Child element of the **dtr** element. The end date and time of this detour. |
| **rtpidatafeed** | (Multi-feed only) Child element of the **dtr** element. The name of the data feed that this detour was retrieved from. |

## XML Schema:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name=”bustime-response” type=”bustime-response”/>
	<xs:complexType name="bustime-response">
	   <xs:sequence>
	        <xs:element name="dtr" maxOccurs="unbounded" minOccurs="0">
	           <xs:complexType>
	            <xs:sequence>
	                <xs:element type="xs:string" name="id" minOccurs="1" maxOccurs="1"/>
	                <xs:element type="xs:int" name="ver" minOccurs="1" maxOccurs="1"/>
	                <xs:element type="xs:int" name="st" minOccurs="1" maxOccurs="1"/>
	                <xs:element type="xs:string" name="desc" minOccurs="1" maxOccurs="1"/>
	                <xs:element name="rtdirs" minOccurs="1" maxOccurs="1">
	                  <xs:complexType>
	                    <xs:sequence>
	                      <xs:element name="rtdir" minOccurs="0" maxOccurs="unbounded">
	                        <xs:complexType>
	                          <xs:sequence>
	                            <xs:element type="xs:string" name="rt" minOccurs="1"
	        maxOccurs="1"/>
	                            <xs:element type="xs:string" name="dir" minOccurs="1"
	        maxOccurs="1"/>
	                          </xs:sequence>
	                        </xs:complexType>
	                      </xs:element>
	                    </xs:sequence>
	                  </xs:complexType>
	                </xs:element>
	                <xs:element type="xs:string" name="startdt" minOccurs="1" maxOccurs="1"/>
	                <xs:element type="xs:string" name="enddt" minOccurs="1" maxOccurs="1"/>
	                <xs:element type="xs:string" name="rtpidatafeed" minOccurs="0"
	                maxOccurs="1"/>
	            </xs:sequence>
	           </xs:complexType>
	        </xs:element>
	   </xs:sequence>
	</xs:complexType>
	<xs:complexType name="error">
	    <xs:sequence>
	         <xs:element name="msg" type="xs:string" minOccurs="1" maxOccurs="1"/>
	         <xs:element name="rt" type="xs:string" minOccurs="0" maxOccurs="1"/>
	         <xs:element name="rtdir" type="xs:string" minOccurs="0" maxOccurs="1"/>
	         <xs:element name="rtpidatafeed" type="xs:string" minOccurs="0" maxOccurs="1"/>
	   </xs:sequence>
	</xs:complexType>
</xs:schema>
```

## Examples:

**Request**

http://localhost:8080/bustime/api/v3/getdetours?key=89dj2he89d8j3j3ksjhdue93j

**Response**

```
<bustime-response>
	<dtr>
		<id>84A97FD3-0741-4004-884D-0ABB22DAFA28</id>
		<ver>2</ver>
		<st>0</st>
		<desc>IVD MultiRoute detour 47</desc>
		<rtdirs>
			<rtdir>
				<rt>72</rt>
				<dir>NORTHBOUND</dir>
			</rtdir>
		</rtdirs>
		<startdt>20180404 08:45</startdt>
		<enddt>20180430 03:00</enddt>
```

```
                <rtpidatafeed>bustime</rtpidatafeed>
        </dtr>
        <dtr>
                <id>329E1F2D-A848-43E9-8F90-4FB00E643786</id>
                <ver>1</ver>
                <st>1</st>
                <desc>IVD Multiroute Detour S47/62</desc>
                <rtdirs>
                        <rtdir>
                                <rt>800</rt>
                                <dir>EASTBOUND</dir>
                        </rtdir>
                        <rtdir>
                                <rt>72M</rt>
                                <dir>NORTHBOUND</dir>
                        </rtdir>
                </rtdirs>
                <startdt>20180404 09:06</startdt>
                <enddt>20180430 03:00</enddt>
                <rtpidatafeed>bustime</rtpidatafeed>
        </dtr>
</bustime-response>
```

**Request:**

http://localhost:8080/bustime/api/v3/getdetours?key=89dj2he89d8j3j3ksjhdue93j&rt=2&format=json

**Response:**

```
{
        "bustime-response": {
                "dtrs": [
                        {
                                "id": "84A97FD3-0741-4004-884D-0ABB22DAFA28",
                                "ver": 2,
                                "st": 0,
                                "desc": "IVD MultiRoute detour 47",
                                "rtdirs": [
                                        {
                                                "rt": "72",
                                                "dir": "NORTHBOUND"
                                        }
                                ],
                                "startdt": "20180404 08:45",
                                "enddt": "20180430 03:00",
                                "rtpidatafeed": "bustime"
                        },
                        {
                                "id": "329E1F2D-A848-43E9-8F90-4FB00E643786",
                                "ver": 1,
                                "st": 1,
                                "desc": "IVD Multiroute Detour S47/62",
                                "rtdirs": [
                                        {
                                                "rt": "800",
                                                "dir": "EASTBOUND"
                                        },
                                        {
                                                "rt": "72M",
                                                "dir": "NORTHBOUND"
                                        }
                                ],
                                "startdt": "20180404 09:06",
                                "enddt": "20180430 03:00",
                                "rtpidatafeed": "bustime"
                        }
                ]
        }
}
```

**Remarks:**

Use the **getdetours** request to retrieve a list of active detours in the system. Detours are considered "active" if they are currently affecting the current service day, even if the start time has not yet been reached or the end time has already passed.

If a detour is canceled or expired, it will still appear in this result. This is to handle cases where a vehicle is still running a canceled or expired detour and the developer wishes to alert users that the detour is technically still in effect.

If the client application is to support detours, it is recommended that detours are requested frequently in case a new version is added or a detour is canceled. If a current detour or new version is added (or removed), the client should consider requesting new stop and pattern data for the given route/direction combination in case data has been changed by the detour.

> **Note:** Data feeds with a source of "NEXTBUS" or "SYNCROMATICS" do not support this call.

# 4 Version 3 Release Notes

Version 3 of the Developer API contains a number of changes:

- The URL of the request changes.
- Most calls now support an **rtpidatafeed** parameter to query desired feeds of multi-feed systems
- In a multi-feed system, some calls now return an **rtpidatafeed** element in their results
- The results of some calls are now affected by detours which introduces a new **getdetours** call.
- The results of some calls are now affected by disruption management changes
- Standardization of format of the Route Directions call
- Changes to the Real Time Passenger Information call
- Miscellaneous fixes

## 4.1 *Calling Version 3*

Version 3 of the API is used by including "v3"in the request URL as follows:

http://localhost:8080/bustime/api/v3/getroutes?key=89dj2he89d8j3j3ksjhdue93j

## 4.2 *Inclusion of "rtpidatafeed" parameter in most calls*

Version 3 of the API greatly enhances support of systems with multiple configured feeds. A "multi-feed" system is one which services more than one agency, source, or data transmission. API users can determine if their working system is multi-feed by using the **getrtpidatafeeds** call. If the call returns more than one feed, then the system is multi-feed, even if only one feed is enabled.

A feed's enabled state is relevant to the user, however. . The enabled state of a feed is returned in the **getrtpidatafeeds** call. Making any call with an **rtpidatafeed** parameter of a disabled feed's name will result in an "Invalid RTPI Data Feed parameter" error. A Feed must be enabled in order to offer any data through the API. Also, some calls now *require* an **rtpidatafeed** parameter when working within a multi-feed system in order to properly determine what the user is requesting.

The following calls now support or better support the **rtpidatafeed** parameter:

- Vehicles
- Routes
- Route Directions
- Stops
- Patterns
- Predictions

- Service Bulletins

API users should review the reference for each call's handling of this new parameter.

## 4.3 *Inclusion of "rtpidatafeed" element for multi-feed systems*

In a multi-feed system, some calls will return an **rtpidatafeed** element within the results. Generally, this element denotes the feed that its parent's data belongs to. This element helps API users differentiate objects such as routes and stops with ids that are present across multiple feeds.

The following calls now return an **rtpidatafeed** element within a multi-feed system:

- Vehicles
- Routes
- Service Bulletins

## 4.4 *Introduction of the Detours call*

Some calls such as Stops and Patterns can now be affected by detours. These calls will reference detour ids which can be referenced in the new Detours call. See section 1.8 and the reference for the Detours call for more information.

## 4.5 *Introduction of Disruption Management changes*

Some calls are now affected by disruption management changes. For example, a prediction can now be marked as canceled if the vehicle will skip the associated stop. If the developer wishes to support disruption management, recurring requests to route data calls will be needed. Additionally, a new Dynamic Changes call will be created to support further changes. See section 1.8 for more information.

## 4.6 *Standardization of the Route Directions call*

Version 2 introduced the multi-feed concept to the Route Directions call. In that version, the results for a multi-feed system had localization data but a single feed did not. In v3, the Route Directions call will always be formatted to show locale-specific data. See the reference for Route Directions for examples of this format.

## 4.7 *Changes to Real Time Passenger Information call*

Version 3 introduces some new elements and element name changes in order to provide developers with more valuable and more accurate information about RTPI feeds:

- The `agency e`lement is now `displayname`
- The call now returns disabled feeds in addition to enabled ones
- The `enabled` boolean element has been added

## 4.8 *Miscellaneous Fixes*

- **Predictions**
    - Using this call in a multi-feed system now appropriately returns "No Service Scheduled" for given **stpid**s

  - All given **stpid**s/**vid**s are now in some way represented in the result, either with predictions or an error
- **Service Bulletins**
  - Results are now properly filtered by combinations of **rt**, **rtdir**, and **stpid** instead of being expanded by them
  - An error is now properly returned if a given **stpid** is not along the given **rt** and **rtdir**
  - Any invalid parameters given for this call are now ignored and return an error; bulletins will be returned for whatever valid parameters remain
- **Stops -** Passing in a rtpidatafeed parameter along with stpid(s) no longer results in an empty response
- Other Notes
  - The **dir** and **rtdir** parameters now use the id of the direction instead of the localized name
  - Much of the core API code has been optimized and primed for versioning, which should increase response time for users
  - New error messages have been added to support new functionality in this version
  - To allow easier transitioning from legacy versions to v3, the inconsistency of pluralizations of JSON arrays has not been changed

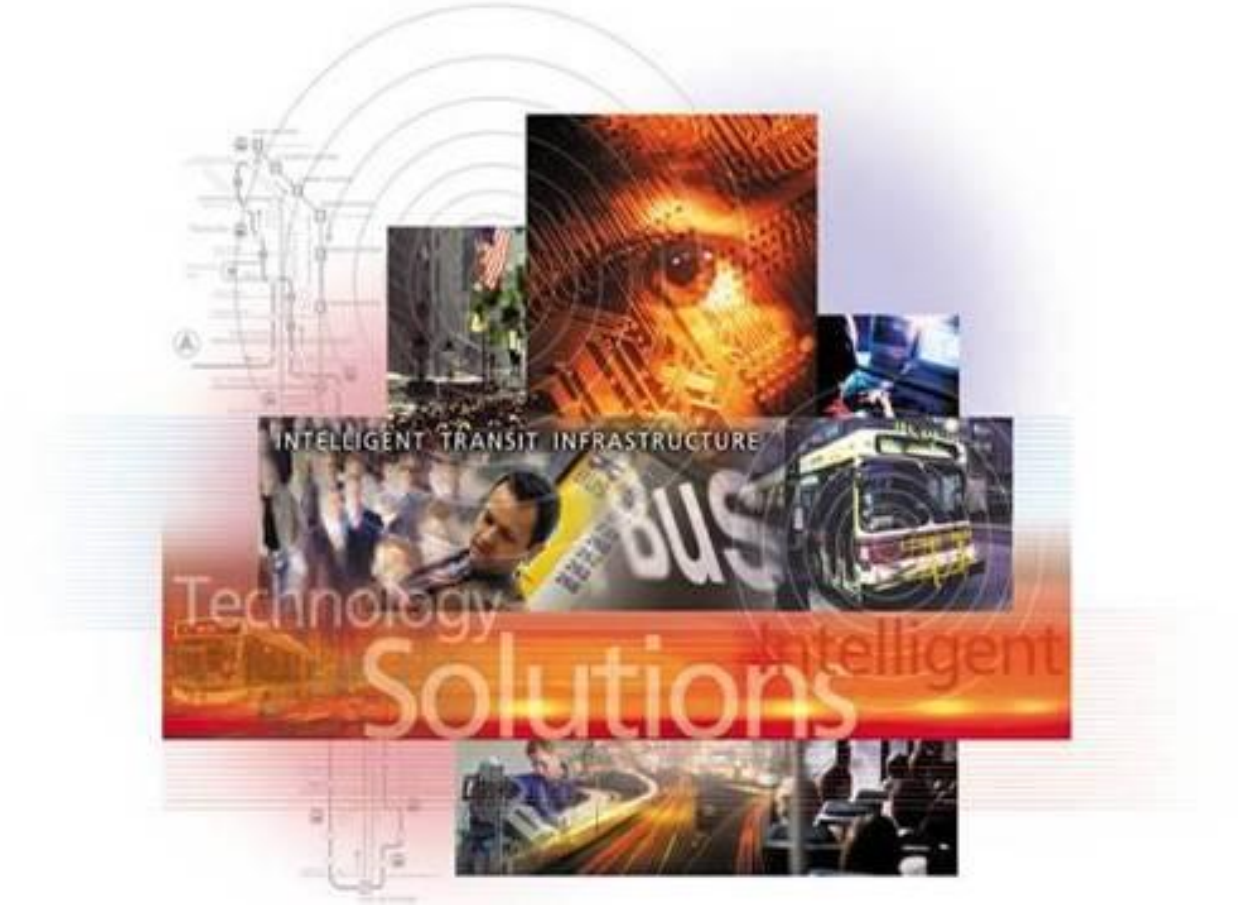
# 5 Error Descriptions

This section describes all possible error responses that can be received from the BusTime® Developer API.

| Error Message | Related API Calls | Description |
|---|---|---|
| Internal server error - Unable to complete request at this time | All | The most general error message, given when we cannot find a more specific error message to send. |
| No API access permitted | All | The Developer API has been disabled by the Transit Authority. |
| No API access key supplied | All | The 'key=<DevKey>' parameter is missing from the API request. |
| Invalid API access key supplied | All | The given Developer key is not assigned to any users. |
| No version requested | All | The request URL is missing the version. |
| Unsupported version requested | All | The request URL contains an unsupported version. |
| Unsupported function | N/A | The request contains a function name that is not supported by the API. |
| Transaction limit for current day has been exceeded. | All | The user, identified by the Developer Key, has already reached the maximum number of API calls allowed for the day. |
| Invalid locale parameter | All | (Only in BusTime version 3.0+) The requested locale string is not in a proper format. The proper format is "la" where la is a legal ISO |

| Error Message | Related API Calls | Description |
|---|---|---|
| | | 639 code. |
| Format parameter must be xml or json | All | The 'format' parameter is invalid. The value must be "xml" or "json". |
| **Error Message** | **Related API Calls** | **Description** |
| No data found for parameter(s) | All except **gettime** | No results were found that matched the given parameters. |
| No parameter provided | **getpattern** | Required 'rt' or 'pid' parameters are missing. |
| No parameter provided | **getpredictions** | The required 'stpid' or 'vid' parameters are missing. |
| No parameter provided | **getservicebulletins** | The required 'rt' or 'stpid' parameters are missing. |
| dir parameter missing | **getstops** | The required 'dir' parameter is missing. |
| rt parameter missing | **getdirections**, **getstops**, **getservicebulletins** | The required 'rt' parameter is missing. |
| Either rt or vid parameter must be specified | **getvehicles** | The request is required to contain either a 'rt' or 'vid' parameter. |
| Invalid parameter provided | **getpatterns**, **getpredictions,** **getdetours** | The listed parameter(s) does not match any known ID. |
| Maximum number of pid identifiers exceeded | **getpattern** | The 'pid' parameter contains more than 10 pattern IDs. |
| Invalid top parameter provided | **getpredictions** | The 'top' parameter is not an integer or contains extra characters. For instance "top=10" is legal but "top=10." is not. |
| Maximum number of <x> identifiers exceeded | **getpredictions**, **getvehicles** | The 'stpid' or 'vid' parameter contains too many IDs. <x> shows the maximum allowed in a single request. |
| No arrival times | **getpredictions** | The given stop has no scheduled arrival times. |
| No service scheduled | **getpredictions** | The given stop has no service scheduled. |
| Invalid RTPI Data Feed parameter | All except **gettime** and **getlocalelist** | The given 'rtpidatafeed' is an invalid or disabled feed. |
| No RTPI Data Feed parameter provided | **getdirections**, **getstops**, **getpatterns**, **getpredictions**, **getservicebulletins** | The required 'rtpidatafeed' parameter is missing. |
| The rtpidatafeed does not support this function | **getvehicles**, **getservicebulletins,** **getdetours** | The given 'rtpidatafeed' is a valid feed but does not support the call's functionality. |

Clever Devices




## America's Leader in Transit Technology

Clever Devices Ltd.
300 Crossways Park Drive
Woodbury, New York 11797
Phone: (516) 433-6100
Fax: (516) 433-5088
*www.cleverdevices.com*