Introduction

Welcome to TSL API v1.0 documentation. The API is freely available for everyone to use and does not require authentication. The following list describes the queries you can make to our API, and the data returned. In order to minimise overhead and maximise efficiency we provide all data in the form of JavaScript Object Notation (JSON) with the exception of crowdedness data which is provided as a string.
Currently the server is hosted at `stud-tfl.cs.ucl.ac.uk` and all queries should be made by appending the query to the server's address.

Disclaimer

Transport data is provided by TfL and Twitter feeds are provided by courtesy of Twitter (REST API v1). TSL is not responsible for any inconsistencies of data provided by TfL and Twitter. We are the owners of the API and we reserve the rights to change access to the API at any time without notice. Although we moderate comments, we are not responsible for any offensive commentaries submitted by other clients of the API.
TSL is a non-profit organisation providing all data free of charge. TSL is not affiliated with TfL, Twitter or any other organization unless explicitly stated.

Bus Countdown

Bus countdown data is provided in the form of a JSON array. Each bus countdown element is a separate JSON object.The key-value pairs available in each object are:

Route	The route number of the bus
Destination	The destination of the bus
Time	Time in which the bus is expected to arrive

Bus countdown can be requested at
`/bus?stop=#####`
where ##### is the 5 digit bus stop code

An example query:
`http://stud-tfl.cs.ucl.ac.uk/bus?stop=53365`
This query returns the following data:
```
[{
      "Route":"490",
      "Destination":"Heathrow T5",
      "Time":"6 min"
},
{
      "Route":"482",
      "Destination":"Heathrow T5",
      "Time":"17 min"
},
{
      "Route":"490",
      "Destination":"Heathrow T5",
      "Time":"21 min"
},
{
      "Route":"490",
      "Destination":"Heathrow T5",
      "Time":"29 min"
}]
```

Bus Stop Query

Bus stop location data is provided in the form of a JSON array. Each bus stop element is a separate JSON object. The key-value pairs available in each object are:

stopName	The name of the bus stop
stopCode	The five digit stop code for the bus stop
stopIndicator	The identifying marker letter(s) that are displayed on the bus stop
long	The longitude coordinate for the bus stop
lat	The latitude coordinate for the bus stop

Bus stop locations can be requested at
`/stops?loc=######,$$$$$$,%%%`
where `######` is the latitude coordinate, `$$$$$$` is the longitude coordinate and `%%%` is the radius in meters.

An example query:
`http://stud-tfl.cs.ucl.ac.uk/stops?loc=51.5239,-0.133101,300`
This query returns the following data:

```
[{
    "stopName":"Euston Square Station",
    "stopCode":"50975",
    "stopIndicator":"P",
    "long":"-0.134858",
    "lat":"51.525813"
},
{
    "stopName":"University College Hospital",
    "stopCode":"76205",
    "stopIndicator":"Z",
    "long":"-0.13655",
    "lat":"51.523916"
},
{
    "stopName":"Torrington Place",
    "stopCode":"73135",
    "stopIndicator":"C",
    "long":"-0.133016",
    "lat":"51.522916"
},
{
    "stopName":"Euston Station",
    "stopCode":"51664",
    "stopIndicator":"H",
    "long":"-0.13295",
    "lat":"51.526449"
}]
```

Tube Countdown

Tube countdown data is provided in the form of a single JSON object. The object contains the following fields:

LineName	Specifies line name		
StationName	Specifies station name		
Platform	JSON array of platforms available at station		
	PlatformName	Name of the train platform	
	Trains	JSON array of train countdown data	
		Destination	Train destination
		TimeTo	Train countdown

Tube countdown can be requested at
`/tube?stop=@,$$$`
where $$$ is the 3 letter station code and @ is the 1 letter line code

An example query:
http://stud-tfl.cs.ucl.ac.uk/tube?stop=P,RSQ
This query returns the following data (only a fragment of data returned is shown):

```
{
      "LineName": "Piccadilly Line",
      "StationName": "Russell Square",
      "Platform":
            [{
                  "PlatformName": "Eastbound - Platform 1",
                  "Trains":
                        [{
                              "Destination": "Cockfosters",
                              "TimeTo": "1 min"
                        },
                        {
                              "Destination": "Cockfosters",
                              "TimeTo": "3 min"
                        },
                        {
                              "Destination": "Arnos Grove",
                              "TimeTo": "7 min"
                        },
                        {
                              "Destination": "Cockfosters",
                              "TimeTo": "9 min"
                        },
                        {
                              "Destination": "Cockfosters",
                              "TimeTo": "12 min"
                        },
                        {
                              "Destination:" "Cockfosters",
                              "TimeTo": "16 min"
                        },
                        {
                              "Destination:" "Arnos Grove",
                              "TimeTo": "19 min"
                        },
                        {
                              "Destination": "Cockfosters",
                              "TimeTo": "22 min"
                        },
                        {
                             "Destination": "Cockfosters",
                             "TimeTo": "24 min"
                        },
                        {
                             "Destination": "Cockfosters",
                             "TimeTo": "28 min"
                        }
                   ]
          },
          …
}
```

Line Status

Line status data is provided in the form of a JSON array. Each line element is a separate JSON object. The key-value pairs available in each object are:

_id	The internal MongoDB ID for the line (do not use)
lineID	The TFL ID number for the line
lineName	The name of the line
statusDescription	The current status of each line
statusDetails	Further details on the status of each line. If the status is Good Service, then this will be blank.

Line status can be requested at
`/lines`

An example query:
http://stud-tfl.cs.ucl.ac.uk/lines
This query returns the following data that has been shortened for display purposes:

```
[{
    "_id":"5132a740f919b47a0f000005",
    "lineID":"8",
    "lineName":"Hammersmith and City",
    "statusDescription":"Good Service",
    "statusDetails":""
},
{
    "_id":"5132a740f919b47a0f00000b",
    "lineID":"12",
    "lineName":"Waterloo and City",
    "statusDescription":"Good Service",
    "statusDetails":""
},
{
    "_id":"5132a740f919b47a0f000003",
    "lineID":"7",
    "lineName":"Circle",
    "statusDescription":"Good Service",
    "statusDetails":""
},
...
]
```

Bike Docks

Bike dock data is provided in the form of a JSON array. Each dock element is a separate JSON object. The key-value pairs available in each object are:

id	The TFL ID number for the dock
name	The name of the dock
lat	The latitude coordinate for the bike dock
log	The longitude coordinate for the bike dock
locked	Whether the dock is available for use or not. If false, then it is available. If true, then it is not available.
nbBikes	The number of bikes currently available at the bike dock
nbEmptyDocks	The number of empty docks at the bike dock
dbDocks	The total number of docks at the bike dock
_id	The internal MongoDB ID for the dock (do not use)

Bike dock locations can be requested at
`/bike?loc=######,$$$$$$,%%%`
where `######` is the latitude coordinate, `$$$$$$` is the longitude coordinate and `%%%` is the radius in meters

An example query:
http://stud-tfl.cs.ucl.ac.uk/bike?loc=51.5239,-0.133101,150
This query returns the following data:
```
[{
    "id":"19",
    "name":"Taviton Street, Bloomsbury",
    "lat":51.52505093,
    "long":-0.131161087,
    "locked":"false",
    "nbBikes":"18",
    "nbEmptyDocks":"12",
    "dbDocks":"30",
    "_id":"5135e4957d46a55802009f87"
},{
    "id":"65",
    "name":"Gower Place , Euston",
    "lat":51.52522753,
    "long":-0.13518856,
    "locked":"false",
    "nbBikes":"11",
    "nbEmptyDocks":"6",
    "dbDocks":"17",
    "_id":"5135e4957d46a55802009fb4"
}]
```

Crowdedness

Line status data is provided in the form of a string that describes the current crowdedness.

Crowdedness can be requested at
`/crowd?stop=###,$$$$`
where ### is the 3 digit national location code (NLC) for the station and $$$$ is the time in 24-hour international/military time format (e.g. 0000 for midnight)

An example query:
http://stud-tfl.cs.ucl.ac.uk/crowd?stop=625,1256
This query returns the following data:
```
"CROWDED"
```

Weather

Weather data is provided in the form of a single JSON object with the following key-value pairs:

Temperature	The current temperature in Celcius

WeatherDesc	A text description of the current weather
IconURL	A direct link to the icon of the current weather

Weather can be requested at
`/weather?loc=######,$$$$$$`
where ###### is the latitude coordinate and $$$$$$ is the longitude coordinate.

An example query:
`http://stud-tfl.cs.ucl.ac.uk/weather?loc=51.5239,-0.133101`
This query returns the following data that has been modified slightly for display purposes:

```
{
    "Temperature":"5",
    "WeatherDesc":"Partly Cloudy",
    "IconURL":"http://goo.gl/ZqXeM"
}
```

Tube Line Ratings

Tube ratings can be both posted to and retrieved from the API. Ratings are refreshed daily at 1:10AM local UK time. Furthermore user ratings are refreshed three times a day so that re-rating can be implemented on the client side accordingly at 12:10PM, 7:10PM and 1:10AM local UK time.
Line ratings can be retrieved in two forms: overall ratings and ratings per user.
Overall ratings are provided in the form of a JSON array. The key-value pairs available in each object are:

0	Number of “0” ratings
1	Number of “1” ratings
2	Number of “2” ratings
3	Number of “0” ratings
4	Number of “3” ratings
5	Number of “4” ratings
_id	The internal MongoDB ID for the line (do not use)
line	Name of the Tube line

Overall ratings are available at:
`/getratings?fetchall`

An example query:

http://stud-tfl.cs.ucl.ac.uk/lines
An example fragment of data returned:

```
[
      {
            0: 0,
            1: 1,
            2: 0,
            3: 0,
            4: 0,
            5: 0,
            "_id": "5137407debd89e15c3affcfb",
            "line": "Piccadilly"
},
...
]
```

Ratings per user are provided in the form of a single JSON object. We store them as they are provided, therefore it is up to the client to choose whether they want to encrypt or hash the username. We recommend the use of SHA-1 hashing if the userNames are not displayed in your application or RSA encryption otherwise. The key-value pairs available in the object are:

Bakerloo	Line rating integer
Central	Line rating integer
Circle	Line rating integer
DLR	Line rating integer
District	Line rating integer
Hammersmith_and_City	Line rating integer
Jubilee	Line rating integer
Metropolitan	Line rating integer
Northern	Line rating integer
Overground	Line rating integer
Piccadilly	Line rating integer
Victoria	Line rating integer
Waterloo_and_city	Line rating integer
_id	The internal MongoDB ID for the line (do not

	use)
userName	User name of the user who provided the rating

Ratings per user can be requested at:
`/getratings?fetchforuser=*username*`
where `*username*` is the username for which ratings are requested.

An example query:
http://stud-tfl.cs.ucl.ac.uk/getratings?fetchforuser=example
This query returns the following data:

```
{
      "Bakerloo": null,
      "Central": null,
      "Circle": null,
      "DLR": null,
      "District": null,
      "Hammersmith_and_City": null,
      "Jubilee": null,
      "Metropolitan": null,
      "Northern": null,
      "Overground": null,
      "Piccadilly": 1,
      "Victoria": null,
      "Waterloo_and_City": null,
      "_id": "5149d5a157921c980e00000f",
      "userName": "example"
}
```

We allow clients to submit ratings to our API as well. This can be done by appending the rating to the following URL:
`/postratings?foruser=*username*,Piccadilly=#`
where *username* is the username of the user providing the rating and # is a rating integer in the range 0-5.

Comments

Comments about individual Tube stations can be both posted to and retrieved from the API. We store them as they are provided therefore it is up to the client to choose whether they want to encrypt or hash the username. We recommend the use of SHA-1 hashing if the userNames are not displayed in your application or RSA encryption otherwise. Comments are refreshed manually based on their number and timestamp. We reserve the right to delete comments without notice.

Comments are retrieved based on a Tube Station. One can query for all comments at an individual station or only the ones for an individual user.

All comments per station are returned as a JSON array. The first object contains a reply from the server. The server returns

```
{
      "reply":"OK"
}
```

as the first object if comments exist. In this situation, the following JSON objects are always comments. The key-value pairs available in an object are:

userName	User name of the user who provided the comment
stationCode	3 digit NLC code for the station
comment	Textual representation of the comment. URL-encoded (has to be both encoded and decoded on the client side)
created_at	Date of creation of the comment (ISO time stamp)
_id	The internal MongoDB ID for the line (do not use)

If there are no comments at a particular station the following reply is returned as the only object in the array:

```
{
      "reply":"No comments for station"
}
```

All comments per station can be requested at:
`/getcomments?fetchallforstation=###`
where ### is the 3 digit NLC code for the station

An example query:
`http://stud-tfl.cs.ucl.ac.uk/getcomments?fetchallforstation=625`
The query returns the following data (example):

```
[
      {
            "reply": "OK"
      },
      {
            "userName": "f4f63e32d2a140719f77e6f80780b7063e4a01af",
            "stationCode": "625",
            "comment":
      "very+crowded+today%2C+avoid+St+all+cost%21%21%21",
            "created_at": "2013-03-18T13:42:12.194Z",
            "_id": "514719b457921c980e000002"
      },
      {
            "userName": "example",
            "stationCode": "625",
            "comment": "ok",
            "created_at": "2013-03-20T16:13:55.681Z",
            "_id": "5149e04357921c980e000011"
      }
]
```

Comments per station for a particular user follow exactly the same format, the only difference being that only comments for a single user name are returned.

Comments per station for an individual user can be requested at:
`/getcomments?fetchforuser=*username*,atstation=###`
where `*username*` is the username for which comments are requested and `###` is the 3 digit NLC code for the station

We allow clients to submit comments to our API as well. Once a rating is submitted, a JSON array with a single JSON object is returned to confirm that a comment has been accepted. This is in the following format:

```
[
      {
            "reply":"OK"
      }
]
```

Comments can be submitted by appending the rating to the following URL:
`/postcomment?forstation=###,user=*username*,comment=*comment*`
where `###` is the 3 digit NLC code for the station, `*username*` is the username of the user who posted the comment and `*comment*` is the comment text (this requires URL encoding).

Twitter Feeds

The API allows clients to retrieve Twitter feeds about individual Tube Lines and individual Tube stations. The API returns a maximum of 10 Tweets currently. We reserve the right to increase the number of Tweets returned with no previous notice.
In both cases the data is returned in the form of a JSON array. The key-value pairs available in each object are:

created_at	Tweet creation date

from_user	Twitter username of the person who submitted the Tweet
text	Tweet text (**not** URL encoded)

Tweets per station can be requested at:
`/gettwitter?fetchforstation=*stationname*`
where `*stationname*` is the string station name. This should **not** contain the word “station”. The string should be URL encoded. For example, for Russel Square Station the following string should be used: `Russel%20Square`.

Tweets per line can be requested at:
`/gettwitter?fetchforline=*linename*`
where `*linename*` is the string line name. This should **not** contain the word “line”. The string should be URL encoded. For example, for Hammersmith and City line the following string should be used: `Hammersmith%20and%20City`.

An example query:
http://stud-tfl.cs.ucl.ac.uk/gettwitter?fetchforstation=Euston
The query returns the following data (example mock fragment of data returned):
```
[
      {
            "created_at":"Wed, 20 Mar 2013 17:14:31 +0000",
            "from_user":"example",
            "text":"example Twitter status feed"
      },
      ...
]
```