BusBuddy

Generated by Doxygen 1.8.3.1

Sun Apr 21 2013 21:55:03

Contents

1 Namespace Index

1.1 Packages

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

2 Hierarchical Index

2.1 Class Hierarchy

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

3 Class Index

3.1 Class List

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

4 Namespace Documentation

4.1 Package alert

The Alert Module.

4.1.1 Detailed Description

The Alert Module.

4.2 Package common

This package contains common BusBuddy objects and utilities to be used by all modules.

Classes

- class BaseController

  This is a base class to be extended by each of the controller classes.
- class BusBuddyBadRequestException

  This exception object represents internal errors which may occur as a result of an error in the client's request.
- class BusBuddyException

  This exception object is an abstract base class.
- class BusBuddyForbiddenException

  This exception object represents internal errors which may occur as a result of attempts to access a resource without authorization.
- class BusBuddyInternalException

  This exception object represents internal errors which may occur, which are generally not due to the specifics of what appears to be a valid request.
- class BusBuddyNotFoundException

  This exception object represents the error that occurs when a resource cannot be found.
- class HashUtility

  This is a utility class to handle secure hashes.
- class MessageDeliveryUtility

  This is a utility class to handle message delivery, such as through email or SMS.
- interface Specification$< T >$

  A Generic Specification to be used for chaining business validation rules together.

4.2.1 Detailed Description

This package contains common BusBuddy objects and utilities to be used by all modules.

4.3 Package tracking

The Tracking Module.

Classes

- class AlertFactory

  The Alert Factory handles the creation of a user alert.
- interface AlertSpecification

  Interface for Alert Specifications which contain the business logic used to determine if an alert should be triggered for a vehicle.
- class AlertZoneLogic

  Alert Zone Logic implements the business logic to determine is a vehicle is within a zone where an alert needs to be sent to a user who has registered for tracking alerts.
- class BusVehicle
- class CommercialTracking

  Implements Subject Location Tracking for retrieving GPS location updates from outside commercial tracking services.
- class DelayAlertLogic

  Implements the business logic to determine if the vehicle is behind schedule or not reporting GPS updates and send a notification to the transit company.
- class GPSLocationObserver

  Observer Pattern - Observer interface for GPS location tracking.
- class GPSLocationTracking

  GPSLocationTracking - interface Subject of the Observer Pattern Defines methods for an observer GPS Device to register and receive updates on vehicle location.

- class GPSPuller

  GPS Puller is a concrete implementation of GPS Location tracker for obtaining coordinates directly from a GPS device installed in a registered vehicle.
- class GPSPusher

  Implements Subject Location Tracking for retrieving GPS location updates from registered vehicles.
- class GPSVehicleTracker

  Implementation of the Observer, update the transit vehicle GPS location.
- interface ITrackingService
- class TrackingAlertObserver

  Abstract class defining the methods for the tracking alert observer.
- class TrackingDelayAlert
- class TrackingLocationAlert
- class TrackingServiceController
- class TransitVehicle

  Abstract transit vehicle class contains the common data for all types of vehicles and the Subject GPS Tracking and the GPS observer to receive GPS location updates.
- class TransitVehicleFactory

  Transit Vehicle Factory encapsulates the complexity of creating a new vehicle.
- class VehicleObject

  Value Object containing vehicle information obtained when the user registers a vehicle using the user interface.
- class VehicleRepository

  Repository for information on vehicles registered on a route.

4.3.1 Detailed Description

The Tracking Module.

4.4 Package transit

The Transit Module is an interface to get Route/Fare/Detour information from a TransitProvider.

Classes

- class AbstractFeedParserTemplate

  A Template Method pattern to allow for the import of data from different TransitProviders in potentially different formats.
- class Detour

  A disruption in service due to an unexpected event.
- class Fare

  An immutable Value Object representing the cost, or 'fare,' required to ride a TransitVehicle on a particular Route.
- class GoogleTransitServiceAdapter

  An Adapter Class to allow a {} service to appear as a TransitService.
- interface GoogleTransitServiceAPI

  A client to Google's `Maps API`.
- class GTFSFeedParser

  A AbstractFeedParserTemplate implementation designed to parse `GTFS` format ZIP files into Routes.
- class InvalidRouteParseException

  An InvalidRouteParseException indicates an invalid batch of parsed Routes has been been detected.
- class ITeamTransitService

  The iTeam implementation of the TransitService that exposes Transit data via a REST Service.
- class ITeamTripService

  An iTeam implementation of the TripService that exposes Trip data via a REST Service.

- class Location

  An immutable Value Object representing a physical point on the geographic coordinate system.
- class PersistedTransitFeed

  An implementation of the TransitFeed interface that communicates with a RouteRepository to retrieve its data.
- class Route

  A Route is a TransitVehicle path of travel, or a "Line," as referred to by a TransitProvider.
- class RouteDisruptionAlert

  An Alert indicating a disruption of normal Route availability or scheduling.
- interface RouteRepository

  A Repository Pattern supporting lifecycle operations of Routes, such as Read, Save, Delete, and Query functionality.
- class RouteSpecification

  A Specification Pattern class for validating a Route.
- class Stop

  A point on a Route in which a TransitVehicle will stop to pick up and drop off passengers.
- interface TransitFeed

  A TransitFeed is an abstraction over a service or set of services that provide information about Routes.
- class TransitInfo

  An immutable Value Object describing metadata about a TransitService.
- class TransitProvider

  A TransitProvider is a description of a company or organization that is the producer of public transportation services.
- interface TransitService

  The TransitService is an interface to get Route/Fare/Detour information from a TransitProvider.
- class Trip

  A Trip is considered an ordered collection of Routes going from a starting point to an ending point.
- interface TripService

  A Service to calculate a collection of Routes, or a Trip, allowing for a continuous transit path from a start Location to an end Location.

4.4.1 Detailed Description

The Transit Module is an interface to get Route/Fare/Detour information from a TransitProvider. The main module interface, the TransitService, provides a consistent interface for the application logic to query for this information.

From a design perspective, there are two main tasks performed by the Transit Module: Consuming Transit Information from a TransitProvider, and Providing Transit Information to Users.

4.5 Package user

This package contains the objects used by the User Module of the BusBuddy application.

Classes

- class ITeamUserLoginService

  This is the iTeam's implementation of UserLoginService.
- class ITeamUserManagementService
- class Session

  This class represents a single session for a user of the system, and all of the state data associated with that session.
- class SessionRepository

  This class is responsible for handling database access for Sessions, and to construct, persist, and retrieve Session objects.
- class User

This class represents a single user of the system, and all of the state data associated with that user.

- interface UserLoginService

  This is the generic BusBuddy UserLoginService interface.
- interface UserManagementService

  This is the generic BusBuddy UserManagementService interface.
- class UserRepository

  This class is responsible for handling database access for User objects, and to construct, persist, and retrieve User objects.
- enum UserType

4.5.1 Detailed Description

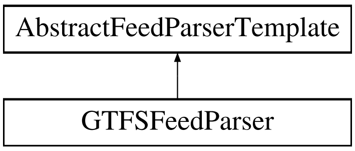
This package contains the objects used by the User Module of the BusBuddy application.

5 Class Documentation

5.1 AbstractFeedParserTemplate Class Reference

A Template Method pattern to allow for the import of data from different TransitProviders in potentially different formats.

Inheritance diagram for AbstractFeedParserTemplate:



Public Member Functions

- RouteRepository **getRouteRepository** ()
- void **setRouteRepository** (RouteRepository routeRepository)
- Specification< Route > **getRouteSpecification** ()
- void **setRouteSpecification** (Specification< Route > routeSpecification)

Protected Member Functions

- void start (URL location) throws InvalidRouteParseException

  The start method initiates the process and calls the appropriate methods in the appropriate order.
- InputStream loadFeed (URL location)

  Converts the resource URL into an InputStream for further processing.
- abstract Set< Route > parseFeed (InputStream feed)

  Parses the feed InputStream into a Set of Routes.
- boolean validate (Route route)

  Allow subclasses to validate Routes as they are parsed.
- void saveRoutes (Set< Route > routes)

  Save the Routes to the RouteRepository.

Private Attributes

- RouteRepository routeRepository

  The RouteRepository dependency allows for the persistence of the parsed Routes.
- Specification< Route > routeSpecification

  This Specification allows subclasses to validate Routes as they are parsed.

5.1.1 Detailed Description

A Template Method pattern to allow for the import of data from different TransitProviders in potentially different formats.

The algorithm sequence is as follows:

1. A URL of a resource location is passed into the start(URL) method. This method initiates the parsing/transformation process.
2. The start(URL) method calls the method loadFeed(URL) to establish the InputStream.
3. The InputStream returned by loadFeed(URL) is passed into the abstract parseFeed(InputStream) method. Subclasses will implement this as necessary to produce the resulting Routes
4. The newly created Routes are saved to the RouteRepository via the saveRoutes(Set<Route>) method.

5.1.2 Member Function Documentation

5.1.2.1 InputStream loadFeed (URL *location*) `[protected]`

Converts the resource URL into an InputStream for further processing.

Precondition

location exists and has been validated.

Parameters

location	The resource location

Returns

The resulting InputStream

5.1.2.2 abstract Set<Route> parseFeed (InputStream *feed*) `[protected]`, `[pure virtual]`

Parses the feed InputStream into a Set of Routes.

Subclasses will implement this abstract method with the appropriate parsing logic for the particular input format.

Parameters

feed	The resource InputStream

Returns

The resulting Set of Routes

Implemented in GTFSFeedParser.

5.1.2.3 void saveRoutes (Set< Route > *routes*) [protected]

Save the Routes to the RouteRepository.

Precondition

routes may be an empty Set, but must not be null.

Parameters

routes	The Set of Routes to persist.

5.1.2.4 void start (URL *location*) throws InvalidRouteParseException [protected]

The start method initiates the process and calls the appropriate methods in the appropriate order.

Exceptions

InvalidRouteParseException	if any of the parsed Routes fail to validate via the given routeSpecification.

Parameters

location	The input data resource location. This may be a local file or a remote resource.

5.1.2.5 boolean validate (Route *route*) [protected]

Allow subclasses to validate Routes as they are parsed.

Subclasses are encouraged to use this method

Parameters

route	the route

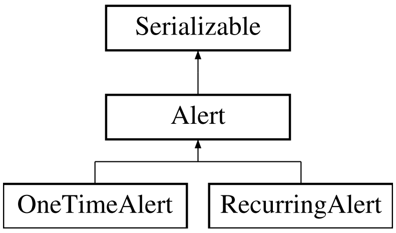
Returns

true, if successful

5.2 Alert Class Reference

This is a base Alert Model that has most of the common information about an Alert.

Inheritance diagram for Alert:



Public Member Functions

- String getAlertGuid ()
- void setAlertGuid (String alertGuid)

- String getDescription ()
- void setDescription (String description)
- Date getCreatedDateTime ()
- void setCreatedDateTime (Date createdDateTime)
- Date getStartDateTime ()
- void setStartDateTime (Date startDateTime)
- Date getExpireDateTime ()
- void setExpireDateTime (Date expireDateTime)
- AlertStatus getStatus ()
- void setStatus (AlertStatus status)
- int getErrorCount ()
- void setErrorCount (int errorCount)
- AlertNotificationType getAlertType ()
- void setAlertType (AlertNotificationType alertType)
- AlertRunType getAlertRunType ()
- void setAlertRunType (AlertRunType alertRunType)

Private Attributes

- String alertGuid

  A unique identifier for Alert.
- String description

  A text description about the alert that the user or other modules want to remember.
- Date createdDateTime

  DateTime that the alert was created.
- Date startDateTime

  DateTime that the alert should start running.
- Date expireDateTime

  DateTime that the alert would expire.
- AlertStatus Status

  current status of the alert.
- int errorCount

  Count of error occurrence when the alert was ran.
- AlertNotificationType alertType

  Notification type of alert.
- AlertRunType alertRunType

  Run type of alert e.g., one time or recurring.

Static Private Attributes

- static final long **serialVersionUID** = -5671884600600864426L

5.2.1 Detailed Description

This is a base Alert Model that has most of the common information about an Alert.

OneTimeAlert and RecurringAlert extends this Alert Model. Alert can only be created from AlertFactory and then manipulated from AlertRepository.

5.2.2 Member Function Documentation

5.2.2.1 String getAlertGuid ( )

Returns

the alertGuid

5.2.2.2 AlertRunType getAlertRunType ( )

Returns

the alertRunType

5.2.2.3 AlertNotificationType getAlertType ( )

Returns

the alertType

5.2.2.4 Date getCreatedDateTime ( )

Returns

the createdDateTime

5.2.2.5 String getDescription ( )

Returns

the description

5.2.2.6 int getErrorCount ( )

Returns

the errorCount

5.2.2.7 Date getExpireDateTime ( )

Returns

the expireDateTime

5.2.2.8 Date getStartDateTime ( )

Returns

the startDateTime

5.2.2.9 AlertStatus getStatus ( )

Returns

the status

5.2.2.10 void setAlertGuid (String *alertGuid*)

Parameters

alertGuid	the alertGuid to set

5.2.2.11 void setAlertRunType (AlertRunType *alertRunType*)

Parameters

alertRunType	the alertRunType to set

5.2.2.12 void setAlertType (AlertNotificationType *alertType*)

Parameters

alertType	the alertType to set

5.2.2.13 void setCreatedDateTime (Date *createdDateTime*)

Parameters

createdDateTime	the createdDateTime to set

5.2.2.14 void setDescription (String *description*)

Parameters

description	the description to set

5.2.2.15 void setErrorCount (int *errorCount*)

Parameters

errorCount	the errorCount to set

5.2.2.16 void setExpireDateTime (Date *expireDateTime*)

Parameters

expireDateTime	the expireDateTime to set

5.2.2.17 void setStartDateTime (Date *startDateTime*)

Parameters

startDateTime	the startDateTime to set

5.2.2.18 void setStatus (AlertStatus *status*)

Parameters

status	the status to set

5.2.3 Member Data Documentation

5.2.3.1 AlertRunType alertRunType `[private]`

Run type of alert e.g., one time or recurring.

Value is defined by AlertRunType

5.2.3.2 AlertNotificationType alertType `[private]`

Notification type of alert.

Depends upon the value as specified in AlertNotificationType

5.2.3.3 AlertStatus Status [private]

current status of the alert.

The value depends upon AlertStatus enum.

5.3 AlertExecuteStrategyFactory Class Reference

Static Public Member Functions

- static AlertService **getAlertService** (AlertRequestModel requestModel)

5.4 AlertFactory Class Reference

The Alert Factory handles the creation of a user alert.

Public Member Functions

- TrackingAlertObserver createAlertObserver (TransitVehicle vehicle)

5.4.1 Detailed Description

The Alert Factory handles the creation of a user alert.

The necessary values for an alert will be entered by a registered user from the BusBuddy User Interface. See UserTrackingAlertObject for input parameter details.

5.4.2 Member Function Documentation

5.4.2.1 TrackingAlertObserver createAlertObserver (TransitVehicle *vehicle*)

1. Determine what type of tracking alert observer to create

1. Determine what rules are needed and add an Alert Specification to this alert. Configuration for alert logic will be obtained based on a configuration file.

5.5 AlertFactory Class Reference

Public Member Functions

- Alert **createAlert** (Alert alert)

5.6 AlertInitiator Enum Reference

Public Attributes

- **UserModule**
- **TrackingModule**
- **TransitModule**

5.7 AlertNotificationType Enum Reference

Public Attributes

- **PlannedDisruption**
- **UnplannedDisruption**
- **ScheduleInformation**

5.8 AlertRecurringType Enum Reference

Public Attributes

- **Yearly**
- **Monthly**
- **Weekly**
- **Daily**

5.9 AlertRepository Class Reference

Public Member Functions

- Alert saveAlert (Alert alertModel)

  This methods take an Alert and saves it to the database.
- boolean deleteAlert (Alert alertModel)

  This method deletes the alert that is being passed.
- Alert updateAlert (Alert alertModel)

  This method is used to update the alert with new information.
- List< Alert > getAlertByDateTime (Date dateTimeToFetch, int offsetMinute)

  This method fetches all the alerts that is to be run in next couple of minutes of given date and time.
- List< Alert > **getAlertByRoute** (String routeId)
- List< Alert > **getAlertByUserId** (String userId)

5.9.1 Member Function Documentation

5.9.1.1 boolean deleteAlert (Alert *alertModel*)

This method deletes the alert that is being passed.

Precondition

the alertModel being passed at least needs to have an ID defined.

Postcondition

the alert will be removed from the system and can no longer be accessed.

Parameters

alertModel	. The alert that is to be deleted.

Returns

A boolean to indicate whether the delete was success or not.

5.9.1.2 List<Alert> getAlertByDateTime (Date *dateTimeToFetch,* int *offsetMinute*)

This method fetches all the alerts that is to be run in next couple of minutes of given date and time.

e.g., if DateTime is NOW and offset is 5 minutes. Then it fetches all the alerts that is to be run in next 5 minutes.

Parameters

dateTimeToFetch	. DateTime when the alert is supposed to run.
offsetMinute	

Returns

Returns a list of alert that is to be run in next couple of minutes (offsetMinute) of given date time.

5.9.1.3 Alert saveAlert (Alert *alertModel*)

This methods take an Alert and saves it to the database.

Parameters

alertModel	

Returns

The saved object with updated property.

Save the alert via Hibernate.

5.9.1.4 Alert updateAlert (Alert *alertModel*)

This method is used to update the alert with new information.

Precondition

the alert must exist in the system.

Parameters

alertModel	

Returns

Returns the updated model back to the method that is calling.

5.10 AlertRequestController Class Reference

Public Member Functions

- AlertResponseModel **processUserAlertRequest** (UserAlertRequestModel userAlertRequest)

Private Attributes

- ISessionHandler **sessionHandler**

5.11 AlertRequestModel Class Reference

5.12 AlertResponseModel Class Reference

Private Attributes

- String **requestCompete**
- String **errorMessage**

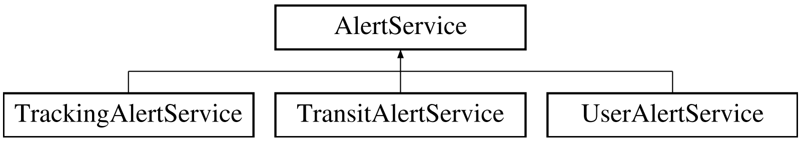
5.13 AlertRunType Enum Reference

Public Attributes

- **Onetime**
- **Recurring**

5.14 AlertService Class Reference

Inheritance diagram for AlertService:



Public Member Functions

- abstract AlertResponseModel **createAlert** (AlertRequestModel requestModel)
- AlertResponseModel **saveAlert** (Alert alertModel)
- AlertResponseModel **deleteAlert** (Alert alertModel)
- AlertResponseModel **updateAlert** (Alert alertModel)

Package Attributes

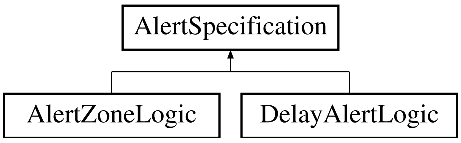
- IAlertExecuteStrategy **alertExecuteStrategy**
- AlertRepository **alertRepository**

5.15 AlertServiceFactory Class Reference

5.16 AlertSpecification Interface Reference

Interface for Alert Specifications which contain the business logic used to determine if an alert should be triggered for a vehicle.

Inheritance diagram for AlertSpecification:



Public Member Functions

- boolean **inAlertZone** (Date lastUpdateTime, Location vehicleLocation)

5.16.1 Detailed Description

Interface for Alert Specifications which contain the business logic used to determine if an alert should be triggered for a vehicle.

Alert specifications are referenced in the vehicle tracking observer and used by the subject to determine when to send an alert. This is to reduce the number of false positive alerts.

5.17 AlertStatus Enum Reference

Public Attributes

- **Active**
- **Deactive**
- **Running**
- **Expired**
- **Error**

5.18 UserTrackingAlertObject.AlertType Enum Reference

Public Attributes

- **TRACKING**
- **DELAY**

5.19 ITrackingService.AlertType Enum Reference

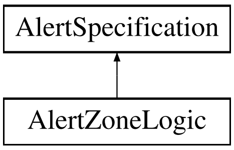
Public Attributes

- **LOCATION**
- **DELAY**

5.20 AlertZoneLogic Class Reference

Alert Zone Logic implements the business logic to determine is a vehicle is within a zone where an alert needs to be sent to a user who has registered for tracking alerts.

Inheritance diagram for AlertZoneLogic:



Public Member Functions

- boolean **inAlertZone** (Date lastUpdateTime, Location vehicleLocation)

5.20.1 Detailed Description

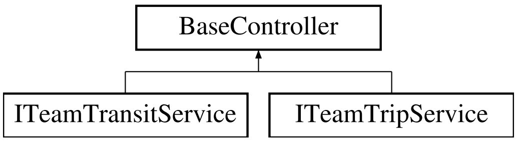
Alert Zone Logic implements the business logic to determine is a vehicle is within a zone where an alert needs to be sent to a user who has registered for tracking alerts.

This logic is designed to guarantee that an alert will be sent to the user before the vehicle has passes the desired stop.

5.21 BaseController Class Reference

This is a base class to be extended by each of the controller classes.

Inheritance diagram for BaseController:



Public Member Functions

- ResponseEntity< String > handleBusBuddyException (BusBuddyException e)

  This method handles cases where BusBuddyException is thrown from controller methods.
- ResponseEntity< String > handleGenericException (BusBuddyException e)

  This method handles cases where a generic Exception is thrown from controller methods (other than BusBuddyException).

5.21.1 Detailed Description

This is a base class to be extended by each of the controller classes.

This provides a means to handle exceptions that need to be thrown back up to the user. It could be modified to add other common logic that apply to multiple controllers.

5.21.2 Member Function Documentation

5.21.2.1 ResponseEntity<String> handleBusBuddyException (BusBuddyException *e*)

This method handles cases where BusBuddyException is thrown from controller methods.

It will format the exception for the user, and return the correct HTTP status code, based on the code stored within the exception.

Parameters

e	exception which was thrown

Returns

ResponseEntity object

5.21.2.2 ResponseEntity<String> handleGenericException (BusBuddyException *e*)

This method handles cases where a generic Exception is thrown from controller methods (other than BusBuddyException).

It will format the exception for the user, and return a generic HTTP 500. Since handled exceptions should result in a BusBuddyException, if this happens, it is unexpected behavior and should be treated as an internal error.

Parameters

e	exception which was thrown

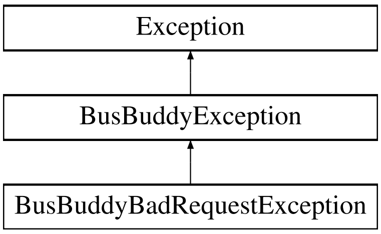
Returns

ResponseEntity object

5.22 BusBuddyBadRequestException Class Reference

This exception object represents internal errors which may occur as a result of an error in the client's request.

Inheritance diagram for BusBuddyBadRequestException:



Public Member Functions

- **BusBuddyBadRequestException** (String message)
- **BusBuddyBadRequestException** (Throwable cause)
- **BusBuddyBadRequestException** (String message, Throwable cause)
- HttpStatus getHttpCode ()

  This method returns the HTTP status code associated with this exception.

Static Private Attributes

- static final long **serialVersionUID** = -5974225882272455539L

Additional Inherited Members

5.22.1 Detailed Description

This exception object represents internal errors which may occur as a result of an error in the client's request.

5.22.2 Member Function Documentation

5.22.2.1 HttpStatus getHttpCode ( ) [virtual]

This method returns the HTTP status code associated with this exception.

Returns

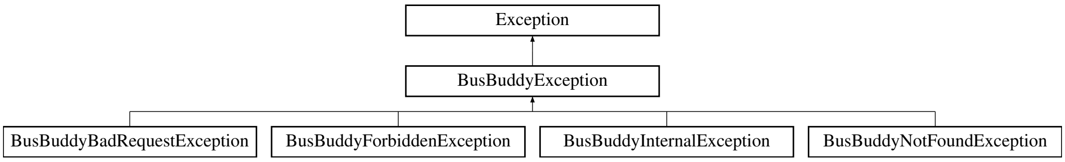
HTTP 400 Bad Request

Implements BusBuddyException.

5.23 BusBuddyException Class Reference

This exception object is an abstract base class.

Inheritance diagram for BusBuddyException:


Exception
BusBuddyException
BusBuddyBadRequestException
BusBuddyForbiddenException
BusBuddyInternalException
BusBuddyNotFoundException

Protected Member Functions

- **BusBuddyException** (String message)
- **BusBuddyException** (Throwable cause)
- **BusBuddyException** (String message, Throwable cause)
- abstract HttpStatus getHttpCode ()

  This method returns a Spring HTTP status code object representing the HTTP status code tied to this exception.

Static Private Attributes

- static final long **serialVersionUID** = 5906063726935813830L

5.23.1 Detailed Description

This exception object is an abstract base class.

Other excpetions within the BusBuddy application will extend this class. This provides a common base for all application exceptions.

5.23.2 Member Function Documentation

5.23.2.1 abstract HttpStatus getHttpCode ( ) [protected], [pure virtual]

This method returns a Spring HTTP status code object representing the HTTP status code tied to this exception.

Returns

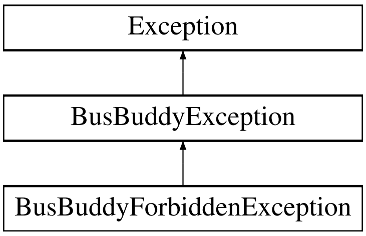
HTTP Status Code object

Implemented in BusBuddyForbiddenException, BusBuddyInternalException, BusBuddyBadRequestException, and BusBuddyNotFoundException.

5.24 BusBuddyForbiddenException Class Reference

This exception object represents internal errors which may occur as a result of attempts to access a resource without authorization.

Inheritance diagram for BusBuddyForbiddenException:



Public Member Functions

- **BusBuddyForbiddenException** (String message)
- **BusBuddyForbiddenException** (Throwable cause)
- **BusBuddyForbiddenException** (String message, Throwable cause)
- HttpStatus getHttpCode ()

  This method returns the HTTP status code associated with this exception.

Static Private Attributes

- static final long **serialVersionUID** = -4463973248172436949L

Additional Inherited Members

5.24.1 Detailed Description

This exception object represents internal errors which may occur as a result of attempts to access a resource without authorization.

5.24.2 Member Function Documentation

5.24.2.1 HttpStatus getHttpCode ( ) [virtual]

This method returns the HTTP status code associated with this exception.

Returns

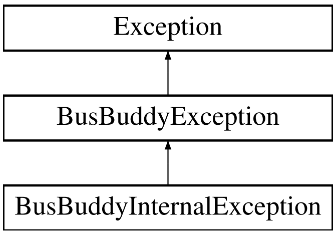
HTTP 403 Forbidden

Implements BusBuddyException.

5.25 BusBuddyInternalException Class Reference

This exception object represents internal errors which may occur, which are generally not due to the specifics of what appears to be a valid request.

Inheritance diagram for BusBuddyInternalException:



Public Member Functions

- **BusBuddyInternalException** (String message)
- **BusBuddyInternalException** (Throwable cause)
- **BusBuddyInternalException** (String message, Throwable cause)
- HttpStatus getHttpCode ()

  This method returns the HTTP status code associated with this exception.

Static Private Attributes

- static final long **serialVersionUID** = 4549592428602851924L

Additional Inherited Members

5.25.1 Detailed Description

This exception object represents internal errors which may occur, which are generally not due to the specifics of what appears to be a valid request.

5.25.2 Member Function Documentation

5.25.2.1 HttpStatus getHttpCode ( ) [virtual]

This method returns the HTTP status code associated with this exception.

Returns

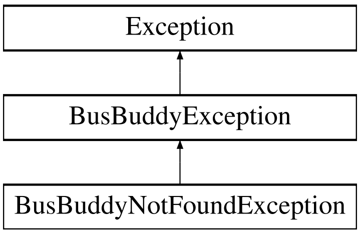
HTTP 500 Internal Server Error

Implements BusBuddyException.

5.26 BusBuddyNotFoundException Class Reference

This exception object represents the error that occurs when a resource cannot be found.

Inheritance diagram for BusBuddyNotFoundException:



Public Member Functions

- **BusBuddyNotFoundException** (String message)
- **BusBuddyNotFoundException** (Throwable cause)
- **BusBuddyNotFoundException** (String message, Throwable cause)
- HttpStatus getHttpCode ()

  This method returns the HTTP status code associated with this exception.

Static Private Attributes

- static final long **serialVersionUID** = -5490492502661128777L

Additional Inherited Members

5.26.1 Detailed Description

This exception object represents the error that occurs when a resource cannot be found.

5.26.2 Member Function Documentation

5.26.2.1 HttpStatus getHttpCode ( ) [virtual]

This method returns the HTTP status code associated with this exception.

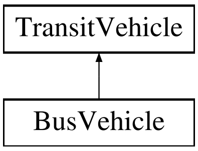
Returns

HTTP 404 Not Found

Implements BusBuddyException.

5.27 BusVehicle Class Reference

Inheritance diagram for BusVehicle:



Public Member Functions

- BusVehicle ()

  Constructor for bus type vehicles, perform any initializations unique to buses.
- void registerTrackingAlert (TrackingAlertObserver ao)

  Register any user alerts for this vehicle.
- void unregisterTrackingAlert (TrackingAlertObserver ao)

  Unregister any user alert currently tracking this bus.
- void checkForAlerts ()

  When the bus GPS position is updated, determine if any user alerts need to be sent.

Private Attributes

- ArrayList< TrackingAlertObserver > alertList

  List of alerts registered for this vehicle.

5.27.1 Member Data Documentation

5.27.1.1 ArrayList<TrackingAlertObserver> alertList [private]

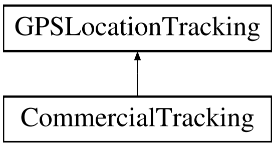
List of alerts registered for this vehicle.

Note alerts may be tracking or delay alerts

5.28 CommercialTracking Class Reference

Implements Subject Location Tracking for retrieving GPS location updates from outside commercial tracking services.

Inheritance diagram for CommercialTracking:



Classes

- class CommercialTrackingHolder

  Commercial Tracking Holder is loaded on the first execution of CommercialTracking.getInstance() or the first access to CommercialTracking.INSTANCE, not before (lazy instantiation).

Public Member Functions

- void registerGPSDevice (GPSLocationObserver gpsObs)

  Register a GPS Device to the list to poll for updates.
- void unregisterGPSDevice (GPSLocationObserver gpsObs)

  Remove a GPS device from the list currently being polled for updates.
- void pollGPSDevice ()

  Continuously poll the registered GPS devices for location updates.

Static Public Member Functions

- static CommercialTracking **getInstance** ()

Private Member Functions

- CommercialTracking ()

  Only need one Commercial Tracking Service running to track by polling all registered GPS devices.

Private Attributes

- ArrayList< GPSLocationObserver > gpsObserver

  Array list of GPS devices registered for updates.

5.28.1 Detailed Description

Implements Subject Location Tracking for retrieving GPS location updates from outside commercial tracking services.

5.28.2 Constructor & Destructor Documentation

5.28.2.1 CommercialTracking ( ) [private]

Only need one Commercial Tracking Service running to track by polling all registered GPS devices.

Constructor, creates ArrayList<GPSLocationObserver> to hold registered observers.

5.29 CommercialTracking.CommercialTrackingHolder Class Reference

Commercial Tracking Holder is loaded on the first execution of CommercialTracking.getInstance() or the first access to CommercialTracking.INSTANCE, not before (lazy instantiation).

Static Public Attributes

- static final CommercialTracking **INSTANCE** = new CommercialTracking()

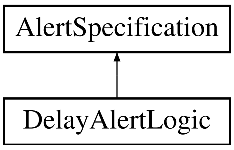
5.29.1 Detailed Description

Commercial Tracking Holder is loaded on the first execution of CommercialTracking.getInstance() or the first access to CommercialTracking.INSTANCE, not before (lazy instantiation).

5.30 DelayAlertLogic Class Reference

Implements the business logic to determine if the vehicle is behind schedule or not reporting GPS updates and send a notification to the transit company.

Inheritance diagram for DelayAlertLogic:



Public Member Functions

- boolean inAlertZone (Date lastUpdateTime, Location vehicleLocation)

  Implements Subject Location Tracking for retrieving GPS location updates from outside commercial tracking services.

5.30.1 Detailed Description

Implements the business logic to determine if the vehicle is behind schedule or not reporting GPS updates and send a notification to the transit company.

5.31 Detour Class Reference

A disruption in service due to an unexpected event.

Public Member Functions

- String **getCause** ()
- void **setCause** (String cause)
- int **getEstimatedDelay** ()
- void **setEstimatedDelay** (int estimatedDelay)
- Set< Stop > **getAffectedStops** ()
- void **setAffectedStops** (Set< Stop > affectedStops)

Private Attributes

- String cause

  A text-based description of the cause of the Detour, intended to be displayed to customers.
- int estimatedDelay

  The estimated time (in minutes) that each of the Stops in the affectedStops will be delayed.
- Set< Stop > affectedStops

  All Stops that are subject to the noted estimatedDelay.

5.31.1 Detailed Description

A disruption in service due to an unexpected event.

A Detour may not affect all Stops in a Route, as a Detour may only alter portions of the Route. Any affected Stop will be listed in the affectedStops attribute.

5.31.2 Member Data Documentation

5.31.2.1 String cause [private]

A text-based description of the cause of the Detour, intended to be displayed to customers.

If null or blank, the cause is considered Unspecified or Unknown.

5.32 Fare Class Reference

An immutable Value Object representing the cost, or 'fare,' required to ride a TransitVehicle on a particular Route.

Public Member Functions

- BigDecimal **getRegularFare** ()
- void setRegularFare (BigDecimal regularFare)
- BigDecimal **getDiscountedFare** ()
- void setDiscountedFare (BigDecimal discountedFare)

Private Attributes

- BigDecimal regularFare

  The normally applied fare.
- BigDecimal discountedFare

  A discounted fare for children, elderly, or other adjustment criteria as supplied by the TransitProvider.

5.32.1 Detailed Description

An immutable Value Object representing the cost, or 'fare,' required to ride a TransitVehicle on a particular Route.

5.32.2 Member Function Documentation

5.32.2.1 void setDiscountedFare (BigDecimal *discountedFare*)

Precondition

{regularFare} >= 0

5.32.2.2 void setRegularFare (BigDecimal *regularFare*)

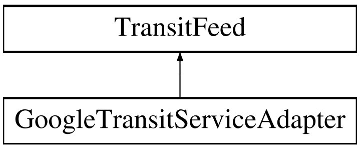
Precondition

regularFare >= 0

5.33 GoogleTransitServiceAdapter Class Reference

An Adapter Class to allow a {} service to appear as a TransitService.

Inheritance diagram for GoogleTransitServiceAdapter:



Public Member Functions

- GoogleTransitServiceAdapter (GoogleTransitServiceAPI googleTransitServiceAPI)

  Instantiates a new GoogleTransitServiceAdapter with a {} to delegate calls to.
- Route getRoute (String routeId)

  Gets a Route by its unique identifier.
- Set< Route > getRoutes (Location pickup, Location dropoff, int distance)

  *Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.*
- GoogleTransitServiceAPI **getGoogleTransitServiceAPI** ()
- void **setGoogleTransitServiceAPI** (GoogleTransitServiceAPI googleTransitServiceAPI)

Private Attributes

- GoogleTransitServiceAPI googleTransitServiceAPI

  The {} to adapt as a TransitService.

5.33.1 Detailed Description

An Adapter Class to allow a {} service to appear as a TransitService.

5.33.2 Constructor & Destructor Documentation

5.33.2.1 GoogleTransitServiceAdapter (GoogleTransitServiceAPI *googleTransitServiceAPI*)

Instantiates a new GoogleTransitServiceAdapter with a {} to delegate calls to.

Parameters

googleTransitServiceAPI	the google transit service api

5.33.3 Member Function Documentation

5.33.3.1 Route getRoute (String *routeId*)

Gets a Route by its unique identifier.

Precondition

routeId is not null or blank.

Postcondition

The Route is returned if the **routeId** is found, else null.

Parameters

routeId	The unique identifier of the Route

Returns

The matching Route, or null if not found

Implements TransitFeed.

5.33.3.2 Set<Route> getRoutes (Location *pickup*, Location *dropoff*, int *distance*)

Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.

Precondition

pickup is not null or blank.
dropoff is not null or blank.
distance is non-negative.

Parameters

pickup	The requested pickup Location
pickup	The requested dropoff Location
distance	The distance (in miles) that each Route can deviate from the requested **pickup** or **dropoff** Location. For each Route returned, neither its start or end Location can differ from the requested **pickup** or **dropoff** Location by more than the value of the **distance** parameter.

Returns

The matching Routes

Implements TransitFeed.

5.34 GoogleTransitServiceAPI Interface Reference

A client to Google's `Maps API`.

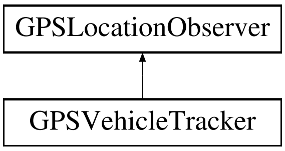
5.34.1 Detailed Description

A client to Google's `Maps API`.

5.35 GPSLocationObserver Class Reference

Observer Pattern - Observer interface for GPS location tracking.

Inheritance diagram for GPSLocationObserver:



Public Member Functions

- abstract void gpsUpdate (int gpsID, Location newLocation)

  Observer Pattern update method to update transit vehicle GPS location.
- Location **getGPSLocation** ()

Protected Member Functions

- void **setGPSLocation** (Location gpsLocation)

Protected Attributes

- GPSLocationTracking gpsDevice

  Observer Pattern Subject.
- int gpsID

  GPS Device ID being tracked.
- Location gpsLocation

  Current GPS latitude and longitude from GPS tracker.

5.35.1 Detailed Description

Observer Pattern - Observer interface for GPS location tracking.

5.35.2 Member Function Documentation

5.35.2.1 abstract void gpsUpdate (int *gpsID,* Location *newLocation*) `[pure virtual]`

Observer Pattern update method to update transit vehicle GPS location.

Parameters

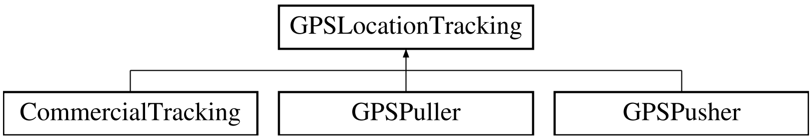
gpsID	- integer device ID from the GPS unit being tracked
latitude	- double new latitude from GPS device
longitude	- double new longitude from GPS device

Implemented in GPSVehicleTracker.

5.36 GPSLocationTracking Class Reference

GPSLocationTracking - interface Subject of the Observer Pattern Defines methods for an observer GPS Device to register and receive updates on vehicle location.

Inheritance diagram for GPSLocationTracking:



Public Member Functions

- abstract void registerGPSDevice (GPSLocationObserver gpsObs)

  registerGPSDevice - register a GPS device with the Location Tracking Service
- abstract void unregisterGPSDevice (GPSLocationObserver gpsObs)

  unregisterGPSDevice - remove a vehicle from list.
- abstract void pollGPSDevice ()

  pollGPSDevice - continuously poll registered GPS Devices for location updates

5.36.1 Detailed Description

GPSLocationTracking - interface Subject of the Observer Pattern Defines methods for an observer GPS Device to register and receive updates on vehicle location.

5.36.2 Member Function Documentation

5.36.2.1 abstract void registerGPSDevice (GPSLocationObserver *gpsObs*) [pure virtual]

registerGPSDevice - register a GPS device with the Location Tracking Service

Parameters

GPSLocationObserver	- Vehicle location to notify when new vehicle GPS location is received

Implemented in GPSPusher, CommercialTracking, and GPSPuller.

5.36.2.2 abstract void unregisterGPSDevice (GPSLocationObserver *gpsObs*) [pure virtual]

unregisterGPSDevice - remove a vehicle from list.

Stop updating vehicle location.

Parameters

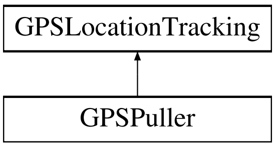
gpsObs	GPSLocationObserver - vehicle to remove from notification list

Implemented in GPSPusher, CommercialTracking, and GPSPuller.

5.37 GPSPuller Class Reference

GPS Puller is a concrete implementation of GPS Location tracker for obtaining coordinates directly from a GPS device installed in a registered vehicle.

Inheritance diagram for GPSPuller:



Classes

- class GPSPullerHolder

  GPS Puller Holder is loaded on the first execution of GPSPuller.getInstance() or the first access to GPSPuller.INSTANCE, not before (lazy instantiation).

Public Member Functions

- void registerGPSDevice (GPSLocationObserver gpsObs)

  Register a GPS Device to the list to poll for updates.
- void unregisterGPSDevice (GPSLocationObserver gpsObs)

  Remove a GPS device from the list currently being polled for updates.
- void pollGPSDevice ()

  Continuously poll the registered GPS devices for location updates.

Static Public Member Functions

- static GPSPuller **getInstance** ()

Private Member Functions

- GPSPuller ()

  Only need one GPS Puller Service running to track by polling all registered GPS devices.

Private Attributes

- ArrayList$<$ GPSLocationObserver $>$ gpsObserver

  Array list of GPS devices registered for updates.

5.37.1 Detailed Description

GPS Puller is a concrete implementation of GPS Location tracker for obtaining coordinates directly from a GPS device installed in a registered vehicle.

GPS Puller is implemented as a singleton to limit the number of system resources consumed. GPS Puller uses the system infrastructure to establish a wireless network connection to the physical GPS device and retrieve update coordinates. The necessary information to contact the device is provided through the user interface when a vehicle is registered to a route.

5.37.2 Constructor & Destructor Documentation

5.37.2.1 GPSPuller ( ) [private]

Only need one GPS Puller Service running to track by polling all registered GPS devices.

Constructor, creates ArrayList<GPSLocationObserver> to hold registered observers.

5.38 GPSPuller.GPSPullerHolder Class Reference

GPS Puller Holder is loaded on the first execution of GPSPuller.getInstance() or the first access to GPSPuller.INSTANCE, not before (lazy instantiation).

Static Public Attributes

- static final GPSPuller **INSTANCE** = new GPSPuller()

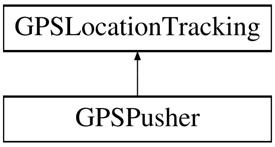
5.38.1 Detailed Description

GPS Puller Holder is loaded on the first execution of GPSPuller.getInstance() or the first access to GPSPuller.INSTANCE, not before (lazy instantiation).

5.39 GPSPusher Class Reference

Implements Subject Location Tracking for retrieving GPS location updates from registered vehicles.

Inheritance diagram for GPSPusher:



Classes

- class GPSPusherHolder

  GPS Pusher Holder is loaded on the first execution of GPSPusher.getInstance() or the first access to GPSPusher.INSTANCE, not before (lazy instantiation).

Public Member Functions

- void registerGPSDevice (GPSLocationObserver gpsObs)

  Register a GPS Device to the list to poll for updates.
- void unregisterGPSDevice (GPSLocationObserver gpsObs)

  Remove a GPS device from the list currently being polled for updates.
- void pollGPSDevice ()

  Continuously poll the registered GPS devices for location updates.

Static Public Member Functions

- static GPSPusher getInstance ()

    Create a single instance of the GPS Listener for receiving GPS updates from devices that periodically push updated directly from the device.

Private Member Functions

- GPSPusher ()

    Only need one GPS Pusher Service running to track by polling all registered GPS devices.

Private Attributes

- ArrayList< GPSLocationObserver > gpsObserver

    Array list of GPS devices registered for updates.

5.39.1 Detailed Description

Implements Subject Location Tracking for retrieving GPS location updates from registered vehicles.

GPSPusher uses system infrastructure resources to set up a network listener to receive updates directly from the GPS device. GPS Pusher is implemented as a singleton to limit the number of system resources consumed. GPS Puser receives the necessary configuration information (e.g. port) from the user interface when the GPS device is registered.

5.39.2 Constructor & Destructor Documentation

5.39.2.1 GPSPusher ( ) `[private]`

Only need one GPS Pusher Service running to track by polling all registered GPS devices.

Constructor, creates ArrayList<GPSLocationObserver> to hold registered observers. < List of GPS devices currently registered and waiting for updates

5.39.3 Member Function Documentation

5.39.3.1 static GPSPusher getInstance ( ) `[static]`

Create a single instance of the GPS Listener for receiving GPS updates from devices that periodically push updated directly from the device.

Returns

GPSPusher reference to the listener for incoming GPS updates from registered devices.

5.40 GPSPusher.GPSPusherHolder Class Reference

GPS Pusher Holder is loaded on the first execution of GPSPusher.getInstance() or the first access to GPSPusher.-INSTANCE, not before (lazy instantiation).

Static Public Attributes

- static final GPSPusher **INSTANCE** = new GPSPusher()

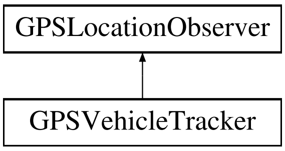
5.40.1 Detailed Description

GPS Pusher Holder is loaded on the first execution of GPSPusher.getInstance() or the first access to GPSPusher.-INSTANCE, not before (lazy instantiation).

5.41 GPSVehicleTracker Class Reference

Implementation of the Observer, update the transit vehicle GPS location.

Inheritance diagram for GPSVehicleTracker:



Public Member Functions

- GPSVehicleTracker (GPSLocationTracking gpsDevice)

  Register the Transit Vehicle GPS device with GPS location tracking.
- void gpsUpdate (int gpsID, Location newLocation)

  Notify method to get the new GPS coordinates from GPS location tracking.

Additional Inherited Members

5.41.1 Detailed Description

Implementation of the Observer, update the transit vehicle GPS location.

5.41.2 Constructor & Destructor Documentation

5.41.2.1 GPSVehicleTracker (GPSLocationTracking *gpsDevice*)

Register the Transit Vehicle GPS device with GPS location tracking.

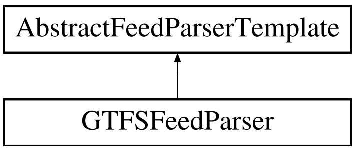
Parameters

gpsDevice	- GPSLocationTracking Subject being observed

5.42 GTFSFeedParser Class Reference

A AbstractFeedParserTemplate implementation designed to parse `GTFS` format ZIP files into Routes.

Inheritance diagram for GTFSFeedParser:



Protected Member Functions

- Set< Route > parseFeed (InputStream feed)

  Parse the `GTFS` *format ZIP files into Routes.*

Additional Inherited Members

5.42.1 Detailed Description

A AbstractFeedParserTemplate implementation designed to parse `GTFS` format ZIP files into Routes.

5.42.2 Member Function Documentation

5.42.2.1 Set<Route> parseFeed (InputStream *feed*) [protected], [virtual]

Parse the `GTFS` format ZIP files into Routes.

See Also

AbstractFeedParserTemplate::parseFeed(InputStream)

Implements AbstractFeedParserTemplate.

5.43 HashUtility Class Reference

This is a utility class to handle secure hashes.

Static Public Member Functions

- static String hash (String input)

  This is a method that will take an input string, securely hash it, and return the hashed String using the SHA-512 algorithm.

5.43.1 Detailed Description

This is a utility class to handle secure hashes.

5.43.2 Member Function Documentation

5.43.2.1 static String hash (String *input*) [static]

This is a method that will take an input string, securely hash it, and return the hashed String using the SHA-512 algorithm.

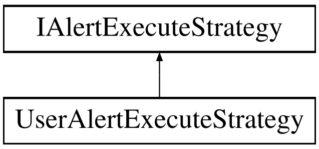
Parameters

input	

Returns

5.44 IAlertExecuteStrategy Interface Reference

Inheritance diagram for IAlertExecuteStrategy:



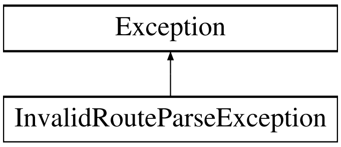
Public Member Functions

- boolean **execute** ()

5.45 InvalidRouteParseException Class Reference

An InvalidRouteParseException indicates an invalid batch of parsed Routes has been been detected.

Inheritance diagram for InvalidRouteParseException:



Public Member Functions

- Set< Route > **getRouteBatch** ()
- void **setRouteBatch** (Set< Route > routeBatch)

Protected Member Functions

- InvalidRouteParseException (Set< Route > routeBatch)

  Instantiates a new invalid route parse exception.

Private Attributes

- Set< Route > routeBatch

  The failed Route batch.

Static Private Attributes

- static final long serialVersionUID = -4399874766965916500L

  The Constant serialVersionUID.

5.45.1 Detailed Description

An InvalidRouteParseException indicates an invalid batch of parsed Routes has been been detected.

Note that one ore more of the referenced Routes are invalid, but not necessarily all of them are invalid.

5.45.2 Constructor & Destructor Documentation

5.45.2.1 InvalidRouteParseException (Set< Route > *routeBatch*) [protected]

Instantiates a new invalid route parse exception.

Parameters

routeBatch	the route batch

5.45.3 Member Data Documentation

5.45.3.1 Set<Route> routeBatch [private]

The failed Route batch.

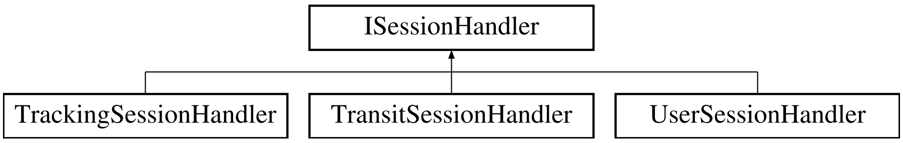
Handlers of this Exception may choose to re-validate, fix, and/or retry the operation with an adjusted batch.

5.45.3.2 final long serialVersionUID = -4399874766965916500L [static], [private]

The Constant serialVersionUID.

5.46 ISessionHandler Interface Reference

Inheritance diagram for ISessionHandler:



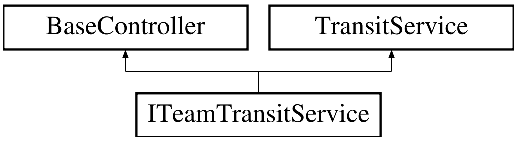
Public Member Functions

- boolean **verifySessionToken** (String sessionToken)

5.47 ITeamTransitService Class Reference

The iTeam implementation of the TransitService that exposes Transit data via a REST Service.

Inheritance diagram for ITeamTransitService:



Public Member Functions

- Route getRoute (String routeId)

  Gets a Route by its unique identifier.
- Set< Route > getRoutes (Location pickup, Location dropoff, int distance)

  *Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.*
- TransitInfo getTransitInfo ()

  Gets metadata about the Transit Authority providing the information retrieved from this service.
- URL getServiceURL ()

  The URL that uniquely identifies this TransitService.
- TransitFeed **getTransitFeed** ()
- void **setTransitFeed** (TransitFeed transitFeed)

Private Attributes

- TransitFeed transitFeed

  The TransitFeed used to provide data to this TransitService implementation.

5.47.1 Detailed Description

The iTeam implementation of the TransitService that exposes Transit data via a REST Service.

5.47.2 Member Function Documentation

5.47.2.1 Route getRoute (String *routeId*)

Gets a Route by its unique identifier.

Precondition

> **routeId** is not null or blank.

Postcondition

> The Route is returned if the **routeId** is found, else null.

Parameters

routeId	The unique identifier of the Route

Returns

> The matching Route, or null if not found

Implements TransitService.

5.47.2.2 Set<Route> getRoutes (Location *pickup*, Location *dropoff*, int *distance*)

Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.

Precondition

> **pickup** is not null or blank.
> **dropoff** is not null or blank.
> **distance** is non-negative.

Parameters

pickup	The requested pickup Location
pickup	The requested dropoff Location
distance	The distance (in miles) that each Route can deviate from the requested **pickup** or **dropoff** Location. For each Route returned, neither its start or end Location can differ from the requested **pickup** or **dropoff** Location by more than the value of the **distance** parameter.

Returns

The matching Routes

Implements TransitService.

5.47.2.3 URL getServiceURL ( )

The URL that uniquely identifies this TransitService.

In a REST environment, this might be the root of the REST API path. In a SOAP environment, it could represent a SOAP endpoint.

Returns

The URL of this service

Implements TransitService.

5.47.2.4 TransitInfo getTransitInfo ( )

Gets metadata about the Transit Authority providing the information retrieved from this service.

Returns

The TransitInfo of the Transit Authority of this service.

Implements TransitService.

5.47.3 Member Data Documentation

5.47.3.1 TransitFeed transitFeed [private]

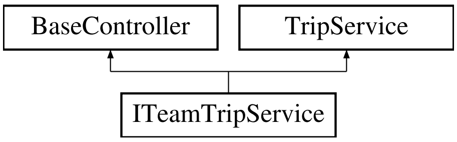
The TransitFeed used to provide data to this TransitService implementation.

Note that this TransitFeed implementation may be aggregate of many TransitFeeds, an Adapter to another API, or other implementation.

5.48 ITeamTripService Class Reference

An iTeam implementation of the TripService that exposes Trip data via a REST Service.

Inheritance diagram for ITeamTripService:



Public Member Functions

- Trip calculateTrip (Location start, Location end)

  *Calculate an optimal Trip given a **start** Location and an **end** Location.*
- TransitService **getTransitService** ()
- void **setTransitService** (TransitService transitService)

Private Attributes

- TransitService transitService

  The TransitService used to provide the Route data used in the Trip calculations.

5.48.1 Detailed Description

An iTeam implementation of the TripService that exposes Trip data via a REST Service.

Note: The actual Trip calculation algorithm is not specified here and is beyond the scope of this project.

5.48.2 Member Function Documentation

5.48.2.1 Trip calculateTrip (Location *start,* Location *end*)

Calculate an optimal Trip given a **start** Location and an **end** Location.

Parameters

start	The requested start Location of the Trip.
end	The requested end Location of the Trip.

Returns

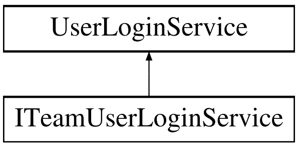
The calculated Trip

Implements TripService.

5.49 ITeamUserLoginService Class Reference

This is the iTeam's implementation of UserLoginService.

Inheritance diagram for ITeamUserLoginService:



Public Member Functions

- String login (String username, String password) throws BusBuddyException
- void logout (String sessionToken) throws BusBuddyException
- User getUser (String sessionToken) throws BusBuddyException
- String createAlertSession (String sessionToken) throws BusBuddyException
- void sendUsername (String email) throws BusBuddyException

- void sendUsername (short countryCode, String mobile) throws BusBuddyException
- void **resetPassword** (String username, String email)
- void **resetPassword** (String username, short countryCode, String mobile)

Protected Attributes

- SessionRepository **sessionRepository**
- UserRepository **userRepository**

5.49.1 Detailed Description

This is the iTeam's implementation of UserLoginService.

5.49.2 Member Function Documentation

5.49.2.1 String createAlertSession (String *sessionToken*) throws **BusBuddyException**

See Also

UserLoginService.createAlertSession

Implements UserLoginService.

5.49.2.2 User getUser (String *sessionToken*) throws **BusBuddyException**

See Also

UserLoginService.getUser

Implements UserLoginService.

5.49.2.3 String login (String *username,* String *password*) throws **BusBuddyException**

See Also

UserLoginService.login

Implements UserLoginService.

5.49.2.4 void logout (String *sessionToken*) throws **BusBuddyException**

See Also

UserLoginService.logout

Implements UserLoginService.

5.49.2.5 void sendUsername (String *email*) throws **BusBuddyException**

See Also

UserLoginService.sendUsername(String)

Implements UserLoginService.

5.49.2.6 void sendUsername (short *countryCode,* String *mobile*) throws **BusBuddyException**

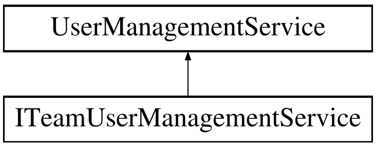
See Also

UserLoginService.sendUsername(short, String)

Implements UserLoginService.

5.50 ITeamUserManagementService Class Reference

Inheritance diagram for ITeamUserManagementService:



Public Member Functions

- User **createUser** (User userToCreate, String password)
- User **findUserByUsername** (String sessionToken, String username)
- User **findUserByEmail** (String sessionToken, String email)
- User **findUserByMobile** (String sessionToken, short countryCode, String mobile)
- void **updateUser** (String sessionToken, User newUserData, String password)
- void **deleteUser** (String sessionToken, User userToDelete)

Protected Member Functions

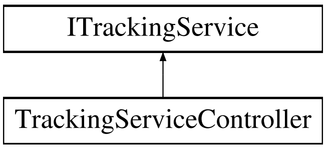
- boolean **checkPermissions** (String sessionToken)

Protected Attributes

- UserRepository **userRepository**

5.51 ITrackingService Interface Reference

Inheritance diagram for ITrackingService:



Classes

- enum AlertType

Public Member Functions

- void **registerVehicleOnRoute** (URL url, int gpsDeviceID)
- void **unregisterVehicleFromRoute** (String url, int gpsDeviceID)
- void **addUserTrackingAlert** (UserTrackingAlertObject utao)
- void **startTrackingController** ()
- Location **getTransitVehicleLocation** (int gpsDeviceID)

5.52 Location Class Reference

An immutable Value Object representing a physical point on the geographic coordinate system.

Public Member Functions

- Location (double latitude, double longitude)

  Instantiates a new immutable Location with the given latitude and longitude.
- double **getLatitude** ()
- double **getLongitude** ()

Private Attributes

- double latitude

  The latitude of the point.
- double longitude

  The longitude of the point.

5.52.1 Detailed Description

An immutable Value Object representing a physical point on the geographic coordinate system.

5.52.2 Constructor & Destructor Documentation

5.52.2.1 Location (double *latitude,* double *longitude*)

Instantiates a new immutable Location with the given latitude and longitude.

Parameters

latitude	The point latitude
longitude	The point longitude

5.53 MessageDeliveryUtility Class Reference

This is a utility class to handle message delivery, such as through email or SMS.

Static Public Member Functions

- static void sendEmail (String to, String from, String subject, String htmlBody) throws BusBuddyInternalException

  This method sends an HTML e-mail.
- static void sendSms (short countryCode, String mobileNumber, String message)

  This method sends an SMS text message.

5.53.1 Detailed Description

This is a utility class to handle message delivery, such as through email or SMS.

5.53.2 Member Function Documentation

5.53.2.1 static void sendEmail (String *to,* String *from,* String *subject,* String *htmlBody*) throws BusBuddyInternalException `[static]`

This method sends an HTML e-mail.

Parameters

to	recipient address
from	sender address
subject	subject line
htmlBody	HTML body of the message

Exceptions

BusBuddyInternalException	This exception is thrown if there is an error sending the e-mail.

5.53.2.2 static void sendSms (short *countryCode,* String *mobileNumber,* String *message*) [static]

This method sends an SMS text message.

Precondition

The mobile number must be a String consisting entirely of digits.

Parameters

countryCode	country code for the recipient
mobileNumber	mobile number to send to
message	body of the message to send

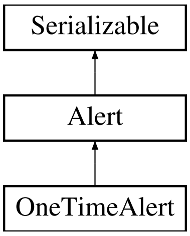
Exceptions

BusBuddyInternalException	This exception is thrown if there is an error sending the message.

5.54 OneTimeAlert Class Reference

This is a model of alert that is to be run one time only.

Inheritance diagram for OneTimeAlert:



Public Member Functions

- Date getDateExecuted ()
- void setDateExecuted (Date dateExecuted)

Private Attributes

- Date dateExecuted

  Date when it was executed.

Static Private Attributes

- static final long **serialVersionUID** = 8851691556082123516L

5.54.1 Detailed Description

This is a model of alert that is to be run one time only.

This can be configure by User (e.g., catch bus to Boston at 9am on MM/DD/YYYY) or by any other module (e.g., Route to MSP downtown on MM/DD/YYYY is going to be rerouted). This class extends the Alert.

5.54.2 Member Function Documentation

5.54.2.1 Date getDateExecuted ( )

Returns

the dateExecuted

5.54.2.2 void setDateExecuted (Date *dateExecuted*)

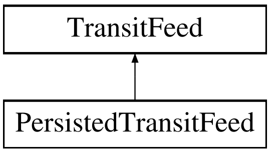
Parameters

dateExecuted	the dateExecuted to set

5.55 PersistedTransitFeed Class Reference

An implementation of the TransitFeed interface that communicates with a RouteRepository to retrieve its data.

Inheritance diagram for PersistedTransitFeed:



Public Member Functions

- Route getRoute (String routeId)

  Gets a Route by its unique identifier.
- Set< Route > getRoutes (Location pickup, Location dropoff, int distance)

  *Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.*
- RouteRepository **getRouteRepository** ()
- void **setRouteRepository** (RouteRepository routeRepository)

Private Attributes

- RouteRepository routeRepository

  The RouteRepository responsible for providing data.

5.55.1 Detailed Description

An implementation of the TransitFeed interface that communicates with a RouteRepository to retrieve its data.

This implementation is appropriate when a retrieving data from a TransitProvider that does not already supply an external API that can be used at runtime. If the data needs to be parsed and imported into a RouteRepository, this implementation will expose that persisted data as a TransitFeed.

5.55.2 Member Function Documentation

5.55.2.1 Route getRoute (String *routeId*)

Gets a Route by its unique identifier.

Precondition

routeId is not null or blank.

Postcondition

The Route is returned if the **routeId** is found, else null.

Parameters

routeId	The unique identifier of the Route

Returns

The matching Route, or null if not found

Implements TransitFeed.

5.55.2.2 Set<Route> getRoutes (Location *pickup,* Location *dropoff,* int *distance*)

Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.

Precondition

pickup is not null or blank.
dropoff is not null or blank.
distance is non-negative.

Parameters

pickup	The requested pickup Location
pickup	The requested dropoff Location
distance	The distance (in miles) that each Route can deviate from the requested **pickup** or **dropoff** Location. For each Route returned, neither its start or end Location can differ from the requested **pickup** or **dropoff** Location by more than the value of the **distance** parameter.

Returns

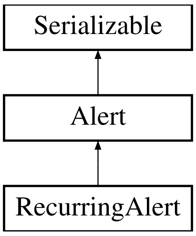
The matching Routes

Implements TransitFeed.

5.56 RecurringAlert Class Reference

This is a model of alert that is to be run multiple times.

Inheritance diagram for RecurringAlert:



Public Member Functions

- Date getSuspendDateTime ()
- void setSuspendDateTime (Date suspendDateTime)
- Date getResumeDateTime ()
- void setResumeDateTime (Date resumeDateTime)
- Date getLastSuccessfullyRanOnDateTime ()
- void setLastSuccessfullyRanOnDateTime (Date lastSuccessfullyRanOnDateTime)
- List< RecurringData > getRecurringData ()
- void setRecurringData (List< RecurringData > recurringData)
- int getRepeatEvery ()
- void setRepeatEvery (int repeatEvery)
- AlertRecurringType getAlertRecurringType ()
- void setAlertRecurringType (AlertRecurringType alertRecurringType)

Static Public Member Functions

- static long getSerialversionuid ()

Private Attributes

- Date suspendDateTime

  DateTime when the alert is to be suspended temporarily.
- Date resumeDateTime

  DateTime when the alert is to be resumed.
- Date lastSuccessfullyRanOnDateTime

  DateTime of last successful run.
- List< RecurringData > recurringData

  List of RecurringData that holds the information about when the alert should actually run.
- int repeatEvery

  Parameter to signify the skip count.
- AlertRecurringType alertRecurringType

  Type of recurring alert.

Static Private Attributes

- static final long **serialVersionUID** = -475174398668611743L

5.56.1 Detailed Description

This is a model of alert that is to be run multiple times.

Depending on User or other modules, the alert will run yearly, monthly, daily in specified hour and minute.

5.56.2 Member Function Documentation

5.56.2.1 AlertRecurringType getAlertRecurringType ( )

Returns

the alertRecurringType

5.56.2.2 Date getLastSuccessfullyRanOnDateTime ( )

Returns

the lastSuccessfullyRanOnDateTime

5.56.2.3 List< RecurringData > getRecurringData ( )

Returns

the recurringData

5.56.2.4 int getRepeatEvery ( )

Returns

the repeatEvery

5.56.2.5 Date getResumeDateTime ( )

Returns

the resumeDateTime

5.56.2.6 static long getSerialversionuid ( ) [static]

Returns

the serialversionuid

5.56.2.7 Date getSuspendDateTime ( )

Returns

the suspendDateTime

5.56.2.8 void setAlertRecurringType (AlertRecurringType *alertRecurringType*)

Parameters

alertRecurringType	the alertRecurringType to set

5.56.2.9 void setLastSuccessfullyRanOnDateTime (Date *lastSuccessfullyRanOnDateTime*)

Parameters

lastSuccessfullyRanOnDateTime	the lastSuccessfullyRanOnDateTime to set

5.56.2.10 void setRecurringData (List< RecurringData > *recurringData*)

Parameters

recurringData	the recurringData to set

5.56.2.11 void setRepeatEvery (int *repeatEvery*)

Parameters

repeatEvery	the repeatEvery to set

5.56.2.12 void setResumeDateTime (Date *resumeDateTime*)

Parameters

resumeDateTime	the resumeDateTime to set

5.56.2.13 void setSuspendDateTime (Date *suspendDateTime*)

Parameters

suspendDateTime	the suspendDateTime to set

5.56.3 Member Data Documentation

5.56.3.1 AlertRecurringType alertRecurringType [private]

Type of recurring alert.

Value is as defined in AlertRecurringType

5.56.3.2 int repeatEvery [private]

Parameter to signify the skip count.

Valid value is >0 If alert is to occur every Monday and the repeatEvery is set to 2, then it will repeat once every 2 week.

5.57 RecurringData Class Reference

Public Member Functions

- int getDayOfYear ()
- void setDayOfYear (int dayOfYear)
- int getDayOfMonth ()
- void setDayOfMonth (int dayOfMonth)
- int getDayOfWeek ()
- void setDayOfWeek (int dayOfWeek)
- int getStartMinute ()
- void setStartMinute (int startMinute)
- int getStartHour ()
- void setStartHour (int startHour)

Private Attributes

- int dayOfYear

  Day of year that the alert should run.

- int dayOfMonth

  Day of month that the alert should run.
- int dayOfWeek

  Day of week that the alert should run.
- int startHour

  The exact hour when the alert should run.
- int startMinute

  The exact minute when the alert should run.

5.57.1 Member Function Documentation

5.57.1.1 int getDayOfMonth ( )

Returns

the dayOfMonth

5.57.1.2 int getDayOfWeek ( )

Returns

the dayOfWeek

5.57.1.3 int getDayOfYear ( )

Returns

the dayOfYear

5.57.1.4 int getStartHour ( )

Returns

the startHour

5.57.1.5 int getStartMinute ( )

Returns

the startMinute

5.57.1.6 void setDayOfMonth (int *dayOfMonth*)

Parameters

dayOfMonth	the dayOfMonth to set

5.57.1.7 void setDayOfWeek (int *dayOfWeek*)

Parameters

dayOfWeek	the dayOfWeek to set

5.57.1.8 void setDayOfYear (int *dayOfYear*)

Parameters

dayOfYear	the dayOfYear to set

5.57.1.9 void setStartHour (int *startHour*)

Parameters

startHour	the startHour to set

5.57.1.10 void setStartMinute (int *startMinute*)

Parameters

startMinute	the startMinute to set

5.57.2 Member Data Documentation

5.57.2.1 int dayOfMonth [private]

Day of month that the alert should run.

Valid value is from 1-28.

5.57.2.2 int dayOfWeek [private]

Day of week that the alert should run.

e.g., 1 = Sunday and 7 = Saturday.

5.57.2.3 int dayOfYear [private]

Day of year that the alert should run.

Valid value = 1-365

5.57.2.4 int startHour [private]

The exact hour when the alert should run.

Valid value is from 0 - 23

5.57.2.5 int startMinute [private]

The exact minute when the alert should run.

Valid value is from 0-59.

5.58 Route Class Reference

A Route is a TransitVehicle path of travel, or a "Line," as referred to by a TransitProvider.

Public Member Functions

- List< Stop > **getStops** ()
- void **setStops** (List< Stop > stops)
- String **getRouteId** ()
- void **setRouteId** (String routeId)
- String **getRouteName** ()
- void **setRouteName** (String routeName)
- Set< Detour > **getDetours** ()

- void **setDetours** (Set< Detour > detours)

Private Attributes

- String routeId

  A unique identifier for this Route.
- String routeName

  Text to display in maps and other literature to denote this Route.
- List< Stop > stops

  And ordered list of Stops to be visited in this Route.
- Set< Detour > detours

  A set of Detours, or disruptions in Route availability and/or Stop schedule.

5.58.1 Detailed Description

A Route is a TransitVehicle path of travel, or a "Line," as referred to by a TransitProvider.

A Route can be thought of as an ordered list of Stops.

Note that Routes may add/remove stops, change Stop times, or be disrupted by Detours, while still remaining the same Route.

5.58.2 Member Data Documentation

5.58.2.1 Set<Detour> detours `[private]`

A set of Detours, or disruptions in Route availability and/or Stop schedule.

These Detours represent disruptions that are current at the time of retrieval of this Route.

5.58.2.2 String routeName `[private]`

Text to display in maps and other literature to denote this Route.

Uniqueness is not enforced, but this name should provide enough context to allow users to distinguish this Route.

5.58.2.3 List<Stop> stops `[private]`

And ordered list of Stops to be visited in this Route.

Stops must be visited in order unless there is a disruption in service, in which case clients can expect a Route-DistruptionAlert and/or an Detour.

5.59 RouteDisruptionAlert Class Reference

An Alert indicating a disruption of normal Route availability or scheduling.

Public Member Functions

- URL **getTransitServiceUrl** ()
- void **setTransitServiceUrl** (URL transitServiceUrl)
- String **getRouteId** ()
- void **setRouteId** (String routeId)

Private Attributes

- URL transitServiceUrl

  The URL callback of the originating TransitService.
- String routeId

  The unique identifier of the affected Route.

5.59.1 Detailed Description

An Alert indicating a disruption of normal Route availability or scheduling.

Clients interested in more specific information about the disruption, including cause and affected Stops, should use the getTransitServiceUrl() method to establish a link to the appropriate TransitService, and then obtain the affected Route using the routeId from the getRouteId() method.

Once retrieved, current Detour information can be accessed via the Route#getDetours() method on the given Route. This method, upon subsequent retrievals of the Route, will return an empty set when all Detours have cleared.

5.59.2 Member Data Documentation

5.59.2.1 String routeId `[private]`

The unique identifier of the affected Route.

This can be used in the TransitService method TransitService#getRoute(String) to retrieve more information about the disruption.

5.59.2.2 URL transitServiceUrl `[private]`

The URL callback of the originating TransitService.

Clients should use this URL to obtain further distruption information, such as Detours of the affected Route.

5.60 RouteRepository Interface Reference

A Repository Pattern supporting lifecycle operations of Routes, such as Read, Save, Delete, and Query functionality.

Public Member Functions

- void delete (String routeId)

  *Deletes the Route corresponding to the given **routeId**.*
- void save (Route route)

  Saves the Route to the Repository.
- void save (Set< Route > routes)

  Saves all of the Routes to the Repository.
- Route read (String routeId)

  Read a single Route from the Repository by its identifier.
- Collection< Route > getAll ()

  Retrieves all available Routes in the Repository.

5.60.1 Detailed Description

A Repository Pattern supporting lifecycle operations of Routes, such as Read, Save, Delete, and Query functionality.

5.60.2 Member Function Documentation

5.60.2.1 void delete (String *routeId*)

Deletes the Route corresponding to the given **routeId**.

Precondition

A Route with the given **routeId** exists in the Repository.

Postcondition

A Route with the given **routeId** is removed from the Repository and is no longer available for retrieval.

Parameters

routeId	

5.60.2.2 Collection<**Route**> getAll ( )

Retrieves all available Routes in the Repository.

Returns

All available Routes.

5.60.2.3 **Route** read (String *routeId*)

Read a single Route from the Repository by its identifier.

If no Route is found with the requested **routeId**, a null value is returned.

Parameters

routeId	The identifier of the requested Route

Returns

The requested Route

5.60.2.4 void save (**Route** *route*)

Saves the Route to the Repository.

Precondition

The Route has been validated with all appropriate business rules.

See Also

RouteSpecification

Postcondition

The Route is available for retrieval by id and also by appropriate Queries.

Parameters

route	The Route to save.

5.60.2.5 void save (Set< Route > *routes*)

Saves all of the Routes to the Repository.

Precondition

The Routes have been validated with all appropriate business rules.

See Also

RouteSpecification

Postcondition

The Routes are available for retrieval by id and also by appropriate Queries.

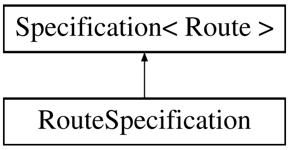
Parameters

route	The Route to save.

5.61 RouteSpecification Class Reference

A Specification Pattern class for validating a Route.

Inheritance diagram for RouteSpecification:



Public Member Functions

- boolean isSatisfiedBy (Route candidate)

  Validates the given Route with the appropriate validation rules of this Specification.
- Specification< Route > **or** (Specification< Route > specification)
- Specification< Route > **and** (Specification< Route > specification)
- Specification< Route > **not** (Specification< Route > specification)

5.61.1 Detailed Description

A Specification Pattern class for validating a Route.

Validation can happen in this class, or business rules can be combined using the and(Specification), or #not(-Specification) methods and separate Specifications. A Route

This Specification is necessary because Routes aren't necessarily created by a controlled Factory, but as a result of parsing of input files via subclasses of the AbstractFeedParserTemplate. Because creation of Routes isn't centralized, it is necessary to centralize the validation rules so that subclasses of AbstractFeedParserTemplate have access to it.

This Specification must validate the following conditions:

1. Each Route has an non-null routeId
2. Each Route has an non-blank routeName
3. Each Route has one or more Stops

5.61.2 Member Function Documentation

5.61.2.1 boolean isSatisfiedBy (Route *candidate*)

Validates the given Route with the appropriate validation rules of this Specification.

Parameters

candidate	The Route to be validated

Returns

True if a valid Route, False if not

5.62 Session Class Reference

This class represents a single session for a user of the system, and all of the state data associated with that session.

Protected Member Functions

- String getSessionToken ()
  This retrieves the session token.
- Calendar getCreationTime ()
  This retrieves the time that the session was created.
- Calendar getExpirationTime ()
  This retrieves the time that the session is set to expire.
- void setExpirationTime (Calendar expirationTime)
  This sets the time that the session should expire.
- boolean isAlertSession ()
  This checks to see if the session is an alert session.
- boolean isValid ()
  This checks to see if the session is valid.
- void setValid (boolean valid)
  This sets whether the session is valid.
- int getUserId ()
  This gets the ID of the user linked to this session.

Package Functions

- Session (String sessionToken, Calendar creationTime, boolean isAlertSession, int userId)
  Create a new session object.

Private Attributes

- final String **sessionToken**
- final Calendar **creationTime**
- Calendar **expirationTime**
- final boolean **isAlertSession**
- boolean **valid**
- final int **userId**

5.62.1 Detailed Description

This class represents a single session for a user of the system, and all of the state data associated with that session.

The session should already exist in the database before instantiating this object. A session grants a user access to the data associated with that user. Sessions expire after a certain point, and can also be invalidated by a user logging out. Some sessions are designed to be longer lasting, for use with alerts. This object is not visible to clients - when they must pass a session reference, they do so by passing around the sessionToken.

5.62.2 Constructor & Destructor Documentation

5.62.2.1 Session (String *sessionToken*, Calendar *creationTime*, boolean *isAlertSession*, int *userId*) `[package]`

Create a new session object.

It is not visible to clients, as User objects should only be constructed through the UserRepository. The parameters taken by the constructor cannot be changed once the session is created.

Parameters

sessionToken	unique session token
creationTime	time that the session was created
isAlertSession	true if this is an alert session, false otherwise
userId	user ID that the session is linked to

5.62.3 Member Function Documentation

5.62.3.1 Calendar getCreationTime ( ) `[protected]`

This retrieves the time that the session was created.

Returns

session creation time

5.62.3.2 Calendar getExpirationTime ( ) `[protected]`

This retrieves the time that the session is set to expire.

Returns

session expiration time

5.62.3.3 String getSessionToken ( ) `[protected]`

This retrieves the session token.

Returns

session token

5.62.3.4 int getUserId ( ) [protected]

This gets the ID of the user linked to this session.

Returns

user's ID number

5.62.3.5 boolean isAlertSession ( ) [protected]

This checks to see if the session is an alert session.

Returns

true if it is, false otherwise

5.62.3.6 boolean isValid ( ) [protected]

This checks to see if the session is valid.

Returns

true if it is, false otherwise

5.62.3.7 void setExpirationTime (Calendar *expirationTime*) [protected]

This sets the time that the session should expire.

Parameters

expirationTime	expiration time to set

5.62.3.8 void setValid (boolean *valid*) [protected]

This sets whether the session is valid.

Parameters

valid	true if it is, false otherwise

5.63 SessionRepository Class Reference

This class is responsible for handling database access for Sessions, and to construct, persist, and retrieve Session objects.

Package Functions

- Session createSession (User user, boolean isAlertSession) throws BusBuddyInternalException

  This creates a new session for the given user.
- Session getSession (String sessionToken) throws BusBuddyInternalException, BusBuddyForbiddenException

  This method gets a session from the database.
- void killSession (String sessionToken) throws BusBuddyInternalException, BusBuddyNotFoundException

  This method invalidates a session in the database.
- void **killAllSessions** (String userId)

5.63.1 Detailed Description

This class is responsible for handling database access for Sessions, and to construct, persist, and retrieve Session objects.

5.63.2 Member Function Documentation

5.63.2.1 Session createSession (User *user*, boolean *isAlertSession*) throws BusBuddyInternalException [package]

This creates a new session for the given user.

Precondition

The User object parameter must be a valid user retrieved from the database.

Postcondition

A session is created in the database, and the object representing that session is returned.

Parameters

user	This is the user to create the session for.
isAlertSession	This is set to true if this should be a long-lived session, for an alert. Otherwise, set to false for a normal session.

Returns

The method returns the newly created Session object.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.

5.63.2.2 Session getSession (String *sessionToken*) throws BusBuddyInternalException, BusBuddyForbiddenException [package]

This method gets a session from the database.

In addition, since this method is only called when there is an it will update the expiration date on the session.

Precondition

The sessionToken parameter must be a valid session identifier in the database.

Postcondition

The session's expiration date will have been pushed back due to this activity in the session.

Parameters

sessionToken	This is the session token that identifies the session.

Returns

Session object represented by the session token that was passed in.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.
BusBuddyForbiddenException	This exception is thrown if the session token is invalid or the session is expired.

5.63.2.3 void killSession (String *sessionToken*) throws **BusBuddyInternalException, BusBuddyNotFoundException** [package]

This method invalidates a session in the database.

Precondition

The sessionToken parameter must be a valid session identifier in the database.

Postcondition

The session will be invalidated and future calls using that sessionToken will fail.

Parameters

sessionToken	This is the session token that identifies the session.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.
BusBuddyNotFoundException	This exception is thrown if the session token is invalid.

5.64 SessionVerificationFactory Class Reference

Static Public Member Functions

- static ISessionHandler **getSessionTokenVerificationStrategy** (AlertInitiator alertInitiator)

5.65 Specification< T > Interface Reference

A Generic Specification to be used for chaining business validation rules together.

Public Member Functions

- boolean isSatisfiedBy (T candidate)

  Checks if the given candidate satisfies the specification.
- Specification< T > or (Specification< T > specification)

  Returns a Specification representing the 'OR' boolean operation of the Specifications.
- Specification< T > and (Specification< T > specification)

  Returns a Specification representing the 'AND' boolean operation of the Specifications.
- Specification< T > not (Specification< T > specification)

  Returns a Specification representing the 'NOT' boolean operation of the Specifications.

5.65.1 Detailed Description

A Generic Specification to be used for chaining business validation rules together.

Parameters

<T>	The candidate Type accepted by the Specification.

5.65.2 Member Function Documentation

5.65.2.1 Specification<T> and (Specification< T > *specification*)

Returns a Specification representing the 'AND' boolean operation of the Specifications.

Parameters

specification	The Specification to apply the 'AND' operation to.

Returns

The 'AND' Specification

5.65.2.2 boolean isSatisfiedBy (T *candidate*)

Checks if the given candidate satisfies the specification.

Parameters

candidate	The candidate

Returns

true, if is satisfied by the candidate

5.65.2.3 Specification<T> not (Specification< T > *specification*)

Returns a Specification representing the 'NOT' boolean operation of the Specifications.

Parameters

specification	The Specification to apply the 'NOT' operation to.

Returns

The 'NOT' Specification

5.65.2.4 Specification<T> or (Specification< T > *specification*)

Returns a Specification representing the 'OR' boolean operation of the Specifications.

Parameters

specification	The Specification to apply the 'OR' operation to.

Returns

The 'OR' Specification

5.66 Stop Class Reference

A point on a Route in which a TransitVehicle will stop to pick up and drop off passengers.

Public Member Functions

- Set$<$ Date $>$ getStopTimes (Date begin, Date end)

  Reports the expected times in which a TransitVehicle will be at the given Stop for a given time period.
- Location **getLocation** ()
- void **setLocation** (Location location)
- String **getDescription** ()
- void **setDescription** (String description)

Private Attributes

- String **description**
- Location location

  The physical location of the Stop.

5.66.1 Detailed Description

A point on a Route in which a TransitVehicle will stop to pick up and drop off passengers.

A Stop also is responsible for providing a set of the times in which the TransitVehicle will be at the Stop.

A Stop is identified within the context of a single Route. This means that two Routes may share the same physical Stop location, but maintain different schedules.

5.66.2 Member Function Documentation

5.66.2.1 Set$<$Date$>$ getStopTimes (Date *begin,* Date *end*)

Reports the expected times in which a TransitVehicle will be at the given Stop for a given time period.

Precondition

> **begin** $<$ **end**.

Parameters

begin	The start of the reporting time period. All Stop Times returned will be on (or after) this time. If null, assume to be the current time.
end	The end of the reporting time period. All Stop Times returned will before this time.

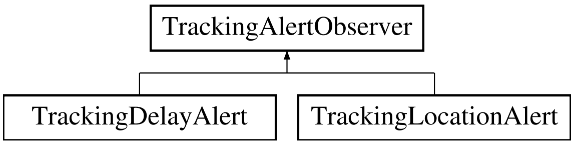
Returns

> Stop Times associated with this Stop that satisfy the begin and end criteria.

5.67 TrackingAlertObserver Class Reference

Abstract class defining the methods for the tracking alert observer.

Inheritance diagram for TrackingAlertObserver:



Public Member Functions

- AlertSpecification getSpec ()

  Return the specification to use to determine if a vehicle is in an alert zone.
- abstract void updateAlert ()

  The observer pattern update method called from the subject TransitVehicle when a vehicle is determined to be in an alert zone and a user needs to be notified.

Protected Member Functions

- void setSpec (AlertSpecification spec)

  Set the alert specification.

Private Attributes

- UserTrackingAlertObject utao

  Value Object containing the items necessary for an alert.
- AlertSpecification spec

  The business logic specification of how to determine if an alert needs to be sent for a vehicle.

5.67.1 Detailed Description

Abstract class defining the methods for the tracking alert observer.

5.67.2 Member Function Documentation

5.67.2.1 void setSpec (**AlertSpecification** *spec*) `[protected]`

Set the alert specification.

Parameters

spec	AlertSpecification - the rules used by the subject to determine if an alert is necessary.

5.67.2.2 abstract void updateAlert ( ) `[pure virtual]`

The observer pattern update method called from the subject TransitVehicle when a vehicle is determined to be in an alert zone and a user needs to be notified.

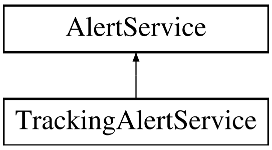
Alert notifications are actually sent using the AlertService

Implemented in TrackingLocationAlert, and TrackingDelayAlert.

5.68 TrackingAlertRequestModel Class Reference

5.69 TrackingAlertService Class Reference

Inheritance diagram for TrackingAlertService:



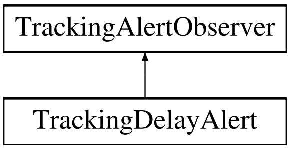
Public Member Functions

- AlertResponseModel **createAlert** (AlertRequestModel requestModel)
- AlertResponseModel **saveAlert** (Alert alertModel)
- AlertResponseModel **deleteAlert** (Alert alertModel)
- AlertResponseModel **updateAlert** (Alert alertModel)

Additional Inherited Members

5.70 TrackingDelayAlert Class Reference

Inheritance diagram for TrackingDelayAlert:



Public Member Functions

- void updateAlert ()

  Receives the notification indicating that a vehicle is in the alert zone.

Additional Inherited Members

5.70.1 Member Function Documentation

5.70.1.1 void updateAlert ( ) `[virtual]`

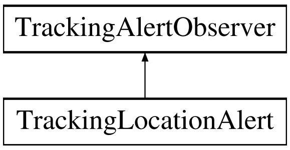
Receives the notification indicating that a vehicle is in the alert zone.

Use the AlertService to contact the registered user.

Implements TrackingAlertObserver.

5.71 TrackingLocationAlert Class Reference

Inheritance diagram for TrackingLocationAlert:



Public Member Functions

- TrackingLocationAlert (TransitVehicle vehicle)

  Tracking Location Alert constructor.
- void updateAlert ()

  Vehicle is in vicinity where user registered to be notified, call Alert Service.

Additional Inherited Members

5.71.1 Constructor & Destructor Documentation

5.71.1.1 TrackingLocationAlert (TransitVehicle *vehicle*)

Tracking Location Alert constructor.

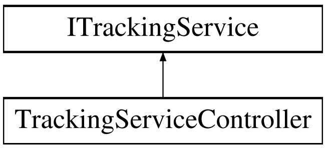
Associates user information with vehicle to monitor.

Parameters

vehicle	TransitVehicle Vehicle to add alert monitoring
alert	- TrackingAlert User contact details and vehicle location indicating when user wants to receive alert.

5.72 TrackingServiceController Class Reference

Inheritance diagram for TrackingServiceController:



Public Member Functions

- void registerVehicleOnRoute (URL url, int gpsDeviceID)

  Create a vehicle when a user registers a vehicle on a route through the user interface.
- void unregisterVehicleFromRoute (String url, int gpsDeviceID)

  Remove a vehicle from the vehicle repository when the vehicle is no longer in service.
- void addUserTrackingAlert (UserTrackingAlertObject utao)

  Add a new user alert.
- void startTrackingController ()

  Gets a list of saved alerts from the AlertService and restores then on tracking module startup.
- Location getTransitVehicleLocation (int gpsDeviceID)

  Find locations of the specified GPS device ID.

Package Attributes

- TransitVehicleFactory **transitFactory** = new TransitVehicleFactory()
- AlertFactory **alertFactory** = new AlertFactory()

5.72.1 Member Function Documentation

5.72.1.1 void addUserTrackingAlert (UserTrackingAlertObject *utao*)

Add a new user alert.

Necessary inputs are entered by the user on the User Interface and made available to the Tracking Controller through UserTrackingAlertObject

1. Verify that there is a vehicle registered on the routed requested by the user.
2. Get a list of vehicles on the route from the vehicle repository
3. Create a new Tracking Alert Observer
4. Add an alert specification containing the business rules to determine if bus is in alert zone.
5. Register the user alert observer to the vehicles

Find the vehicles registered on this route

Create an alert for this user request and register this alert with the vehicle(s) the user is watching.

Implements ITrackingService.

5.72.1.2 void unregisterVehicleFromRoute (String *url,* int *gpsDeviceID*)

Remove a vehicle from the vehicle repository when the vehicle is no longer in service.

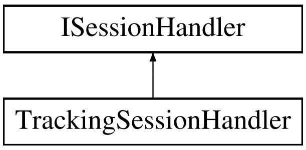
Parameters

url	- URL transit company and route information for this vehicle
gpsDeviceID	- integer GPS device ID installed in vehicle, must match the ID the vehicle registered with.

Implements ITrackingService.

5.73 TrackingSessionHandler Class Reference

Inheritance diagram for TrackingSessionHandler:



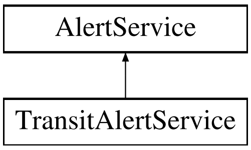
Public Member Functions

- boolean **verifySessionToken** (String sessionToken)

5.74 TransitAlertRequestModel Class Reference

5.75 TransitAlertService Class Reference

Inheritance diagram for TransitAlertService:



Public Member Functions

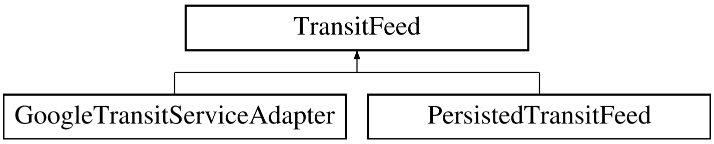
- AlertResponseModel **createAlert** (AlertRequestModel requestModel)
- AlertResponseModel **saveAlert** (Alert alertModel)
- AlertResponseModel **deleteAlert** (Alert alertModel)
- AlertResponseModel **updateAlert** (Alert alertModel)

Additional Inherited Members

5.76 TransitFeed Interface Reference

A TransitFeed is an abstraction over a service or set of services that provide information about Routes.

Inheritance diagram for TransitFeed:



Public Member Functions

- Route getRoute (String routeId)

  Gets a Route by its unique identifier.
- Set< Route > getRoutes (Location pickup, Location dropoff, int distance)

  *Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.*

5.76.1 Detailed Description

A TransitFeed is an abstraction over a service or set of services that provide information about Routes.

This differs from the TransitService interface in that a TransitFeed does not expose provenance information such as the method. Because of this, a single TransitService (or TransitProvider) could use more than one TransitFeed to expose their Transit data. For example, a TransitProvider could utilize two different database storage schemes, each one represented as a separate TransitFeed, and then aggregate the two into one TransitService.

5.76.2 Member Function Documentation

5.76.2.1 Route getRoute (String *routeId*)

Gets a Route by its unique identifier.

Precondition

routeId is not null or blank.

Postcondition

The Route is returned if the **routeId** is found, else null.

Parameters

routeId	The unique identifier of the Route

Returns

The matching Route, or null if not found

Implemented in GoogleTransitServiceAdapter, and PersistedTransitFeed.

5.76.2.2 Set<Route> getRoutes (Location *pickup,* Location *dropoff,* int *distance*)

Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.

Precondition

pickup is not null or blank.
dropoff is not null or blank.
distance is non-negative.

Parameters

pickup	The requested pickup Location
pickup	The requested dropoff Location
distance	The distance (in miles) that each Route can deviate from the requested **pickup** or **dropoff** Location. For each Route returned, neither its start or end Location can differ from the requested **pickup** or **dropoff** Location by more than the value of the **distance** parameter.

Returns

The matching Routes

Implemented in GoogleTransitServiceAdapter, and PersistedTransitFeed.

5.77 TransitInfo Class Reference

An immutable Value Object describing metadata about a TransitService.

Public Member Functions

- String **getTransitAuthorityName** ()
- void **setTransitAuthorityName** (String transitAuthorityName)
- URL **getWebsite** ()
- void **setWebsite** (URL website)
- byte[] **getLogo** ()
- void **setLogo** (byte[] logo)

Private Attributes

- String transitAuthorityName

  The name of the governing body of the associated TransitService.
- URL website

  A resolvable URL reference to the official Transit Authority web site.
- byte[] logo

  A byte array of the Transit Authority logo, if any.

5.77.1 Detailed Description

An immutable Value Object describing metadata about a TransitService.

Each TransitService is required to supply the following information.

5.77.2 Member Data Documentation

5.77.2.1 byte [] logo [private]

A byte array of the Transit Authority logo, if any.

Allowed formats are unspecified, as image format parsing/conversion is beyond the scope of this project.

5.77.2.2 String transitAuthorityName [private]

The name of the governing body of the associated TransitService.

This can be a Federal, State, or Local governing body responsible for the transit activity associated with the TransitService.

5.77.2.3 URL website [private]

A resolvable URL reference to the official Transit Authority web site.

Where possible, this site should contain contact info and links to policy, specialized transit requirements, or other information.

5.78 TransitProvider Class Reference

A TransitProvider is a description of a company or organization that is the producer of public transportation services.

Public Member Functions

- String **getProviderId** ()
- void **setProviderId** (String providerId)
- String **getName** ()
- void **setName** (String name)

Private Attributes

- String providerId

  A unique identifier that globally identifies this TransitProvider.
- String name

  A text description of the TransitProvider.

5.78.1 Detailed Description

A TransitProvider is a description of a company or organization that is the producer of public transportation services.

5.78.2 Member Data Documentation

5.78.2.1 String name [private]

A text description of the TransitProvider.

This is the text that will be displayed on guides, Route maps, and advertisements.

5.78.2.2 String providerId [private]

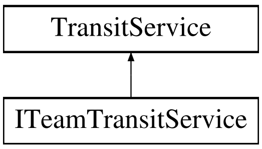
A unique identifier that globally identifies this TransitProvider.

The actual identifier type is unspecified and left to implementations. It may be the same as the name, if that is sufficient to provide uniqueness.

5.79 TransitService Interface Reference

The TransitService is an interface to get Route/Fare/Detour information from a TransitProvider.

Inheritance diagram for TransitService:



Public Member Functions

- Route getRoute (String routeId)

  Gets a Route by its unique identifier.
- Set< Route > getRoutes (Location pickup, Location dropoff, int distance)

  *Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.*
- TransitInfo getTransitInfo ()

  Gets metadata about the Transit Authority providing the information retrieved from this service.
- URL getServiceURL ()

  The URL that uniquely identifies this TransitService.

5.79.1 Detailed Description

The TransitService is an interface to get Route/Fare/Detour information from a TransitProvider.

This service will provide a consistent interface for the application logic to query to get this information.

5.79.2 Member Function Documentation

5.79.2.1 Route getRoute (String *routeId*)

Gets a Route by its unique identifier.

Precondition

routeId is not null or blank.

Postcondition

The Route is returned if the **routeId** is found, else null.

Parameters

routeId	The unique identifier of the Route

Returns

The matching Route, or null if not found

Implemented in ITeamTransitService.

5.79.2.2 Set<**Route**> getRoutes (**Location** *pickup,* **Location** *dropoff,* int *distance*)

Gets all available Routes that match a **pickup** or **dropoff** Location by not more than a given **distance**.

Precondition

pickup is not null or blank.
dropoff is not null or blank.
distance is non-negative.

Parameters

pickup	The requested pickup Location
pickup	The requested dropoff Location
distance	The distance (in miles) that each Route can deviate from the requested **pickup** or **dropoff** Location. For each Route returned, neither its start or end Location can differ from the requested **pickup** or **dropoff** Location by more than the value of the **distance** parameter.

Returns

The matching Routes

Implemented in ITeamTransitService.

5.79.2.3 URL getServiceURL ( )

The URL that uniquely identifies this TransitService.

In a REST environment, this might be the root of the REST API path. In a SOAP environment, it could represent a SOAP endpoint.

Returns

The URL of this service

Implemented in ITeamTransitService.

5.79.2.4 **TransitInfo** getTransitInfo ( )

Gets metadata about the Transit Authority providing the information retrieved from this service.

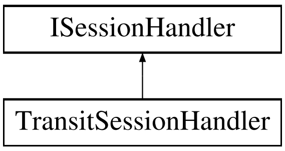
Returns

The TransitInfo of the Transit Authority of this service.

Implemented in ITeamTransitService.

5.80 TransitSessionHandler Class Reference

Inheritance diagram for TransitSessionHandler:



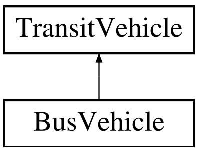
Public Member Functions

- boolean **verifySessionToken** (String sessionToken)

5.81 TransitVehicle Class Reference

Abstract transit vehicle class contains the common data for all types of vehicles and the Subject GPS Tracking and the GPS observer to receive GPS location updates.

Inheritance diagram for TransitVehicle:



Public Member Functions

- abstract void registerTrackingAlert (TrackingAlertObserver ao)

  Transit Vehicle is also the subject for tracking user subscribed alerts.
- abstract void **unregisterTrackingAlert** (TrackingAlertObserver ao)
- void checkForAlerts ()

  The Observer Pattern Subject notify method extended to limit the number of alerts issued by check conditions prior to actually triggering an alert.
- void triggerAlert (TrackingAlertObserver ao)

  The conditions in the Alert Specification were met, send update to the observer.
- void addAlertSpecification ()

  Add an alert specification AlertSpecification to this vehicle.
- void removeAlertSpecifcation ()

  Remove an alert specification from a transit vehicle.
- String toString ()

  Provide a generic method to output Transit Vehicle information.

Private Attributes

- VehicleObject vehicle

  Value Object holding vehicle details.
- GPSLocationObserver gpsObserver

  Observer that update the GPS coordinates of the vehicle as they are received.
- ArrayList< AlertSpecification > alertSpecification

  Rules to determine if this vehicle is in an alert zone.

5.81.1 Detailed Description

Abstract transit vehicle class contains the common data for all types of vehicles and the Subject GPS Tracking and the GPS observer to receive GPS location updates.

5.81.2 Member Function Documentation

5.81.2.1 void addAlertSpecification ( )

Add an alert specification AlertSpecification to this vehicle.

A vehicle may have these alerts:

- one or more users registered for location based alerts
- transit company registered for delay alerts, or loss of GPS signal alerts

5.81.2.2 void checkForAlerts ( )

The Observer Pattern Subject notify method extended to limit the number of alerts issued by check conditions prior to actually triggering an alert.

The checkForAlerts method uses AlertSpecification to determine if the observing vehicle should be notified. Calls triggerAlert.

5.82 TransitVehicleFactory Class Reference

Transit Vehicle Factory encapsulates the complexity of creating a new vehicle.

Public Member Functions

- TransitVehicle createTransitVehicle (URL url, int gpsDeviceID)

Protected Member Functions

- int getVehicleGPSDeviceID (URL url)

  Retrieve the GPS Device ID from repository of vehicles registered for route identified by type and URL.

Private Member Functions

- int getGPSTypeFromURL (URL url)

  Parse the input URL for information about connecting to GPS device in this vehicle.

5.82.1 Detailed Description

Transit Vehicle Factory encapsulates the complexity of creating a new vehicle.

Inputs are obtained from the user interface when a vehicle is registered by a user.

5.82.2 Member Function Documentation

5.82.2.1 TransitVehicle createTransitVehicle (URL *url,* int *gpsDeviceID*)

Determine what type of vehicle is needed.

Determine what type of GPS tracking is available on this vehicle and register with the appropriate GPSLocationTracking service.

5.82.2.2 int getGPSTypeFromURL (URL *url*) [private]

Parse the input URL for information about connecting to GPS device in this vehicle.

Parameters

url	- URL from User Interface, contains GPS connection information.

Returns

integer type of GPS Device Commercial Service, GPS Pusher, or GPS Puller.

5.82.2.3 int getVehicleGPSDeviceID (URL *url*) [protected]

Retrieve the GPS Device ID from repository of vehicles registered for route identified by type and URL.

Parameters

url	- URL identifying the transit company

Returns

integer GPS Device ID

5.83 Trip Class Reference

A Trip is considered an ordered collection of Routes going from a starting point to an ending point.

Public Member Functions

- Collection< Route > **getRoutes** ()
- void **setRoutes** (Collection< Route > routes)

Private Attributes

- Collection< Route > routes

  The ordered collection of Routes that when combined make a navigable Trip.

5.83.1 Detailed Description

A Trip is considered an ordered collection of Routes going from a starting point to an ending point.

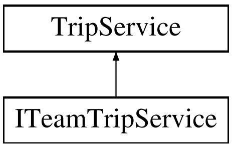
A Trip can be thought of as a composition of Routes, and the TripService is the service that composes them.

5.84 TripInformation Class Reference

5.85 TripService Interface Reference

A Service to calculate a collection of Routes, or a Trip, allowing for a continuous transit path from a start Location to an end Location.

Inheritance diagram for TripService:



Public Member Functions

- Trip calculateTrip (Location start, Location end)

    *Calculate an optimal Trip given a **start** Location and an **end** Location.*

5.85.1 Detailed Description

A Service to calculate a collection of Routes, or a Trip, allowing for a continuous transit path from a start Location to an end Location.

5.85.2 Member Function Documentation

5.85.2.1 Trip calculateTrip (Location *start,* Location *end*)

Calculate an optimal Trip given a **start** Location and an **end** Location.

Parameters

start	The requested start Location of the Trip.
end	The requested end Location of the Trip.

Returns

The calculated Trip

Implemented in ITeamTripService.

5.86 User Class Reference

This class represents a single user of the system, and all of the state data associated with that user.

Public Member Functions

- int getUserId ()

    This retrieves the user identifier.
- String getUsername ()

This retrieves the user's username.

- boolean isForcePasswordChange ()

  This checks to see if the password is in a state where it needs to be changed.
- String getFirstName ()

  This retrieves the user's first name.
- void setFirstName (String firstName)

  This sets the user's first name.
- String getEmail ()

  This retrieves the user's e-mail.
- void setEmail (String email)

  This sets the user's e-mail.
- Short getCountryCode ()

  This gets the user's country code.
- void setCountryCode (Short countryCode)

  This sets the user's country code.
- String getMobile ()

  This gets the user's mobile phone number.
- void setMobile (String mobile)

  This sets the user's mobile phone number.
- UserType getUserType ()

  This retrieves the type of the current user.
- void setUserType (UserType userType)

  This sets the type of the current user.

Protected Member Functions

- String getPasswordHash ()

  This retrieves the password hash for this user.
- void setPasswordHash (String passwordHash)

  This sets the password hash for this user.
- void setForcePasswordChange (boolean forcePasswordChange)

  This sets the state indicating if the password is in a state where it needs to be changed.

Package Functions

- User (int userId, String username)

  This constructs a new User object.

Private Attributes

- final int **userId**
- final String **username**
- String **passwordHash**
- boolean **forcePasswordChange**
- String **firstName**
- String **email**
- Short **countryCode**
- String **mobile**
- UserType **userType**

5.86.1 Detailed Description

This class represents a single user of the system, and all of the state data associated with that user.

The user should already exist in the database before instantiating this object.

5.86.2 Constructor & Destructor Documentation

5.86.2.1 User (int *userId*, String *username*) [package]

This constructs a new User object.

It is not visible to clients, as User objects should only be constructed through the UserRepository.

Parameters

userId	This is the user's unique identifier, which should match the database.
username	This is the user's username. It cannot be changed.

5.86.3 Member Function Documentation

5.86.3.1 Short getCountryCode ( )

This gets the user's country code.

Returns

user's country code

5.86.3.2 String getEmail ( )

This retrieves the user's e-mail.

Returns

user's e-mail

5.86.3.3 String getFirstName ( )

This retrieves the user's first name.

Returns

user's first name

5.86.3.4 String getMobile ( )

This gets the user's mobile phone number.

Postcondition

The mobile phone number returned should be a String containing only digits.

Returns

user's mobile phone number

5.86.3.5 String getPasswordHash ( ) [protected]

This retrieves the password hash for this user.

It has decreased visibility and is ignored when serializing responses, as this data should not be shared beyond this module.

Returns

hash of the user's password

5.86.3.6 int getUserId ( )

This retrieves the user identifier.

Returns

user identifier

5.86.3.7 String getUsername ( )

This retrieves the user's username.

Returns

username

5.86.3.8 UserType getUserType ( )

This retrieves the type of the current user.

Returns

user type

5.86.3.9 boolean isForcePasswordChange ( )

This checks to see if the password is in a state where it needs to be changed.

Returns

true if it is, false if it is not

5.86.3.10 void setCountryCode (Short *countryCode*)

This sets the user's country code.

Parameters

countryCode	user's country code

5.86.3.11 void setEmail (String *email*)

This sets the user's e-mail.

Parameters

email	user's e-mail

5.86.3.12 void setFirstName (String *firstName*)

This sets the user's first name.

Parameters

firstName	user's first name

5.86.3.13 void setForcePasswordChange (boolean *forcePasswordChange*) [protected]

This sets the state indicating if the password is in a state where it needs to be changed.

This is ignored during deserialization, as it should never be set from outside this module. it is never

Parameters

forcePasswordChange	true if it should be set, false if it should be cleared

5.86.3.14 void setMobile (String *mobile*)

This sets the user's mobile phone number.

Precondition

The mobile parameter should be a String containing only digits.

Parameters

mobile	user's mobile phone number

5.86.3.15 void setPasswordHash (String *passwordHash*) [protected]

This sets the password hash for this user.

It has decreased visibility and is ignored when deserializing requests, as this data should not be set outside this module.

Parameters

passwordHash	hash of the user's password

5.86.3.16 void setUserType (UserType *userType*)

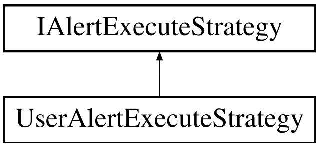
This sets the type of the current user.

Parameters

userType	user type

5.87 UserAlertExecuteStrategy Class Reference

Inheritance diagram for UserAlertExecuteStrategy:



Public Member Functions

- boolean **execute** ()

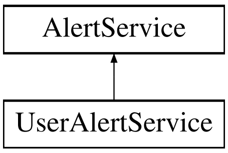
Package Attributes

- AlertRepository **alertRepository**

5.88 UserAlertRequestModel Class Reference

5.89 UserAlertService Class Reference

Inheritance diagram for UserAlertService:



Public Member Functions

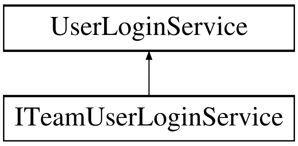
- AlertResponseModel **createAlert** (AlertRequestModel requestModel)
- AlertResponseModel **saveAlert** (Alert alertModel)
- AlertResponseModel **deleteAlert** (Alert alertModel)
- AlertResponseModel **updateAlert** (Alert alertModel)

Additional Inherited Members

5.90 UserLoginService Interface Reference

This is the generic BusBuddy UserLoginService interface.

Inheritance diagram for UserLoginService:



Public Member Functions

- String login (String username, String password) throws BusBuddyException

  This method handles the login process.

- void logout (String sessionToken) throws BusBuddyException

  This method logs a user out, invalidating their session in the database.
- User getUser (String sessionToken) throws BusBuddyException

  This method retrieves the user tied to a supplied session token.
- String createAlertSession (String sessionToken) throws BusBuddyException

  This method creates a new session to be used by an alert.
- void sendUsername (String email) throws BusBuddyException

  This method sends a user his or her username via e-mail.
- void sendUsername (short countryCode, String mobile) throws BusBuddyException

  This method sends a user his or her username via SMS.
- void **resetPassword** (String username, String email) throws BusBuddyException
- void **resetPassword** (String username, short countryCode, String mobile) throws BusBuddyException

5.90.1 Detailed Description

This is the generic BusBuddy UserLoginService interface.

This interface contains methods dealing with user login and session management. It is one of three interfaces that a user module implementation must implement. It can be implemented as a service or as a service client.

5.90.2 Member Function Documentation

5.90.2.1 String createAlertSession (String *sessionToken*) throws BusBuddyException

This method creates a new session to be used by an alert.

Since creation of an alert requires an active user session, this takes an active sessionToken as a parameter. It will then create a new alert session for the same user as the active session. This "alert session" will be long-lived, so it won't expire like the main session. This will allow the Alert module to use this sessionToken when the alert executes.

Precondition

The session token must be linked to an active and valid session, which must be linked to an active account.

Postcondition

The returned session token points to a valid alert session for this user, which will not expire. The base session's expiration time will be advanced based on this activity against the session.

Parameters

sessionToken	The session token identifying the session that is creating the new alert session.

Returns

Session token representing the new alert seession.

Exceptions

BusBuddyBadRequestException	This exception is thrown if the session token is blank.
BusBuddyForbiddenException	This exception is thrown if the session token is invalid, linked to an expired session, or the user does not have permission to be signed in.
BusBuddyInternalException	This exception is thrown if an internal error prevents processing of the request.

Implemented in ITeamUserLoginService.

5.90.2.2 User getUser (String *sessionToken*) throws BusBuddyException

This method retrieves the user tied to a supplied session token.

It will also update the expiration time on the session to keep it valid.

Precondition

The session token must be linked to an active and valid session, which must be linked to an active account.

Postcondition

The returned session token points to a valid session for this user. The expiration time will be advanced based on this activity against the session.

Parameters

sessionToken	The session token identifying the session that the user information should be retrieved for.

Returns

User object for the user linked to the session represented by the session token parameter.

Exceptions

BusBuddyBadRequestException	This exception is thrown if the session token is blank.
BusBuddyForbiddenException	This exception is thrown if the session token is invalid, linked to an expired session, or the user does not have permission to be signed in.
BusBuddyInternalException	This exception is thrown if an internal error prevents processing of the request.

Implemented in ITeamUserLoginService.

5.90.2.3 String login (String *username*, String *password*) throws BusBuddyException

This method handles the login process.

A username and password are supplied. A valid session is created for this user.

Precondition

Login credentials must be valid and linked to an active account, or a .common.BusBuddyForbiddenException will be thrown.

Postcondition

The returned session token points to a valid session for this user.

Parameters

username	Username of the user to login as.
password	Password of the user to login as.

Returns

session token of the new session

Exceptions

BusBuddyBadRequestException	This exception is thrown if the username or password are blank.
BusBuddyForbiddenException	This exception is thrown if the credentials are incorrect, or the user does not have permission to sign in.
BusBuddyInternalException	This exception is thrown if an internal error prevents processing of the request.

Implemented in ITeamUserLoginService.

5.90.2.4 void logout (String *sessionToken*) throws BusBuddyException

This method logs a user out, invalidating their session in the database.

Precondition

The sessionToken parameter must be a valid session identifier in the database.

Postcondition

The session will be invalidated and future calls using that sessionToken will fail.

Parameters

sessionToken	This is the session token that identifies the session.

Exceptions

BusBuddyNotFoundException	This exception is thrown if the session token is blank or missing on the request..
BusBuddyNotFoundException	This exception is thrown if the session token is invalid.
BusBuddyInternalException	This exception is thrown if an internal error prevents processing of the request.

Implemented in ITeamUserLoginService.

5.90.2.5 void sendUsername (String *email*) throws BusBuddyException

This method sends a user his or her username via e-mail.

Precondition

The e-mail address provided must be linked to a valid and active account.

Postcondition

An e-mail has been sent to the user, containing the user's username.

Parameters

email	E-mail address of the account to send to.

Exceptions

BusBuddyBadRequestException	This exception is thrown if the e-mail address is blank or invalid.
BusBuddyForbiddenException	This exception is thrown if the e-mail address is linked to an account that is suspended or deleted.
BusBuddyNotFoundException	This exception is thrown if the e-mail address doesn't link to a valid user.
BusBuddyInternalException	This exception is thrown if an internal error prevents processing of the request.

Implemented in ITeamUserLoginService.

5.90.2.6 void sendUsername (short *countryCode*, String *mobile*) throws BusBuddyException

This method sends a user his or her username via SMS.

Precondition

The mobile details provided must be linked to a valid and active account.

Postcondition

An e-mail has been sent to the user, containing the user's username.

Parameters

email	E-mail address of the account to send to.

Exceptions

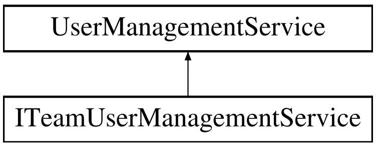
BusBuddyBadRequestException	This exception is thrown if the e-mail address is blank or invalid.
BusBuddyForbiddenException	This exception is thrown if the e-mail address is linked to an account that is suspended or deleted.
BusBuddyNotFoundException	This exception is thrown if the e-mail address doesn't link to a valid user.
BusBuddyInternalException	This exception is thrown if an internal error prevents processing of the request.

Implemented in ITeamUserLoginService.

5.91 UserManagementService Interface Reference

This is the generic BusBuddy UserManagementService interface.

Inheritance diagram for UserManagementService:



Public Member Functions

- User **createUser** (User userToCreate, String password) throws BusBuddyException
- User **findUserByUsername** (String sessionToken, String username) throws BusBuddyException

- User **findUserByEmail** (String sessionToken, String email)
- User **findUserByMobile** (String sessionToken, short countryCode, String mobile) throws BusBuddyException
- void **updateUser** (String sessionToken, User newUserData, String password) throws BusBuddyException
- void **deleteUser** (String sessionToken, User userToDelete) throws BusBuddyException

5.91.1 Detailed Description

This is the generic BusBuddy UserManagementService interface.

This interface contains methods dealing with user account management. It is one of three interfaces that a user module implementation must implement. It can be implemented as a service or as a service client.

5.92 UserRepository Class Reference

This class is responsible for handling database access for User objects, and to construct, persist, and retrieve User objects.

Package Functions

- User **createUser** (String username, String password)
- User getUserById (int userId) throws BusBuddyInternalException, BusBuddyNotFoundException

  This method attempts to retrieve a user by id number.
- User getUserByUsername (String username) throws BusBuddyInternalException, BusBuddyNotFoundException

  This method attempts to retrieve a user by username.
- User getUserByEmail (String email) throws BusBuddyInternalException, BusBuddyNotFoundException

  This method attempts to retrieve a user by e-mail address.
- User getUserByMobile (short countryCode, String mobile) throws BusBuddyInternalException, BusBuddyNotFoundException

  This method attempts to retrieve a user by mobile phone number.
- void **updateUser** (User newUserData)
- void **deleteUser** (User userToDelete)

5.92.1 Detailed Description

This class is responsible for handling database access for User objects, and to construct, persist, and retrieve User objects.

5.92.2 Member Function Documentation

5.92.2.1 User getUserByEmail (String *email*) throws **BusBuddyInternalException, BusBuddyNotFoundException** `[package]`

This method attempts to retrieve a user by e-mail address.

It is not case sensitive. The method will take an e-mail address, read the details from the database, and construct a user object with the given details.

Precondition

A user with the supplied e-mail address exists within the database.

Postcondition

A user will be returned whose e-mail address matches the supplied e-mail address parameter.

Parameters

email	This is the e-mail address to look up.

Returns

The user with the given e-mail address.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.
BusBuddyNotFoundException	This exception is thrown when the requested user record could not be found.

5.92.2.2 User getUserById (int *userId*) throws BusBuddyInternalException, BusBuddyNotFoundException [package]

This method attempts to retrieve a user by id number.

The method will take a user id, read the details from the database, and construct a user object with the given details.

Precondition

A user with the supplied user id exists within the database.

Postcondition

A user will be returned whose user id matches the supplied userId parameter.

Parameters

userId	This is the user ID to look up.

Returns

The user with the given ID.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.
BusBuddyNotFoundException	This exception is thrown when the requested user record could not be found.

5.92.2.3 User getUserByMobile (short *countryCode,* String *mobile*) throws BusBuddyInternalException, BusBuddyNotFoundException [package]

This method attempts to retrieve a user by mobile phone number.

The method will take a mobile phone number, read the details from the database, and construct a user object with the given details.

Precondition

A user with the supplied mobile phone number exists within the database.

Postcondition

A user will be returned whose mobile phone details match the supplied parameters.

Parameters

countryCode	This is the country code of the user's mobile phone number.
mobile	This is the remainder of the user's mobile phone numer. This string should consist entirely of digits.

Returns

The user with the given mobile phone details.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.
BusBuddyNotFoundException	This exception is thrown when the requested user record could not be found.

5.92.2.4 User getUserByUsername (String *username*) throws BusBuddyInternalException, BusBuddyNotFoundException [package]

This method attempts to retrieve a user by username.

It is not case sensitive. The method will take a username, read the details from the database, and construct a user object with the given details.

Precondition

A user with the supplied username exists within the database.

Postcondition

A user will be returned whose username matches the supplied username parameter.

Parameters

username	This is the username to look up.

Returns

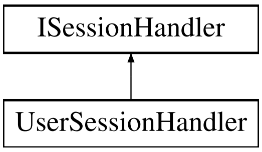
The user with the given username.

Exceptions

BusBuddyInternalException	This exception is thrown when there is a database error.
BusBuddyNotFoundException	This exception is thrown when the requested user record could not be found.

5.93 UserSessionHandler Class Reference

Inheritance diagram for UserSessionHandler:



Public Member Functions

- boolean **verifySessionToken** (String sessionToken)

5.94 UserSessionInformation Class Reference

Public Member Functions

- String getUserId ()
- void setUserId (String userId)
- String getUserSessionToken ()
- void setUserSessionToken (String userSessionToken)

Private Attributes

- String userId

  User ID of the user that initiated the alert.
- String userSessionToken

  A long lived session token to validate the authenticity of request to UserModule.

5.94.1 Member Function Documentation

5.94.1.1 String getUserId ( )

Returns

the userId

5.94.1.2 String getUserSessionToken ( )

Returns

the userSessionToken

5.94.1.3 void setUserId (String *userId*)

Parameters

userId	the userId to set

5.94.1.4 void setUserSessionToken (String *userSessionToken*)

Parameters

userSessionToken	the userSessionToken to set

5.94.2 Member Data Documentation

5.94.2.1 String userSessionToken `[private]`

A long lived session token to validate the authenticity of request to UserModule.

This is required to get the current user information in order to alert the user.

5.95 UserTrackingAlertObject Class Reference

User tracking alert information obtained from the user interface when the user registers for an alert.

Classes

- enum AlertType

Public Member Functions

- int getRouteID ()

  Return the route identifier.
- void setRouteID (int routeID)

  Set the route identifier, obtained by translating the user interface route description drop down to the transit information route id.
- Location getStopLocation ()

  Get the latitude and longitude of the vehicle stop.
- void setStopLocation (Location stopLocation)

  Set the vehicle stop GPS coordinates with information uploaded by a transit company.
- Date getScheduledTime ()

  The time a vehicle is scheduled to be at this stop.
- void setScheduledTime (Date scheduledTime)

  Time vehicle is expected at a stop.
- Date **getAlertTime** ()
- void **setAlertTime** (Date alertTime)
- String **getUserContactInfo** ()
- void **setUserContactInfo** (String userContactInfo)
- AlertType **getType** ()
- void **setType** (AlertType type)
- URL **getTransitCoInfo** ()
- void **setTransitCoInfo** (URL transitCoInfo)

Private Attributes

- URL **transitCoInfo**
- int **routeID**
- Location **stopLocation**
- Date **scheduledTime**
- Date **alertTime**
- String **userContactInfo**
- AlertType **type**

5.95.1 Detailed Description

User tracking alert information obtained from the user interface when the user registers for an alert.

5.95.2 Member Function Documentation

5.95.2.1 int getRouteID ( )

Return the route identifier.

Returns

- integer the route ID this where vehicle is registered.

5.95.2.2 Date getScheduledTime ( )

The time a vehicle is scheduled to be at this stop.

Returns

- Date and time the vehicle is expected to arrive at a stop.

5.95.2.3 Location getStopLocation ( )

Get the latitude and longitude of the vehicle stop.

Returns

- Location value object with GPS latitude and longitude coordinates of the stop.

5.95.2.4 void setRouteID (int *routeID*)

Set the route identifier, obtained by translating the user interface route description drop down to the transit information route id.

Parameters

routeID	- integer route identification

5.95.2.5 void setScheduledTime (Date *scheduledTime*)

Time vehicle is expected at a stop.

Obtained from information uploaded by a transit company. Used for determining delay alerts.

Parameters

scheduledTime	Date and time the vehicle is expected to arrive at a stop.

5.95.2.6 void setStopLocation (Location *stopLocation*)

Set the vehicle stop GPS coordinates with information uploaded by a transit company.

Parameters

stopLocation	- Location GPS coordinates of stop location

5.96 UserType Enum Reference

Public Attributes

- **NORMAL_USER**
- **SYSTEM_ADMINISTRATOR**
- **SUSPENDED_USER**

5.97 VehicleObject Class Reference

Value Object containing vehicle information obtained when the user registers a vehicle using the user interface.

Public Member Functions

- int **getGPSDeviceID** ()
- String **getGPSDeviceInfo** ()
- String **getTransitCoURL** ()
- int **getCurrentRoute** ()

Protected Member Functions

- void **setGPSDeviceID** (int gpsDeviceID)
- void **setGPSDeviceInfo** (String gpsDeviceInfo)
- void **setTransitCoURL** (String transitCoURL)
- void **setCurrentRoute** (int currentRoute)

Private Attributes

- int gpsDeviceID

  GPS hardware device ID.
- String gpsDeviceInfo

  GPS device contact information, commercial web URL, GPS wireless connection or port number.
- String transitCoURL

  Transit company operating this vehicle.
- int currentRoute

  Current route number.

5.97.1 Detailed Description

Value Object containing vehicle information obtained when the user registers a vehicle using the user interface.

5.98 VehicleRepository Class Reference

Repository for information on vehicles registered on a route.

Public Member Functions

- VehicleRepository ()

  Create the initial repository for saving vehicles registered with the tracking service.
- void addVehicle (TransitVehicle vehicle)

  Add a vehicle to the repository.
- void removeVehicle (int gpsDeviceID)

  Remove a vehicle from the repository.
- void updateVehicle (TransitVehicle vehicle)

  A vehicle may switch routes, update an existing vehicle in the repository.
- TransitVehicle findVehicle (int gpsDeviceID)

  Find a vehicle currently stored in the repository based on the unique GPS device ID.

Static Public Member Functions

- static ArrayList< TransitVehicle > findVehiclesByRoute (URL transitCoURL, int routeID)

  Find all vehicles from a transit company registered on a route.

Private Attributes

- ArrayList< TransitVehicle > vehicleList = null

  List of vehicles currently registered and available in this repository.

5.98.1 Detailed Description

Repository for information on vehicles registered on a route.

5.98.2 Member Function Documentation

5.98.2.1 TransitVehicle findVehicle (int *gpsDeviceID*)

Find a vehicle currently stored in the repository based on the unique GPS device ID.

Parameters

gpsDeviceID	- integer GPS device ID

Returns

VehicleObject matching vehicle or null if no matching vehicle found.

5.98.2.2 static ArrayList<TransitVehicle> findVehiclesByRoute (URL *transitCoURL,* int *routeID*) [static]

Find all vehicles from a transit company registered on a route.

Parameters

transitCoURL	URL or the transit company
routeID	- integer route that vehicle is currently registered on.

Returns

ArrayList<TransitVehicle> of all vehicles for transit company registered on the route or null if no matching vehicles found.

5.98.2.3 void removeVehicle (int *gpsDeviceID*)

Remove a vehicle from the repository.

Parameters

gpsDeviceID	- integer the GPS id if the vehicle to remove.

5.98.2.4 void updateVehicle (TransitVehicle *vehicle*)

A vehicle may switch routes, update an existing vehicle in the repository.

Parameters

vehicle	- VehicleObject new vehicle information from driver through UI

Index