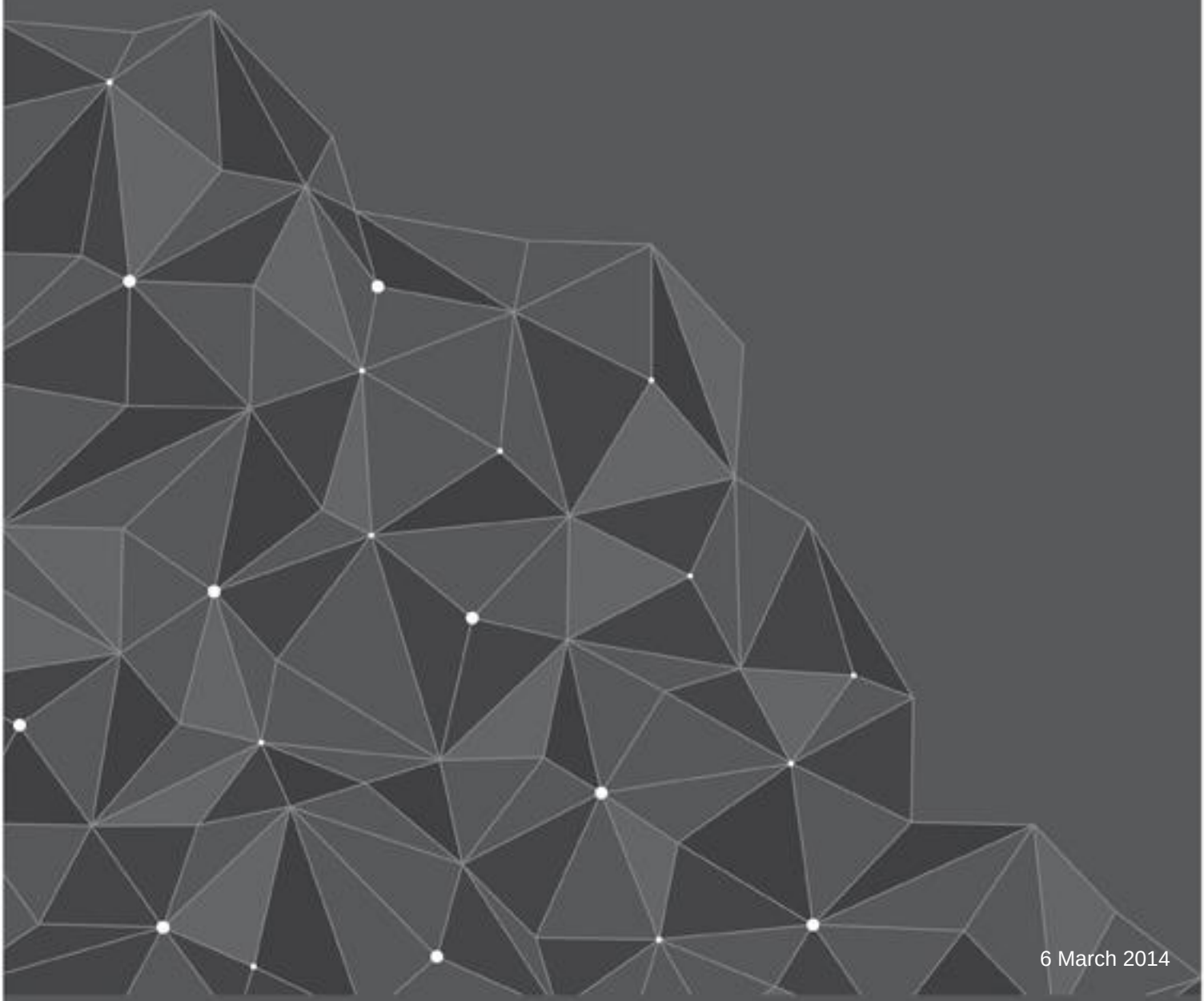
PUBLIC TRANSPORT VICTORIA

Timetable API



6 March 2014

Introduction

Hello and welcome

Hi and welcome to the PTV Timetable API. The API has been created to provide public transport timetable data to the public in the most dynamic and efficient way.

By providing an API, rather than an enormous data file or database, PTV hopes to maximise both the opportunities for re-use of public transport data and the potential for innovation.

Licence

Disclaimer

Your use is your responsibility

The data provided to you by PTV is provided “as is” and PTV is not liable for how you use this data, how third parties use or rely on this data or any errors contained within the data. You are responsible for determining whether the data is suitable for your particular usage and purposes.

What you get with the PTV Timetable API

Our API gives you direct access to the PTV timetable data. The API allows you to query locations for timetable, line and stop data for all train, tram and bus, V/Line rail and coach, and NightRider services. It also includes access to myki ticket outlet data.

> In this document, “stop” means any train station, tram stop or bus stop. For more information, check out the **Guide to understanding public transport data**.

The data will be updated weekly to take into account any planned timetable changes, for example, due to holidays or planned disruptions. (Any changes to the timetable notified by public transport operators on the day of operation will not be picked up.)

> PTV timetable data consists of the scheduled timetable. It is **not** real-time data nor service disruption information.

The PTV timetable API is the same API currently used by PTV for our website and smartphone apps. PTV enhances these products by integrating its timetable data with TramTracker data, the Google geocoding API (which allows for address searching) and PTV’s own journey planner service.

> The following are **not included** in the PTV Timetable API:
>
> TramTracker data – this is available through **YarraTrams**
>
> An address geocoding API – available through search providers such as **Google**
>
> The PTV journey planner – this is not raw data but rather a service PTV provides

Do’s and don’ts

Timetables, stops, lines and even ticket outlets change frequently so to get the most out of the PTV Timetable API, we recommend you use it dynamically. That’s the only way to ensure you’re accessing the most up-to-date data and providing it to your audience through the app or service you create.

Do use the API dynamically to get the most up-to-date data for your audience.

Do not cache the data. If you want a dump of timetable data for a specific point in time, **please contact us**.

Do not hammer our servers. Do not use the API to make multiple requests for large sets of data in short periods of time.

Audience

The PTV Timetable API is for everyone. All members of the public, whether they are students, hobbyist app developers or companies, can access PTV timetable data using our API.

PTV assumes that you know how to use APIs and do not provide instructions on how to code in any specific programming languages.

PTV **does not provide technical support** for the API.

All information required to use the API is included in this document.

What's in the document

Getting started

- First steps: getting your key
- Quickstart Guide
- Quick Reference Guide
- Use Case Maps

Overview

- Main features of the API
- API Structure
- API Interface
- Errors

Reference

- A description of the request and response, data specification and examples for each API below:
  - Health Check
  - Stops Nearby
  - Transport POIs by Map
  - Search
  - Broad Next Departures
  - Specific Next Departures
  - Stopping Pattern
  - Stops on a Line
- JSON object structure
- Data Quality Statement

Need help?

Check the **Glossary** for explanations of common terms and acronyms

Take a look at the **Guide to understanding public transport data** – it's been designed to help you make sense of the data that you access through the API

Try the **FAQs** – it has answers to some common questions

Contact PTV using the comment and feedback link on the DataVic site

Getting started

First steps: getting your security key and developer ID

You'll need to pass along a signature and a developer ID – or "devid" - with every request using HTTP GET.

To calculate the signature, you'll need **the request, which includes yourdeveloper ID, and akey**.

The key consists of a 128bit GUID.

> The **key**and the **request (includingdeveloper ID)** are used to calculate a **signature** for every request.

How to register for a key and developer ID

> Send an email to APIKeyRequest@ptv.vic.gov.auwith the following information in the subject line of the email:
>
> - "PTV Timetable API – request for key"

> Once we've got your email request, we'll send you a key and a developer ID by return email.

> A high volume of requests may result in a delay in providing you with your key and developer ID. We'll try to get it to you as soon as we can.

> We'll also add your email address to our API mailing list so we can keep you informed about the API.

> The "APIKeyRequest" email address is only used to send you the key and developer ID and any relevant notifications. To provide feedback on the API check out our **contact details**.
>
> PTV **does not provide technical support** on the API.

Privacy

Your email address is the only bit of information about you that PTV will hold in its register. You can view PTV's privacy policy at http://www.ptv.vic.gov.au/privacy.

Quick start guide

Once you have obtained your key and developer ID you can get started. The first thing you need to do is to calculate a **signature**.

How to calculate a signature

> The signature value is a **HMAC-SHA1 hash of the completed request** (minus the base URL but including your developer ID, known as "devid") **and the key**:

- signature = crypto.HMACSHA1(request,key)

> The calculation of a signature is based on a **case-sensitive** reading of the request message. This means that the request message used to calculate the signature must not be modified later on in your code or the **signature will not work**. If you do modify the case of the request message, you will need to calculate a new signature.

For example, "http://timetableapi.ptv.vic.gov.au/v2/**h**ealth**c**heck?devid=ABCXYZ" and "http://timetableapi.ptv.vic.gov.au/v2/**H**ealth**C**heck?devid=ABCXYZ" require different signatures to be calculated; **the same signature will not work for both requests**.

> The signature itself is also case-sensitive

Example of a request message for signature calculation:

The request URL for the Stops Nearby API is:

base URL/v2/nearme/latitude/%@/longitude/%@?devid=%@&signature=%@

A sample request message used to calculate a signature would be:

http://timetableapi.ptv.vic.gov.au/v2/nearme/latitude/-37.82392124423254/longitude/144.9462017431463?devid=0000001

Refer to the **Appendix** for some sample code for calculating a signature.

Performing the Health Check

The first API you need to call is the Health Check.

The Health Check will test a number of the key services that deliver the PTV Timetable API and let you know if there are any problems with connectivity, availability or reachability.

It will also test the time on your system to make sure that your clock is in sync with our clock.

The output is in the JSON format.

Health Check request URL:

http://timetableapi.ptv.vic.gov.au/v2/healthcheck?timestamp=%@&devid=%@&signature=%@

"%@" in the request URL represents a parameter

Parameters

timestamp = *optional:* the date and time of the request in ISO 8601UTC format
e.g. 2014-02-28T05:24:25Z

devid = *optional:* the developer ID supplied in your email from PTV

signature = *optional:*the customised message digest calculated using the method in the**Quick start guide**

> While all parameters for this API are optional, if you don't include them the **securityTokenOK** and **clientClockOK** response will return **"false"**.

Response output:

```
{
  "securityTokenOK": boolean,
  "clientClockOK": boolean,
  "memcacheOK": boolean,
  "databaseOK": boolean,
}
```

where a "true" value indicates service connectivity and availability, and "false" indicates a problem. For more information on this API, check out **Errors** and the **Reference**.

Congratulations

Once you've calculated a signature and performed the health check successfully you are ready to access the timetable, line and stop data available through the PTV Timetable API.

All systems are go!

Quick reference guide

The PTV Timetable API lets you access stop, line and timetable data for all metropolitan and regional services in Victoria. The APIs are as follows:

Health Check

This API returns a health report on the timely availability, connectivity and reachability of the key services that deliver our timetable data to web clients.

Stops Nearby

The Stops Nearby API returns up to 30 stops nearest to a specified coordinate.

Transport POIs by Map

This API returns a list of transport points of interest (POIs) in a region described by latitude and longitude coordinates. POIs can be any or all of stations, stops or myki ticket outlets.

Search

The Search API returns all stops and lines that include the search term.

Broad Next Departures

This API returns departure times from a stop, irrespective of what line the service is on or in what direction the service is running.

Specific Next Departures

This API returns all departure times from a stop for a specific line and in a specific direction.

Stopping Pattern

The Stopping Pattern API returns all the times for stops that a particular vehicle will stop at on a specific service run (that is, specific line, direction and point in time).

Stops on a Line

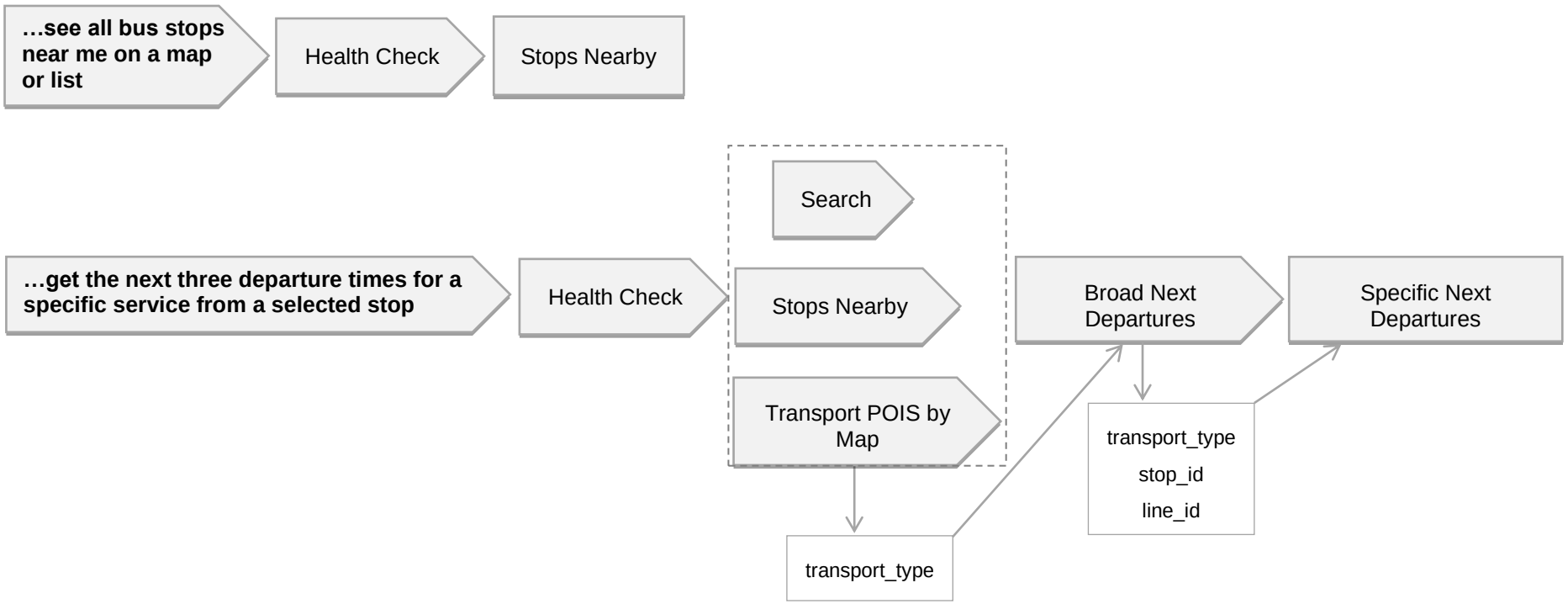
The Stops on a Line API returns all the stops along a specific line.

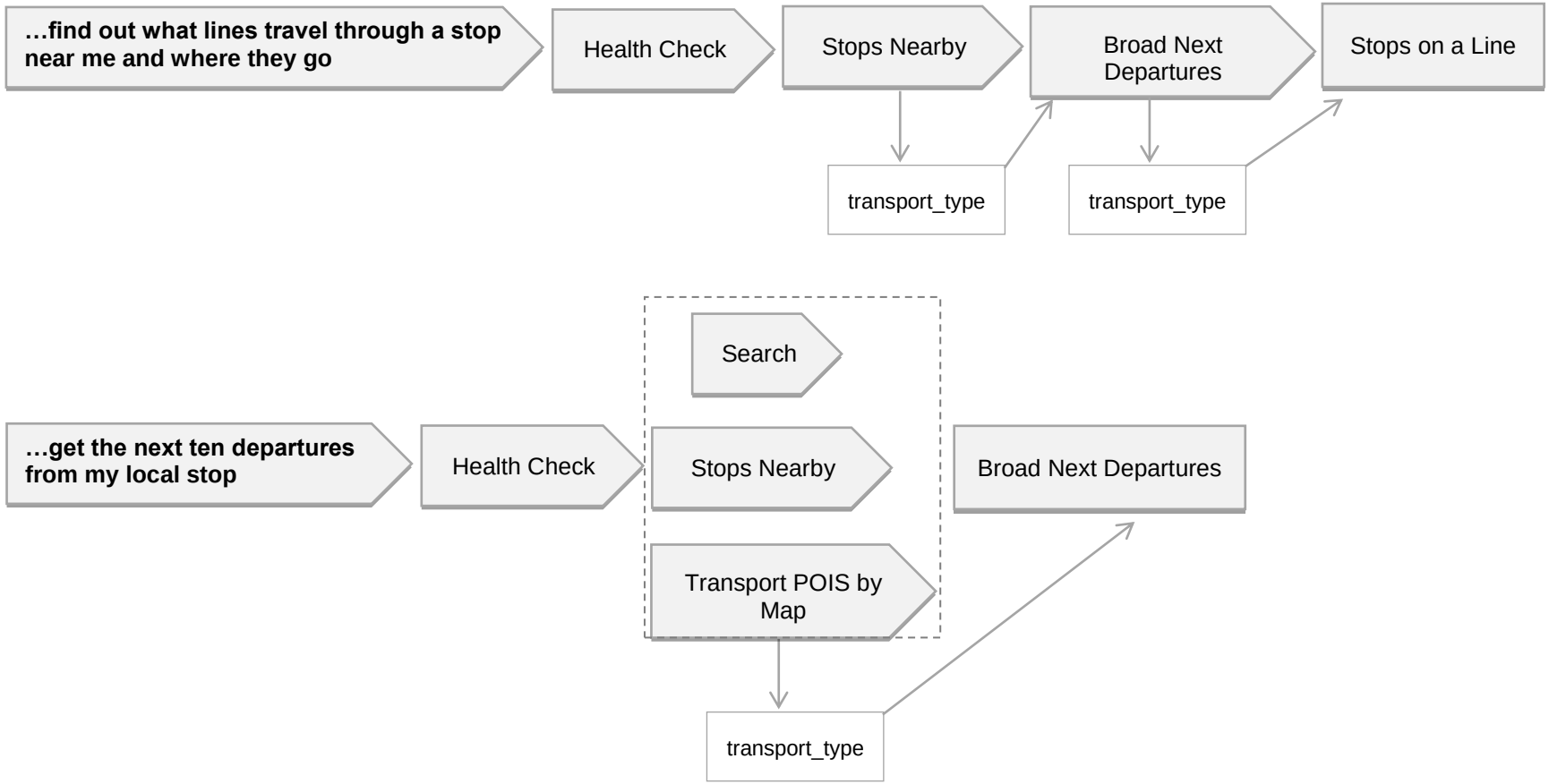
Use case maps

To give you a taste of what you can do with the PTV Timetable API, we've created a small list of use case maps that show the sequence of APIs required to obtain particular information.

> The maps below only show the data outputs that are used as inputs into the next API. **They don't show all the inputs and outputs for each API.** For more detailed information on the APIs, check out the section on the **API Structure**as well as the **Reference**.

I want to…




…find out what lines travel through a stop near me and where they go
Health Check
Stops Nearby
Broad Next Departures
Stops on a Line
transport_type
transport_type
Search
…get the next ten departures from my local stop
Health Check
Stops Nearby
Broad Next Departures
Transport POIS by Map
transport_type
…find myki ticket outlets in my area
Health Check
Transport POIs by Map
…see on a map which bus stops are near my local train station
Health Check
Transport POIs by Map

Overview

Main features

Stateless

Public transport timetable data is fast-changing, time-based data so our API is REST-like (and therefore stateless).

Format

The API functions via a request and response format whereby parameters are passed in a request and a response with the relevant data received accordingly.

Output

The responses you receive from the API will be represented in JSON. The format is that of a JSON object with a name for each attribute.

For more information on JSON, see http://www.json.org/

Authentication

A unique key and developer ID is used to calculate a signature for every request that you make.

For more information about how to get a key and developer ID and how to calculate a signature, check out the **Getting Started** section.

DateTime and time zone

All DateTimes are stored and reported in UTC. The ISO8601 format (e.g. 2011-09-13T16:09:54Z) is used throughout the API. The DateTimes are returned as strings since JSON does not have a DateTime object in the specification.

Versioning

The PTV Timetable API uses semantic versioning.

The **current version of the API is 2.0.0** but the URL only includes the relevant part of the version number (in this case, the major part).

Structure

The PTV Timetable API is structured to allow you to build information dynamically as you need it, based on the output of each API called.

For example, the Stops Nearby API requires inputs that are not specific to public transport (i.e. latitude and longitude coordinates). The output, however, includes stop_id and transport_typedata. You can then pass that data through the Broad Next Departures API to obtain

line_id, direction_id, run_id and timetable data. You can then use these outputs as inputs into other APIs.

The table below summarises the **relevant subset** of inputs and outputs of each API:

API	Inputs	Outputs include
Stops Nearby	lat / lon,signature	transport_type, stop_id, lat / lon
Transport POIs by Map	transport_type, lat / lon, grid depth, limit,signature	transport_type, stop_id, lat / lon
Search	search term,signature	transport_type, stop_id, lat / lon, line_id
Broad Next Departures	transport_type,stop_id, limit,signature	transport_type, stop_id, lat / lon, line_id, direction_id, run_id, timetable time
Specific Next Departures	transport_type,line_id,stop_id,direction_id, limit, for_utc,signature	transport_type, stop_id, lat / lon, line_id, direction_id, run_id, timetable time
Stopping Pattern	transport_type,run_id,stop_id, for_utc,signature	transport_type, stop_id, lat / lon, line_id, direction_id, run_id, timetable time
Stops on a Line	transport_type,line_id,signature	transport_type, stop_id, lat / lon

> For a **full list** of the relevant parameters and output data for each API, check out the **Reference** section.

Interface

You access the PTV Timetable API through an HTTP interface, as follows:

base URL / version number / API name / query string

The base URL is http://timetableapi.ptv.vic.gov.au

The version number, API name and query string are provided in the **Reference** section, under each API.

> **"%@"** in the request URL represents a parameter

Errors

Error trapping through Health Check

Calling the **Health Check API** at the start of each sequence of APIs flushes out any system problems.

A return of **true** or **false** for the following attributes reveals their status (where “true” means the system is okay, and “false” reveals a problem):

securityTokenOK – i.e. your key/signature is working
(if it returns “false” check your logic and ensure you have a valid key)

clientClockOK – i.e. your clock is synchronised with our clock within three minutes
(this is for your information only; if it returns “false” it may affect the way you present dates and times)

memcacheOK – performance cache is working well
(if it returns “false” your queries will be slow)

databaseOK – availability of the data
(if it returns “false” your queries won’t work)

For more information on the Health Check API, check out the **Quick start guide** and the **Reference** section.

HTTP status codes

Since the PTV Timetable API uses a HTTP interface, any of the following standard HTTP status codes may be returned:

200 – no error; system okay

403 – access denied (will be returned when the wrong signature is used)

404 – requested resource not found (check your URL, including parameters, is correct)

500 – internal server error (check your URL, including parameters, is correct)

For more information, see http://en.wikipedia.org/wiki/List_of_HTTP_status_codes.

Reference

Health Check

Version Number

2.0.0

Description

A check on the timely availability, connectivity and reachability of the services that deliver security, caching and data to web clients. A health status report is returned.

> It's good practice to call the Health Check APIevery time you make a sequence of calls to the API.

Request URL

base URL
/v2/healthcheck?timestamp=%@&devid=%@&signature=%@

Parameters

timestamp = *optional:* the date and time of the request in ISO 8601UTC format
e.g. 2013-11-13T05:24:25Z

devid = *optional:* the developer ID supplied in your email from PTV

signature = *optional:*the customised message digest calculated using the method in the**Quick start guide**

> While all parameters for this API are optional, if you don't include them the **securityTokenOK** and **clientClockOK** response will return **"false"**.

Response

The response is made up of the following JSON objects:

securityTokenOK *boolean*
– indicates whether your key is valid/signature is calculated correctly

clientClockOK *boolean*
– indicates whether your clock is synchronised with our clock within 3 minutes

memcacheOK *boolean*
– indicates status of the performance cache

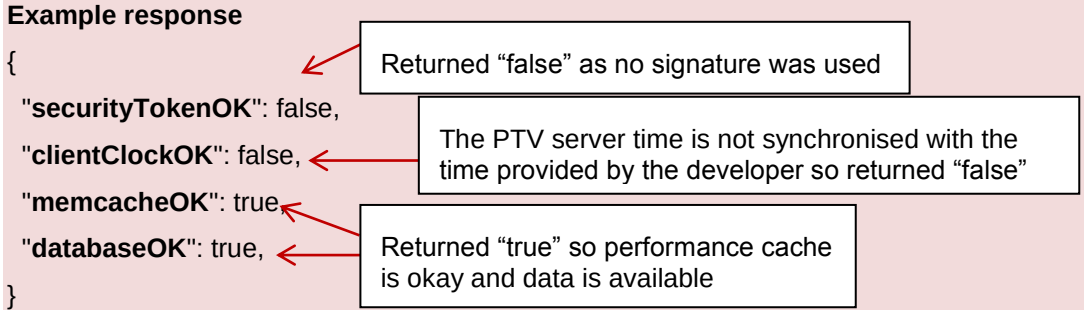
databaseOK *boolean*
– indicates availability of the data

> Refer to **Errors** for more information on using Health Check to trap errors.

Example request

http://timetableapi.ptv.vic.gov.au/v2/healthcheck?timestamp=2014-01-22T03:28:33Z

Example response



Stops Nearby

Version Number

2.0.0

Description

Stops Nearby returnsup to 30 stops nearest to a specified coordinate.

> **“Stops”** includes train stations as well as tram and bus stops.

Applicable stops are returned as a collection in the JSON format.

> There are **no spatial constraints** on how Stops Nearby retrieves stops. It will always return up to 30 stops near the passed latitude and longitude coordinates, even if some of those stops are (relatively) far away.

Request URL

base URL
/v2/nearme/latitude/%@/longitude/%@?devid=%@&signature=%@

Parameters

latitude = prescribed latitude, expressed in decimal degrees.
e.g. -37.82392124423254

longitude = prescribed longitude, expressed in decimal degrees.
e.g. 144.9462017431463

devid = the developer ID supplied in your email from PTV

signature = the customised message digest calculated using the method in the **Quick start guide**

Response

Returns an array of JSON **"result"** objects for which the "type" equals **"stop"**. A "stop" object is embedded within each "result". Stops are ordered by distance.

For more information on the data structures, check out the **JSON object structure**.

The **"stop"** object has these attributes:

suburb *string*
– the suburb name
– e.g. "Belgrave"

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either "train", "tram", "bus", "vline" or "nightrider"

stop_id *numeric string*
– the unique identifier of each stop
– e.g. "2825"

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "20-Barkly Square/115 Sydney Rd (Brunswick)"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.81603

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.9824

distance *decimal number*
– not used in the context of this API (it is a legacy attribute of unknown worth)

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the**"location_name"** is the name of the station – e.g. "Belgrave Station".

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the "stop_id").

Example use case

Janelle is creating an app for tourists in Melbourne and wants to use the PTV Timetable API to access public transport data.

First off, she wants tourists to be able to see all public transport stops near them on a list or on a map, no matter where they are, so Janelle uses the Stops Nearby API.

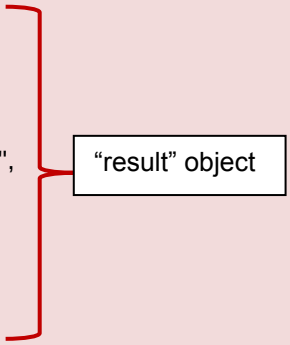
Example location: Brunton Avenue, Richmond, VIC 3002, Australia; -37.817993 , 144.981916

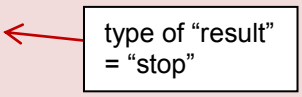
Example request

http://timetableapi.ptv.vic.gov.au/v2/nearme/latitude/-37.817993/longitude/144.981916?devid=4&signature=20F0ED441F888A604A7760BA42ECE94333AD279BD

Example response


```
[
  {
    "result": {
      "suburb": "East Melbourne",
      "transport_type": "tram",
      "stop_id": 2825,
      "location_name": "Clarendon St/Wellington Pde #11 ",
      "lat": -37.81603,
      "lon": 144.9824,
      "distance": 4.08647838E-06
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "East Melbourne",
      "transport_type": "nightrider",
      "stop_id": 24276,
      "location_name": "Clarendon St/Wellington Pde ",
      "lat": -37.81626,
      "lon": 144.9833,
      "distance": 4.92775143E-06
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "East Melbourne",
      "transport_type": "train",
      "stop_id": 1104,
      "location_name": "Jolimont-MCG ",
      "lat": -37.81653,
      "lon": 144.9841,
      "distance": 6.88463842E-06
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "East Melbourne",
      "transport_type": "tram",
      "stop_id": 2823,
      "location_name": "Jolimont Rd/Wellington Pde #10 ",
      "lat": -37.8157463,
      "lon": 144.979782,
      "distance": 9.586347E-06
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Melbourne City",
      "transport_type": "tram",
```



```
    "stop_id": 2171,
    "location_name": "7B-Rod Laver Arena/Melbourne Park ",
    "lat": -37.81959,
    "lon": 144.979126,
    "distance": 1.03426455E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "tram",
    "stop_id": 2824,
    "location_name": "Powlett St/Wellington Pde #12 ",
    "lat": -37.8163261,
    "lon": 144.985016,
    "distance": 1.24015E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "tram",
    "stop_id": 2140,
    "location_name": "7C-MCG - Hisense Arena/Melbourne Park ",
    "lat": -37.8224335,
    "lon": 144.983643,
    "distance": 2.27052114E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "tram",
    "stop_id": 2826,
    "location_name": "Simpson St/Wellington Pde #13 ",
    "lat": -37.81654,
    "lon": 144.986984,
    "distance": 2.77356339E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "tram",
    "stop_id": 3041,
    "location_name": "Lansdowne St/Wellington Pde #9 ",
    "lat": -37.81545,
    "lon": 144.977158,
    "distance": 2.910643E-05
  },
```

```
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "tram",
    "stop_id": 2482,
    "location_name": "7A-William Barak Bridge/Melbourne Park ",
    "lat": -37.8181648,
    "lon": 144.976212,
    "distance": 3.26682275E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "tram",
    "stop_id": 3123,
    "location_name": "Lansdowne St/Wellington Pde #9 ",
    "lat": -37.81547,
    "lon": 144.976654,
    "distance": 3.40131E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "bus",
    "stop_id": 18125,
    "location_name": "Olympic Park/Olympic Bvd ",
    "lat": -37.8238525,
    "lon": 144.982132,
    "distance": 3.43570864E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "bus",
    "stop_id": 18124,
    "location_name": "Melbourne Park/Olympic Bvd ",
    "lat": -37.8238258,
    "lon": 144.983185,
    "distance": 3.56446326E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "bus",
```

```
    "stop_id": 18123,
    "location_name": "Yarra Park/Olympic Bvd ",
    "lat": -37.8242,
    "lon": 144.986191,
    "distance": 5.682951E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "tram",
    "stop_id": 2877,
    "location_name": "Spring St/Flinders St #8 ",
    "lat": -37.81532,
    "lon": 144.974609,
    "distance": 6.05528621E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "tram",
    "stop_id": 3000,
    "location_name": "Punt Rd/Wellington Pde #14 ",
    "lat": -37.81739,
    "lon": 144.989685,
    "distance": 6.06124959E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Melbourne City",
    "transport_type": "tram",
    "stop_id": 2100,
    "location_name": "7D-AAMI Park/Melbourne Park ",
    "lat": -37.8241348,
    "lon": 144.986847,
    "distance": 6.208822E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "bus",
    "stop_id": 25361,
    "location_name": "Wellington Pde/Hoddle St ",
    "lat": -37.8172722,
    "lon": 144.989822,
    "distance": 6.310261E-05
  },
```

```
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "bus",
    "stop_id": 12372,
    "location_name": "Rowena Pde/Punt Rd ",
    "lat": -37.8211479,
    "lon": 144.9895,
    "distance": 6.75393749E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "tram",
    "stop_id": 2827,
    "location_name": "Punt Rd/Bridge Rd #14 ",
    "lat": -37.81753,
    "lon": 144.990219,
    "distance": 6.902158E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "bus",
    "stop_id": 12373,
    "location_name": "Bridge Rd/Hoddle St ",
    "lat": -37.81719,
    "lon": 144.9902,
    "distance": 6.940239E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "nightrider",
    "stop_id": 24271,
    "location_name": "Hoddle St/Bridge Rd ",
    "lat": -37.81766,
    "lon": 144.990509,
    "distance": 7.379541E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "bus",
```

```
    "stop_id": 18126,
    "location_name": "Punt Rd/Olympic Bvd ",
    "lat": -37.8245926,
    "lon": 144.987579,
    "distance": 7.558921E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "bus",
    "stop_id": 25365,
    "location_name": "West Richmond Railway Station/Hoddle St ",
    "lat": -37.8149567,
    "lon": 144.990265,
    "distance": 7.88514662E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "bus",
    "stop_id": 23445,
    "location_name": "Eades St/Victoria Pde ",
    "lat": -37.8091,
    "lon": 144.981567,
    "distance": 7.916382E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "nightrider",
    "stop_id": 23445,
    "location_name": "Eades St/Victoria Pde ",
    "lat": -37.8091,
    "lon": 144.981567,
    "distance": 7.916382E-05
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "tram",
    "stop_id": 2477,
    "location_name": "Smith St/Victoria Pde #15 ",
    "lat": -37.8089752,
    "lon": 144.9827,
    "distance": 8.19254637E-05
  },
```

```
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Richmond",
      "transport_type": "bus",
      "stop_id": 12384,
      "location_name": "West Richmond Railway Station/Hoddle St ",
      "lat": -37.81497,
      "lon": 144.9905,
      "distance": 8.279046E-05
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Melbourne City",
      "transport_type": "tram",
      "stop_id": 3320,
      "location_name": "Collins St/Spring St #0 ",
      "lat": -37.8134956,
      "lon": 144.974,
      "distance": 8.284038E-05
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Fitzroy",
      "transport_type": "tram",
      "stop_id": 2246,
      "location_name": "Smith St/Victoria Pde #15 ",
      "lat": -37.80887,
      "lon": 144.982452,
      "distance": 8.35470055E-05
    },
    "type": "stop"
  }
]
```


Transport POIs by Map

Version Number

2.0.0

Description

Transport POIs by Map returns a set of **locations** consisting of **stops** and/or myki ticket **outlets** (collectively known as points of interest – i.e. POIs) within a region demarcated on a map through a set of latitude and longitude coordinates.

Through the **poi** parameter, the API can return any combination of POIs (e.g. ticket outlets only, bus stops only, tram stops and ticket outlets only, all of the above, and so on).

Where POIs are geographically dispersed they are returned in a list; where they are geographically concentrated they can be returned in a **cluster**, depending on the map **griddepth** that is sent in the request.

Have a play around with the griddepth parameter to see what best suits the device you are developing for.

If you set griddepth to zero it will not cluster.

You can also set a **limit** of how many stops are listed in a cluster. The API will return what the total number of POIs is, however it will only return data for as many POIs are set by the limit. Check out the example response below for a better understanding of how this works.

When there are more POIs in a cluster than the limit, the POIs returned will be determined by a business rule that is hard coded at the server end. The order of priority is V/Line stops first, followed by train, tram, bus, NightRider and, last of all, ticket outlets.

Request URL

base URL
/v2/poi/%@/lat1/%@/long1/%@/lat2/%@/long2/%@/griddepth/%@/limit/%@?devid=%@
&signature=%@

Parameters

poi = a comma separated list of numbers representing the types of POIs you want returned, defined as follows:

0	Train (metropolitan)
1	Tram
2	Bus (metropolitan and regional, but not V/Line)
3	V/Line regional train and coach
4	NightRider
100	Ticket outlet

e.g. “0,1,2,4,100” would return train, tram, bus, NightRider & ticket outlets

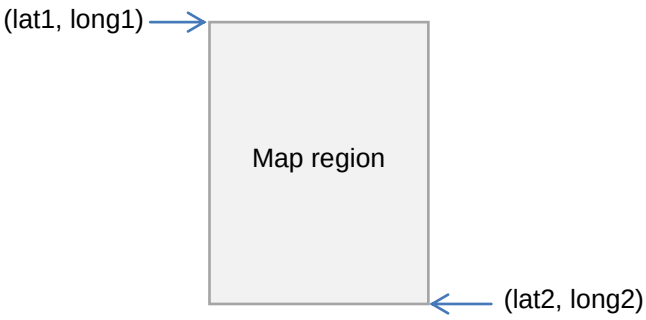
lat1 = latitude at the top left corner of a region depicted on a map, expressed in decimal degrees.*
e.g. -37.82392124423254

long1 = longitude at the top left corner of a region depicted on a map, expressed in decimal degrees.*
e.g. 144.9462017431463

lat2 = latitude at the bottom right corner of a region depicted on a map, expressed in decimal degrees.*
e.g. -37.81540959390813

long2 = longitude at the bottom right corner of a region depicted on a map, expressed in decimal degrees.*
e.g. 144.9542017407848

The coordinate pairs (lat1, long1) and (lat2, long2) are two diagonally opposite corners of the map region of interest, namely:



griddepth = the number of cells per block of cluster grid (between 0-20 inclusive).
e.g. “1” would look like this:

…while “2” would look like:

limit = the minimum number of POIs (stops or outlets) required to create a cluster, as well as the maximum number of POIs returned as part of a cluster in the JSON response (for example, if the limit is “4”, at least 4 POIs are required to form a cluster; and in the JSON response, if there are 7 total locations in a cluster, only 4 will be listed in the response)
e.g. 4

devid = the developer ID supplied in your email from PTV

signature = the customised message digest calculated using the method in the **Quick start guide**

Response

Returns a list of JSON objects which are either **“locations”** or **“clusters”**; “clusters” have their own list of “locations” within them.

“locations” have either a **“stop” or “outlet” (i.e. ticket outlet) object** embedded within them.

For more information on the data structures, check out the **JSON object structure**.

Each **stop and outlet “location”** object has the following attributes:

suburb *string*
– the suburb name
– e.g. “Belgrave”

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "20-Barkly Square/115 Sydney Rd (Brunswick)"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.82005

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.95047

distance *decimal number*
– returns zero in the context of this API

“stop” objects have the following extra attributes:

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “vline” or “nightrider”

stop_id *numeric string*
– the unique identifier of each stop
– e.g. “2171”

While **“outlet”** objects have the following extra attributes:

outlet_type *string (limited values)*
– either “stop” meaning a myki card machine at a station or stop or “retail” meaning a shop of some kind
– e.g. “retail”

business_name *string*
– the business name of the outlet
– e.g. “IGA Victoria Harbour”

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the “location_name” is the name of the station – e.g. “Belgrave Station”.

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number (which is the number that appears on the signage at the stop or in the timetable; not the same as the “stop_id”).

For each set of locations and clusters, it will also return the following objects:

minLat *decimal number*
– the minimum latitude value of all of the locations in the cluster, including those that are not returned (i.e. they are beyond the limit set)**
– e.g. -37.81959

minLong *decimal number*
– the minimum longitude value of all of the locations in the cluster, including

those that are not returned (i.e. they are beyond the limit set)**
– e.g. 144.979126

maxLat *decimal number*
– the maximum latitude value of all of the locations in the cluster, including those that are not returned (i.e. they are beyond the limit set)**
– e.g. -37.8134956

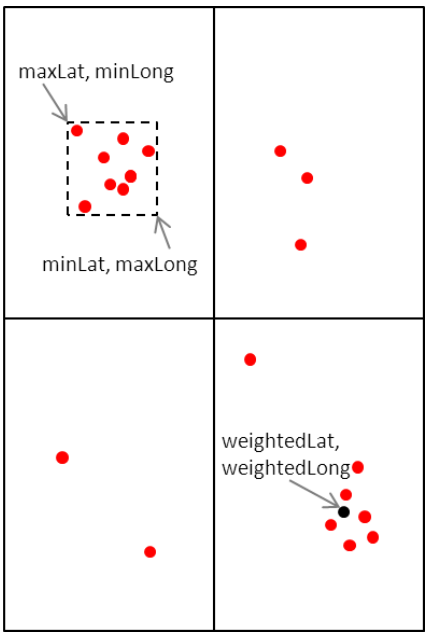
maxLong *decimal number*
– the maximum longitude value of all of the locations in the cluster, including those that are not returned (i.e. they are beyond the limit set)**
– e.g. 144.9854

weightedLat *decimal number*
– latitude at the point that is the average of all POIs returned in a grid cell**
– e.g. -37.81671

weightedLong *decimal number*
– longitudeat the point that is the average of all POIs returned in a grid cell**
– e.g. 144.982849

totalLocations *integer*
– the total number of locations within the region described above
– e.g. 7

** The set of coordinates above describe the following points (sample only):



Example use case

Janelle wants to develop her app further and allow tourists to see public transport stops in an entire region that the tourist has selected on a map. She wantsthe tourists to be able to specify which mode of stops they see (i.e. train, tram, bus, V/Line or NightRider) and also to be able to see myki ticket outlets if they want. Janelle uses the Transport POIs by Map API to do this.

Example location: Area around Brunton Avenue, Richmond, VIC 3002, Australia

Example POIs selected: Train, Tram, Bus, Ticket Outlet

Example request

http://timetableapi.ptv.vic.gov.au/v2/poi/0,1,2,100/lat1/-37.82205143151239/long1/144.9779160007277/lat2/-37.81393456848758/long2/144.9859159992726/griddepth/3/limit/6?devid=4&signature=2BELL8A77A14452DEC110FD849906EBE4F10DC7B

Example response


```
{
  "minLat": -37.81959,
  "minLong": 144.979126,
  "maxLat": -37.8157463,
  "maxLong": 144.9854,
  "weightedLat": -37.81671,
  "weightedLong": 144.982849,
  "totalLocations": 7,
  "locations": [
    {
      "suburb": "Melbourne City",
      "transport_type": "tram",
      "stop_id": 2171,
      "location_name": "7B-Rod Laver Arena/Melbourne Park ",
      "lat": -37.81959,
      "lon": 144.979126,
      "distance": 0.0
    },
    {
      "suburb": "East Melbourne",
      "transport_type": "tram",
      "stop_id": 2823,
      "location_name": "Jolimont Rd/Wellington Pde #10 ",
      "lat": -37.8157463,
      "lon": 144.979782,
      "distance": 0.0
    },
    {
      "suburb": "East Melbourne",
      "transport_type": "tram",
      "stop_id": 2825,
      "location_name": "Clarendon St/Wellington Pde #11 ",
      "lat": -37.81603,
      "lon": 144.9824,
      "distance": 0.0
```


The total "locations" found is 7.

List of "locations" starts here

"location" that is a "stop" – it has a "transport_type" and a "stop_id"

```
  },
  {
    "suburb": "East Melbourne",
    "transport_type": "train",
    "stop_id": 1104,
    "location_name": "Jolimont-MCG ",
    "lat": -37.81653,
    "lon": 144.9841,
    "distance": 0.0
  },
  {
    "outlet_type": "Stop",
    "suburb": "East Melbourne",
    "business_name": "Jolimont Station",
    "location_name": "Wellington Cres",
    "lat": -37.81653,
    "lon": 144.9841,
    "distance": 0.0
  },
  {
    "suburb": "East Melbourne",
    "transport_type": "tram",
    "stop_id": 2824,
    "location_name": "Powlett St/Wellington Pde #12 ",
    "lat": -37.8163261,
    "lon": 144.985016,
    "distance": 0.0
  },
  {
    "outlet_type": "Retail",
    "suburb": "East Melbourne",
    "business_name": "7-Eleven MCG Melbourne",
    "location_name": "142 Wellington Parade",
    "lat": -37.8162231,
    "lon": 144.9854,
    "distance": 0.0
  }
 ],
 "clusters": []
}
```


"location" that is a ticket "outlet" at a train station ("outlet_type" = "stop" and "business_name" is the station name, i.e. "Jolimont Station")

"location" that is a ticket "outlet" at a shop ("outlet_type" = "Retail")

Zero "clusters" of POIs

Search

Version Number

2.0.0

Description

The Search API returns all stops and lines that match the input search text.

Request URL

base URL
/v2/search/%@?&devid=%@&signature=%@

Parameters

search = search text
e.g. “Alamein”

devid = the developer ID supplied in your email from PTV

signature = the customised message digest calculated using the method in the **Quick start guide**

Response

Returns an array of JSON **“result”** objects for which the “type” equals either **“stop” or “line”**.

A **“stop”** object or **“line”** object is embedded within each **“result”** depending on its type.

For more information on the data structures, check out the **JSON object structure**.

“stop” objects have these attributes:

suburb *string*
– the suburb name
– e.g. “Richmond”

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “vline” or “nightrider”

stop_id *numeric string*
– the unique identifier of each stop
– e.g. “12373”

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "Bridge Rd/Hoddle St"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.81719

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.9902

distance *decimal number*
– returns zero in the context of this API

“line” objects have these attributes:

transport_type *string*
– the mode of transport serviced by the line
– e.g. can be either “train”, “tram”, “bus”, “vline” or “nightrider”

line_id *numeric string*
– the unique identifier of each line
– e.g. “1818”

line_name *string*
– the name of the line
– e.g. "970 - City - Frankston - Mornington - Rosebud via Nepean Highway & Frankston Station "

line_number *string*
– the line number that is presented to the public (i.e. not the “line_id”)
– e.g. “970”

For train lines, the line_number will be the same as the line_name (for example, “Alamein”).

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the “location_name” is the name of the station – e.g. “Belgrave Station”.

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the “stop_id”).

Example use case

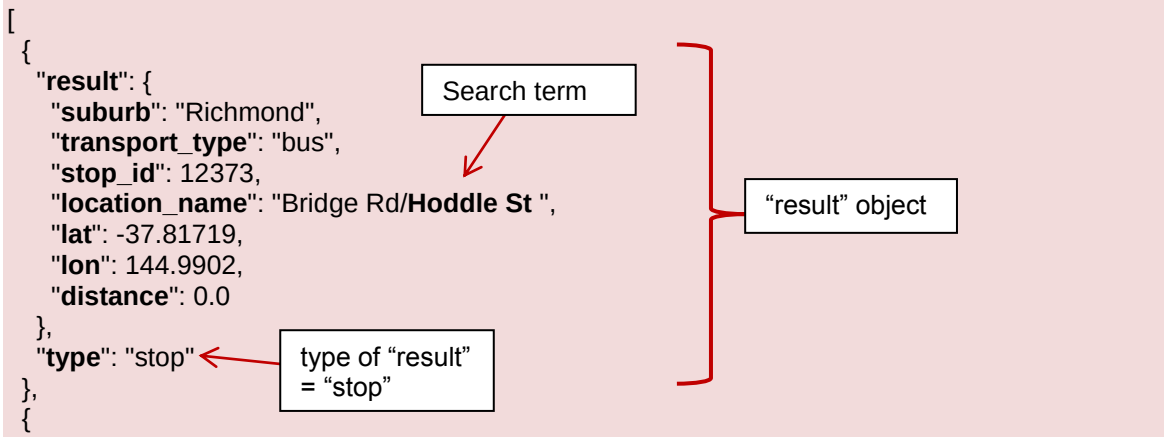
Janelle’s next development for the tourist app is to add a search function that allows tourists to find any stations or stops, as well as any train lines, tram routes or bus routes, by inputting some text. She uses the Search API.

Example search term: “Hoddle St”

Example request

http://timetableapi.ptv.vic.gov.au/v2/search/Hoddle%20St?&devid=4&signature=93121A8B16A7158DB8169DBC405CF7405A05F2C0

Example response




```
[
  {
    "result": {
      "suburb": "Richmond",
      "transport_type": "bus",
      "stop_id": 12373,
      "location_name": "Bridge Rd/Hoddle St ",
      "lat": -37.81719,
      "lon": 144.9902,
      "distance": 0.0
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Clifton Hill",
      "transport_type": "bus",
      "stop_id": 10285,
      "location_name": "Clifton Hill Railway Station/Hoddle St ",
      "lat": -37.7886963,
      "lon": 144.995071,
      "distance": 0.0
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Abbotsford",
      "transport_type": "bus",
      "stop_id": 12427,
      "location_name": "Gipps St/Hoddle St ",
      "lat": -37.80475,
      "lon": 144.992355,
      "distance": 0.0
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "East Melbourne",
      "transport_type": "bus",
      "stop_id": 25376,
      "location_name": "Grey St/Hoddle St ",
      "lat": -37.81243,
      "lon": 144.990723,
      "distance": 0.0
    },
    "type": "stop"
  },
  {
    "result": {
      "suburb": "Richmond",
      "transport_type": "nightrider",
```

```
    "stop_id": 24271,
    "location_name": "Hoddle St/Bridge Rd ",
    "lat": -37.81766,
    "lon": 144.990509,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Essendon",
    "transport_type": "bus",
    "stop_id": 24972,
    "location_name": "Hoddle St/Fletcher St ",
    "lat": -37.7557068,
    "lon": 144.923187,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Essendon",
    "transport_type": "tram",
    "stop_id": 2676,
    "location_name": "Hoddle St/Fletcher St #39 ",
    "lat": -37.7554665,
    "lon": 144.92276,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Abbotsford",
    "transport_type": "bus",
    "stop_id": 12841,
    "location_name": "Hoddle St/Johnston St ",
    "lat": -37.7999763,
    "lon": 144.993561,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Collingwood",
    "transport_type": "bus",
    "stop_id": 13002,
    "location_name": "Hoddle St/Johnston St ",
    "lat": -37.79976,
    "lon": 144.992477,
    "distance": 0.0
  },
```

```
  "type": "stop"
},
{
  "result": {
    "suburb": "Warrnambool",
    "transport_type": "bus",
    "stop_id": 31224,
    "location_name": "Hoddle St/Morriss Rd ",
    "lat": -38.36617,
    "lon": 142.46698,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Clifton Hill",
    "transport_type": "bus",
    "stop_id": 10869,
    "location_name": "Hoddle St/North Tce ",
    "lat": -37.79012,
    "lon": 144.994385,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Collingwood",
    "transport_type": "tram",
    "stop_id": 2472,
    "location_name": "Hoddle St/Victoria Pde #18 ",
    "lat": -37.8097343,
    "lon": 144.990967,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "tram",
    "stop_id": 2471,
    "location_name": "Hoddle St/Victoria St #18 ",
    "lat": -37.80977,
    "lon": 144.991074,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Abbotsford",
    "transport_type": "bus",
```

```
    "stop_id": 13952,
    "location_name": "Johnston St/Hoddle St ",
    "lat": -37.800415,
    "lon": 144.993057,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Collingwood",
    "transport_type": "bus",
    "stop_id": 25429,
    "location_name": "Johnston St/Hoddle St ",
    "lat": -37.7994537,
    "lon": 144.992889,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Collingwood",
    "transport_type": "bus",
    "stop_id": 20219,
    "location_name": "Keele St/Hoddle St ",
    "lat": -37.79744,
    "lon": 144.993317,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Abbotsford",
    "transport_type": "bus",
    "stop_id": 12415,
    "location_name": "Langridge St/Hoddle St ",
    "lat": -37.8069649,
    "lon": 144.992,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Clifton Hill",
    "transport_type": "bus",
    "stop_id": 12460,
    "location_name": "Noone St/Hoddle St ",
    "lat": -37.79351,
    "lon": 144.994141,
    "distance": 0.0
  },
```

```
  "type": "stop"
},
{
  "result": {
    "suburb": "Clifton Hill",
    "transport_type": "bus",
    "stop_id": 12471,
    "location_name": "North Tce/Hoddle St ",
    "lat": -37.79044,
    "lon": 144.994675,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Clifton Hill",
    "transport_type": "bus",
    "stop_id": 25440,
    "location_name": "Parslow St/Hoddle St ",
    "lat": -37.7932968,
    "lon": 144.99411,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "transport_type": "tram",
    "line_id": 5313,
    "line_name": "Route 31 - Hoddle Street - Victoria Harbour Docklands",
    "line_number": "Route 31"
  },
  "type": "line"
},
{
  "result": {
    "suburb": "Abbotsford",
    "transport_type": "bus",
    "stop_id": 12449,
    "location_name": "Truro St/Hoddle St ",
    "lat": -37.79768,
    "lon": 144993561,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Abbotsford",
    "transport_type": "bus",
    "stop_id": 12438,
    "location_name": "Vere St/Hoddle St ",
    "lat": -37.8028221,
```


Search term

"result" object

type of "result" = "line"

```
    "lon": 144.9927,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Collingwood",
    "transport_type": "bus",
    "stop_id": 25418,
    "location_name": "Vere St/Hoddle St ",
    "lat": -37.80218,
    "lon": 144.992462,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "bus",
    "stop_id": 25387,
    "location_name": "Victoria Pde/Hoddle St ",
    "lat": -37.8105965,
    "lon": 144.9911,
    "distance": 0.0
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "Richmond",
    "transport_type": "bus",
    "stop_id": 12405,
    "location_name": "Victoria Pde/Hoddle St ",
    "lat": -37.8103256,
    "lon": 144.991333,
    "distance": 00
  },
  "type": "stop"
},
{
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "bus",
    "stop_id": 25361,
    "location_name": "Wellington Pde/Hoddle St ",
    "lat": -37.8172722,
    "lon": 144.989822,
    "distance": 0.0
  },
  "type": "stop"
},
{
```

```
  "result": {
    "suburb": "Richmond",
    "transport_type": "bus",
    "stop_id": 12384,
    "location_name": "West Richmond Railway Station/Hoddle St ",
    "lat": -37.81497,
    "lon": 144.9905,
    "distance": 0.0
  },
  "type": "stop"
 },
 {
  "result": {
    "suburb": "East Melbourne",
    "transport_type": "bus",
    "stop_id": 25365,
    "location_name": "West Richmond Railway Station/Hoddle St ",
    "lat": -37.8149567,
    "lon": 144.990265,
    "distance": 0.0
  },
  "type": "stop"
 },
 {
  "result": {
    "suburb": "Richmond",
    "transport_type": "bus",
    "stop_id": 12394,
    "location_name": "York St/Hoddle St ",
    "lat": -37.8127022,
    "lon": 144.990967,
    "distance": 0.0
  },
  "type": "stop"
 }
]
```


Broad Next Departures

Version Number

2.0.0

Description

Broad Next Departuresreturns the next departure times at a prescribed stop irrespective of the line and direction of the service.

For example, if the stop is Camberwell Station, Broad Next Departures will return the times for all three lines (Belgrave, Lilydale and Alamein) running in both directions (towards the city and away from the city).

Through the **"limit"** parameter you can choose to return the very next departure or all departures for the day from that point in time.

Request URL

base URL
/v2/mode/%@/stop/%@/departures/by-destination/limit/%@?devid=%@&signature=%@

Parameters

mode = a number representing the **transport_type** of the stop, defined as follows:

0 Train (metropolitan)
1 Tram
2 Bus (metropolitan and regional, but not V/Line)
3 V/Line train and coach
4 NightRider

e.g. "2"

stop = the **stop_id** of the stop
e.g. "1108"

limit = the number of next departure times to be returned, i.e. "5" will return the next five departure times (notes: "0" will return departures for the entire day; "1" will limit it to the very next departure, even if this is a few days away)
e.g. 2

devid = the developer ID supplied in your email from PTV

signature = the customised message digest calculated using the method in the **Quick start guide**

Response

Returns a collection of JSON timetable **"values"** that have a **"platform"** and **"run"** object embedded within them.

The **"platform"** objects have a **"stop"** and **"direction"** object in them, and the **"direction"** object has a **"line"** object within it.

For more information on the data structures, check out the **JSON object structure**.

Timetable **"values"** have the following attributes:

time_timetable_utc *date and time expressed in ISO 8601 UTC format*
– the scheduled time of the service at the stop
– e.g. "2013-11-18T03:21:00Z"

time_realtime_utc *date and time expressed in ISO 8601 UTC format*
– a place holder for the real-time of the service at the stop (for potential future implementation; as no real-time feeds are provided at this time, this returns “null”)
– e.g. “null”

flags *Character*
– a stop may have zero or more flags associated with it, delimited by a “-“ character; examples include:

RR = Reservations Required
GC = Guaranteed Connection
DOO = Drop Off Only
PUO = Pick Up Only
MO = Mondays only
TU = Tuesdays only
WE = Wednesdays only
TH = Thursdays only
FR = Fridays only
SS = School days only

note: ignore “E” flag

– e.g. “RR-PUO”

“run” objects have the following attributes:

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “vline” or “nightrider”

run_id *numeric string*
– the unique identifier of each run
– e.g. “1464”

num_skipped *integer*
– the number of stops skipped for the run, applicable to train;a number greater than zero indicates either a limited express or express service
– e.g. 0

destination_id *numeric string*
– the **stop_id** of the destination, i.e. the last stop for the run
– e.g. “1044”

destination_name *string*
– the **location_name** of the destination, i.e. the last stop for the run
– e.g. “Craigieburn”

“platform” objects have the following attributes:

realtime_id *string*
– a place holder for the stop’s real-time feed system ID (for potential future implementation; as no real-time feeds are provided at this time, this returns “0”)
– e.g. “0”

“stop” objects have these attributes:

suburb *string*
– the suburb name
– e.g. “Belgrave”

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “V/Line” or “NightRider”

stop_id *numeric string*
– the unique identifier of each stop
– e.g. “1234”

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "20-Barkly Square/115 Sydney Rd (Brunswick)"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.82005

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.95047

distance *decimal number*
–returns zero in the context of this API

“direction” objects have the following attributes:

linedir_id *numeric string*
– unique identifier ofa particular line and direction
– e.g. “21”

direction_id *numeric string*
– unique identifier of a direction (e.g. “0” signifies “city”)
– e.g. “0”

direction_name *string*
– name of the direction of the service
– e.g. "City (Flinders Street)"

“line” objects have these attributes:

transport_type *string*
– the mode of transport serviced by the line
– e.g. can be either “train”, “tram”, “bus”, “V/Line” or “NightRider”

line_id *numeric string*
– the unique identifier of each line
– e.g. “1818”

line_name *string*
– the name of the line
– e.g. "970 - City - Frankston - Mornington - Rosebud via Nepean Highway & Frankston Station "

line_number ..... *string*
– the line number that is presented to the public (i.e. not the “line_id”)
– e.g. “970”

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the “location_name” is the name of the station – e.g. “Belgrave Station”.

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the “stop_id”).

For train lines, the line_number will be the same as the line_name (for example, “Alamein”).

Example use case

Janelle has decided to add some timetable information to the tourist app. The next development lets tourists see the next departure times for any of the stations or stops that the tourist selects from a map or list.

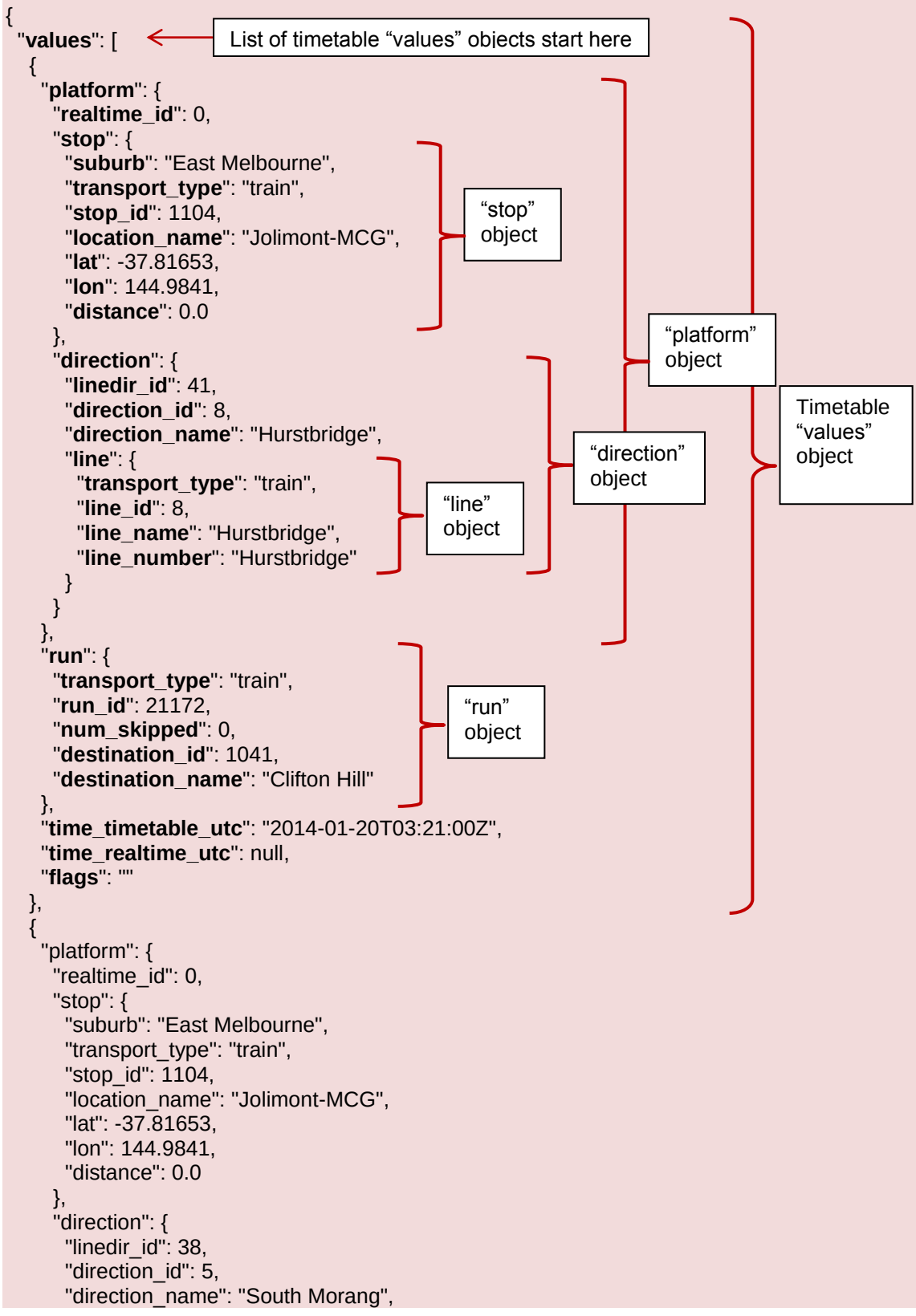
Janelle uses the Broad Next Departures API to show the departure times for stops found via any of the three methods available (Stops Nearby, Transport POIs by Map or Search).

Example stop selected: Jolimont - MCG Train Station (stop_id: 1104)

Example request

http://timetableapi.ptv.vic.gov.au/v2/mode/0/stop/1104/departures/by-destination/limit/1?devid=4&signature=2BEBBA8A77A24452DEC040F849906EBE4F10DA7D

Example response



```
      "line": {
        "transport_type": "train",
        "line_id": 5,
        "line_name": "South Morang",
        "line_number": "South Morang"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 13975,
    "num_skipped": 0,
    "destination_id": 1224,
    "destination_name": "South Morang"
  },
  "time_timetable_utc": "2014-01-20T03:21:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "East Melbourne",
      "transport_type": "train",
      "stop_id": 1104,
      "location_name": "Jolimont-MCG",
      "lat": -37.81653,
      "lon": 144.9841,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 26,
      "direction_id": 0,
      "direction_name": "City (Flinders Street)",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21045,
    "num_skipped": 0,
    "destination_id": 1155,
    "destination_name": "Parliament"
  },
  "time_timetable_utc": "2014-01-20T03:23:00Z",
  "time_realtime_utc": null,
  "flags": ""
}
```

```
 ]
}
```

Specific Next Departures

Version Number

2.0.0

Description

Specific Next Departures returns the times for the next departures at a prescribed stop for a specific mode, line and direction.

For example, if the stop is Camberwell Station, Specific Next Departures returns only the times for one line running in one direction (for example, the Belgrave line running towards the city).

> Through the **"limit"** parameter you can choose how many departure times to display. Setting the limit to zero will return departures for the entire day which works best when setting the date and time to midnight local time (in UTC format).

> The optional parameter **"for_utc"** allows you to set the time when departures should be returned from (the default time is the time of the query).

Request URL

base URL
/v2/mode/%@/line/%@/stop/%@/directionid/%@/departures/all/limit/%@?for_utc=%@
&devid=%@&signature=%@

Parameters

mode = a number representing the **transport_type** of the stop, defined as follows:

0	Train (metropolitan)
1	Tram
2	Bus (metropolitan and regional, but not V/Line)
3	V/Line train and coach
4	NightRider

e.g. "0"

line = the **line_id** of the requested services
e.g. "3"

stop = the **stop_id** of the stop
e.g. "1108"

directionid = the **direction_id** of the requested services
e.g. “0”

limit = the number of next departure times to be returned, i.e. “5” will return the next five departure times (notes: “0” will return departures for the entire day; “1” will limit it to the very next departure, even if this is a few days away)
e.g. 2

for_utc = *optional:* the date and time of the request in ISO 8601 UTC format
e.g. 2013-11-13T07:08:03Z

devid = the developer ID supplied in your email from PTV

signature = the customised message digest calculated using the method in the **Quick start guide**

Response

Returns a collection of JSON timetable **“values”** that have a **“platform”** and **“run”** object embedded within them.

The **“platform”** objects have a **“stop”** and **“direction”** object in them, and the **“direction”** object has a **“line”** object within it.

For more information on the data structures, check out the **JSON object structure**.

Timetable **“values”** have the following attributes:

time_timetable_utc *date and time expressed in ISO 8601 UTC format*
– the scheduled time of the service at the stop
– e.g. "2013-11-18T03:21:00Z"

time_realtime_utc *date and time expressed in ISO 8601 UTC format*
– a place holder for the real-time of the service at the stop (for potential future implementation; as no real-time feeds are provided at this time, this returns “null”)
– e.g. “null”

flags *Character*
– a stop may have zero or more flags associated with it, delimited by a “-“ character; examples include:

RR = Reservations Required
GC = Guaranteed Connection
DOO = Drop Off Only
PUO = Pick Up Only
MO = Mondays only
TU = Tuesdays only
WE = Wednesdays only
TH = Thursdays only
FR = Fridays only
SS = School days only

note: ignore “E” flag

– e.g. “RR-PUO”

“run” objects have the following attributes:

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “vline” or “nightrider”

run_id *numeric string*
– the unique identifier of each run
– e.g. “1464”

num_skipped *integer*
– the number of stops skipped for the run, applicable to train; a number greater than zero indicates either a limited express or express service
– e.g. 0

destination_id *numeric string*
– the **stop_id** of the destination, i.e. the last stop for the run
– e.g. “1044”

destination_name *string*
– the **location_name** of the destination, i.e. the last stop for the run
– e.g. “Craigieburn”

“platform” objects have the following attributes:

realtime_id *string*
– a place holder for the stop’s real-time feed system ID (for potential future implementation; as no real-time feeds are provided at this time, this returns “0”)
– e.g. “0”

“stop” objects have these attributes:

suburb *string*
– the suburb name
– e.g. “Belgrave”

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “V/Line” or “NightRider”

stop_id *numeric string*
– the unique identifier of each stop
– e.g. “1234”

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "20-Barkly Square/115 Sydney Rd (Brunswick)"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.82005

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.95047

distance *decimal number*
–returns zero in the context of this API

“direction” objects have the following attributes:

linedir_id *numeric string*
– unique identifier of a particular line and direction
– e.g. “21”

direction_id *numeric string*
– unique identifier of a direction
– e.g. “0”

direction_name *string*
– name of the direction of the service (e.g. “0” signifies “city”)
– e.g. "City (Flinders Street)"

“line” objects have these attributes:

transport_type *string*
– the mode of transport serviced by the line
– e.g. can be either “train”, “tram”, “bus”, “V/Line” or “NightRider”

line_id *numeric string*
– the unique identifier of each line
– e.g. “1818”

line_name *string*
– the name of the line
– e.g. "970 - City - Frankston - Mornington - Rosebud via Nepean Highway & Frankston Station "

line_number *string*
– the line number that is presented to the public (i.e. not the “line_id”)
– e.g. “970”

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the “location_name” is the name of the station – e.g. “Belgrave Station”.

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the “stop_id”).

For train lines, the line_number will be the same as the line_name (for example, “Alamein”).

Example use case

Janelle’s next enhancement for the tourist app is to let tourists choose which departure times they see for any given stop, by selecting the line and direction.

This will mean that if a stop or station has multiple routes or lines stopping there (for example, Flinders Street Station), the tourist won’t be bombarded with a confusing list of departure times for multiple lines.

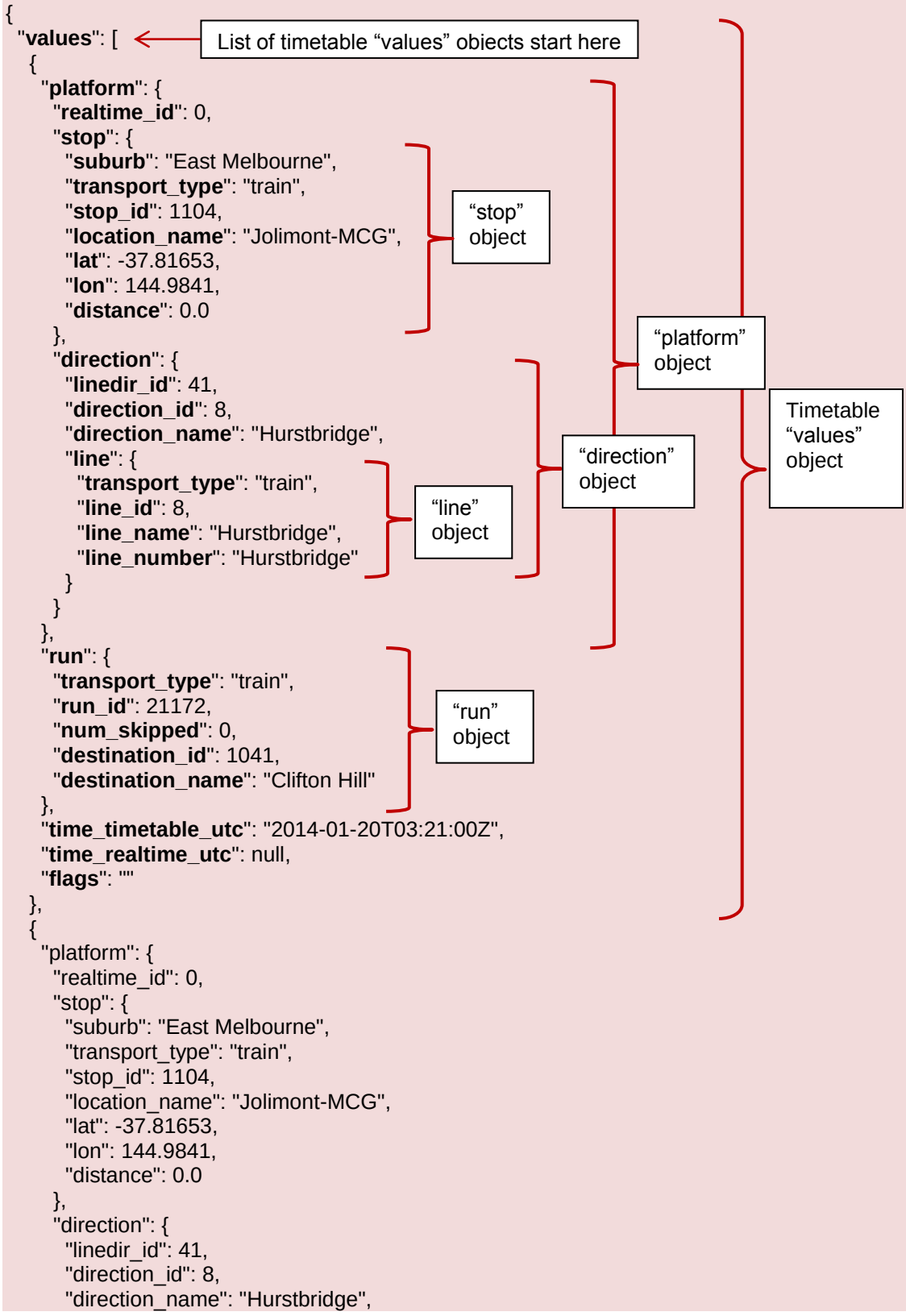
Building on the other APIs, Janelle uses the Specific Next Departures API to do this.

Example selection: Jolimont-MCG Train Station towards Hurstbridge

Example request

http://timetableapi.ptv.vic.gov.au/v2/mode/0/line/8/stop/1104/directionid/8/departures/all/limit/5?for_utc=2014-01-20T03:18:08Z&devid=4&signature=2BEBB8A77A24452FAF110FD849906EBE4F10DC7B

Example response



```
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 1062,
    "destination_name": "Eltham"
  },
  "time_timetable_utc": "2014-01-20T03:31:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "East Melbourne",
      "transport_type": "train",
      "stop_id": 1104,
      "location_name": "Jolimont-MCG",
      "lat": -37.81653,
      "lon": 144.9841,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 41,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21174,
    "num_skipped": 0,
    "destination_id": 1041,
    "destination_name": "Clifton Hill"
  },
  "time_timetable_utc": "2014-01-20T03:41:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
```

```
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "East Melbourne",
      "transport_type": "train",
      "stop_id": 1104,
      "location_name": "Jolimont-MCG",
      "lat": -37.81653,
      "lon": 144.9841,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 41,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21175,
    "num_skipped": 0,
    "destination_id": 1100,
    "destination_name": "Hurstbridge"
  },
  "time_timetable_utc": "2014-01-20T03:51:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "East Melbourne",
      "transport_type": "train",
      "stop_id": 1104,
      "location_name": "Jolimont-MCG",
      "lat": -37.81653,
      "lon": 144.9841,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 41,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
```

```
          "line_name": "Hurstbridge",
          "line_number": "Hurstbridge"
        }
      }
    },
    "run": {
      "transport_type": "train",
      "run_id": 21176,
      "num_skipped": 0,
      "destination_id": 1041,
      "destination_name": "Clifton Hill"
    },
    "time_timetable_utc": "2014-01-20T04:01:00Z",
    "time_realtime_utc": null,
    "flags": ""
  }
 ]
}
```


Stopping Pattern

Version Number

2.0.0

Description

The Stopping Pattern API returns the stopping pattern for a specific run (i.e. transport service) from a prescribed stop at a prescribed time. The stopping pattern is comprised of timetable values ordered by stopping order.

Request URL

base URL
/v2/mode/%@/run/%@/stop/%@/stopping-pattern?for_utc=%@&devid=%@&signature=%@

Parameters

mode = a number representing the **transport_type** of the stop, defined as follows:

0 Train (metropolitan)
1 Tram
2 Bus (metropolitan and regional, but not V/Line)
3 V/Line train and coach
4 NightRider

e.g. “2”

run	=	the **run_id** of the requested run e.g. “1464”
stop	=	the **stop_id** of the stop e.g. “1108”
for_utc	=	the date and time of the request in ISO 8601 UTC format e.g. 2013-11-13T05:24:25Z
devid	=	the developer ID supplied in your email from PTV
signature	=	the customised message digest calculated using the method in the Quick start guide

Response

Returns a collection of JSON timetable **“values”** that have a **“platform”** and **“run”** object embedded within them.

The **“platform”** objects have a **“stop”** and **“direction”** object in them, and the **“direction”** object has a **“line”** object within it.

> The order of the timetable **“value”** objects reflects the stopping pattern.

For more information on the data structures, check out the JSON object structure.

Timetable **“values”** have the following attributes:

time_timetable_utc *date and time expressed in ISO 8601 UTC format*
– the scheduled time of the service at the stop
– e.g. "2013-11-18T03:21:00Z"

time_realtime_utc *date and time expressed in ISO 8601 UTC format*
– a place holder for the real-time of the service at the stop (for potential future implementation; as no real-time feeds are provided at this time, this returns “null”)
– e.g. “null”

flags *Character*
– a stop may have zero or more flags associated with it, delimited by a “-“ character; examples include:

RR = Reservations Required
GC = Guaranteed Connection
DOO = Drop Off Only
PUO = Pick Up Only
MO = Mondays only
TU = Tuesdays only
WE = Wednesdays only
TH = Thursdays only
FR = Fridays only
SS = School days only

note: ignore “E” flag

– e.g. “RR-PUO”

“run” objects have the following attributes:

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “vline” or “nightrider”

run_id *numeric string*
– the unique identifier of each run
– e.g. “1464”

num_skipped *integer*
– the number of stops skipped for the run, applicable to train;a number greater than zero indicates either a limited express or express service
– e.g. 0

destination_id *numeric string*
– the **stop_id** of the destination, i.e. the last stop for the run
– e.g. “1044”

destination_name *string*
– the **location_name** of the destination, i.e. the last stop for the run
– e.g. “Craigieburn”

“platform” objects have the following attributes:

realtime_id *string*
– a place holder for the stop’s real-time feed system ID (for potential future implementation; as no real-time feeds are provided at this time, this returns “0”)
– e.g. “0”

“stop” objects have these attributes:

suburb *string*
– the suburb name
– e.g. “Belgrave”

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either “train”, “tram”, “bus”, “V/Line” or “NightRider”

stop_id *numeric string*
– the unique identifier of each stop
– e.g. “1234”

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "20-Barkly Square/115 Sydney Rd (Brunswick)"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.82005

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.95047

distance *decimal number*
–returns zero in the context of this API

“direction” objects have the following attributes:

linedir_id *numeric string*
– unique identifier ofa particular line and direction
– e.g. “21”

direction_id *numeric string*
– unique identifier of a direction
– e.g. “0”

direction_name *string*
– name of the direction of the service (e.g. “0” signifies “city”)
– e.g. "City (Flinders Street)"

“line” objects have these attributes:

transport_type *string*
– the mode of transport serviced by the line
– e.g. can be either “train”, “tram”, “bus”, “V/Line” or “NightRider”

line_id *numeric string*
– the unique identifier of each line
– e.g. “1818”

line_name *string*
– the name of the line
– e.g. "970 - City - Frankston - Mornington - Rosebud via Nepean Highway & Frankston Station "

line_number *string*
– the line number that is presented to the public (i.e. not the “line_id”)
– e.g. “970”

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the “location_name” is the name of the station – e.g. “Belgrave Station”.

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the “stop_id”).

For train lines, the line_number will be the same as the line_name (for example, “Alamein”).

Example use case

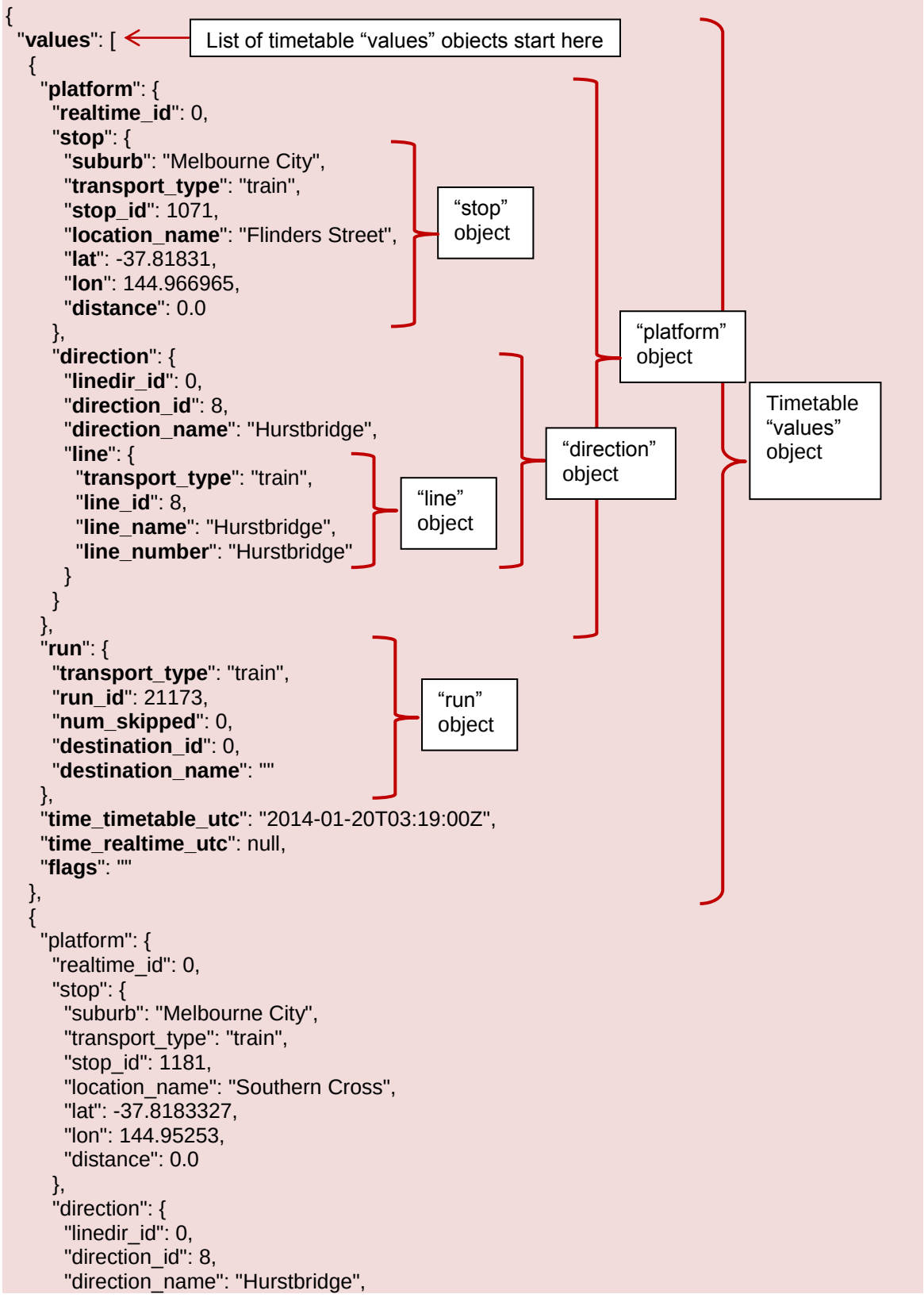
Next Janelle wants to enhance her tourist app so it can show a basic timetable for any of the departure times selected by tourists (i.e. returned through Broad Next Departures). She uses the Stopping Pattern API to do this.

Example selection: 2:31pm service departing Jolimont-MCG Train Station on the Hurstbridge Line towards Eltham

Example request

http://timetableapi.ptv.vic.gov.au/v2/mode/0/run/21173/stop/1104/stopping-pattern?for_utc=2014-01-20T03:18:08Z&devid=4&signature=2CACC8A77A24452DEC110FD948906EBE4F10DC7B

Example response



```
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:22:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Melbourne City",
      "transport_type": "train",
      "stop_id": 1068,
      "location_name": "Flagstaff",
      "lat": -37.8119774,
      "lon": 144.955658,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:24:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
```

```
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Melbourne City",
      "transport_type": "train",
      "stop_id": 1120,
      "location_name": "Melbourne Central",
      "lat": -37.8099365,
      "lon": 144.9626,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:26:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Melbourne City",
      "transport_type": "train",
      "stop_id": 1155,
      "location_name": "Parliament",
      "lat": -37.8110542,
      "lon": 144.9729,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
```

```
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:28:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "East Melbourne",
      "transport_type": "train",
      "stop_id": 1104,
      "location_name": "Jolimont-MCG",
      "lat": -37.81653,
      "lon": 144.9841,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:31:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
```

```
    "stop": {
      "suburb": "Richmond",
      "transport_type": "train",
      "stop_id": 1207,
      "location_name": "West Richmond",
      "lat": -37.8149452,
      "lon": 144.991425,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:32:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Richmond",
      "transport_type": "train",
      "stop_id": 1145,
      "location_name": "North Richmond",
      "lat": -37.8103943,
      "lon": 144.9925,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
```

```
    }
   },
   "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
   },
   "time_timetable_utc": "2014-01-20T03:34:00Z",
   "time_realtime_utc": null,
   "flags": ""
  },
  {
   "platform": {
    "realtime_id": 0,
    "stop": {
     "suburb": "Abbotsford",
     "transport_type": "train",
     "stop_id": 1043,
     "location_name": "Collingwood",
     "lat": -37.8045235,
     "lon": 144.993744,
     "distance": 0.0
    },
    "direction": {
     "linedir_id": 0,
     "direction_id": 8,
     "direction_name": "Hurstbridge",
     "line": {
      "transport_type": "train",
      "line_id": 8,
      "line_name": "Hurstbridge",
      "line_number": "Hurstbridge"
     }
    }
   },
   "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
   },
   "time_timetable_utc": "2014-01-20T03:35:00Z",
   "time_realtime_utc": null,
   "flags": ""
  },
  {
   "platform": {
    "realtime_id": 0,
    "stop": {
     "suburb": "Abbotsford",
     "transport_type": "train",
```

```
      "stop_id": 1201,
      "location_name": "Victoria Park",
      "lat": -37.7991562,
      "lon": 144.994446,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:37:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Clifton Hill",
      "transport_type": "train",
      "stop_id": 1041,
      "location_name": "Clifton Hill",
      "lat": -37.7886543,
      "lon": 144.995422,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
```

```
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
   },
   "time_timetable_utc": "2014-01-20T03:39:00Z",
   "time_realtime_utc": null,
   "flags": ""
  },
  {
   "platform": {
    "realtime_id": 0,
    "stop": {
     "suburb": "Northcote",
     "transport_type": "train",
     "stop_id": 1209,
     "location_name": "Westgarth",
     "lat": -37.7806168,
     "lon": 144.999237,
     "distance": 0.0
    },
    "direction": {
     "linedir_id": 0,
     "direction_id": 8,
     "direction_name": "Hurstbridge",
     "line": {
      "transport_type": "train",
      "line_id": 8,
      "line_name": "Hurstbridge",
      "line_number": "Hurstbridge"
     }
    }
   },
   "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
   },
   "time_timetable_utc": "2014-01-20T03:41:00Z",
   "time_realtime_utc": null,
   "flags": ""
  },
  {
   "platform": {
    "realtime_id": 0,
    "stop": {
     "suburb": "Northcote",
     "transport_type": "train",
     "stop_id": 1053,
     "location_name": "Dennis",
     "lat": -37.7791824,
```

```
      "lon": 145.00824,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:43:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Fairfield",
      "transport_type": "train",
      "stop_id": 1065,
      "location_name": "Fairfield",
      "lat": -37.7791748,
      "lon": 145.0169,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
```

```
    "destination_id": 0,
    "destination_name": ""
   },
   "time_timetable_utc": "2014-01-20T03:44:00Z",
   "time_realtime_utc": null,
   "flags": ""
  },
  {
   "platform": {
    "realtime_id": 0,
    "stop": {
     "suburb": "Alphington",
     "transport_type": "train",
     "stop_id": 1004,
     "location_name": "Alphington",
     "lat": -37.7783966,
     "lon": 145.03125,
     "distance": 0.0
    },
    "direction": {
     "linedir_id": 0,
     "direction_id": 8,
     "direction_name": "Hurstbridge",
     "line": {
      "transport_type": "train",
      "line_id": 8,
      "line_name": "Hurstbridge",
      "line_number": "Hurstbridge"
     }
    }
   },
   "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
   },
   "time_timetable_utc": "2014-01-20T03:46:00Z",
   "time_realtime_utc": null,
   "flags": ""
  },
  {
   "platform": {
    "realtime_id": 0,
    "stop": {
     "suburb": "Ivanhoe",
     "transport_type": "train",
     "stop_id": 1050,
     "location_name": "Darebin",
     "lat": -37.7749634,
     "lon": 145.038483,
     "distance": 0.0
    },
```

```
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:48:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Ivanhoe",
      "transport_type": "train",
      "stop_id": 1101,
      "location_name": "Ivanhoe",
      "lat": -37.768898,
      "lon": 145.045425,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
```

```
  "time_timetable_utc": "2014-01-20T03:50:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Eaglemont",
      "transport_type": "train",
      "stop_id": 1056,
      "location_name": "Eaglemont",
      "lat": -37.7635841,
      "lon": 145.05394,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:51:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Heidelberg",
      "transport_type": "train",
      "stop_id": 1093,
      "location_name": "Heidelberg",
      "lat": -37.7570763,
      "lon": 145.060684,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
```

```
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:53:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Rosanna",
      "transport_type": "train",
      "stop_id": 1168,
      "location_name": "Rosanna",
      "lat": -37.7428741,
      "lon": 145.066147,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T03:56:00Z",
  "time_realtime_utc": null,
  "flags": ""
```

```
  },
  {
    "platform": {
      "realtime_id": 0,
      "stop": {
        "suburb": "Macleod",
        "transport_type": "train",
        "stop_id": 1117,
        "location_name": "Macleod",
        "lat": -37.72601,
        "lon": 145.069153,
        "distance": 0.0
      },
      "direction": {
        "linedir_id": 0,
        "direction_id": 8,
        "direction_name": "Hurstbridge",
        "line": {
          "transport_type": "train",
          "line_id": 8,
          "line_name": "Hurstbridge",
          "line_number": "Hurstbridge"
        }
      }
    },
    "run": {
      "transport_type": "train",
      "run_id": 21173,
      "num_skipped": 0,
      "destination_id": 0,
      "destination_name": ""
    },
    "time_timetable_utc": "2014-01-20T03:59:00Z",
    "time_realtime_utc": null,
    "flags": ""
  },
  {
    "platform": {
      "realtime_id": 0,
      "stop": {
        "suburb": "Watsonia",
        "transport_type": "train",
        "stop_id": 1203,
        "location_name": "Watsonia",
        "lat": -37.7109528,
        "lon": 145.0838,
        "distance": 0.0
      },
      "direction": {
        "linedir_id": 0,
        "direction_id": 8,
        "direction_name": "Hurstbridge",
        "line": {
          "transport_type": "train",
```

```
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T04:02:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Greensborough",
      "transport_type": "train",
      "stop_id": 1084,
      "location_name": "Greensborough",
      "lat": -37.70395,
      "lon": 145.108246,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T04:08:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
```

```
    "realtime_id": 0,
    "stop": {
      "suburb": "Montmorency",
      "transport_type": "train",
      "stop_id": 1130,
      "location_name": "Montmorency",
      "lat": -37.7152977,
      "lon": 145.1215,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
      }
    }
  },
  "run": {
    "transport_type": "train",
    "run_id": 21173,
    "num_skipped": 0,
    "destination_id": 0,
    "destination_name": ""
  },
  "time_timetable_utc": "2014-01-20T04:11:00Z",
  "time_realtime_utc": null,
  "flags": ""
},
{
  "platform": {
    "realtime_id": 0,
    "stop": {
      "suburb": "Eltham",
      "transport_type": "train",
      "stop_id": 1062,
      "location_name": "Eltham",
      "lat": -37.71355,
      "lon": 145.147827,
      "distance": 0.0
    },
    "direction": {
      "linedir_id": 0,
      "direction_id": 8,
      "direction_name": "Hurstbridge",
      "line": {
        "transport_type": "train",
        "line_id": 8,
        "line_name": "Hurstbridge",
        "line_number": "Hurstbridge"
```

```
        }
      }
    },
    "run": {
      "transport_type": "train",
      "run_id": 21173,
      "num_skipped": 0,
      "destination_id": 0,
      "destination_name": ""
    },
    "time_timetable_utc": "2014-01-20T04:15:00Z",
    "time_realtime_utc": null,
    "flags": "E"
  }
 ]
}
```


Stops on a Line

Version Number

2.0.0

Description

The Stops on a Line API returns a list of all the stops for a requested line, ordered by location name.

Request URL

base URL
/v2/mode/%@/line/%@/stops-for-line?devid=%@&signature=%@

Parameters

mode = a number representing the **transport_type** of the stop, defined as follows:

0 Train (metropolitan)
1 Tram
2 Bus (metropolitan and regional, but not V/Line)
3 V/Line train and coach
4 NightRider

e.g. “2”

line = the **line_id** of the requested line
e.g. “1818”

devid = the developer ID supplied in your email from PTV

signature = the customised message digest calculated using the method in the **Quick start guide**

Response

Returns a collection of JSON **"stop"** objects, with the attributes below, ordered by **location_name**:

suburb *string*
– the suburb name
– e.g. "Belgrave"

transport_type *string*
– the mode of transport serviced by the stop
– e.g. can be either "train", "tram", "bus", "V/Line" or "NightRider"

stop_id *numeric string*
– the unique identifier of each stop
– e.g. "1108"

location_name *string*
– the name of the stop based on a concise geographic description
– e.g. "20-Barkly Square/115 Sydney Rd (Brunswick)"

lat *decimal number*
– geographic coordinate of latitude
– e.g. -37.82005

lon *decimal number*
– geographic coordinate of longitude
– e.g. 144.95047

distance *decimal number*
– not used; returns zero

GPS coordinates for stops are mostlyto 6 decimal places. This identifies a location to sub meter accuracy.

For train stations, the "location_name" is the name of the station – e.g. "Belgrave Station".

For tram and bus stops, it is a concise geographic descriptor that is determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

Depending on the content of those fields the location name can be Landmark/Travel Street, or Cross Street/Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the "stop_id").

For more information on the data structures, check out the **JSON object structure**.

Example use case

Janelle's last development for the app is to help tourists understand where different trains, trams and buses go – especially bus routes which tend to be a bit less obvious.

Building on the previous APIs, she uses the Stops on a Line API to do this.

Example selection: Route 901 - Frankston - Melbourne Airport (SMARTBUS Service)

Example request

http://timetableapi.ptv.vic.gov.au/v2/mode/2/line/7531/stops-for-line?devid=4&signature=2BFFB8A77A24452CED110FD869906EBE4F10DC7B

Example response

Actual response returns 336 stops; only a selection is shown here

"stop" object


```
[
 {
  "suburb": "Plenty",
  "transport_type": "bus",
  "stop_id": 28066,
  "location_name": "200 Yan Yean Rd ",
  "lat": -37.6616554,
  "lon": 145.124359,
  "distance": 0.0
 },
 {
  "suburb": "Wantirna South",
  "transport_type": "bus",
  "stop_id": 15968,
  "location_name": "500 Stud Rd ",
  "lat": -37.8816528,
  "lon": 145.232788,
  "distance": 0.0
 },
 {
  "suburb": "Scoresby",
  "transport_type": "bus",
  "stop_id": 15150,
  "location_name": "500 Stud Rd ",
  "lat": -37.8827934,
  "lon": 145.233047,
  "distance": 0.0
 },
 {
  "suburb": "Dandenong South",
  "transport_type": "bus",
  "stop_id": 19958,
  "location_name": "Abbotts Rd/Frankston-Dandenong Rd ",
  "lat": -38.0308762,
  "lon": 145.211945,
  "distance": 0.0
```

```
},
{
  "suburb": "Wantirna",
  "transport_type": "bus",
  "stop_id": 15980,
  "location_name": "Ainsdale Ave/Boronia Rd ",
  "lat": -37.8485641,
  "lon": 145.2304,
  "distance": 0.0
},
{
  "suburb": "Montmorency",
  "transport_type": "bus",
  "stop_id": 30223,
  "location_name": "Airlie Rd/Para Rd ",
  "lat": -37.7228165,
  "lon": 145.113159,
  "distance": 0.0
},
{
  "suburb": "Blackburn",
  "transport_type": "bus",
  "stop_id": 22000,
  "location_name": "Albert St/Railway Rd ",
  "lat": -37.8198967,
  "lon": 145.152145,
  "distance": 0.0
},
{
  "suburb": "Nunawading",
  "transport_type": "bus",
  "stop_id": 30228,
  "location_name": "Alexander St/Whitehorse Rd ",
  "lat": -37.81692,
  "lon": 145.189316,
  "distance": 0.0
},
{
  "suburb": "Carrum Downs",
  "transport_type": "bus",
  "stop_id": 19965,
  "location_name": "Amayla Cres/Frankston-Dandenong Rd ",
  "lat": -38.10478,
  "lon": 145.168015,
  "distance": 0.0
},
{
  "suburb": "Dandenong South",
  "transport_type": "bus",
  "stop_id": 19950,
  "location_name": "Amberley Cres/Frankston-Dandenong Rd ",
  "lat": -38.00275,
  "lon": 145.21785,
  "distance": 0.0
```

```
},
{
  "suburb": "Wantirna",
  "transport_type": "bus",
  "stop_id": 15976,
  "location_name": "Amesbury Ave/Boronia Rd ",
  "lat": -37.8513374,
  "lon": 145.239075,
  "distance": 0.0
},
{
  "suburb": "Doncaster East",
  "transport_type": "bus",
  "stop_id": 21008,
  "location_name": "Andersons Creek Rd/Blackburn Rd ",
  "lat": -37.77661,
  "lon": 145.164108,
  "distance": 0.0
},
{
  "suburb": "Plenty",
  "transport_type": "bus",
  "stop_id": 29697,
  "location_name": "Aqueduct Rd/Diamond Creek Rd ",
  "lat": -37.6784325,
  "lon": 145.1248,
  "distance": 0.0
},
{
  "suburb": "Ringwood",
  "transport_type": "bus",
  "stop_id": 15990,
  "location_name": "Arlington St/Wantirna Rd ",
  "lat": -37.81924,
  "lon": 145.227249,
  "distance": 0.0
},
{
  "suburb": "Scoresby",
  "transport_type": "bus",
  "stop_id": 15148,
  "location_name": "Armin St/Stud Rd ",
  "lat": -37.88876,
  "lon": 145.233,
  "distance": 0.0
},
{
  "suburb": "South Morang",
  "transport_type": "bus",
  "stop_id": 25503,
  "location_name": "Armstrong Rd/Kurrak Rd ",
  "lat": -37.65187,
  "lon": 145.109146,
  "distance": 0.0
```

```
},
{
  "suburb": "Blackburn",
  "transport_type": "bus",
  "stop_id": 30212,
  "location_name": "Ashburn Pl/Whitehorse Rd ",
  "lat": -37.81816,
  "lon": 145.162079,
  "distance": 0.0
},
{
  "suburb": "Templestowe",
  "transport_type": "bus",
  "stop_id": 21648,
  "location_name": "Atkinson St/Williamsons Rd ",
  "lat": -37.75579,
  "lon": 145.134674,
  "distance": 0.0
},
{
  "suburb": "Templestowe",
  "transport_type": "bus",
  "stop_id": 22124,
  "location_name": "Aumann Dr/Reynolds Rd ",
  "lat": -37.7623444,
  "lon": 145.157822,
  "distance": 0.0
},
{
  "suburb": "Rowville",
  "transport_type": "bus",
  "stop_id": 15956,
  "location_name": "Avalon Rd/Stud Rd ",
  "lat": -37.9222755,
  "lon": 145.234085,
  "distance": 0.0
},
{
  "suburb": "Greensborough",
  "transport_type": "bus",
  "stop_id": 10883,
  "location_name": "Avandina Cres/Diamond Creek Rd ",
  "lat": -37.6934547,
  "lon": 145.11232,
  "distance": 0.0
},
{
  "suburb": "Dallas",
  "transport_type": "bus",
  "stop_id": 26077,
  "location_name": "Avoca St/Pascoe Vale Rd ",
  "lat": -37.6668053,
  "lon": 144.923279,
  "distance": 0.0
```

```
},
{
  "suburb": "Doncaster East",
  "transport_type": "bus",
  "stop_id": 21000,
  "location_name": "Avocet St/Blackburn Rd ",
  "lat": -37.79577,
  "lon": 145.160645,
  "distance": 0.0
},
{
  "suburb": "Mitcham",
  "transport_type": "bus",
  "stop_id": 30209,
  "location_name": "Barkly Tce/Whitehorse Rd ",
  "lat": -37.81583,
  "lon": 145.198563,
  "distance": 0.0
},
{
  "suburb": "Coolaroo",
  "transport_type": "bus",
  "stop_id": 25778,
  "location_name": "Barry Rd/Pascoe Vale Rd ",
  "lat": -37.66433,
  "lon": 144.922226,
  "distance": 0.0
},
{
  "suburb": "Carrum Downs",
  "transport_type": "bus",
  "stop_id": 19969,
  "location_name": "Bawden St/Frankston-Dandenong Rd - East ",
  "lat": -38.0915,
  "lon": 145.1794,
  "distance": 0.0
},
{
  "suburb": "Carrum Downs",
  "transport_type": "bus",
  "stop_id": 28676,
  "location_name": "Bawden St/Frankston-Dandenong Rd - West ",
  "lat": -38.09126,
  "lon": 145.179047,
  "distance": 0.0
},
{
  "suburb": "Montmorency",
  "transport_type": "bus",
  "stop_id": 28932,
  "location_name": "Beleura Gr/Main Rd ",
  "lat": -37.7256432,
  "lon": 145.123184,
  "distance": 0.0
```

```
 },
 {
   "suburb": "Rowville",
   "transport_type": "bus",
   "stop_id": 13764,
   "location_name": "Bergins Rd/Stud Rd ",
   "lat": -37.93004,
   "lon": 145.232834,
   "distance": 0.0
 },
 {
   "suburb": "Doncaster East",
   "transport_type": "bus",
   "stop_id": 21003,
   "location_name": "Beverley St/Blackburn Rd ",
   "lat": -37.791996,
   "lon": 145.161362,
   "distance": 0.0
 }
]
```

JSON object structure

The diagrams below show the structure of the JSON objects returned from the API calls.

> **Stop data** is returned through three different pathways: as a **stop** object, as part of a "**locations"** object, and as a type of "**result"** object.
>
> **Line data** is returned through two pathways: as a **"line"** object and as a type of **"result"** object.
>
> **Timetable data** is returned through **"value"** objects.

"result" objects

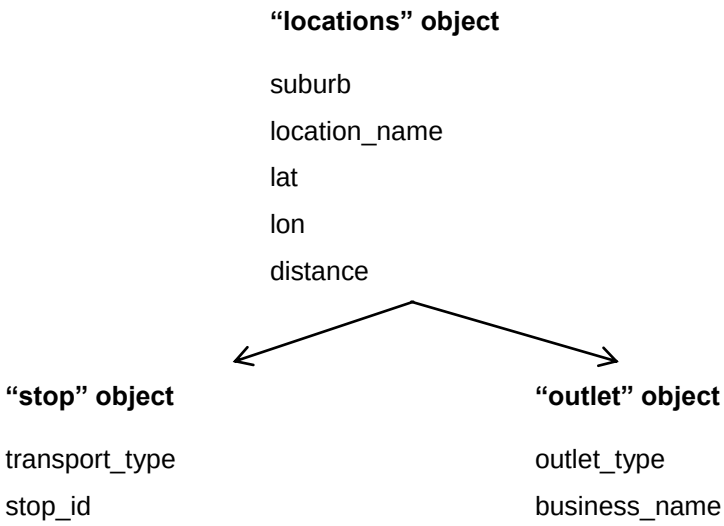
These are returned through the **Stops Nearby** and**Search** APIs. The data structure looks like this:

"result" object

"stop" object:	"line" object:
suburb	transport_type
transport_type	line_id
stop_id	line_name
location_name	line_number
lat	
lon	
distance	
type = "stop"	**type = "line"**

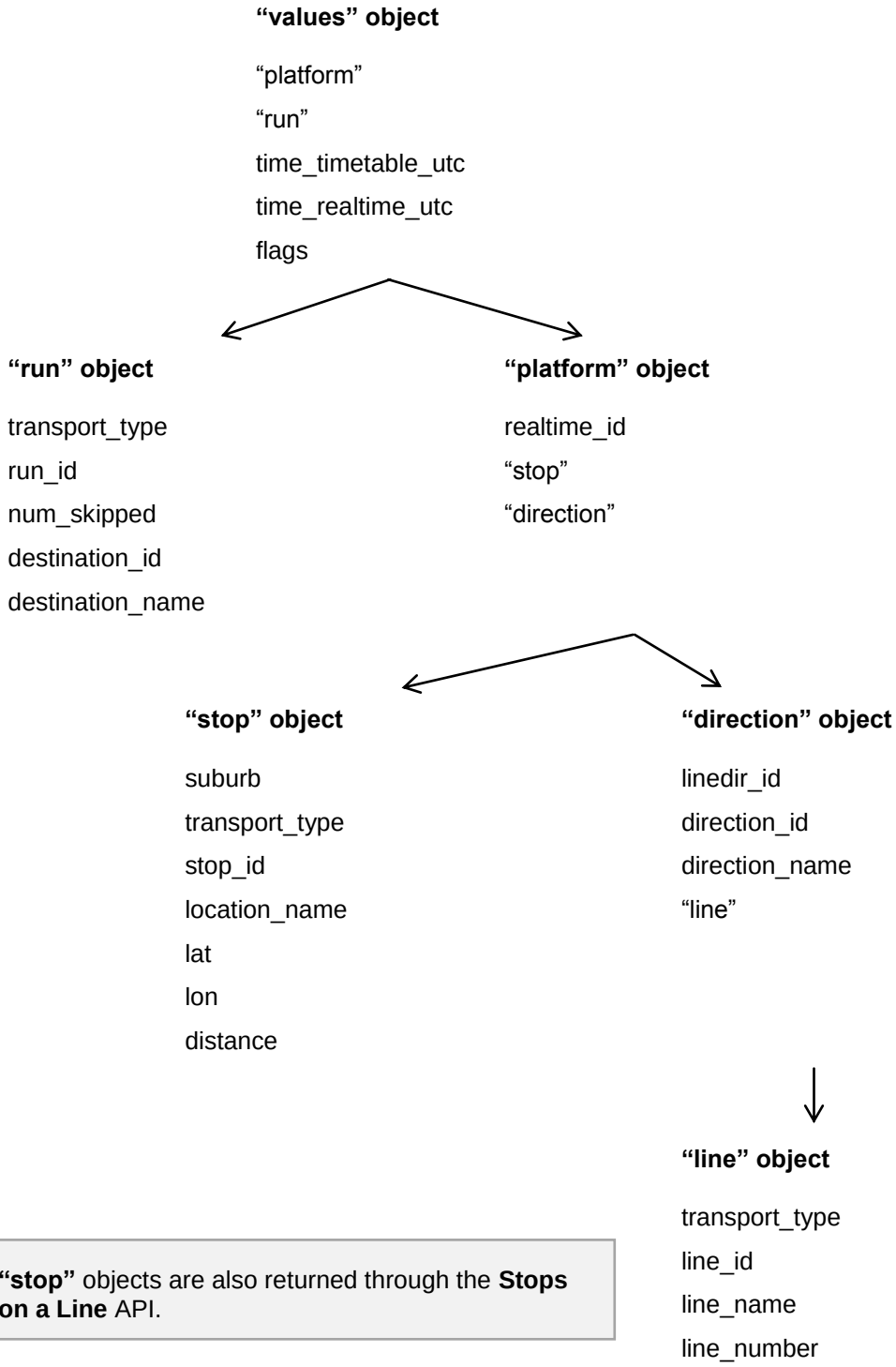
“locations” objects

These are returned through the **Transport POIs by Map** API. The data structure looks like this:



“values” objects

These are returned via the three APIs which return timetable data: **Broad Next Departures**, **Stopping Pattern** and **Specific Next Departures**. The data structure looks like this:



“stop” objects are also returned through the **Stops on a Line** API.

Data Quality Statement

Data Source:	**PTV Timetable API**
Institutional Environment:	**Data Collector(s):** Data is created and collected by Victorian public transport train, tram and bus operators and by Public Transport Victoria. **Collection authority:** Public Transport Victoria **Data Compiler(s):** Public Transport Victoria (a government agency) compiles the data. **Additional information:** Timetable data changes frequently (for example, over summer, or for planned works) and is updated on an as needs basis. Changes are advertised on the PTV website. As the data is accessed through an API, the data released is always up-to-date.
Relevance:	**Data topic:** The data collected is the public transport timetable for services in the state of Victoria (including Melbourne metropolitan services). **Level of geography:** The state of Victoria. **Key Data Items:** Timetable, station/stop locations, line/route paths, and myki ticket outlets. **Additional information:** The PTV Timetable API does not provide access to the PTV journey planner, the Yarra Trams TramTRACKER feed, nor the Google Geocoding API (used by PTV to search addresses).
Timeliness:	**Data collected:** The data is collected on an as needs basis and is compiled weekly for release. **Data available:** The data is made accessible through the PTV Timetable API, its availability is current. **Referenced Period:** Approximately 14 days. **Additional information:** Public transport timetable, stop/station, route and line data changes frequently. In order to ensure you access the most up-to-date data, it is strongly recommended you use the API dynamically.
Accuracy:	**Method of Collection:** The data is collected both manually and through electronic file. **Data Adjustments:** The data contains interpolated times for stops

	on bus routes that are not timing points. **Collection size:** All public transport services in the State of Victoria. **Additional information:** The timetable data released is consistent with the data on the PTV website and through the PTV apps.
Coherence:	**Consistency over time:** The data is consistent over time in that the same set of data (i.e. pertaining to services provided by public transport operators) is consistently collected and released. **Consistency of jurisdictions:** Unknown. **Time series:** There is not a consistent time series for this data. Timetable data changes frequently; PTV only keeps the current timetable data, it does not archive the old data once it has changed.
Interpretability:	**Context:** The PTV Timetable API gives the public access to raw public transport timetable data. It is not a journey planner service.
Accessibility:	**Additional information:** PTV has also released other public transport data through the DataVic website: www.data.vic.gov.au.

Getting help

Glossary

cluster is a group of geographically concentrated POIs

Cross Street is a street that intersects the street a tram or bus is travelling along (i.e. the Travel Street)

Google Geocoding API is a Google API that provides access to a service which converts addresses into geographic coordinates (and vice versa)

journey planner is a PTV service that allows people to plan journeys from one specific point to another

landmark is a prominent or easily identifiable building or other place that is used to mark a location, for example, a shopping centre, school, hospital, or park

line is a collection of route variations or paths that travel in the same direction

location is the physical place of a stop or outlet, described by the “location_name” attribute

outletis a myki ticket outlet; this can be a retail outlet (i.e. shop) or a stop outlet (i.e. machine located at a station, tram stop or bus stop)

platform a data object returned through the API made up of stop, line and direction information

POIstops and/or myki ticket outlets (collectively known as points of interest – i.e. POIs)

route is a collection of route variations or paths that travel in the same direction; in the data, a “route” is called a “line”

run is a specific service i.e. the transport type or mode, line, direction and time are all specified

stop is any of the following: a train station, tram stop, bus stop, or even a bus bay (e.g. at a shopping centre)

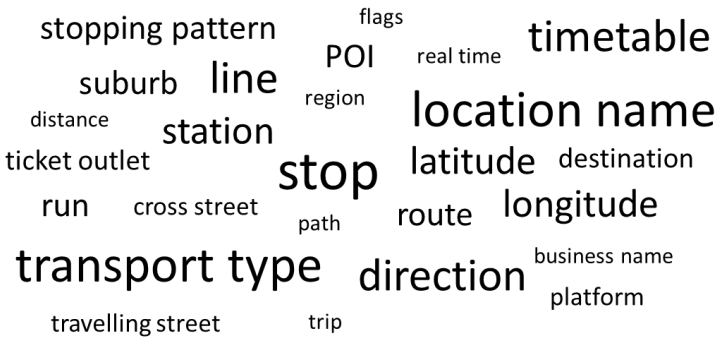
stopping pattern is the sequence of stops that a vehicle actually stops at on a line for any particular run

TramTRACKER is a real-time feed for tram services provided by Yarra Trams

Travel Street is the street that a tram or bus is travelling along at any given point in its run

UTC stands for Coordinated Universal Time; it is the primary time standard that regulates clocks and time

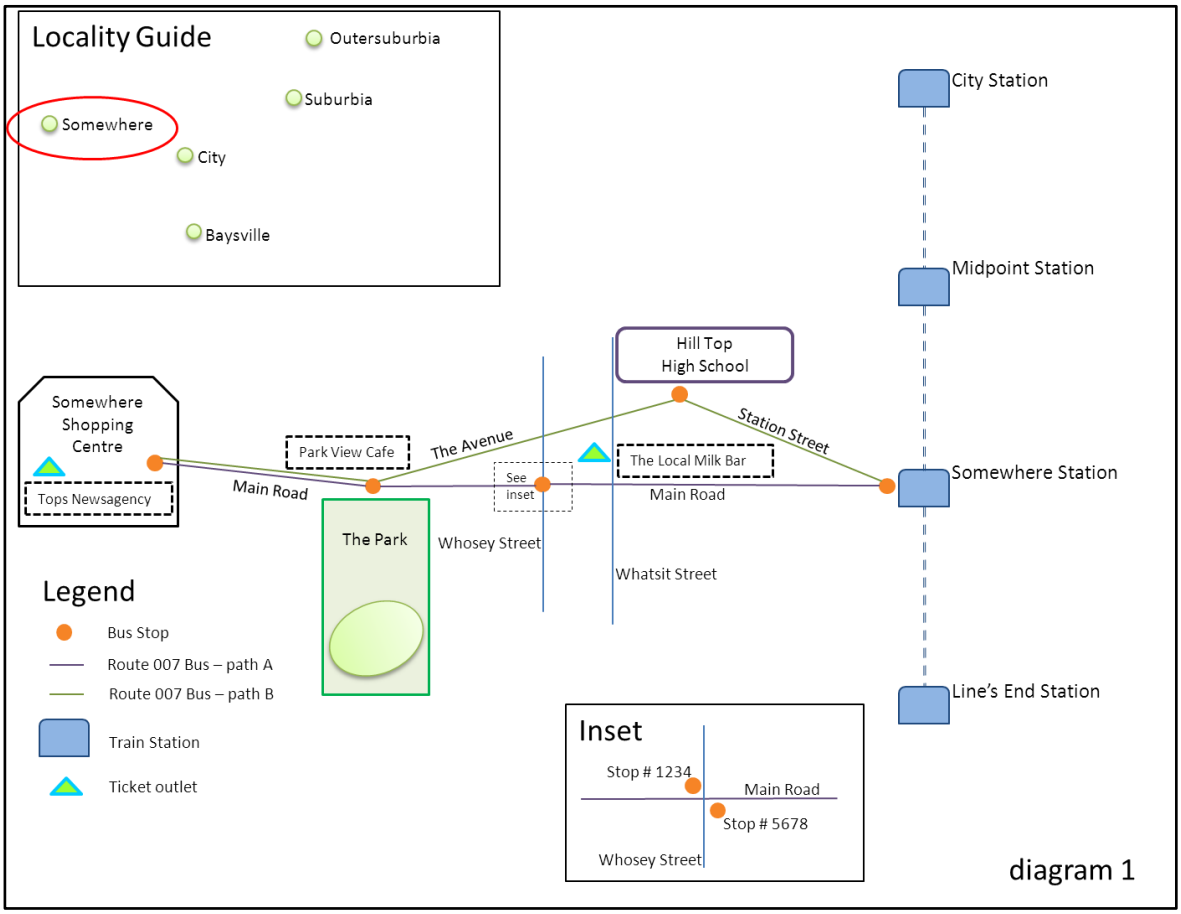
Guide to understanding public transport data



This part of the document is designed to give you a crash course in public transport data concepts and terms relevant to the API. It's not meant to be a comprehensive guide to all public transport data in the known universe.

We want to help you understand the data that is returned through the API so you can get the most out of it.

We've created a fictional town called Somewhere (diagram 1) to help illustrate some of the key concepts.



Public transport is easy to understand, right? There's trains, trams and buses, some routes, a timetable, some stations and stops…it's all pretty simple stuff, right?

Well, yes. And also no.

Public transport seems pretty simple on the surface, but the data that underpins it is a very complex beast. You see, there's public transport data, and then there's how it's presented to the public. A lot of effort goes into making public transport 'nice' for the public!

Let's start with the concept of a **stop**. In our data, a stop can be a tram stop, a bus stop, a bus bay at a shopping centre or junction, or a train station.

So all of the bus stops and the train stations in Somewhere (diagram 2) would be returned as stops.

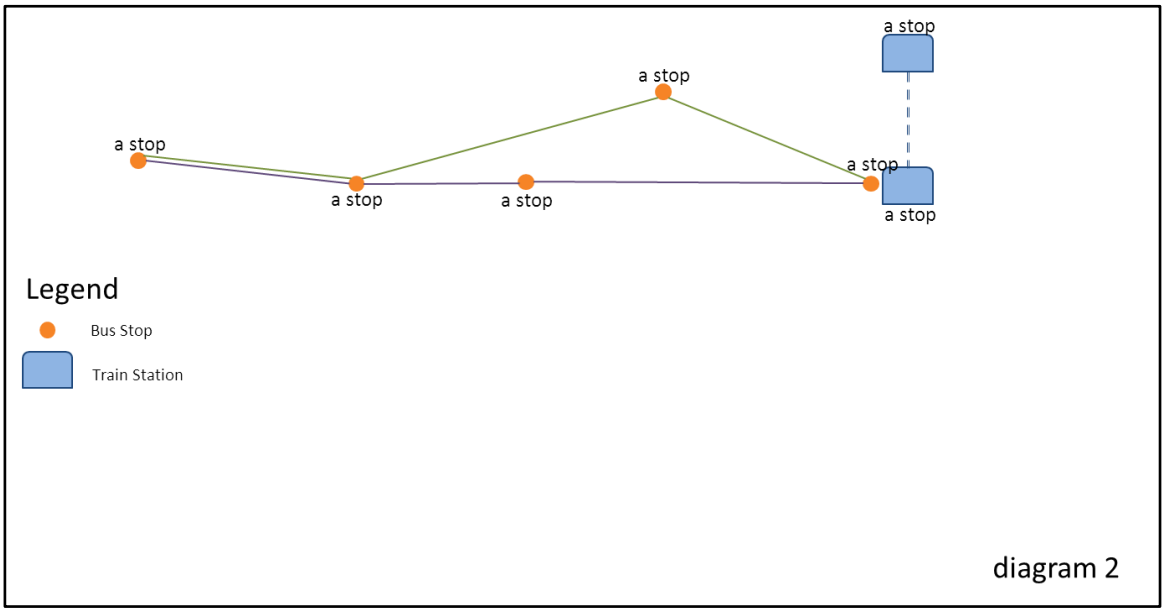


diagram 2

You can identify what kind of stop it is through the **transport_type** attribute. These can be train, tram, bus, V/Line train and coach, or NightRider.

All stops have a unique identifier (**stop_id**attribute) and also a **location name**. Location names describe as concisely as possible the physical location of the stop.

For train stations, the location name is the name of the station – for example, "Somewhere Station".

For tram and bus stops, location names are determined by a hierarchy of available stop information. The hierarchy is:

Landmark > Cross Street > Travel Street

where Travel Street is the street the vehicle is travelling on, and Cross Street is a street that intersects, or crosses, it.

Depending on the content of those fields in our database, the location name can be Landmark_Travel Street, or Cross Street_Travel Street, or just Travel Street, together with the suburb. Tram stop location names also include a stop number at the start (which is the number that appears on the signage at the stop or in the timetable; not the same as the "stop_id").

Diagram 3 shows the bus stop location names in Somewhere.

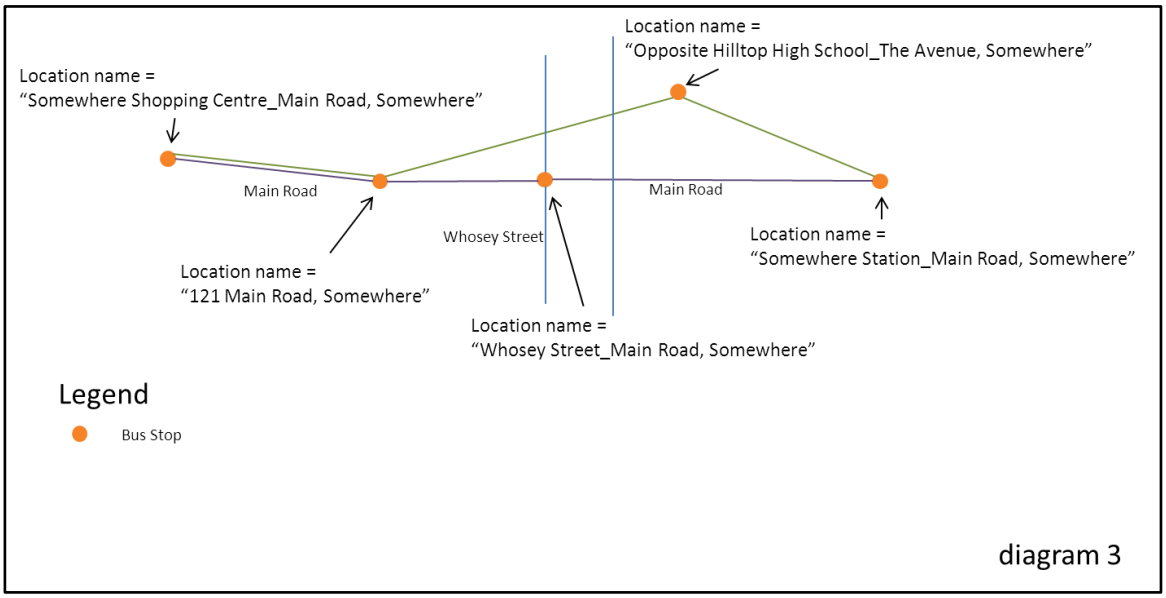


diagram 3

On a map, a stop may appear to be a single point on a road (tram or bus) or building (train). When you zoom in, however, you will notice that it may in fact be made up of several separate stops, each with its own unique stop ID.

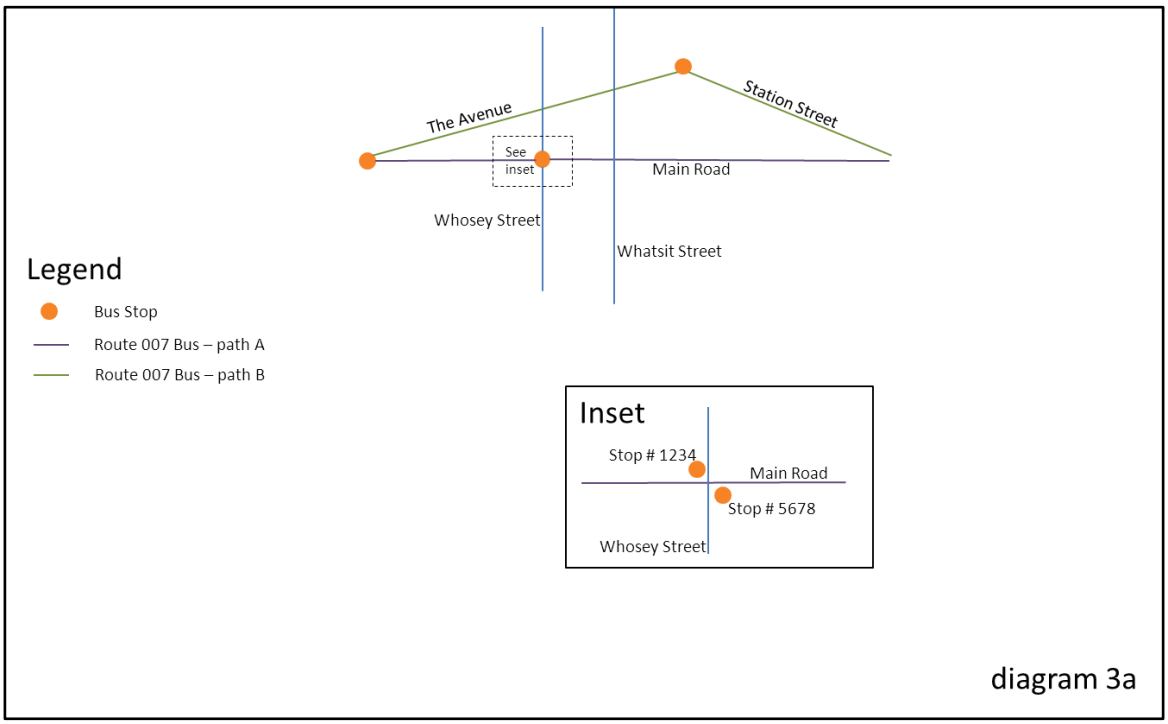


diagram 3a

Stops and **retailticket outlets** make up public transport points of interest (**POIs**). Ticket outlet POIs also have a location name, as well as a **businessname**. The town of Somewhere has two ticket outlet POIs and 6 stop POIs (diagram 4).

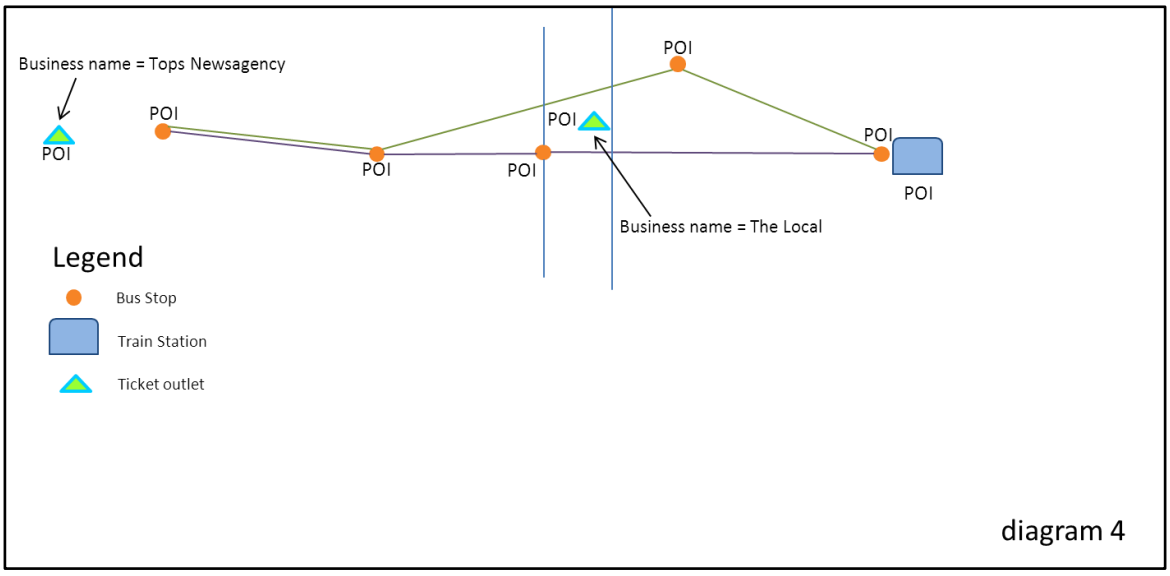


diagram 4

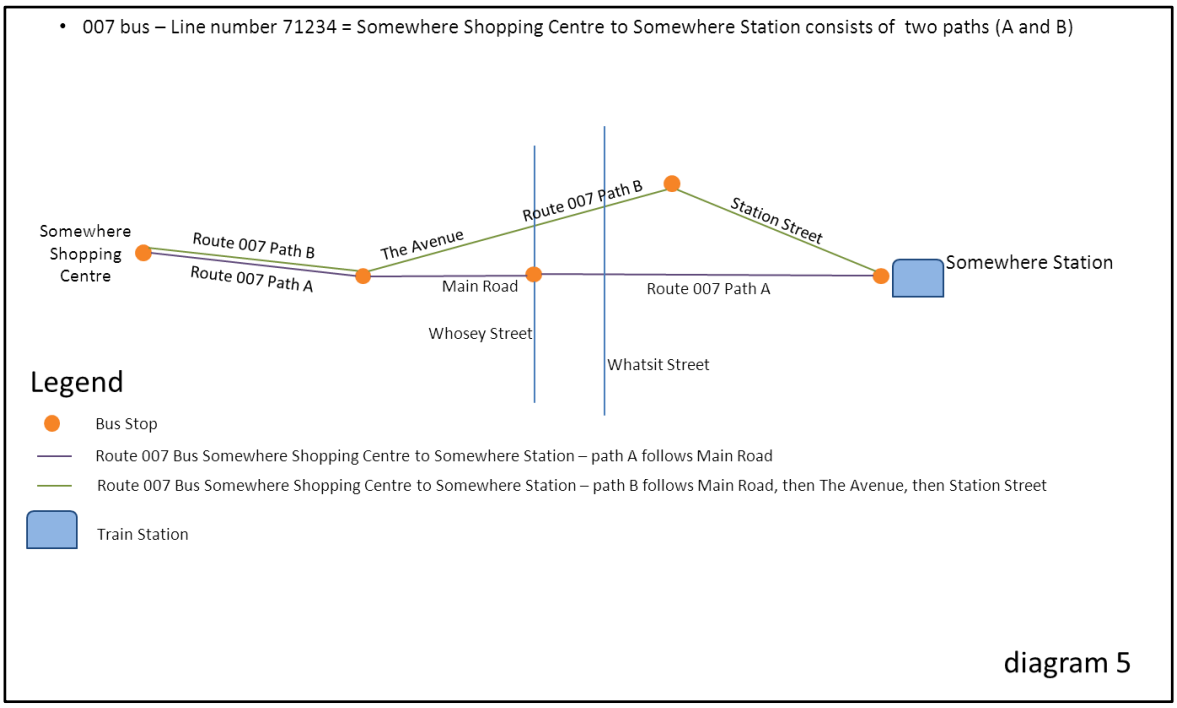
Next let's look at routes. The public knows train services by the name of the line (for example, "Alamein line"), and tram and bus routes by a number, or a name. For example, the Route 112 tram or the Ballarat-Bendigo coach.

In public transport data, however, each route is made up of multiple route variations, which are the geographic paths that a vehicle takes under the name of a route. Each route variation or path is made up of a sequence of stops in a particular order (and a path may pass the same stop more than once).

There can be different paths at different times of day, in different directions or when a vehicle (usually a bus) deviates to a school or shopping centre.

A collection of route variations or paths make up a **line**. Mostly lines run in two directions, however they sometimes run in a loop. In our data a line can be any of the following transport type: train, tram, bus, V/Line rail and coach, or NightRider.

In Somewhere, the Route 007 bus is a line made up of two paths (diagram 5).



The sequence of stops that a vehicle actually stops at on a path for any particular trip (also known as a service or **run**) is known as a **stopping pattern**. For example, a vehicle may stop at all stops on a particular path, or it may travel express and skip some stops on the path.

A **timetable** overlays times onto the movement of a vehicle on a particular run. The times take into account the path that the vehicle takes, as well as the stopping pattern.

Variations are explained through the use of **flags**. A flag might indicate that a particular stopping pattern or path is taken on school days only, for example, or only ona particular day of the week. It can also indicate that reservations are required for a service or that the service is drop off or pick up only at a particular stop.

The Route 007 bus that runs from Somewhere Shopping Centre to Somewhere Station has two paths and three stopping patterns. These are indicated in the timetable for that route (diagram 6).

- **Timetable for 007 bus Somewhere Shopping Centre to Somewhere Station**

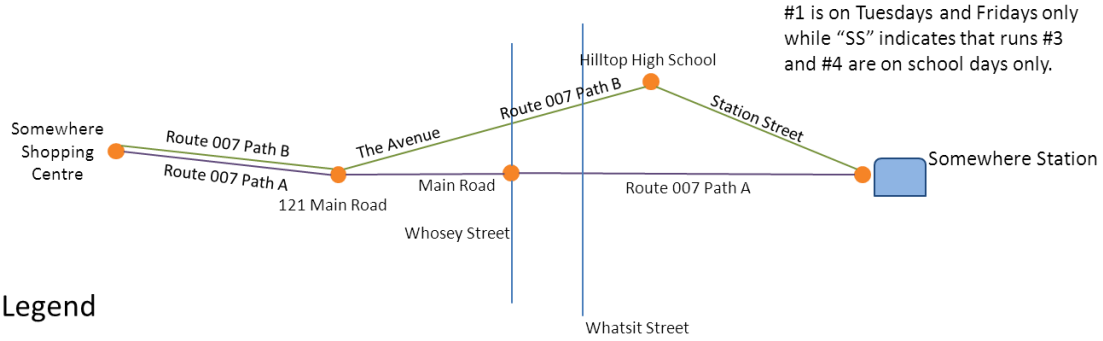
flags

	TU,FR		SS	SS	
Somewhere Shopping Centre	7:26	7:48	8:04	8:25	8:37
121 Main Road	7:37	7:59	8.17	—	8:50
Hilltop High School			8:32	8:48	
Whosey Street	7:47	8:09			9:00
Somewhere Station	7:56	8:20	8:41	8:57	9:13
	Run #1	Run #2	Run #3	Run #4	Run #5

The timetable shows one stopping pattern for path A, and two stopping patterns for path B:

- Runs #1, #2 and #5 stop at all stops along path A
- Run #3 stops at all stops along path B
- Run #4 runs express along path B from Somewhere Shopping Centre to Hilltop High School

The flags "TU,FR" indicate that run #1 is on Tuesdays and Fridays only while "SS" indicates that runs #3 and #4 are on school days only.



Legend

TU,FR,SS Examples of flags appearing in a timetable; in this case TU means "Tuesdays only," "FR" means "Fridays only" and SS means "School days only"

Bus Stop

Route 007 Bus Somewhere Shopping Centre to Somewhere Station – path A follows Main Road

Route 007 Bus Somewhere Shopping Centre to Somewhere Station – path B follows Main Road, then The Avenue, then Station Street

Train Station

diagram 6

We hope you now understand the main timetable data concepts a little better, to help you get the most out of our API.

FAQs

Q. What does the PTV Timetable API do?

A. The PTV Timetable API provides a way to directly and dynamically access the most up-to-date stop, line and timetable data held by PTV. By creating an API we are increasing the opportunities for developers to take our data and re-use it in innovative way.

Q. Who is the PTV Timetable API for?

A. Our API is for everyone who wants to take our data and re-use it in a web or smartphone app.

Q. Can I use the API to download all the timetable data?

A.The API is not designed to download all the timetable data at once. It works most effectively when used dynamically within an app as that is the way to guarantee you're always accessing – and providing – the most up-to-date data. If you want to use the API to get a dump of all of the data, **please contact us**.

Q. Some of the timetable results I get using the API are different to those the PTV journey planner provides – why?

A. The PTV journey planner is coded with a number of business rules that reflect public transport operational requirements. This can include noting which multi-modal services are guaranteed connections, for example, or requiring passengers to board a regional train half an hour before it is due to leave.

The API accesses raw timetable data, not journey planner results. Your timetable results will reflect the actual timetable.

Q. Is the timetable data available in a GTFS format?

A. Timetable data is not available in GTFS format. The operational needs of our business dictate the format of our data and the methods to access it. This is why the API we're releasing is the same as the API that we use for our own online products.

Q. What programming languages can I use with the API?

A. The PTV Timetable API can work with all programming languages. Because it is a programming language agnostic interface, as long as the language you are using supports HTTP protocols, you can use our API.

Q. Is real-time data available?

A. PTV gets real-time data for trams through Yarra Trams' TramTRACKER system. Contact **Yarra Trams** to find out more.

There are currently no real-time data feeds for trains and buses.

Contact details

Got a question?

Check our **FAQs** – we may already have it covered.

If you're unsure about some of the public transport terms returned in the data, take a look through our **Guide to understanding public transport data**. Alternatively, check the **Glossary** if there is other terminology you're unsure about.

If you still can't find what you're looking for, you can send us your question – or feedback – through the PTV Timetable API page on **data.vic.gov.au**.

Just click the **Add comment** button at the bottom of the page, thenunder "Message Type", select **Feedback** from the two radio buttons. Fill in the form and DataVic will then send your question or feedback through to us.

> If you choose to leave a **Comment** on the DataVic page, it will be moderated but we won't get back to you about it. If you choose **Feedback**, on the other hand, we will get back to you if you require a response but it won't be published on our DataVic page.
>
> PTV **does not provide technical support** for the API.

If you want to ask about the release of other data sets, you can do so through the **Suggest a dataset** form on data.vic.gov.au.

API mailing list

Once you have registered for a key, you'll automatically be signed up to our mailing list. It's our way of keeping you up-to-date with information on version updates, maintenance outages or other information about the API.

> We'll be monitoring the use of our API to make sure our mailing list is current and sustainable. **If you haven't used the API for over 3 months, we maydisable your key and remove you from the list** – but you can always register for a new key if you need one.

Appendix

Sample code for creating a signature

You'll need to pass along a signature and a developer ID – or "devid" – withevery request using HTTP GET.

The signature value is a HMAC-SHA1 hash of the completed request (minus the base URL but including your developer ID, known as "devid") and the key.

Example in .net C#

The following is the .net C# code snippet for the signature calculation.

Note: key values are used for example purposes only.

```
string key = "9c132d31-6a30-4cac-8d8b-8a1970834799"; // supplied by PTV
int developerId = 2; // supplied by PTV
string url = "/v2/mode/2/line/787/stops-for-line"; // the PTV api method we want

// add developer id
url = string.Format("{0}{1}devid={2}",url,url.Contains("?") ? "&" : "?",developerId);
System.Text.ASCIIEncoding encoding = new System.Text.ASCIIEncoding();
// encode key
byte[] keyBytes = encoding.GetBytes(key);
// encode url
byte[] urlBytes = encoding.GetBytes(url);
byte[] tokenBytes = new
System.Security.Cryptography.HMACSHA1(keyBytes).ComputeHash(urlBytes);
var sb = new System.Text.StringBuilder();
// convert signature to string
Array.ForEach<byte>(tokenBytes, x => sb.Append (x.ToString("X2")));
// add signature to url
url = string.Format("{0}&signature={1}",url,sb.ToString());

// extra code to add base URL – the resultant url should be:
// http://timetableapi.ptv.vic.gov.au/v2/mode/2/line/787/stops-for-
line?devid=2&signature=D5474F344CDAA7B92F2253169F6C1D66C1A15001
```

Example in Java

The following is the Java code snippet for the signature calculation.

Note: key values are used for example purposes only.

```
/**
 * Generates a signature using the HMAC-SHA1 algorithm
 *
 * @param privateKey - Developer Key supplied by PTV
 * @param uri - request uri (Example :/v2/HealthCheck)
 * @param developerId - Developer ID supplied by PTV
 * @return Unique Signature Value
 */
public String generateSignature(final String privateKey, final String uri, final int developerId)
{
    String encoding = "UTF-8";
    String HMAC_SHA1_ALGORITHM = "HmacSHA1";
    String signature;
    StringBuffer uriWithDeveloperID = new StringBuffer();
    uriWithDeveloperID.append(uri).append(uri.contains("?") ? "&" :
"?").append("devid="+developerId);
    try
    {
        byte[] keyBytes = privateKey.getBytes(encoding);
        byte[] uriBytes = uriWithDeveloperID.toString().getBytes(encoding);
        Key signingKey = new SecretKeySpec(keyBytes, HMAC_SHA1_ALGORITHM);
        Mac mac = Mac.getInstance(HMAC_SHA1_ALGORITHM);
        mac.init(signingKey);
        byte[] signatureBytes = mac.doFinal(uriBytes);
        StringBuffer buf = new StringBuffer(signatureBytes.length * 2);
        for (byte signatureByte : signatureBytes)
        {
            int intVal = signatureByte & 0xff;
            if (intVal < 0x10)
            {
                buf.append("0");
            }
```

```
            buf.append(Integer.toHexString(intVal));
        }
        signature = buf.toString();
    }
    catch (UnsupportedEncodingException e)
    {
        throw new RuntimeException(e);
    }
    catch (NoSuchAlgorithmException e)
    {
        throw new RuntimeException(e);
    }
    catch (InvalidKeyException e)
    {
        throw new RuntimeException(e);
    }
    return signature.toString().toUpperCase();
}


/**
 * Generate full URL using generateSignature() method
 *
 * @param privateKey - Developer Key supplied by PTV (Example :  "92dknhh31-6a30-4cac-8d8b-8a1970834799");
 * @param uri - request uri (Example :"/v2/mode/2/line/787/stops-for-line)
 * @param developerId - Developer ID supplied by PTV( int developerId )
 * @return - Full URL with Signature
 */
public String generateCompleteURLWithSignature(final String privateKey, final String uri, final int developerId)
{


    String baseURL="http://timetableapi.ptv.vic.gov.au";
    StringBuffer url = new StringBuffer(baseURL).append(uri).append(uri.contains("?") ? "&" : "?").append("devid="+developerId).append("&signature="+generateSignature(privateKey, uri, developerId));
    return url.toString();
```

```
}
```

Example in Objective C

The following is the Objective C code snippet for the signature calculation.

Note: key values are used for example purposes only.

```
-(NSURL*) generateURLWithDevIDAndKey:(NSString*)urlPath {

   NSString *hardcodedURL = @”http://timetableapi.ptv.vic.gov.au”;
   NSString *hardcodedDevID = @”developerID provided by PTV”;
   NSString *hardcodedkey = @”developer key provided by PTV”;

/*    urlPath = @"http://timetableapi.ptv.vic.gov.au/v2/mode/2/line/787/stops-for-line";
*/
   NSRange deleteRange ={0,[hardcodedURL length]};
   NSMutableString *urlString = [[NSMutableString alloc]initWithString:urlPath];
   [urlString deleteCharactersInRange:deleteRange];
   if( [urlString containsString:@"?"])
      [urlString appendString:@"&"];
   else
      [urlString appendString:@"?"];

   [urlString appendFormat:@"devid=%@",hardcodedDevID];


   const char *cKey  = [hardcodedkey cStringUsingEncoding:NSUTF8StringEncoding];
   const char *cData = [urlString cStringUsingEncoding:NSUTF8StringEncoding];
   unsigned char cHMAC[CC_SHA1_DIGEST_LENGTH];
   CCHmac(kCCHmacAlgSHA1, cKey, strlen(cKey), cData, strlen(cData), cHMAC);

   NSString *hash;
```

```
    NSMutableString* output = [NSMutableString
stringWithCapacity:CC_SHA1_DIGEST_LENGTH * 2];


    for(int i = 0; i < CC_SHA1_DIGEST_LENGTH; i++)
        [output appendFormat:@"%02x", cHMAC[i]];
    hash = output;


    NSString* signature = [hash uppercaseString];
    NSString *urlSuffix = [NSString stringWithFormat:@"devid=%@&signature=%@",
hardcodedDevID,signature];


    NSURL *url = [NSURL URLWithString:urlPath];
    NSString *urlQuery = [url query];
    if(urlQuery != nil && [urlQuery length] > 0){
        url = [NSURL URLWithString:[NSString stringWithFormat:@"%@&%@",urlPath,urlSuffix]];
    }else{
        url = [NSURL URLWithString:[NSString stringWithFormat:@"%@?%@",urlPath,urlSuffix]];
    }


    return url;
}
```