# Reading Buses API

V2.0.0

Generated by Doxygen 1.8.18

# Chapter 1

# Namespace Index

## 1.1 Packages

Here are the packages with brief descriptions (if available):

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

# Chapter 3

# Class Index

## 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

# Chapter 4

# Namespace Documentation

## 4.1 ReadingBusesAPI Namespace Reference

### Classes

- class ReadingBuses

  *This is the main class for the library, here you can initialise a singleton instance and then query and use the Reading Buses API.*

## 4.2 ReadingBusesAPI.Bus_Service Namespace Reference

### Classes

- class BusService

  *Stores information about an individual bus services. Related to the "List Of Services" API.*
- class **Services**

  *This classes simply gets all the bus services operated by Reading Buses, by interfacing with the "List Of Services" API.*

## 4.3 ReadingBusesAPI.Bus_Stops Namespace Reference

### Classes

- class BusStop

  *Stores information about a single bus stop. Related to the "List Of Bus Stops" API.*
- class **Locations**

  *This classes simply gets all the buses stops visited by Reading Buses, by interfacing with the "List Of Bus Stops" API.*

## 4.4 ReadingBusesAPI.Journey_Details Namespace Reference

### Classes

- class LiveRecord

  *Used to store information about a buses arrival at a bus stop. Mainly related to the "Stop Predictions" API.*

## 4.5 ReadingBusesAPI.Shared Namespace Reference

### Classes

- class **ParseOperatorConverter**

  *Converts a string short code for an Operator into an Operator Enum and back again for the JSON converter.*
- struct Point

  *Stores an X and Y Position simply.*
- class **UrlConstructor**

  *Returns back the URL needed to make a get command to the Reading Buses Open Data API. You can use this for testing purposes to check the API is returning what you were expecting.*

### Enumerations

- enum Direction { Direction.Inbound, Direction.Outbound }

  *The direction in which a bus is traveling.*
- enum Operators { Operators.ReadingBuses, Operators.Kennections, Operators.NewburyAndDistrict, Operators.Other }

  *An Enum of the Operators Reading Buses owns or manages in their API.*

### 4.5.1 Enumeration Type Documentation

#### 4.5.1.1 Direction

```
enum ReadingBusesAPI.Shared.Direction  [strong]
```

The direction in which a bus is traveling.

**Enumerator**

| | |
|---|---|
| Inbound | For buses traveling inbound. |
| Outbound | For buses traveling outbound. |

#### 4.5.1.2 Operators

```
enum ReadingBusesAPI.Shared.Operators  [strong]
```

An Enum of the Operators Reading Buses owns or manages in their API.

**Enumerator**

| | |
|---|---|
| ReadingBuses | For Reading Buses services |
| Kennections | For Kennections services |

**Enumerator**

| | |
|---|---|
| NewburyAndDistrict | For Newbury And District services |
| Other | For any other operator which is new in the API and has not yet been officially supported in this library. |

## 4.6 ReadingBusesAPI.TimeTable Namespace Reference

### Classes

- class ArchivedBusTimeTable

  *Represents and retrieves information about a actual single time table record, which means information on one bus at one location. Related to the "Tracking History" API.*
- class BusTimeTable

  *Represents and retrieves information about a scheduled/predicted single time table record, which means information on one bus at one location. Related to the "Timetabled Journeys" API.*
- class TimeTableRecord

  *Represents the Raw timetable object data you get from the Timetabled Journeys and Tracking History APIs.*

## 4.7 ReadingBusesAPI.Vehicle_Positions Namespace Reference

### Classes

- class ArchivedPositions

  *Stores information about previous/ archived GPS data on vehicles.*
- class GPSController

  *Helps get live and historical GPS data on vehicles by accessing the "Live Vehicle Positions" API.*
- class LivePosition

  *Used to store live information about a buses GPS position. Related to the "Live Vehicle Positions" API.*

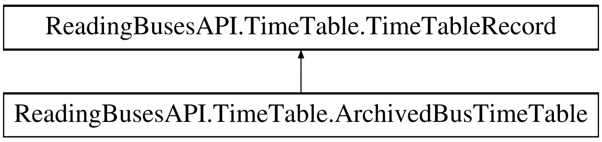
# Chapter 5

# Class Documentation

## 5.1 ReadingBusesAPI.TimeTable.ArchivedBusTimeTable Class Reference

Represents and retrieves information about a actual single time table record, which means information on one bus at one location. Related to the "Tracking History" API.

Inheritance diagram for ReadingBusesAPI.TimeTable.ArchivedBusTimeTable:


ReadingBusesAPI.TimeTable.TimeTableRecord
ReadingBusesAPI.TimeTable.ArchivedBusTimeTable

### Public Member Functions

- double ArrivalLateness ()

  *How late the bus was to arrive at a bus stop.*
- double DepartureLateness ()

  *How late the bus was to departure at a bus stop.*

### Properties

- DateTime? ActArrivalTime `[get, set]`

  *The actual arrival time for the bus.*
- DateTime? ActDepartureTime `[get, set]`

  *The actual departure time for the bus.*

### 5.1.1 Detailed Description

Represents and retrieves information about a actual single time table record, which means information on one bus at one location. Related to the "Tracking History" API.

### 5.1.2 Member Function Documentation

#### 5.1.2.1 ArrivalLateness()

```
double ReadingBusesAPI.TimeTable.ArchivedBusTimeTable.ArrivalLateness ( )
```

How late the bus was to arrive at a bus stop.

**Returns**

The number of seconds the bus was late to arrive by. If no arrival time can be found, 0 is returned.

#### 5.1.2.2 DepartureLateness()

```
double ReadingBusesAPI.TimeTable.ArchivedBusTimeTable.DepartureLateness ( )
```

How late the bus was to departure at a bus stop.

**Returns**

The number of seconds the bus was late to departure by. If no departure time can be found, 0 is returned.

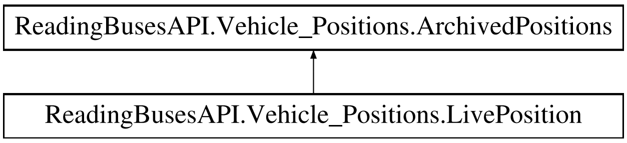
The documentation for this class was generated from the following file:

- TimeTable/ArchivedBusTimeTable.cs

## 5.2 ReadingBusesAPI.Vehicle_Positions.ArchivedPositions Class Reference

Stores information about previous/ archived GPS data on vehicles.

Inheritance diagram for ReadingBusesAPI.Vehicle_Positions.ArchivedPositions:


ReadingBusesAPI.Vehicle_Positions.ArchivedPositions
ReadingBusesAPI.Vehicle_Positions.LivePosition

### Public Member Functions

- Point GetPoint ()

  *Gets the geographical position of the bus.*

## Properties

- Operators OperatorCode `[get, set]`

  *Holds the operators enum value.*
- string Vehicle `[get, set]`

  *Holds the reference/identifier for the vehicle*
- DateTimeOffset Observed `[get, set]`

  *Holds the time it was last seen/ new data was retrieved.*
- string Latitude `[get, set]`

  *Latitude position of the bus*
- string Longitude `[get, set]`

  *longitude position of the bus*

### 5.2.1 Detailed Description

Stores information about previous/ archived GPS data on vehicles.

### 5.2.2 Member Function Documentation

#### 5.2.2.1 GetPoint()

```
Point ReadingBusesAPI.Vehicle_Positions.ArchivedPositions.GetPoint ( )
```

Gets the geographical position of the bus.

**Returns**

A Point Object for the position of the bus.

The documentation for this class was generated from the following file:

- Vehicle Positions/ArchivedPositions.cs

## 5.3 ReadingBusesAPI.Bus_Service.BusService Class Reference

Stores information about an individual bus services. Related to the "List Of Services" API.

## Public Member Functions

- BusService (string serviceNumber)

  *Used to create a snub/ fake object for passing to function calls, if all you need to pass is an service number to the function.*
- BusService (string serviceNumber, Operators operators)

  *Used to create a snub/ fake object for passing to function calls, if all you need to pass is an service number to the function.*
- async Task< string[ ]> GetLocationsActo ()

  *Gets an array of stops the bus service travels too as an array of ActoCode*
- async Task< BusStop[ ]> GetLocations ()

  *Gets an array of 'BusStop' objects the bus service travels too as an array of BusStop objects. If the API is invalid and links to a Bus Stop not in the list of locations it will simply be ignored.*
- async Task< LivePosition[ ]> GetLivePositions ()

  *Gets the Live GPS positions for all Vehicles operating on this service.*
- void PrintLocationsActo ()

  *Prints off all the Acto-codes for bus stops visited by the service.*
- void PrintLocationNames ()

  *Prints off all the names for the bus stops visited by the service.*
- Task< BusTimeTable[ ]> GetTimeTable (DateTime date, BusStop location=null)

  *Gets the full bus time table, for a specific date.*
- Task< IGrouping< string, BusTimeTable >[ ]> GetGroupedTimeTable (DateTime date, BusStop location=null)

  *Gets the time table for this specific bus service, split into groups by the journey code.*
- Task< ArchivedBusTimeTable[ ]> GetArchivedTimeTable (DateTime date, BusStop location=null)

  *Gets the archived real bus departure and arrival times along with their time table history for this service on a specific date.*
- Task< IGrouping< string, ArchivedBusTimeTable >[ ]> GetGroupedArchivedTimeTable (DateTime date, BusStop location=null)

  *Gets the archived real bus departure and arrival times along with their time table history for this service on a specific date, split into groups by the journey code.*

## Properties

- string ServiceId `[get, set]`
- string BrandName `[get, set]`

  *The brand name for the service, used mainly for Reading Buses services, such as Lion, Purple or Orange.*
- Operators OperatorCode `[get, set]`

  *The operator enum value.*

### 5.3.1 Detailed Description

Stores information about an individual bus services. Related to the "List Of Services" API.

### 5.3.2 Constructor & Destructor Documentation

#### 5.3.2.1 BusService() [1/2]

```
ReadingBusesAPI.Bus_Service.BusService.BusService (
            string serviceNumber )
```

Used to create a snub/ fake object for passing to function calls, if all you need to pass is an service number to the function.

**Parameters**

| | |
|---|---|
| *serviceNumber* | ID of the bus service. |

Unless you are doing something very strange, you probably should not need to use this, it is more for testing purposes.

#### 5.3.2.2 BusService() [2/2]

```
ReadingBusesAPI.Bus_Service.BusService.BusService (
            string serviceNumber,
            Operators operators )
```

Used to create a snub/ fake object for passing to function calls, if all you need to pass is an service number to the function.

**Parameters**

| | |
|---|---|
| *serviceNumber* | ID of the bus service. |
| *operators* | The operator who runs the service. |

Unless you are doing something very strange, you probably should not need to use this, it is more for testing purposes.

### 5.3.3 Member Function Documentation

#### 5.3.3.1 GetArchivedTimeTable()

```
Task<ArchivedBusTimeTable[ ]> ReadingBusesAPI.Bus_Service.BusService.GetArchivedTimeTable (
            DateTime date,
            BusStop location = null )
```

Gets the archived real bus departure and arrival times along with their time table history for this service on a specific date.

**Parameters**

| | |
|---|---|
| *date* | the date on which you want a archived timetable data for. This should be a date in the past. |
| *location* | (optional) a specific bus stop you want archived timetables for, if null it will get a timetable for every bus stop on route. |

**Returns**

#### 5.3.3.2 GetGroupedArchivedTimeTable()

```
Task<IGrouping<string, ArchivedBusTimeTable>[]> ReadingBusesAPI.Bus_Service.BusService.Get↩
GroupedArchivedTimeTable (
            DateTime date,
            BusStop location = null )
```

Gets the archived real bus departure and arrival times along with their time table history for this service on a specific date, split into groups by the journey code.

**Parameters**

| | |
|---|---|
| *date* | The date on which you want the time table for. This should be a date in the past. |
| *location* | (optional) The specific bus stop you want time table data for. Leave as null if you want the whole routes timetable. |

**Returns**

A grouping of arrays of time table records based upon journey code.

#### 5.3.3.3 GetGroupedTimeTable()

```
Task<IGrouping<string, BusTimeTable>[]> ReadingBusesAPI.Bus_Service.BusService.GetGrouped↩
TimeTable (
            DateTime date,
            BusStop location = null )
```

Gets the time table for this specific bus service, split into groups by the journey code.

**Parameters**

| | |
|---|---|
| *date* | The date on which you want the time table for. |
| *location* | (optional) The specific bus stop you want time table data for. Leave as null if you want the whole routes timetable. |

**Returns**

A grouping of arrays of time table records based upon journey code.

#### 5.3.3.4 GetLivePositions()

```
async Task<LivePosition[]> ReadingBusesAPI.Bus_Service.BusService.GetLivePositions ( )
```

Gets the Live GPS positions for all Vehicles operating on this service.

**Returns**

An array of GPS data points for all vehicles currently operating on this service.

#### 5.3.3.5 GetLocations()

```
async Task<BusStop[ ]> ReadingBusesAPI.Bus_Service.BusService.GetLocations ( )
```

Gets an array of 'BusStop' objects the bus service travels too as an array of BusStop objects. If the API is invalid and links to a Bus Stop not in the list of locations it will simply be ignored.

**Returns**

An array of BusStop objects for the stops visited by this service.

#### 5.3.3.6 GetLocationsActo()

```
async Task<string[ ]> ReadingBusesAPI.Bus_Service.BusService.GetLocationsActo ( )
```

Gets an array of stops the bus service travels too as an array of ActoCode

**Returns**

An array of Acto-Codes for the stops visited by this services.

#### 5.3.3.7 GetTimeTable()

```
Task<BusTimeTable[ ]> ReadingBusesAPI.Bus_Service.BusService.GetTimeTable (
             DateTime date,
             BusStop location = null )
```

Gets the full bus time table, for a specific date.

**Parameters**

| | |
|---|---|
| *date* | the date on which you want a timetable for. |
| *location* | (optional) a specific bus stop you want timetables for, if null it will get a timetable for every bus stop on route. |

**Returns**

#### 5.3.3.8 PrintLocationNames()

```
void ReadingBusesAPI.Bus_Service.BusService.PrintLocationNames ( )
```

Prints off all the names for the bus stops visited by the service.

#### 5.3.3.9 PrintLocationsActo()

```
void ReadingBusesAPI.Bus_Service.BusService.PrintLocationsActo ( )
```

Prints off all the Acto-codes for bus stops visited by the service.

### 5.3.4 Property Documentation

#### 5.3.4.1 ServiceId

```
string ReadingBusesAPI.Bus_Service.BusService.ServiceId  [get], [set]
```

The service number for the bus service, this is only guaranteed to be unique per operator, not in the API as a whole. For example Reading Buses and Newbury And District both operate a number '2' service.

The documentation for this class was generated from the following file:

- Bus Service/BusService.cs

## 5.4 ReadingBusesAPI.Bus_Stops.BusStop Class Reference

Stores information about a single bus stop. Related to the "List Of Bus Stops" API.

### Public Member Functions

- BusStop (string actoCode)

  *Used to create a snub/ fake object for passing to function calls, if all you need to pass is an acto-code to the function.*
- List< LiveRecord > GetLiveData ()

  *Gets live data from a bus stop.*
- List< BusService > GetServices (Operators busOperator)

  *Finds the 'BusService' object for all of the bus services which visit this stop.*
- Point GetPoint ()

  *Gets the geographical position of the bus stop.*
- Task< BusTimeTable[ ]> GetTimeTable (DateTime date, BusService service=null)

  *Gets time table data at this specific bus stop.*
- Task< ArchivedBusTimeTable[ ]> GetArchivedTimeTable (DateTime date, BusService service=null)

  *Gets the archived real bus departure and arrival times along with their time table history at this specific bus stop.*

## Properties

- string ActoCode `[get, set]`

  *The unique identifier for a bus stop.*
- string CommonName `[get, set]`

  *The public, easy to understand stop name.*
- string Latitude `[get, set]`

  *The latitude of the bus stop*
- string Longitude `[get, set]`

  *The longitude of the bus stop*
- string Bearing `[get, set]`

  *The bearing of the bus stop*
- string Services `[get, set]`
- string GroupName `[get, set]`

  *The Brand/Group of buses that most frequently visit this stop. Such as Purple, for the Purple 17s.*

## 5.4.1 Detailed Description

Stores information about a single bus stop. Related to the "List Of Bus Stops" API.

## 5.4.2 Constructor & Destructor Documentation

### 5.4.2.1 BusStop()

```
ReadingBusesAPI.Bus_Stops.BusStop.BusStop (
            string actoCode )
```

Used to create a snub/ fake object for passing to function calls, if all you need to pass is an acto-code to the function.

**Parameters**

| | |
|---|---|
| *actoCode* | ID of the bus stop. |

Unless you are doing something very strange, you probably should not need to use this, it is more for testing purposes.

## 5.4.3 Member Function Documentation

### 5.4.3.1 GetArchivedTimeTable()

```
Task<ArchivedBusTimeTable[]> ReadingBusesAPI.Bus_Stops.BusStop.GetArchivedTimeTable (
            DateTime date,
            BusService service = null )
```

Gets the archived real bus departure and arrival times along with their time table history at this specific bus stop.

**Parameters**

| | |
|---|---|
| *date* | The date you want time table data for. This should be a date in the past. |
| *service* | (optional) the service you want time table data for specifically. If null, you get time table data for all services at this stop. |

**Returns**

#### 5.4.3.2 GetLiveData()

```
List<LiveRecord> ReadingBusesAPI.Bus_Stops.BusStop.GetLiveData ( )
```

Gets live data from a bus stop.

**Returns**

Returns a list of Live Records, which are individual buses due to arrive at the bus stop.

#### 5.4.3.3 GetPoint()

```
Point ReadingBusesAPI.Bus_Stops.BusStop.GetPoint ( )
```

Gets the geographical position of the bus stop.

**Returns**

A Point Object for the position of the bus stop.

#### 5.4.3.4 GetServices()

```
List<BusService> ReadingBusesAPI.Bus_Stops.BusStop.GetServices (
             Operators busOperator )
```

Finds the 'BusService' object for all of the bus services which visit this stop.

**Parameters**

| | |
|---|---|
| *busOperator* | |

**Returns**

A list of BusService Objects for services which visit this bus stop.

#### 5.4.3.5 GetTimeTable()

```
Task<BusTimeTable[ ]> ReadingBusesAPI.Bus_Stops.BusStop.GetTimeTable (
            DateTime date,
            BusService service = null )
```

Gets time table data at this specific bus stop.

**Parameters**

| | |
|---|---|
| *date* | The date you want time table data for. |
| *service* | (optional) the service you want time table data for specifically. If null, you get time table data for all services at this stop. |

**Returns**

### 5.4.4 Property Documentation

#### 5.4.4.1 Services

```
string ReadingBusesAPI.Bus_Stops.BusStop.Services  [get], [set]
```

The services that travel to this stop, separated by '/'

See BusStop.GetServices(Operators) to get a list of Service Objects.

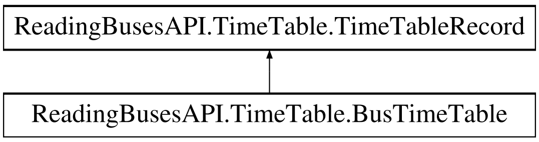
The documentation for this class was generated from the following file:

- Bus Stops/BusStop.cs

## 5.5 ReadingBusesAPI.TimeTable.BusTimeTable Class Reference

Represents and retrieves information about a scheduled/predicted single time table record, which means information on one bus at one location. Related to the "Timetabled Journeys" API.

Inheritance diagram for ReadingBusesAPI.TimeTable.BusTimeTable:


ReadingBusesAPI.TimeTable.TimeTableRecord
ReadingBusesAPI.TimeTable.BusTimeTable

## Additional Inherited Members

### 5.5.1 Detailed Description

Represents and retrieves information about a scheduled/predicted single time table record, which means information on one bus at one location. Related to the "Timetabled Journeys" API.

The documentation for this class was generated from the following file:

- TimeTable/BusTimeTable.cs

## 5.6 ReadingBusesAPI.Vehicle_Positions.GPSController Class Reference

Helps get live and historical GPS data on vehicles by accessing the "Live Vehicle Positions" API.

## Public Member Functions

- async Task< ArchivedPositions[ ]> GetArchivedVehiclePositions (DateTime dateStartTime, TimeSpan? timeSpan, string vehicle=null)

  *Gets historic/archived GPS data for buses on a specific date, filtered either by vehicle ID, or all buses without a time period or both. GPS data is not stored for as long as other forms of data you may fail to get data older than a few months.*
- async Task< LivePosition[ ]> GetLiveVehiclePositions ()

  *Gets live GPS data for all buses currently operating.*
- async Task< LivePosition > GetLiveVehiclePosition (string vehicle)

  *Gets live GPS data for a single buses matching Vehicle ID number.*
- async Task< bool > IsVehicle (string vehicle)

  *Checks if the Vehicle ID Number is currently in service right now.*

### 5.6.1 Detailed Description

Helps get live and historical GPS data on vehicles by accessing the "Live Vehicle Positions" API.

### 5.6.2 Member Function Documentation

#### 5.6.2.1 GetArchivedVehiclePositions()

```
async Task<ArchivedPositions[ ]> ReadingBusesAPI.Vehicle_Positions.GPSController.GetArchived←
VehiclePositions (
            DateTime dateStartTime,
            TimeSpan?  timeSpan,
            string vehicle = null )
```

Gets historic/archived GPS data for buses on a specific date, filtered either by vehicle ID, or all buses without a time period or both. GPS data is not stored for as long as other forms of data you may fail to get data older than a few months.

**Parameters**

| | |
|---|---|
| *dateStartTime* | Vehicle ID Number eg 414 |
| *timeSpan* | (optional) How long a period do you want data for, you can not get multiple days worth of data. If you ask this your result will be automatically truncated to only the start date to midnight. |
| *vehicle* | (optional) Vehicle ID Number eg 414 |

**Returns**

An array of GPS locations at a previous date.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if, you have not choose a date in the past, or the date is too far in the past and so no data exists. Thrown if you have not filtered by either 'timeSpan' or 'vehicle' ID or both. Thrown if the API key is invalid or expired. |

See GPSController.GetLiveVehiclePositions() to get live data instead.

### 5.6.2.2 GetLiveVehiclePosition()

```
async Task<LivePosition> ReadingBusesAPI.Vehicle_Positions.GPSController.GetLiveVehicle↩
Position (
            string vehicle )
```

Gets live GPS data for a single buses matching Vehicle ID number.

**Parameters**

| | |
|---|---|
| *vehicle* | Vehicle ID Number eg 414 |

**Returns**

The GPS point of Vehicle matching your ID provided.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if a vehicle of the ID does not exist or is not currently active. You can check by using the 'IsVehicle' function. |

### 5.6.2.3 GetLiveVehiclePositions()

```
async Task<LivePosition[]> ReadingBusesAPI.Vehicle_Positions.GPSController.GetLiveVehicle↩
Positions ( )
```

Gets live GPS data for all buses currently operating.

**Returns**

An array of GPS locations for all buses operating by Reading Buses currently

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if the API key is invalid or expired. |

#### 5.6.2.4 IsVehicle()

```
async Task<bool> ReadingBusesAPI.Vehicle_Positions.GPSController.IsVehicle (
            string vehicle )
```

Checks if the Vehicle ID Number is currently in service right now.

**Parameters**

| | |
|---|---|
| *vehicle* | Vehicle ID Number eg 414 |

**Returns**

True or False for if the buses GPS can be found or not currently.

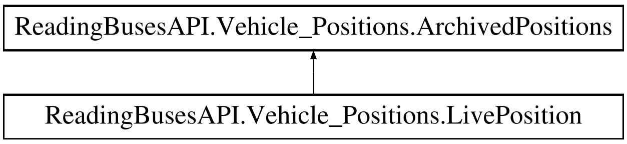
The documentation for this class was generated from the following file:

- Vehicle Positions/GPSController.cs

## 5.7 ReadingBusesAPI.Vehicle_Positions.LivePosition Class Reference

Used to store live information about a buses GPS position. Related to the "Live Vehicle Positions" API.

Inheritance diagram for ReadingBusesAPI.Vehicle_Positions.LivePosition:


ReadingBusesAPI.Vehicle_Positions.ArchivedPositions
ReadingBusesAPI.Vehicle_Positions.LivePosition

### Public Member Functions

- BusService GetService ()

  *Finds the 'BusService' object related to the record.*

## Properties

- string ServiceId `[get, set]`

  *Holds the Service Number for the bus route.*
- string Bearing `[get, set]`

  *bearing direction of the bus*

### 5.7.1 Detailed Description

Used to store live information about a buses GPS position. Related to the "Live Vehicle Positions" API.

### 5.7.2 Member Function Documentation

#### 5.7.2.1 GetService()

```
BusService ReadingBusesAPI.Vehicle_Positions.LivePosition.GetService ( )
```

Finds the 'BusService' object related to the record.

**Returns**

The related 'BusService' object.

The documentation for this class was generated from the following file:

- Vehicle Positions/LivePosition.cs

## 5.8 ReadingBusesAPI.Journey_Details.LiveRecord Class Reference

Used to store information about a buses arrival at a bus stop. Mainly related to the "Stop Predictions" API.

## Public Member Functions

- BusService Service ()

  *Returns the related BusService Object for the Bus LiveRecord.*
- string DisplayTime ()

  *Returns the number of min till bus is due in a min format.*
- double ArrivalMin ()

  *Returns the number of min till the bus is due to arrive.*

## Static Public Member Functions

- static List< LiveRecord > GetLiveData (string actoCode)

  *Gets a list of upcoming arrivals at a specific bus stop. Can throw an exception.*

## Properties

- string ServiceNumber `[get, set]`

  *Holds the Service Number for the bus route.*
- string Destination `[get, set]`

  *Holds the destination for the bus.*
- DateTime SchArrival `[get, set]`

  *Holds scheduled arrival time of the bus at the location.*
- DateTime? ExptArrival `[get, set]`

  *Holds the estimated/ expected arrival time of the bus, if Null no estimated time exists yet.*
- Operators OperatorCode `[get, set]`

  *Holds the operator of the service.*
- string VehicleRef `[get, set]`

  *Holds the Vehicles reference ID or number to identify it.*
- string ViaMessage `[get, set]`

## 5.8.1 Detailed Description

Used to store information about a buses arrival at a bus stop. Mainly related to the "Stop Predictions" API.

## 5.8.2 Member Function Documentation

### 5.8.2.1 ArrivalMin()

```
double ReadingBusesAPI.Journey_Details.LiveRecord.ArrivalMin ( )
```

Returns the number of min till the bus is due to arrive.

**Returns**

The number of min till the bus is due to arrive.

### 5.8.2.2 DisplayTime()

```
string ReadingBusesAPI.Journey_Details.LiveRecord.DisplayTime ( )
```

Returns the number of min till bus is due in a min format.

**Returns**

The number of min until the bus is due to arrive in string format.

### 5.8.2.3 GetLiveData()

```
static List<LiveRecord> ReadingBusesAPI.Journey_Details.LiveRecord.GetLiveData (
             string actoCode ) [static]
```

Gets a list of upcoming arrivals at a specific bus stop. Can throw an exception.

**Parameters**

| | |
|---|---|
| *actoCode* | The Acto-code ID for a specific bus stop. |

**Returns**

A list of Live Records containing details about upcoming buses.

**Exceptions**

| | |
|---|---|
| *NullReferenceException* | Thrown if there is an error with the API. |
| *Exception* | Thrown if you have used an invalid or expired API key. |

#### 5.8.2.4 Service()

```
BusService ReadingBusesAPI.Journey_Details.LiveRecord.Service ( )
```

Returns the related BusService Object for the Bus LiveRecord.

**Returns**

Information about the current bus service object.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Can throw an exception if the service does not exists. This is however very unlikely, if this occurs there is an error in the API, not with your code. |

### 5.8.3 Property Documentation

#### 5.8.3.1 ViaMessage

```
string ReadingBusesAPI.Journey_Details.LiveRecord.ViaMessage  [get], [set]
```

Holds the 'Via' message, which explains where the bus is traveling past on route. Can be null or a place holder value if none exists.

The documentation for this class was generated from the following file:

- Journey Details/LiveRecord.cs

## 5.9 ReadingBusesAPI.Shared.Point Struct Reference

Stores an X and Y Position simply.

### Public Member Functions

- Point (double x, double y)

  *Default constructor*
- override string ToString ()

  *Converts point to string representation.*
- override int GetHashCode ()

  *Generates a unique number for each Point Object.*
- override bool Equals (object obj)

  *Checks if two point objects are the same or not.*
- bool Equals (Point other)

  *Implements logic for checking if two objects are equal.*

### Static Public Member Functions

- static bool operator== (Point point1, Point point2)

  *Checks if two objects are the same.*
- static bool operator!= (Point point1, Point point2)

  *Checks if two objects are the not the same.*

### Properties

- double X `[get]`

  *The X value.*
- double Y `[get]`

  *The Y Value.*

### 5.9.1 Detailed Description

Stores an X and Y Position simply.

### 5.9.2 Constructor & Destructor Documentation

#### 5.9.2.1 Point()

```
ReadingBusesAPI.Shared.Point.Point (
            double x,
            double y )
```

Default constructor

**Parameters**

|  |  |
|---|---|
| *x* | X value of Point. |
| *y* | Y value of Point. |

## 5.9.3 Member Function Documentation

### 5.9.3.1 Equals() [1/2]

```
override bool ReadingBusesAPI.Shared.Point.Equals (
            object obj )
```

Checks if two point objects are the same or not.

**Parameters**

|  |  |
|---|---|
| *obj* | Another object to compare against this object. |

**Returns**

Is it the same object or not.

### 5.9.3.2 Equals() [2/2]

```
bool ReadingBusesAPI.Shared.Point.Equals (
            Point other )
```

Implements logic for checking if two objects are equal.

**Parameters**

|  |  |
|---|---|
| *other* | The other object to check if equal. |

**Returns**

True if equals else false.

### 5.9.3.3 GetHashCode()

```
override int ReadingBusesAPI.Shared.Point.GetHashCode ( )
```

Generates a unique number for each Point Object.

**Returns**

Int value of object.

### 5.9.3.4 operator"!=()

```
static bool ReadingBusesAPI.Shared.Point.operator!= (
            Point point1,
            Point point2 )  [static]
```

Checks if two objects are the not the same.

**Parameters**

| | |
|---|---|
| *point1* | First Point Object. |
| *point2* | Second Point Object. |

**Returns**

True if they are not the same.

### 5.9.3.5 operator==()

```
static bool ReadingBusesAPI.Shared.Point.operator== (
            Point point1,
            Point point2 )  [static]
```

Checks if two objects are the same.

**Parameters**

| | |
|---|---|
| *point1* | First Point Object. |
| *point2* | Second Point Object. |

**Returns**

True if equal else false

### 5.9.3.6 ToString()

```
override string ReadingBusesAPI.Shared.Point.ToString ( )
```

Converts point to string representation.

**Returns**

Point as a string.

The documentation for this struct was generated from the following file:

- Shared/Point.cs

## 5.10 ReadingBusesAPI.ReadingBuses Class Reference

This is the main class for the library, here you can initialise a singleton instance and then query and use the Reading Buses API.

### Public Member Functions

- BusStop GetLocation (string actoCode)

  *Get a bus stop location based upon a bus stops location code*
- BusStop[ ] GetLocations ()

  *All the bus stop locations that Reading Buses Visits*
- bool IsLocation (string actoCode)

  *Checks to see if the acto code for the bus stop exists in the API feed or not.*
- BusService[ ] GetServices ()

  *All the Services Reading Buses Operates*
- BusService[ ] GetServices (string brandName)

  *Returns all services Reading Buses Operates under a brand name, for example "pink" would return "22,25,27,29" services.*
- BusService[ ] GetService (string serviceNumber)

  *Returns a service which matches the Service Number passed, because the Reading Buses API now supports, Kinnections and Newbury and District a service number can no longer be considered unique.*
- BusService GetService (string serviceNumber, Operators operators)

  *Returns a service which matches the Service Number passed and the bus operator.*
- bool IsService (string serviceNumber)

  *Checks to see if a service of that number exists or not in the API feed.*
- bool IsService (string serviceNumber, Operators operators)

  *Checks to see if a service of that number exists or not in the API feed, for a specific bus operator.*
- void PrintServices ()

  *Prints off all the services found by the API which Reading Buses Operates*
- Task< ArchivedBusTimeTable[ ]> GetVehicleTrackingHistory (DateTime date, string vehicle)

  *Gets the archived real bus departure and arrival times along with their time table history for a specific vehicle, on a specific date. This can be used to find how late a vehicle was throughout that day.*

## Static Public Member Functions

- static void SetCache (bool value)

  *Sets if you want to cache data into local files or always get new data from the API, which will take longer.*
- static void SetDebugging (bool value)

  *Sets if you want to debug the library by making requests to a dummy server instead of the real live sever.*
- static void SetWarning (bool value)

  *Sets if you want to print out warning messages to the console screen or not.*
- static void SetFullError (bool value)

  *Sets if you want to print out the full error logs to console, only needed for debugging library errors.*
- static void SetCacheValidityLength (int days)

  *Sets how long to keep Cache data for before invalidating it and getting new data.*
- static void InvalidateCache ()

  *Deletes any Cache data stored, Cache data is deleted automatically after a number of days, use this only if you need to force new data early.*
- static async Task< ReadingBuses > Initialise (string apiKey)

  *Used to initially initialise the ReadingBuses Object, it is recommended you do this in your programs start up.*
- static ReadingBuses GetInstance ()

  *You will never need more than one ReadingBuses object, a singleton is used to ensure you always get the same instance.*

## Properties

- GPSController GPSController `[get]`

  *Stores the GPS controller, which can help get vehicle GPS data.*

### 5.10.1 Detailed Description

This is the main class for the library, here you can initialise a singleton instance and then query and use the Reading Buses API.

```
//Optional
ReadingBuses.SetCache(true);
ReadingBuses Controller =  await ReadingBuses.Initialise("API KEY HERE");
BusService service = Controller.GetService("17"); or ReadingBuses.GetInstance().GetService("17");
```

Cached Data is data stored locally in JSON and XML files, stored in a hidden folder called "cache", in the same directory the program is executed from. This is a copy of the results from an API call, such as the bus services and bus stops, because it is unlikely for this data to change regularly. By default the cached data will be updated every 7 days, but you can request new data or disable cache if you wish. Caching data is however faster as you do not need to keep making API requests for data likely to be the same.

### 5.10.2 Member Function Documentation

#### 5.10.2.1 GetInstance()

`static ReadingBuses ReadingBusesAPI.ReadingBuses.GetInstance ( )  [static]`

You will never need more than one ReadingBuses object, a singleton is used to ensure you always get the same instance.

**Returns**

Returns the ReadingBuses object to be used throughout your program.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if you attempt to get an instance before you have called the "Initialise" function. |

See ReadingBuses.Initialise(string) to initially initialise the ReadingBuses Object singleton.

#### 5.10.2.2 GetLocation()

```
BusStop ReadingBusesAPI.ReadingBuses.GetLocation (
            string actoCode )
```

Get a bus stop location based upon a bus stops location code

**Parameters**

| | |
|---|---|
| *actoCode* | The code of the bus stop |

**Returns**

A Bus Stop object for the Acto Code specified.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if the bus stop does not exist. You should first check with 'IsLocation' If there is any uncertainty. |

See ReadingBuses.IsLocation(string) to check if it is a location.

#### 5.10.2.3 GetLocations()

```
BusStop [] ReadingBusesAPI.ReadingBuses.GetLocations ( )
```

All the bus stop locations that Reading Buses Visits

**Returns**

All the bus stops Reading Buses visits

#### 5.10.2.4 GetService() [1/2]

```
BusService [] ReadingBusesAPI.ReadingBuses.GetService (
            string serviceNumber )
```

Returns a service which matches the Service Number passed, because the Reading Buses API now supports, Kinnections and Newbury and District a service number can no longer be considered unique.

**Parameters**

| | |
|---|---|
| *serviceNumber* | The service number/ID for the service you wish to be returned eg: 17 or 22. |

**Returns**

The services matching the ID.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if the bus services does not exist. You should first check with 'IsService' If there is any uncertainty. |

See ReadingBuses.IsService(string) to check if it is a service.

#### 5.10.2.5 GetService() [2/2]

```
BusService ReadingBusesAPI.ReadingBuses.GetService (
            string serviceNumber,
            Operators operators )
```

Returns a service which matches the Service Number passed and the bus operator.

**Parameters**

| | |
|---|---|
| *serviceNumber* | The service number/ID for the service you wish to be returned eg: 17 or 22. |
| *operators* | The bus operator to search in, for example "ReadingBuses" |

**Returns**

The services matching the ID.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if the bus services does not exist. You should first check with 'IsService' If there is any uncertainty. |

See ReadingBuses.IsService(string) to check if it is a service.

#### 5.10.2.6 GetServices() [1/2]

```
BusService [ ] ReadingBusesAPI.ReadingBuses.GetServices ( )
```

All the Services Reading Buses Operates

**Returns**

All the Services Reading Buses Operates

#### 5.10.2.7 GetServices() [2/2]

```
BusService [ ] ReadingBusesAPI.ReadingBuses.GetServices (
             string brandName )
```

Returns all services Reading Buses Operates under a brand name, for example "pink" would return "22,25,27,29" services.

**Parameters**

| | |
|---|---|
| *brandName* | The brand name for the services you wish to find, eg "pink" or "sky blue". |

**Returns**

An array of Bus Services which are of the brand name specified.

#### 5.10.2.8 GetVehicleTrackingHistory()

```
Task<ArchivedBusTimeTable[ ]> ReadingBusesAPI.ReadingBuses.GetVehicleTrackingHistory (
             DateTime date,
             string vehicle )
```

Gets the archived real bus departure and arrival times along with their time table history for a specific vehicle, on a specific date. This can be used to find how late a vehicle was throughout that day.

**Parameters**

| | |
|---|---|
| *date* | The date you want a report for, must be in the past. |
| *vehicle* | The vehicle ID number |

**Returns**

An array of Archived Bus Departure and arrival times with their timetabled data.

#### 5.10.2.9 Initialise()

```
static async Task<ReadingBuses> ReadingBusesAPI.ReadingBuses.Initialise (
             string apiKey )  [static]
```

Used to initially initialise the ReadingBuses Object, it is recommended you do this in your programs start up.

**Parameters**

| | |
|---|---|
| *apiKey* | The Reading Buses API Key, get your own from `http://rtl2.ods-live.co.uk/cms/apiservice` |

**Returns**

An instance of the library controller. This same instance can be got by calling the "GetInstance" method.

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Can throw an exception if you pass an invalid or expired API Key. |

See ReadingBuses.GetInstance() to get any future instances afterwards.

#### 5.10.2.10 InvalidateCache()

```
static void ReadingBusesAPI.ReadingBuses.InvalidateCache ( )  [static]
```

Deletes any Cache data stored, Cache data is deleted automatically after a number of days, use this only if you need to force new data early.

#### 5.10.2.11 IsLocation()

```
bool ReadingBusesAPI.ReadingBuses.IsLocation (
             string actoCode )
```

Checks to see if the acto code for the bus stop exists in the API feed or not.

**Parameters**

| | |
|---|---|
| *actoCode* | The ID Code for a bus stop. |

**Returns**

True or False depending on if the stop is in the API feed or not.

#### 5.10.2.12 IsService() [1/2]

```
bool ReadingBusesAPI.ReadingBuses.IsService (
             string serviceNumber )
```

Checks to see if a service of that number exists or not in the API feed.

**Parameters**

| | |
|---|---|
| *serviceNumber* | The service number to find. |

**Returns**

True or False for if a service is the API feed or not.

#### 5.10.2.13 IsService() [2/2]

```
bool ReadingBusesAPI.ReadingBuses.IsService (
             string serviceNumber,
             Operators operators )
```

Checks to see if a service of that number exists or not in the API feed, for a specific bus operator.

**Parameters**

| | |
|---|---|
| *serviceNumber* | The service number to find. |
| *operators* | The specific bus operator you want to search in. |

**Returns**

True or False for if a service is the API feed or not.

#### 5.10.2.14 PrintServices()

```
void ReadingBusesAPI.ReadingBuses.PrintServices ( )
```

Prints off all the services found by the API which Reading Buses Operates

#### 5.10.2.15 SetCache()

```
static void ReadingBusesAPI.ReadingBuses.SetCache (
             bool value )  [static]
```

Sets if you want to cache data into local files or always get new data from the API, which will take longer.

**Parameters**

| | |
|---|---|
| *value* | True or False for if you want to get Cache or live data. |

**Exceptions**

| | |
|---|---|
| *InvalidOperationException* | Thrown if you attempt to change the cache options after the library has been instantiated |

#### 5.10.2.16 SetCacheValidityLength()

```
static void ReadingBusesAPI.ReadingBuses.SetCacheValidityLength (
             int days )  [static]
```

Sets how long to keep Cache data for before invalidating it and getting new data.

**Parameters**

| | |
|---|---|
| *days* | The number of days to store the cache data for before getting new data. |

#### 5.10.2.17 SetDebugging()

```
static void ReadingBusesAPI.ReadingBuses.SetDebugging (
             bool value )  [static]
```

Sets if you want to debug the library by making requests to a dummy server instead of the real live sever.

**Parameters**

| | |
|---|---|
| *value* | True or False for if you want to debug or not. |

Unless you are developing or editing library in some way you should not need to use this.

#### 5.10.2.18 SetFullError()

```
static void ReadingBusesAPI.ReadingBuses.SetFullError (
             bool value )  [static]
```

Sets if you want to print out the full error logs to console, only needed for debugging library errors.

**Parameters**

| | |
|---|---|
| *value* | True or False for printing full error logs to console. |

#### 5.10.2.19 SetWarning()

```
static void ReadingBusesAPI.ReadingBuses.SetWarning (
             bool value )  [static]
```

Sets if you want to print out warning messages to the console screen or not.

**Parameters**

| | |
|---|---|
| *value* | True or False for printing warning messages. |

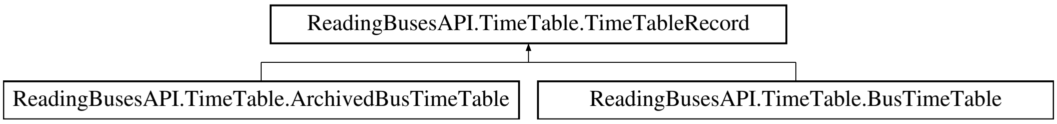
The documentation for this class was generated from the following file:

- ReadingBuses.cs

## 5.11 ReadingBusesAPI.TimeTable.TimeTableRecord Class Reference

Represents the Raw timetable object data you get from the Timetabled Journeys and Tracking History APIs.

Inheritance diagram for ReadingBusesAPI.TimeTable.TimeTableRecord:


ReadingBusesAPI.TimeTable.TimeTableRecord
ReadingBusesAPI.TimeTable.ArchivedBusTimeTable
ReadingBusesAPI.TimeTable.BusTimeTable

### Public Member Functions

- BusService GetService ()

  *Gets the related 'BusService' object relating to the time table record.*

### Properties

- Operators Operator `[get, set]`

  *The operator of the bus services*
- BusStop Location `[get, set]`

  *The 'BusStop' object for the stop relating to the time table record..*
- long Sequence `[get, set]`

  *What number bus stop is this in the buses route, ie 1, is the first stop to visit.*
- Direction Direction `[get, set]`

  *Is this bus heading inbound or outbound.*
- string JourneyCode `[get, set]`
- bool IsTimingPoint `[get, set]`
- DateTime SchArrivalTime `[get, set]`

  *The scheduled arrival time for the bus.*
- DateTime SchDepartureTime `[get, set]`

  *The scheduled departure time for the bus.*

### 5.11.1 Detailed Description

Represents the Raw timetable object data you get from the Timetabled Journeys and Tracking History APIs.

## 5.11.2 Member Function Documentation

### 5.11.2.1 GetService()

```
BusService ReadingBusesAPI.TimeTable.TimeTableRecord.GetService ( )
```

Gets the related 'BusService' object relating to the time table record.

**Returns**

A 'BusService' object for this time table record.

## 5.11.3 Property Documentation

### 5.11.3.1 IsTimingPoint

```
bool ReadingBusesAPI.TimeTable.TimeTableRecord.IsTimingPoint  [get], [set]
```

Is this bus stop a timing point or not.

A timing point is a major bus stop, where the buses is expected to wait if its early and should actually arrive on the scheduled time. All non-timing points times are only estimated scheduled times. A timing point is much more accurate and strict timings.

### 5.11.3.2 JourneyCode

```
string ReadingBusesAPI.TimeTable.TimeTableRecord.JourneyCode  [get], [set]
```

A unique value that groups a selection of time table records across different bus stops to show one loop/ cycle of a bus services route.

The documentation for this class was generated from the following file:

- TimeTable/TimeTableRecord.cs

# Index