

BusBuddy

Generated by Doxygen 1.8.3.1

Fri Apr 26 2013 00:35:04

Contents

1	Namespace Index	1
1.1	Packages	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	7
3.1	Class List	7
4	Namespace Documentation	13
4.1	Package alert.client	13
4.1.1	Detailed Description	13
4.2	Package alert.client.model	13
4.2.1	Detailed Description	13
4.3	Package alert.controller	14
4.3.1	Detailed Description	14
4.4	Package alert.controller.model	14
4.4.1	Detailed Description	15
4.5	Package alert.domain	15
4.5.1	Detailed Description	15
4.6	Package alert.domain.model	15
4.6.1	Detailed Description	16
4.7	Package alert.enums	16
4.7.1	Detailed Description	16
4.8	Package alert.service	16
4.8.1	Detailed Description	17
4.9	Package common	17
4.9.1	Detailed Description	17
4.10	Package tracking	18
4.10.1	Detailed Description	19
4.11	Package transit	19
4.11.1	Detailed Description	20

4.12	Package user	20
4.12.1	Detailed Description	21
5	Class Documentation	23
5.1	AbstractFeedParserTemplate Class Reference	23
5.1.1	Detailed Description	24
5.1.2	Member Function Documentation	24
5.1.2.1	loadFeed	24
5.1.2.2	parseFeed	24
5.1.2.3	saveRoutes	25
5.1.2.4	start	25
5.1.2.5	validate	25
5.2	Alert Class Reference	25
5.2.1	Detailed Description	27
5.2.2	Member Data Documentation	27
5.2.2.1	alertInitiator	27
5.2.2.2	alertRunType	27
5.2.2.3	alertType	27
5.2.2.4	Status	27
5.2.2.5	userInformation	27
5.3	AlertExecuteStrategyFactory Class Reference	27
5.3.1	Detailed Description	28
5.3.2	Member Function Documentation	28
5.3.2.1	getAlertService	28
5.4	AlertFactory Class Reference	28
5.4.1	Detailed Description	28
5.4.2	Member Function Documentation	28
5.4.2.1	createAlert	28
5.4.2.2	createAlert	28
5.5	AlertInitiator Enum Reference	29
5.5.1	Detailed Description	29
5.6	AlertNotificationType Enum Reference	29
5.6.1	Detailed Description	29
5.6.2	Member Data Documentation	29
5.6.2.1	PlannedDisruption	29
5.6.2.2	ScheduleInformation	30
5.6.2.3	UnplannedDisruption	30
5.7	AlertRangeLogic Class Reference	30
5.7.1	Detailed Description	30
5.8	AlertRecurringType Enum Reference	30

5.8.1 Detailed Description	31
5.9 AlertRepository Class Reference	31
5.9.1 Detailed Description	31
5.9.2 Member Function Documentation	31
5.9.2.1 deleteAlert	31
5.9.2.2 getAlertByDateTime	32
5.9.2.3 getAlertByRoute	32
5.9.2.4 getAlertByUserId	32
5.9.2.5 saveAlert	32
5.9.2.6 updateAlert	33
5.10 AlertRequestController Class Reference	33
5.10.1 Detailed Description	33
5.10.2 Member Function Documentation	34
5.10.2.1 processTrackingAlertRequest	34
5.10.2.2 processTransitAlertRequest	34
5.10.2.3 processUserAlertRequest	35
5.10.2.4 verifySession	35
5.11 AlertRequestModel Class Reference	35
5.11.1 Detailed Description	36
5.11.2 Member Data Documentation	36
5.11.2.1 alertInitiator	36
5.12 AlertResponseModel Class Reference	36
5.12.1 Detailed Description	36
5.13 AlertRunType Enum Reference	36
5.13.1 Detailed Description	36
5.14 AlertService Class Reference	37
5.14.1 Detailed Description	37
5.14.2 Member Function Documentation	37
5.14.2.1 createAlert	37
5.14.2.2 deleteAlert	38
5.14.2.3 saveAlert	38
5.14.2.4 sendAlert	38
5.14.2.5 updateAlert	38
5.14.3 Member Data Documentation	39
5.14.3.1 alertExecuteStrategyFactory	39
5.14.3.2 alertRepository	39
5.15 AlertServiceFactory Class Reference	39
5.15.1 Detailed Description	39
5.15.2 Member Function Documentation	40
5.15.2.1 getAlertService	40

5.15.3	Member Data Documentation	40
5.15.3.1	trackingAlertService	40
5.15.3.2	transitAlertService	40
5.15.3.3	userAlertService	40
5.16	AlertSpecification Interface Reference	40
5.16.1	Detailed Description	41
5.16.2	Member Function Documentation	41
5.16.2.1	inAlertRange	41
5.17	AlertStatus Enum Reference	41
5.17.1	Detailed Description	41
5.17.2	Member Data Documentation	42
5.17.2.1	Deactive	42
5.17.2.2	Error	42
5.17.2.3	Expired	42
5.18	AlertType Enum Reference	42
5.18.1	Detailed Description	42
5.19	AlertUserClient Class Reference	42
5.19.1	Detailed Description	43
5.19.2	Member Function Documentation	43
5.19.2.1	getUserInformation	43
5.20	BaseController Class Reference	43
5.20.1	Detailed Description	43
5.20.2	Member Function Documentation	43
5.20.2.1	handleBusBuddyException	44
5.20.2.2	handleGenericException	44
5.21	BusBuddyBadRequestException Class Reference	44
5.21.1	Detailed Description	45
5.21.2	Member Function Documentation	45
5.21.2.1	getHttpCode	45
5.22	BusBuddyConflictException Class Reference	45
5.22.1	Detailed Description	46
5.22.2	Member Function Documentation	46
5.22.2.1	getHttpCode	46
5.23	BusBuddyException Class Reference	46
5.23.1	Detailed Description	46
5.23.2	Member Function Documentation	47
5.23.2.1	getHttpCode	47
5.24	BusBuddyForbiddenException Class Reference	47
5.24.1	Detailed Description	47
5.24.2	Member Function Documentation	47

5.24.2.1	getHttpCode	47
5.25	BusBuddyInternalException Class Reference	48
5.25.1	Detailed Description	48
5.25.2	Member Function Documentation	48
5.25.2.1	getHttpCode	48
5.26	BusBuddyNotFoundException Class Reference	49
5.26.1	Detailed Description	49
5.26.2	Member Function Documentation	49
5.26.2.1	getHttpCode	49
5.27	BusVehicle Class Reference	49
5.27.1	Detailed Description	50
5.27.2	Member Data Documentation	50
5.27.2.1	alertList	50
5.28	CertificateHandler Class Reference	50
5.28.1	Detailed Description	51
5.28.2	Member Function Documentation	51
5.28.2.1	verifySessionToken	51
5.29	CommercialTracking Class Reference	51
5.29.1	Detailed Description	52
5.29.2	Constructor & Destructor Documentation	52
5.29.2.1	CommercialTracking	52
5.29.3	Member Function Documentation	52
5.29.3.1	getInstance	52
5.30	CommercialTracking.CommercialTrackingHolder Class Reference	52
5.30.1	Detailed Description	53
5.31	DelayAlertLogic Class Reference	53
5.31.1	Detailed Description	53
5.32	Detour Class Reference	53
5.32.1	Detailed Description	54
5.32.2	Member Data Documentation	54
5.32.2.1	cause	54
5.33	Fare Class Reference	54
5.33.1	Detailed Description	54
5.33.2	Member Function Documentation	54
5.33.2.1	setDiscountedFare	54
5.33.2.2	setRegularFare	54
5.34	FavoriteTransitService Class Reference	55
5.34.1	Detailed Description	55
5.34.2	Constructor & Destructor Documentation	55
5.34.2.1	FavoriteTransitService	55

5.34.3	Member Function Documentation	55
5.34.3.1	getFavoriteRouteIds	55
5.34.3.2	getTransitServiceUrl	56
5.34.3.3	isFavoriteTransitService	56
5.34.3.4	setFavoriteRouteIds	56
5.34.3.5	setFavoriteTransitService	56
5.35	GoogleTransitServiceAdapter Class Reference	56
5.35.1	Detailed Description	57
5.35.2	Constructor & Destructor Documentation	57
5.35.2.1	GoogleTransitServiceAdapter	57
5.35.3	Member Function Documentation	57
5.35.3.1	getRoute	57
5.35.3.2	getRoutes	58
5.36	GoogleTransitServiceAPI Interface Reference	58
5.36.1	Detailed Description	58
5.37	GPSLocationObject Class Reference	58
5.37.1	Detailed Description	59
5.38	GPSLocationObserver Class Reference	59
5.38.1	Detailed Description	59
5.38.2	Member Function Documentation	59
5.38.2.1	getGPSLocation	59
5.38.2.2	gpsUpdate	60
5.38.2.3	setGPSLocation	60
5.39	GPSLocationTracking Class Reference	60
5.39.1	Detailed Description	60
5.39.2	Member Function Documentation	60
5.39.2.1	registerGPSDevice	60
5.39.2.2	unregisterGPSDevice	61
5.40	GPSPuller Class Reference	61
5.40.1	Detailed Description	62
5.40.2	Constructor & Destructor Documentation	62
5.40.2.1	GPSPuller	62
5.41	GPSPuller.GPSPullerHolder Class Reference	62
5.41.1	Detailed Description	62
5.42	GPSPusher Class Reference	62
5.42.1	Detailed Description	63
5.42.2	Constructor & Destructor Documentation	64
5.42.2.1	GPSPusher	64
5.42.3	Member Function Documentation	64
5.42.3.1	getInstance	64

5.43	GPSPusher.GPSPusherHolder Class Reference	64
5.43.1	Detailed Description	64
5.44	GPSVehicleTracker Class Reference	64
5.44.1	Detailed Description	65
5.44.2	Constructor & Destructor Documentation	65
5.44.2.1	GPSVehicleTracker	65
5.45	GTFSFeedParser Class Reference	65
5.45.1	Detailed Description	65
5.45.2	Member Function Documentation	65
5.45.2.1	parseFeed	65
5.46	HashUtility Class Reference	66
5.46.1	Detailed Description	66
5.46.2	Member Function Documentation	66
5.46.2.1	hash	66
5.47	IAAlertExecuteStrategy Interface Reference	66
5.47.1	Detailed Description	67
5.47.2	Member Function Documentation	67
5.47.2.1	execute	67
5.48	InvalidRouteParseException Class Reference	67
5.48.1	Detailed Description	68
5.48.2	Constructor & Destructor Documentation	68
5.48.2.1	InvalidRouteParseException	68
5.48.3	Member Data Documentation	68
5.48.3.1	routeBatch	68
5.48.3.2	serialVersionUID	68
5.49	ISessionHandler Interface Reference	68
5.49.1	Detailed Description	68
5.49.2	Member Function Documentation	69
5.49.2.1	verifySessionToken	69
5.50	ITeamTransitServiceController Class Reference	69
5.50.1	Detailed Description	70
5.50.2	Member Function Documentation	70
5.50.2.1	getRoute	70
5.50.2.2	getRoutes	70
5.50.2.3	getServiceURL	71
5.50.2.4	getTransitInfo	71
5.50.2.5	handleRouteDisruptionEvent	71
5.50.3	Member Data Documentation	71
5.50.3.1	alertRequestController	71
5.50.3.2	transitFeed	71

5.51 ITeamTripServiceController Class Reference	72
5.51.1 Detailed Description	72
5.51.2 Member Function Documentation	72
5.51.2.1 calculateTrip	72
5.52 ITeamUserFavoritesService Class Reference	72
5.52.1 Detailed Description	73
5.52.2 Member Function Documentation	73
5.52.2.1 readFavorites	73
5.52.2.2 saveFavorites	73
5.53 ITeamUserLoginService Class Reference	73
5.53.1 Detailed Description	74
5.53.2 Member Function Documentation	74
5.53.2.1 checkPermissions	74
5.53.2.2 createAlertSession	75
5.53.2.3 getUser	75
5.53.2.4 login	75
5.53.2.5 logout	75
5.53.2.6 resetPassword	75
5.53.2.7 resetPassword	75
5.53.2.8 setUsername	76
5.53.2.9 setUsername	76
5.54 ITeamUserManagementService Class Reference	76
5.54.1 Detailed Description	76
5.54.2 Member Function Documentation	76
5.54.2.1 createUser	76
5.54.2.2 deleteUser	77
5.54.2.3 findUserByEmail	77
5.54.2.4 findUserByMobile	77
5.54.2.5 findUserByUsername	77
5.54.2.6 updateUser	77
5.55 ITrackingService Interface Reference	77
5.55.1 Detailed Description	78
5.55.2 Member Function Documentation	78
5.55.2.1 addUserTrackingAlert	78
5.55.2.2 getTransitVehicleLocation	78
5.55.2.3 registerVehicleOnRoute	78
5.55.2.4 startTrackingController	79
5.55.2.5 unregisterVehicleFromRoute	79
5.56 Location Class Reference	79
5.56.1 Detailed Description	79

5.56.2	Constructor & Destructor Documentation	79
5.56.2.1	Location	79
5.57	MessageDeliveryUtility Class Reference	80
5.57.1	Detailed Description	80
5.57.2	Member Function Documentation	80
5.57.2.1	sendEmail	80
5.57.2.2	sendSms	80
5.58	OneTimeAlert Class Reference	81
5.58.1	Detailed Description	81
5.58.2	Member Data Documentation	81
5.58.2.1	dateExecuted	81
5.59	PersistedTransitFeed Class Reference	81
5.59.1	Detailed Description	82
5.59.2	Member Function Documentation	82
5.59.2.1	getRoute	82
5.59.2.2	getRoutes	83
5.60	RecurringAlert Class Reference	83
5.60.1	Detailed Description	84
5.60.2	Member Data Documentation	84
5.60.2.1	alertRecurringType	84
5.60.2.2	repeatEvery	84
5.61	RecurringData Class Reference	84
5.61.1	Detailed Description	85
5.61.2	Member Data Documentation	85
5.61.2.1	dayOfMonth	85
5.61.2.2	dayOfWeek	85
5.61.2.3	dayOfYear	85
5.61.2.4	startHour	85
5.61.2.5	startMinute	85
5.62	Route Class Reference	86
5.62.1	Detailed Description	86
5.62.2	Member Data Documentation	86
5.62.2.1	detours	86
5.62.2.2	routeName	86
5.62.2.3	stops	86
5.63	RouteAlertExecuteStrategy Class Reference	87
5.63.1	Detailed Description	87
5.63.2	Member Function Documentation	87
5.63.2.1	execute	87
5.63.3	Member Data Documentation	87

5.63.3.1	alertRepository	87
5.63.3.2	userClient	88
5.64	RouteDisruptionAlert Class Reference	88
5.64.1	Detailed Description	88
5.64.2	Member Data Documentation	88
5.64.2.1	routeld	88
5.64.2.2	transitServiceUrl	88
5.65	RouteDisruptionEvent Class Reference	88
5.65.1	Detailed Description	89
5.65.2	Constructor & Destructor Documentation	89
5.65.2.1	RouteDisruptionEvent	89
5.66	RouteRepository Interface Reference	89
5.66.1	Detailed Description	89
5.66.2	Member Function Documentation	90
5.66.2.1	delete	90
5.66.2.2	getAll	90
5.66.2.3	read	90
5.66.2.4	save	90
5.66.2.5	save	91
5.67	RouteSpecification Class Reference	91
5.67.1	Detailed Description	91
5.67.2	Member Function Documentation	92
5.67.2.1	isSatisfiedBy	92
5.68	Session Class Reference	92
5.68.1	Detailed Description	93
5.68.2	Constructor & Destructor Documentation	93
5.68.2.1	Session	93
5.68.3	Member Function Documentation	93
5.68.3.1	getCreationTime	93
5.68.3.2	getExpirationTime	93
5.68.3.3	getSessionToken	94
5.68.3.4	getUserId	94
5.68.3.5	isAlertSession	94
5.68.3.6	isValid	94
5.68.3.7	setExpirationTime	94
5.68.3.8	setValid	94
5.69	SessionRepository Class Reference	95
5.69.1	Detailed Description	95
5.69.2	Member Function Documentation	95
5.69.2.1	createSession	95

5.69.2.2	getSession	96
5.69.2.3	killSession	96
5.70	SessionTokenHandler Class Reference	97
5.70.1	Detailed Description	97
5.70.2	Member Function Documentation	97
5.70.2.1	verifySessionToken	97
5.71	SessionVerificationFactory Class Reference	97
5.71.1	Detailed Description	97
5.71.2	Member Function Documentation	97
5.71.2.1	getSessionTokenVerificationStrategy	98
5.72	Specification< T > Interface Reference	98
5.72.1	Detailed Description	98
5.72.2	Member Function Documentation	98
5.72.2.1	and	98
5.72.2.2	isSatisfiedBy	98
5.72.2.3	not	99
5.72.2.4	or	99
5.73	Stop Class Reference	99
5.73.1	Detailed Description	100
5.73.2	Member Function Documentation	100
5.73.2.1	getStopTimes	100
5.73.3	Member Data Documentation	100
5.73.3.1	description	100
5.74	TrackingAlertFactory Class Reference	100
5.74.1	Detailed Description	100
5.74.2	Member Function Documentation	101
5.74.2.1	createAlertObserver	101
5.75	TrackingAlertObserver Class Reference	101
5.75.1	Detailed Description	102
5.75.2	Member Function Documentation	102
5.75.2.1	getSpec	102
5.75.2.2	setSpec	102
5.75.2.3	updateAlert	102
5.76	TrackingAlertRequestModel Class Reference	102
5.76.1	Detailed Description	103
5.77	TrackingAlertService Class Reference	103
5.77.1	Detailed Description	103
5.77.2	Member Function Documentation	103
5.77.2.1	createAlert	103
5.77.2.2	sendAlert	103

5.77.2.3	updateAlert	104
5.78	TrackingDelayAlert Class Reference	104
5.78.1	Detailed Description	104
5.78.2	Member Function Documentation	104
5.78.2.1	updateAlert	104
5.79	TrackingLocationAlert Class Reference	105
5.79.1	Detailed Description	105
5.79.2	Constructor & Destructor Documentation	105
5.79.2.1	TrackingLocationAlert	105
5.80	TrackingResponseModel Class Reference	105
5.80.1	Detailed Description	106
5.80.2	Member Function Documentation	106
5.80.2.1	convertJSONAlertInput	106
5.80.2.2	convertJSONVehicleInput	106
5.80.2.3	formatJSONResponse	106
5.81	TrackingServiceController Class Reference	106
5.81.1	Detailed Description	107
5.81.2	Member Function Documentation	107
5.81.2.1	addUserTrackingAlert	107
5.81.2.2	registerVehicleOnRoute	108
5.81.2.3	unregisterVehicleFromRoute	108
5.82	TransitAlertRequestModel Class Reference	108
5.82.1	Detailed Description	108
5.83	TransitAlertService Class Reference	108
5.83.1	Detailed Description	109
5.83.2	Member Function Documentation	109
5.83.2.1	createAlert	109
5.83.2.2	sendAlert	109
5.83.2.3	updateAlert	109
5.84	TransitFeed Interface Reference	110
5.84.1	Detailed Description	110
5.84.2	Member Function Documentation	110
5.84.2.1	getRoute	110
5.84.2.2	getRoutes	111
5.85	TransitInfo Class Reference	111
5.85.1	Detailed Description	111
5.85.2	Member Data Documentation	112
5.85.2.1	logo	112
5.85.2.2	transitAuthorityName	112
5.85.2.3	website	112

5.86 TransitProvider Class Reference	112
5.86.1 Detailed Description	112
5.86.2 Member Function Documentation	113
5.86.2.1 fireRouteDisruptionEvent	113
5.86.2.2 registerObserver	113
5.86.2.3 unregisterObserver	113
5.86.3 Member Data Documentation	113
5.86.3.1 name	113
5.86.3.2 providerId	113
5.87 TransitProviderObserver Interface Reference	113
5.87.1 Detailed Description	114
5.87.2 Member Function Documentation	114
5.87.2.1 handleRouteDisruptionEvent	114
5.88 TransitService Interface Reference	114
5.88.1 Detailed Description	115
5.88.2 Member Function Documentation	115
5.88.2.1 getRoute	115
5.88.2.2 getRoutes	115
5.88.2.3 getServiceURL	115
5.88.2.4 getTransitInfo	116
5.89 TransitVehicle Class Reference	116
5.89.1 Detailed Description	117
5.89.2 Member Function Documentation	117
5.89.2.1 addAlertSpecification	117
5.89.2.2 checkForAlerts	117
5.90 TransitVehicleFactory Class Reference	117
5.90.1 Detailed Description	118
5.90.2 Member Function Documentation	118
5.90.2.1 createTransitVehicle	118
5.90.2.2 getGPSTypeFromURL	118
5.90.2.3 getVehicleGPSDeviceID	118
5.91 Trip Class Reference	118
5.91.1 Detailed Description	119
5.92 TripInformation Class Reference	119
5.92.1 Detailed Description	119
5.92.2 Member Function Documentation	119
5.92.2.1 getRouteIds	119
5.92.3 Member Data Documentation	120
5.92.3.1 tripData	120
5.93 TripService Interface Reference	120

5.93.1 Detailed Description	120
5.93.2 Member Function Documentation	120
5.93.2.1 calculateTrip	120
5.94 User Class Reference	121
5.94.1 Detailed Description	122
5.94.2 Constructor & Destructor Documentation	122
5.94.2.1 User	122
5.94.3 Member Function Documentation	122
5.94.3.1 getCountryCode	122
5.94.3.2 getEmail	122
5.94.3.3 getFirstName	122
5.94.3.4 getMobile	123
5.94.3.5 getPasswordHash	123
5.94.3.6 getUserId	123
5.94.3.7 getUsername	123
5.94.3.8 getUserType	123
5.94.3.9 isForcePasswordChange	124
5.94.3.10 setCountryCode	124
5.94.3.11 setEmail	124
5.94.3.12 setFirstName	124
5.94.3.13 setForcePasswordChange	124
5.94.3.14 setMobile	124
5.94.3.15 setPasswordHash	125
5.94.3.16 setUserType	125
5.95 UserAlertExecuteStrategy Class Reference	125
5.95.1 Detailed Description	125
5.95.2 Member Data Documentation	126
5.95.2.1 alertRepository	126
5.95.2.2 userClient	126
5.96 UserAlertRequestModel Class Reference	126
5.96.1 Detailed Description	126
5.97 UserAlertService Class Reference	126
5.97.1 Detailed Description	127
5.97.2 Member Function Documentation	127
5.97.2.1 createAlert	127
5.97.2.2 sendAlert	127
5.97.2.3 updateAlert	127
5.98 UserFavoritesList Class Reference	127
5.98.1 Detailed Description	128
5.98.2 Constructor & Destructor Documentation	128

5.98.2.1	UserFavoritesList	128
5.98.3	Member Function Documentation	128
5.98.3.1	getFavoriteTransitServices	128
5.98.3.2	getUserId	128
5.98.3.3	setFavoriteTransitServices	128
5.99	UserFavoritesRepository Class Reference	129
5.99.1	Detailed Description	129
5.99.2	Member Function Documentation	129
5.99.2.1	getFavorites	129
5.99.2.2	updateFavorites	129
5.100	UserFavoritesService Interface Reference	130
5.100.1	Detailed Description	130
5.100.2	Member Function Documentation	130
5.100.2.1	readFavorites	130
5.100.2.2	saveFavorites	131
5.101	UserInfo Class Reference	131
5.101.1	Detailed Description	132
5.102	UserLoginService Interface Reference	132
5.102.1	Detailed Description	133
5.102.2	Member Function Documentation	133
5.102.2.1	createAlertSession	133
5.102.2.2	getUser	134
5.102.2.3	login	134
5.102.2.4	logout	135
5.102.2.5	resetPassword	135
5.102.2.6	resetPassword	136
5.102.2.7	sendUsername	136
5.102.2.8	sendUsername	137
5.103	UserManagementService Interface Reference	137
5.103.1	Detailed Description	138
5.103.2	Member Function Documentation	138
5.103.2.1	createUser	138
5.103.2.2	deleteUser	139
5.103.2.3	findUserByEmail	139
5.103.2.4	findUserByMobile	140
5.103.2.5	findUserByUsername	140
5.103.2.6	updateUser	141
5.104	UserRepository Class Reference	142
5.104.1	Detailed Description	142
5.104.2	Member Function Documentation	142

5.104.2.1 createUser	142
5.104.2.2 deleteUser	143
5.104.2.3 getUserByEmail	143
5.104.2.4 getUserById	144
5.104.2.5 getUserByMobile	144
5.104.2.6 getUserByUsername	145
5.104.2.7 updateUser	145
5.105UserSessionInformation Class Reference	146
5.105.1 Detailed Description	146
5.105.2 Member Data Documentation	146
5.105.2.1 userSessionToken	146
5.106UserTrackingAlertObject Class Reference	146
5.106.1 Detailed Description	147
5.106.2 Member Data Documentation	147
5.106.2.1 alertTime	147
5.106.2.2 routeID	147
5.106.2.3 scheduledTime	147
5.106.2.4 stopLocation	147
5.106.2.5 transitCoInfo	147
5.106.2.6 type	148
5.106.2.7 userContactInfo	148
5.107UserType Enum Reference	148
5.107.1 Detailed Description	148
5.108VehicleObject Class Reference	148
5.108.1 Detailed Description	149
5.108.2 Member Data Documentation	149
5.108.2.1 currentRoute	149
5.108.2.2 gpsDeviceID	149
5.108.2.3 gpsDeviceInfo	149
5.108.2.4 transitCoURL	149
5.109VehicleRepository Class Reference	149
5.109.1 Detailed Description	150
5.109.2 Member Function Documentation	150
5.109.2.1 findVehicle	150
5.109.2.2 findVehiclesByRoute	150
5.109.2.3 removeVehicle	150
5.109.2.4 updateVehicle	151

Chapter 1

Namespace Index

1.1 Packages

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

alert.client	This package contains client layer classes that facilitates the call to other modules. We use client layer to communicate and gather information from external system	13
alert.client.model	This package contains model needed for client layer classes	13
alert.controller	This package contains controller layer classes for Alert module which takes in REST request from external sources	14
alert.controller.model	This package contains model needed for controller layer	14
alert.domain	This package contains domain layer classes for Alert Module	15
alert.domain.model	This package contains entity and aggregate needed for domain layer in Alert Module	15
alert.enums	This package contains enums needed for Alert module	16
alert.service	This package contains service layer classes needed for Alert Module	16
common	This package contains common BusBuddy objects and utilities to be used by all modules	17
tracking	The Tracking Module	18
transit	The Transit Module is an interface to get Route/Fare/Detour information from a TransitProvider	19
user	This package contains the objects used by the User Module of the BusBuddy application	20

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

AbstractFeedParserTemplate	23
GTFSFeedParser	65
AlertExecuteStrategyFactory	27
AlertFactory	28
AlertInitiator	29
AlertNotificationType	29
AlertRecurringType	30
AlertRepository	31
AlertRequestController	33
AlertRequestModel	35
TrackingAlertRequestModel	102
TransitAlertRequestModel	108
UserAlertRequestModel	126
AlertResponseModel	36
AlertRunType	36
AlertService	37
TrackingAlertService	103
TransitAlertService	108
UserAlertService	126
AlertServiceFactory	39
AlertSpecification	40
AlertRangeLogic	30
DelayAlertLogic	53
AlertStatus	41
AlertType	42
AlertUserClient	42
BaseController	43
ITeamTransitServiceController	69
ITeamTripServiceController	72
CommercialTracking.CommercialTrackingHolder	52
Detour	53
Exception	
BusBuddyException	46
BusBuddyBadRequestException	44
BusBuddyConflictException	45
BusBuddyForbiddenException	47

BusBuddyInternalException	48
BusBuddyNotFoundException	49
InvalidRouteParseException	67
Fare	54
FavoriteTransitService	55
GoogleTransitServiceAPI	58
GPSLocationObject	58
GPSLocationObserver	59
GPSVehicleTracker	64
GPSLocationTracking	60
CommercialTracking	51
GPSPuller	61
GPSPusher	62
GPSPuller.GPSPullerHolder	62
GPSPusher.GPSPusherHolder	64
HashUtility	66
IAlertExecuteStrategy	66
RouteAlertExecuteStrategy	87
UserAlertExecuteStrategy	125
ISessionHandler	68
CertificateHandler	50
SessionTokenHandler	97
ITrackingService	77
TrackingServiceController	106
Location	79
MessageDeliveryUtility	80
RecurringData	84
Route	86
RouteDisruptionAlert	88
RouteDisruptionEvent	88
RouteRepository	89
Session	92
SessionRepository	95
SessionVerificationFactory	97
Specification< T >	98
Stop	99
TrackingAlertFactory	100
TrackingAlertObserver	101
TrackingDelayAlert	104
TrackingLocationAlert	105
TrackingResponseModel	105
TransitFeed	110
GoogleTransitServiceAdapter	56
PersistedTransitFeed	81
TransitInfo	111
TransitProvider	112
TransitProviderObserver	113
ITeamTransitServiceController	69
TransitService	114
ITeamTransitServiceController	69
TransitVehicle	116
BusVehicle	49
TransitVehicleFactory	117
Trip	118
TripInformation	119

TripService	120
ITeamTripServiceController	72
User	121
UserFavoritesList	127
UserFavoritesRepository	129
UserFavoritesService	130
ITeamUserFavoritesService	72
UserInfo	131
UserLoginService	132
ITeamUserLoginService	73
UserManagementService	137
ITeamUserManagementService	76
UserRepository	142
UserSessionInformation	146
UserTrackingAlertObject	146
UserType	148
VehicleObject	148
VehicleRepository	149
Serializable	
Alert	25
OneTimeAlert	81
RecurringAlert	83
Specification	
RouteSpecification	91

Chapter 3

Class Index

3.1 Class List

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

AbstractFeedParserTemplate	A Template Method pattern to allow for the import of data from different TransitProviders in potentially different formats	23
Alert	This is a base Alert Model that has most of the common information about an Alert	25
AlertExecuteStrategyFactory	This factory is going to pick the best available strategy to execute the alert so that the user can be notified	27
AlertFactory	An alert factory that can create different types of semi-populated alert models depending upon the information	28
AlertInitiator	A list of enums corresponding with modules that initiates the call	29
AlertNotificationType	Notification type of alert,	29
AlertRangeLogic	Alert Range Logic implements the business logic to determine if a vehicle is within a range where an alert needs to be sent to a user who has registered for tracking alerts	30
AlertRecurringType	Represents different recurring type of alert i.e., Yearly, Monthly, Weekly or Daily	30
AlertRepository	A Repository that handles the persistent behavior of the Alert aggregate	31
AlertRequestController	This is a front facing controller that takes in request from other module via REST call and returns a JSON representation data	33
AlertRequestModel	A base model that contains bare minimum information about the alert request	35
AlertResponseModel	This is a basic alert response model that is returned for every alert related requested	36
AlertRunType	Represents the type of alert depending upon the run type whether to run once or to run in a fixed recurring manner	36
AlertService	Alert Service is a base class that is extended by other module specific services	37
AlertServiceFactory	AlertServiceFactory initializes appropriate alert service depending upon the parameter being passed	39

AlertSpecification	Interface for Alert Specifications which contain the business logic used to determine if an alert should be triggered for a vehicle	40
AlertStatus	Represents the status of an alert depending upon its state in its life cycle	41
AlertType	Enumeration of the alert types recognized by bus buddy	42
AlertUserClient	Client layer will handle all the responsibility of sending request to other modules or external sources	42
BaseController	This is a base class to be extended by each of the controller classes	43
BusBuddyBadRequestException	This exception object represents internal errors which may occur as a result of an error in the client's request	44
BusBuddyConflictException	This exception object is thrown when a request would create a conflict which violates constraints set within the system	45
BusBuddyException	This exception object is an abstract base class	46
BusBuddyForbiddenException	This exception object represents internal errors which may occur as a result of attempts to access a resource without authorization	47
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	48
BusBuddyNotFoundException	This exception object represents the error that occurs when a resource cannot be found	49
BusVehicle	Bus Vehicle is a concrete implementation of the abstract Transit Vehicle	49
CertificateHandler	A concrete strategy implementation of ISessionHandler that can verify the certificate token being passed	50
CommercialTracking	Implements Subject GPSLocationTracking for retrieving GPS location updates from outside commercial tracking services	51
CommercialTracking.CommercialTrackingHolder	Commercial Tracking Holder is loaded on the first execution of CommercialTracking.getInstance() or the first access to CommercialTracking.INSTANCE , not before (lazy instantiation)	52
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	53
Detour	A disruption in service due to an unexpected event	53
Fare	An immutable Value Object representing the cost, or 'fare,' required to ride a TransitVehicle on a particular Route	54
FavoriteTransitService	This class is a single transit service in a user's list of favorites	55
GoogleTransitServiceAdapter	An Adapter Class to allow a GoogleTransitServiceAPI service to appear as a TransitService . .	56
GoogleTransitServiceAPI	A client to Google's Maps API	58
GPSLocationObject	GPS Location is a value object used for GPS coordinates and the time of the last update . . .	58
GPSLocationObserver	Observer Pattern - Observer interface for GPS location tracking	59

GPSLocationTracking	
GPSLocationTracking - interface Subject of the Observer Pattern Defines methods for an observer GPS Device to register and receive updates on vehicle location	60
GPSPuller	
GPS Puller is a concrete implementation of GPSLocationTracking for obtaining coordinates directly from a GPS device installed in a registered vehicle	61
GPSPuller.GPSPullerHolder	
GPS Puller Holder is loaded on the first execution of GPSPuller.getInstance() or the first access to GPSPuller.INSTANCE , not before (lazy instantiation)	62
GPSPusher	
Implements Subject GPSLocationTracking for retrieving GPS location updates from registered vehicles	62
GPSPusher.GPSPusherHolder	
GPS Pusher Holder is loaded on the first execution of GPSPusher.getInstance() or the first access to GPSPusher.INSTANCE , not before (lazy instantiation)	64
GPSVehicleTracker	
Implementation of the Observer, update transit vehicle GPS location and time GPS Vehicle Tracker gets the state as new GPS coordinates and time from GPS Location Tracking and updates the transit vehicle	64
GTFSFeedParser	
A AbstractFeedParserTemplate implementation designed to parse GTFS format ZIP files into Routes	65
HashUtility	
This is a utility class to handle secure hashes	66
IAlertExecuteStrategy	
An interface for executing different type of alert based on alert type	66
InvalidRouteParseException	
An InvalidRouteParseException indicates an invalid batch of parsed Routes has been detected	67
ISessionHandler	
An interface to verify the validity of encrypted token being passed	68
ITeamTransitServiceController	
The iTeam implementation of the TransitService that exposes Transit data via a REST Service	69
ITeamTripServiceController	
An iTeam implementation of the TripService that exposes Trip data via a REST Service	72
ITeamUserFavoritesService	
This is the iTeam's implementation of UserFavoritesService	72
ITeamUserLoginService	
This is the iTeam's implementation of UserLoginService	73
ITeamUserManagementService	
This is the iTeam's implementation of UserManagementService	76
ITrackingService	
Interface for the Tracking Service Controller	77
Location	
An immutable Value Object representing a physical point on the geographic coordinate system	79
MessageDeliveryUtility	
This is a utility class to handle message delivery, such as through email or SMS	80
OneTimeAlert	
This is a model of alert that is to be run one time only	81
PersistedTransitFeed	
An implementation of the TransitFeed interface that communicates with a RouteRepository to retrieve its data	81
RecurringAlert	
This is a model of alert that is to be run multiple times	83
RecurringData	
This model holds the information about the date and time the alert needs to run	84
Route	
A Route is a TransitVehicle path of travel, or a "Line," as referred to by a TransitProvider	86

RouteAlertExecuteStrategy	
A concrete implementation of IAlertExecuteStrategy that handles executing alert related to route changes	87
RouteDisruptionAlert	
An Alert indicating a disruption of normal Route availability or scheduling	88
RouteDisruptionEvent	
An event indicating a disruption in normal Route scheduling or service	88
RouteRepository	
A Repository Pattern supporting lifecycle operations of Routes , such as Read, Save, Delete, and Query functionality	89
RouteSpecification	
A Specification Pattern class for validating a Route	91
Session	
This class represents a single session for a user of the system, and all of the state data associated with that session	92
SessionRepository	
This class is responsible for handling database access for Sessions, and to construct, persist, and retrieve Session objects	95
SessionTokenHandler	
A concrete strategy implementation of ISessionHandler that can verify the session token being passed	97
SessionVerificationFactory	
A factory that picks the appropriate strategy ISessionHandler to verify the encrypted token	97
Specification< T >	
A Generic Specification to be used for chaining business validation rules together	98
Stop	
A point on a Route in which a TransitVehicle will stop to pick up and drop off passengers	99
TrackingAlertFactory	
The Alert Factory handles the creation of a user alert	100
TrackingAlertObserver	
Abstract class defining the methods for the tracking alert observer	101
TrackingAlertRequestModel	
This model is a JSON representation of the request from Tracking module	102
TrackingAlertService	
Extends AlertService and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy	103
TrackingDelayAlert	
Tracking Alert Observer implements the abstract tracking alert observer and provides the method to actually send an alert to a registered user that their bus is approaching their stop	104
TrackingLocationAlert	
Concrete implementation of the tracking alert observer	105
TrackingResponseModel	
This is a basic tracking response model that is returned for every tracking related request	105
TrackingServiceController	
Tracking service controller is the concrete implementation of the tracking service interface	106
TransitAlertRequestModel	
This model is a JSON representation of the request from Transit module	108
TransitAlertService	
Extends AlertService and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy	108
TransitFeed	
A TransitFeed is an abstraction over a service or set of services that provide information about Routes	110
TransitInfo	
An immutable Value Object describing metadata about a TransitService	111
TransitProvider	
A TransitProvider is a description of a company or organization that is the producer of public transportation services	112

TransitProviderObserver	
An asynchronous update interface for receiving notifications about TransitProvider Route disruptions	113
TransitService	
The TransitService is an interface to get Route/Fare/Detour information from a TransitProvider	114
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	116
TransitVehicleFactory	
Transit Vehicle Factory encapsulates the complexity of creating a new vehicle	117
Trip	
A Trip is considered an ordered collection of Routes going from a starting point to an ending point	118
TripInformation	
This model stores the information about a trip as a value object	119
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	120
User	
This class represents a single user of the system, and all of the state data associated with that user	121
UserAlertExecuteStrategy	
A concrete implementation of IAlertExecuteStrategy that handles executing alert related to user	125
UserAlertRequestModel	
This model is a JSON representation of the request from User module	126
UserAlertService	
Extends AlertService and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy	126
UserFavoritesList	
This class ties a userId to that user's ordered list of favorites	127
UserFavoritesRepository	
This class is responsible for handling database access for favorites, and to persist and retrieve UserFavoritesList objects	129
UserFavoritesService	
This is the generic BusBuddy UserFavoritesService interface	130
UserInformation	
UserInformation contains the user related data that we get from User Module	131
UserLoginService	
This is the generic BusBuddy UserLoginService interface	132
UserManagementService	
This is the generic BusBuddy UserManagementService interface	137
UserRepository	
This class is responsible for handling database access for User objects, and to construct, persist, and retrieve User objects	142
UserSessionInformation	
A model that stores all the information needed to call user module about user information	146
UserTrackingAlertObject	
User tracking alert information obtained from the user interface when the user registers for an alert	146
UserType	
This is an enumeration of the different statuses that a user can be assigned	148
VehicleObject	
Value Object containing vehicle information obtained when the user registers a vehicle using the user interface	148
VehicleRepository	
Repository for information on vehicles registered on a route	149

Chapter 4

Namespace Documentation

4.1 Package alert.client

This package contains client layer classes that facilitates the call to other modules. We use client layer to communicate and gather information from external system.

Packages

- package [model](#)
This package contains model needed for client layer classes.

Classes

- class [AlertUserClient](#)
Client layer will handle all the responsibility of sending request to other modules or external sources.

4.1.1 Detailed Description

This package contains client layer classes that facilitates the call to other modules. We use client layer to communicate and gather information from external system. This layer contains method that can make REST call to other modules and get back appropriate response. Currently, it is only making REST calls, but we can add new classes to make SOAP or any other type of external request. Hence, it abstracts the interaction with outer system and separates that concern to itself.

4.2 Package alert.client.model

This package contains model needed for client layer classes.

Classes

- class [UserInformation](#)
[UserInformation](#) contains the user related data that we get from User Module.

4.2.1 Detailed Description

This package contains model needed for client layer classes.

4.3 Package alert.controller

This package contains controller layer classes for Alert module which takes in REST request from external sources.

Packages

- package [model](#)

This package contains model needed for controller layer.

Classes

- class [AlertRequestController](#)

This is a front facing controller that takes in request from other module via REST call and returns a JSON representation data.

- class [CertificateHandler](#)

A concrete strategy implementation of [ISessionHandler](#) that can verify the certificate token being passed.

- interface [ISessionHandler](#)

An interface to verify the validity of encrypted token being passed.

- class [SessionTokenHandler](#)

A concrete strategy implementation of [ISessionHandler](#) that can verify the session token being passed.

- class [SessionVerificationFactory](#)

A factory that picks the appropriate strategy [ISessionHandler](#) to verify the encrypted token.

4.3.1 Detailed Description

This package contains controller layer classes for Alert module which takes in REST request from external sources. Controller acts like a facade to most of the modules as the methods are mapped to a certain URI. Other modules can call the REST interface and controller will handle the request. Also, controller handles the session verification depending upon type of encrypted token passed and provides the authorization and authentication. This is achieved by using factory and strategy pattern.

4.4 Package alert.controller.model

This package contains model needed for controller layer.

Classes

- class [AlertRequestModel](#)

A base model that contains bare minimum information about the alert request.

- class [AlertResponseModel](#)

This is a basic alert response model that is returned for every alert related requested.

- class [TrackingAlertRequestModel](#)

This model is a JSON representation of the request from Tracking module.

- class [TransitAlertRequestModel](#)

This model is a JSON representation of the request from Transit module.

- class [UserAlertRequestModel](#)

This model is a JSON representation of the request from User module.

4.4.1 Detailed Description

This package contains model needed for controller layer.

4.5 Package alert.domain

This package contains domain layer classes for Alert Module.

Packages

- package [model](#)

This package contains entity and aggregate needed for domain layer in [Alert](#) Module.

Classes

- class [AlertFactory](#)

An alert factory that can create different types of semi-populated alert models depending upon the information.

- class [AlertRepository](#)

A Repository that handles the persistent behavior of the [Alert](#) aggregate.

4.5.1 Detailed Description

This package contains domain layer classes for Alert Module. Domain layer classes contains factory and repository that are responsible for handling the models lifecycle. Factory pattern is used to create different types of alert depending upon the parameter.

4.6 Package alert.domain.model

This package contains entity and aggregate needed for domain layer in [Alert](#) Module.

Classes

- class [Alert](#)

This is a base [Alert](#) Model that has most of the common information about an [Alert](#).

- class [OneTimeAlert](#)

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

- class [RecurringAlert](#)

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

- class [RecurringData](#)

This model holds the information about the date and time the alert needs to run.

- class [TripInformation](#)

This model stores the information about a trip as a value object.

- class [UserSessionInformation](#)

A model that stores all the information needed to call user module about user information.

4.6.1 Detailed Description

This package contains entity and aggregate needed for domain layer in [Alert](#) Module. Domain layer models are database representation of the alert model. It contains a root aggregate through which we can enforce business rules and get access to information about its child objects. In the future, if we have to add more fields for alert we can probably implement bridge pattern to provide more flexible structure.

4.7 Package alert.enums

This package contains enums needed for Alert module.

Classes

- enum [AlertInitiator](#)
A list of enums corresponding with modules that initiates the call.
- enum [AlertNotificationType](#)
Notification type of alert,.
- enum [AlertRecurringType](#)
Represents different recurring type of alert i.e., Yearly, Monthly, Weekly or Daily.
- enum [AlertRunType](#)
Represents the type of alert depending upon the run type whether to run once or to run in a fixed recurring manner.
- enum [AlertStatus](#)
Represents the status of an alert depending upon its state in its life cycle.

4.7.1 Detailed Description

This package contains enums needed for Alert module.

4.8 Package alert.service

This package contains service layer classes needed for Alert Module.

Classes

- class [AlertExecuteStrategyFactory](#)
This factory is going to pick the best available strategy to execute the alert so that the user can be notified.
- class [AlertService](#)
Alert Service is a base class that is extended by other module specific services.
- class [AlertServiceFactory](#)
[AlertServiceFactory](#) initializes appropriate alert service depending upon the parameter being passed.
- interface [IAlertExecuteStrategy](#)
An interface for executing different type of alert based on alert type.
- class [RouteAlertExecuteStrategy](#)
A concrete implementation of [IAlertExecuteStrategy](#) that handles executing alert related to route changes.
- class [TrackingAlertService](#)
Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.
- class [TransitAlertService](#)

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

- class [UserAlertExecuteStrategy](#)

A concrete implementation of [IAlertExecuteStrategy](#) that handles executing alert related to user.

- class [UserAlertService](#)

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

4.8.1 Detailed Description

This package contains service layer classes needed for Alert Module. Service layer handles the workflow and orchestrates the calls to process the request. Depending upon the modules initiating the call, service implementation can invoke factory method in domain layer to create alert and can manipulate its properties. Also service layer can execute the alert using different strategy. Service layer uses factory and strategy patterns to select appropriate service implementation and strategy type.

4.9 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 [BusBuddyConflictException](#)

This exception object is thrown when a request would create a conflict which violates constraints set within the system.

- 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.

4.9.1 Detailed Description

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

4.10 Package tracking

The Tracking Module.

Classes

- class [AlertRangeLogic](#)
Alert Range Logic implements the business logic to determine if a vehicle is within a range where an alert needs to be sent to a user who has registered for tracking alerts.
- interface [AlertSpecification](#)
Interface for Alert Specifications which contain the business logic used to determine if an alert should be triggered for a vehicle.
- enum [AlertType](#)
Enumeration of the alert types recognized by bus buddy.
- class [BusVehicle](#)
Bus Vehicle is a concrete implementation of the abstract Transit Vehicle.
- class [CommercialTracking](#)
Implements Subject [GPSLocationTracking](#) 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 [GPSLocationObject](#)
GPS Location is a value object used for GPS coordinates and the time of the last update.
- 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 [GPSLocationTracking](#) for obtaining coordinates directly from a GPS device installed in a registered vehicle.
- class [GPSPusher](#)
Implements Subject [GPSLocationTracking](#) for retrieving GPS location updates from registered vehicles.
- class [GPSVehicleTracker](#)
Implementation of the Observer, update transit vehicle GPS location and time GPS Vehicle Tracker gets the state as new GPS coordinates and time from GPS Location Tracking and updates the transit vehicle.
- interface [ITrackingService](#)
Interface for the Tracking Service Controller.
- class [TrackingAlertFactory](#)
The Alert Factory handles the creation of a user alert.
- class [TrackingAlertObserver](#)
Abstract class defining the methods for the tracking alert observer.
- class [TrackingDelayAlert](#)
Tracking Alert Observer implements the abstract tracking alert observer and provides the method to actually send an alert to a registered user that their bus is approaching their stop.
- class [TrackingLocationAlert](#)
Concrete implementation of the tracking alert observer.
- class [TrackingResponseModel](#)
This is a basic tracking response model that is returned for every tracking related request.
- class [TrackingServiceController](#)
Tracking service controller is the concrete implementation of the tracking service interface.

- 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 [UserTrackingAlertObject](#)
User tracking alert information obtained from the user interface when the user registers for an alert.
- 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.

4.10.1 Detailed Description

The Tracking Module. The Tracking Module handles the tracking GPS data from the transit vehicles and initiating alerts to users subscribed to route based messages and triggers transit vehicle delay alerts. The transit vehicle location data is also available to the Maps Module to place the bus icon at the correct position on maps.

4.11 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 [GoogleTransitServiceAPI](#) 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 detected.
- class [ITeamTransitServiceController](#)
The iTeam implementation of the [TransitService](#) that exposes Transit data via a REST Service.
- class [ITeamTripServiceController](#)
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.*

 - class [RouteDisruptionEvent](#)

An event indicating a disruption in normal [Route](#) scheduling or service.
 - 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 [TransitProviderObserver](#)

An asynchronous update interface for receiving notifications about [TransitProvider](#) [Route](#) disruptions.
 - 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](#).

4.11.1 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](#).

4.12 Package user

This package contains the objects used by the [User](#) Module of the BusBuddy application.

Classes

- class [FavoriteTransitService](#)

This class is a single transit service in a user's list of favorites.
- class [ITeamUserFavoritesService](#)

This is the iTeam's implementation of [UserFavoritesService](#).
- class [ITeamUserLoginService](#)

This is the iTeam's implementation of [UserLoginService](#).
- class [ITeamUserManagementService](#)

This is the iTeam's implementation of [UserManagementService](#).
- 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.

- class [UserFavoritesList](#)

This class ties a [userId](#) to that user's ordered list of favorites.

- class [UserFavoritesRepository](#)

This class is responsible for handling database access for favorites, and to persist and retrieve [UserFavoritesList](#) objects.

- interface [UserFavoritesService](#)

This is the generic BusBuddy [UserFavoritesService](#) interface.

- 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](#)

This is an enumeration of the different statuses that a user can be assigned.

4.12.1 Detailed Description

This package contains the objects used by the [User](#) Module of the BusBuddy application.

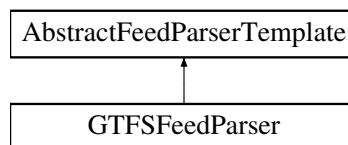
Chapter 5

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](#) method. This method initiates the parsing/transformation process.
2. The [start](#) method calls the method [loadFeed](#) to establish the [InputStream](#).
3. The [InputStream](#) returned by [loadFeed](#) is passed into the abstract [parseFeed](#) 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](#) 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

<i>location</i>	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

<i>feed</i>	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

<i>routes</i>	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

<i>location</i>	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

<i>route</i>	the route
--------------	-----------

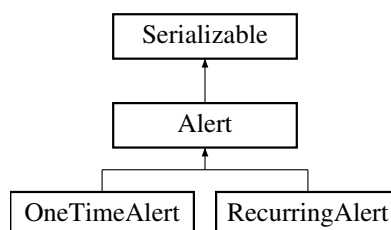
Returns

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](#))
- [UserInformation](#) **getUserInformation** ()
- void **setUserInformation** ([UserInformation](#) [userInformation](#))
- [TripInformation](#) **getTripInformation** ()
- void **setTripInformation** ([TripInformation](#) [tripInformation](#))
- [AlertInitiator](#) **getAlertInitiator** ()
- void **setAlertInitiator** ([AlertInitiator](#) [alertInitiator](#))

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.
- [AlertInitiator](#) [alertInitiator](#)
Module that initiated or created this alert.
- [UserInformation](#) [userInformation](#)
Information about the user if the alert is created by a dedicated user.
- [TripInformation](#) [tripInformation](#)
A HashMap of.

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 Data Documentation

5.2.2.1 **AlertInitiator** alertInitiator [private]

Module that initiated or created this alert.

[AlertInitiator](#)

5.2.2.2 **AlertRunType** alertRunType [private]

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

Value is defined by [AlertRunType](#)

5.2.2.3 **AlertNotificationType** alertType [private]

Notification type of alert.

Depends upon the value as specified in [AlertNotificationType](#)

5.2.2.4 **AlertStatus** status [private]

current status of the alert.

The value depends upon [AlertStatus](#) enum.

5.2.2.5 **UserInfo** userInfo [private]

Information about the user if the alert is created by a dedicated user.

For any alerts generated by Transit or Tracking module, the user can either be admin of that module or the admin of the bus service provider. [UserInfo](#)

5.3 AlertExecuteStrategyFactory Class Reference

This factory is going to pick the best available strategy to execute the alert so that the user can be notified.

Static Public Member Functions

- static [IAlertExecuteStrategy](#) **getAlertService** ([Alert](#) alertModel)

Based on the alertModel information, this method is going to pick the best strategy to send the alert.

5.3.1 Detailed Description

This factory is going to pick the best available strategy to execute the alert so that the user can be notified.

5.3.2 Member Function Documentation

5.3.2.1 static `IAAlertExecuteStrategy getAlertService (Alert alertModel)` [static]

Based on the alertModel information, this method is going to pick the best strategy to send the alert.

Parameters

<i>alertModel</i>	Alert model which is used to pick the best strategy.
-------------------	--

Returns

An implementation of [IAAlertExecuteStrategy](#) that can send the notification message for given alert.

5.4 AlertFactory Class Reference

An alert factory that can create different types of semi-populated alert models depending upon the information.

Public Member Functions

- [Alert createAlert \(AlertRunType runType\)](#)
Creates an alert model depending upon the run Type.
- [Alert createAlert \(AlertRecurringType recurringType\)](#)
Created alert model depending upon the recurringType.

5.4.1 Detailed Description

An alert factory that can create different types of semi-populated alert models depending upon the information.

Any additional create method can be created during implementation phase.

5.4.2 Member Function Documentation

5.4.2.1 `Alert createAlert (AlertRunType runType)`

Creates an alert model depending upon the run Type.

Parameters

<i>runType</i>	AlertRunType enum.
----------------	------------------------------------

Returns

Either a Onetime or Recurring alert Model.

5.4.2.2 `Alert createAlert (AlertRecurringType recurringType)`

Created alert model depending upon the recurringType.

Parameters

<i>recurringType</i>	AlertRecurringType enum
----------------------	---

Returns

A recurring alert Model

5.5 AlertInitiator Enum Reference

A list of enums corresponding with modules that initiates the call.

Public Attributes

- [UserModule](#)
User module initiates the call.
- [TrackingModule](#)
Tracking module initiates the call.
- [TransitModule](#)
Transit module initiates the call.

5.5.1 Detailed Description

A list of enums corresponding with modules that initiates the call.

5.6 AlertNotificationType Enum Reference

Notification type of alert,.

Public Attributes

- [PlannedDisruption](#)
A planned alert that is to be run in the future.
- [UnplannedDisruption](#)
A sudden change in route or plan can trigger this alert.
- [ScheduleInformation](#)
A general type of alert where a user is to be notified of their schedule in a timely fashion.

5.6.1 Detailed Description

Notification type of alert,.

5.6.2 Member Data Documentation

5.6.2.1 PlannedDisruption

A planned alert that is to be run in the future.

Usually this is provided by Transit Module when the bus route is going to be changed in near future.

5.6.2.2 ScheduleInformation

A general type of alert where a user is to be notified of their schedule in a timely fashion.

Generally, user module creates this type of alert.

5.6.2.3 UnplannedDisruption

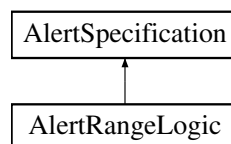
A sudden change in route or plan can trigger this alert.

Usually this type of alert is executed right away. This type of alert is provided by either Transit or Tracking module.

5.7 AlertRangeLogic Class Reference

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

Inheritance diagram for AlertRangeLogic:



Public Member Functions

- boolean **inAlertRange** (GPSLocationObject vehicleLocation)

Provides the logic necessary to determine from the GPS coordinates if the registered user should be alerted.

5.7.1 Detailed Description

Alert Range Logic implements the business logic to determine if a vehicle is within a range 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. GPS coordinates are regularly updated, but not necessarily in real-time. BusBuddy needs to notify the user before the bus reaches the stop. An alert range is a distance range before the stop when the user should be notified.

5.8 AlertRecurringType Enum Reference

Represents different recurring type of alert i.e., Yearly, Monthly, Weekly or Daily.

Public Attributes

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

5.8.1 Detailed Description

Represents different recurring type of alert i.e., Yearly, Monthly, Weekly or Daily.

5.9 AlertRepository Class Reference

A Repository that handles the persistent behavior of the [Alert](#) aggregate.

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 routeld)
This method fetches all the alerts that is linked to the routeld.
- List< [Alert](#) > [getAlertByUserId](#) (String userId)
This method fetches all the alerts that is tied to a user.

5.9.1 Detailed Description

A Repository that handles the persistent behavior of the [Alert](#) aggregate.

It has methods that can alter the lifecycle of the aggregate.

5.9.2 Member Function Documentation

5.9.2.1 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

<i>alertModel</i>	The Alert model that is to be deleted.
-------------------	--

Returns

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

5.9.2.2 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

<i>dateTimeToFetch</i>	DateTime when the alert is supposed to run.
<i>offsetMinute</i>	An int value that is used to fetch alerts within that minute in future.

Returns

Returns a list of [Alert](#) models that is to be run in next couple of minutes (offsetMinute) of given date time.

5.9.2.3 List<Alert> getAlertByRoute (String *routeId*)

This method fetches all the alerts that is linked to the routeId.

Parameters

<i>routeId</i>	The route ID that is being affected.
----------------	--------------------------------------

Returns

A list of [Alert](#) models.

5.9.2.4 List<Alert> getAlertByUserId (String *userId*)

This method fetches all the alerts that is tied to a user.

Parameters

<i>userId</i>	userId that is being affected
---------------	-------------------------------

Returns

A list of [Alert](#) models.

5.9.2.5 Alert saveAlert (Alert *alertModel*)

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

Parameters

<i>alertModel</i>	Alert model to be saved
-------------------	---

Returns

The saved object with updated property.

Save the alert via Hibernate.

5.9.2.6 Alert updateAlert (Alert alertModel)

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

Precondition

the alert must exist in the system.

Parameters

<i>alertModel</i>	A Alert model that needs to be updated
-------------------	--

Returns

Returns the updated [Alert](#) model back to the method that is calling.

5.10 AlertRequestController Class Reference

This is a front facing controller that takes in request from other module via REST call and returns a JSON representation data.

Public Member Functions

- [AlertResponseModel](#) `processUserAlertRequest` ([UserAlertRequestModel](#) userAlertRequest, String encryptedToken)
Processes Alert manipulation request from User module.
- [AlertResponseModel](#) `processTransitAlertRequest` ([TransitAlertRequestModel](#) transitAlertRequest, String encryptedToken)
Processes Alert manipulation request from Transit module.
- [AlertResponseModel](#) `processTrackingAlertRequest` ([TrackingAlertRequestModel](#) trackingAlertRequest, String encryptedToken)
Processes Alert manipulation request from Tracking module.

Private Member Functions

- void `verifySession` ([AlertRequestModel](#) requestModel, String encryptedToken)
Method to verify the validity of session token being passed.

Private Attributes

- [SessionVerificationFactory](#) `sessionVerificationFactory`
Session Verification Factory picks the appropriate type of session verification strategy depending upon which module is initiating the call.

5.10.1 Detailed Description

This is a front facing controller that takes in request from other module via REST call and returns a JSON representation data.

User module calls the alert module to create new alert, delete alert, update alert or to get a list of alert for a user. Transit module can call alert module to create, delete and update a new alert. Tracking module can call alert module to create a new alert. As far as public facing end points go, we have implemented one endpoint for each

module. During implementation there might be different GET, POST, PUT and DELETE methods. But for now, we are putting only one method per module as a placeholder assuming that it can take different actions depending upon the information in the `AlertRequestModel`.

5.10.2 Member Function Documentation

5.10.2.1 `AlertResponseModel processTrackingAlertRequest (TrackingAlertRequestModel trackingAlertRequest, String encryptedToken)`

Processes Alert manipulation request from Tracking module.

Parameters

<i>trackingAlert-Request</i>	{
------------------------------	---

See Also

`TrackingAlertRequestModel`}

Parameters

<i>encryptedToken</i>	An encrypted token that can be validated.
-----------------------	---

Returns

An [AlertResponseModel](#) that has the success/error information.

5.10.2.2 `AlertResponseModel processTransitAlertRequest (TransitAlertRequestModel transitAlertRequest, String encryptedToken)`

Processes Alert manipulation request from Transit module.

Parameters

<i>transitAlert-Request</i>	{
-----------------------------	---

See Also

`TransitAlertRequestModel`}

Parameters

<i>encryptedToken</i>	<ul style="list-style-type: none"> An encrypted token that can be validated.
-----------------------	---

Returns

An [AlertResponseModel](#) that has the success/error information.

5.10.2.3 `AlertResponseModel processUserAlertRequest (UserAlertRequestModel userAlertRequest, String encryptedToken)`

Processes Alert manipulation request from User module.

Parameters

<i>userAlert-Request</i>	An alertRequestModel that has the necessary information regarding creation of an alert. {
--------------------------	---

See Also

UserAlertRequestModel}

Parameters

<i>encryptedToken</i>	An encrypted token that can be validated.
-----------------------	---

Returns

An [AlertResponseModel](#) that has the success/error information.

5.10.2.4 `void verifySession (AlertRequestModel requestModel, String encryptedToken) [private]`

Method to verify the validity of session token being passed.

If token verification fails, it automatically throws an error the request is terminated.

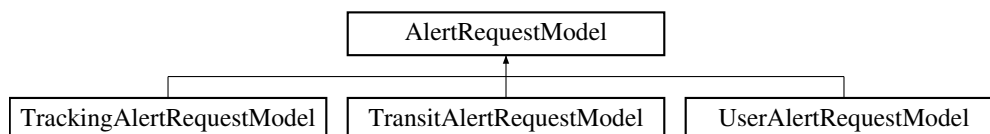
Parameters

<i>requestModel</i>	An object representation of JSON that has the request.
<i>encryptedToken</i>	An encrypted token that can be validated.

5.11 AlertRequestModel Class Reference

A base model that contains bare minimum information about the alert request.

Inheritance diagram for AlertRequestModel:



Public Member Functions

- [AlertInitiator](#) **getAlertInitiator** ()
- void **setAlertInitiator** ([AlertInitiator](#) alertInitiator)

Private Attributes

- [AlertInitiator](#) alertInitiator

Alert Initiator {.

5.11.1 Detailed Description

A base model that contains bare minimum information about the alert request.

Any model that extends this class can add additional data that is needed.

5.11.2 Member Data Documentation

5.11.2.1 **AlertInitiator** alertInitiator [private]

Alert Initiator {.

See Also

AlertInitiator}.

5.12 AlertResponseModel Class Reference

This is a basic alert response model that is returned for every alert related requested.

Private Attributes

- String [status](#)
Status message for the alert process job.
- String [errorMessage](#)
Any error message if the alert request fails.

5.12.1 Detailed Description

This is a basic alert response model that is returned for every alert related requested.

Additional fields can be added as needed during implementation phase.

5.13 AlertRunType Enum Reference

Represents the type of alert depending upon the run type whether to run once or to run in a fixed recurring manner.

Public Attributes

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

5.13.1 Detailed Description

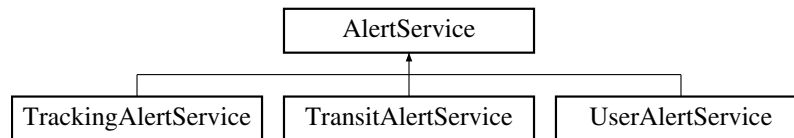
Represents the type of alert depending upon the run type whether to run once or to run in a fixed recurring manner.

OneTime Alert runs only once and is then deactivated. Recurring Alert runs in every certain amount of time depending upon the parameter defined.

5.14 AlertService Class Reference

Alert Service is a base class that is extended by other module specific services.

Inheritance diagram for AlertService:



Public Member Functions

- [AlertResponseModel saveAlert](#) ([Alert](#) alertModel)
Takes in an [Alert](#) model to persist it into db.
- [AlertResponseModel deleteAlert](#) ([Alert](#) alertModel)
Takes in an [Alert](#) Model for deletion.

Protected Member Functions

- abstract [AlertResponseModel createAlert](#) ([AlertRequestModel](#) requestModel)
An abstract class that must be implemented by derived classes to create a new alert model.
- abstract [AlertResponseModel updateAlert](#) ([Alert](#) alertModel)
Takes in a final [Alert](#) Model that needs to be updated in db.
- abstract boolean [sendAlert](#) ()
Finds all the alerts that are initiated by this service and calls [AlertExecuteStrategyFactory](#) to find appropriate strategy to send each type of alert.

Package Attributes

- [AlertExecuteStrategyFactory alertExecuteStrategyFactory](#)
{
- [AlertRepository alertRepository](#)
{

5.14.1 Detailed Description

Alert Service is a base class that is extended by other module specific services.

This abstract class defines signature for basic CRUD operation.

5.14.2 Member Function Documentation

5.14.2.1 `abstract AlertResponseModel createAlert (AlertRequestModel requestModel)` [protected], [pure virtual]

An abstract class that must be implemented by derived classes to create a new alert model.

Precondition

An [alertRequest](#) model must be supplied that must have necessary information to create an alert

Parameters

<i>requestModel</i>	A AlertResponseModel that has necessary information to create a new alert model.
---------------------	--

Returns

Returns an [AlertReponseModel](#) that contains information needed by the caller.

Implemented in [TrackingAlertService](#), [TransitAlertService](#), and [UserAlertService](#).

5.14.2.2 [AlertResponseModel](#) deleteAlert ([Alert](#) *alertModel*)

Takes in an Alert Model for deletion.

Parameters

<i>alertModel</i>	A valid Alert model
-------------------	-------------------------------------

Returns

An [AlertResponseModel](#) that has the success or error message.

5.14.2.3 [AlertResponseModel](#) saveAlert ([Alert](#) *alertModel*)

Takes in an [Alert](#) model to persist it into db.

Parameters

<i>alertModel</i>	A valid Alert model
-------------------	-------------------------------------

Returns

An [AlertResponseModel](#) that has the success or error message.

5.14.2.4 abstract boolean sendAlert () [protected],[pure virtual]

Finds all the alerts that are initiated by this service and calls [AlertExecuteStrategyFactory](#) to find appropriate strategy to send each type of alert.

1. Get all the alerts created by this service/module.
2. Loop through each one of them and call appropriate strategy via StrategyFactory
3. Use execute method in strategy to push notification

Returns

Implemented in [TrackingAlertService](#), [TransitAlertService](#), and [UserAlertService](#).

5.14.2.5 abstract [AlertResponseModel](#) updateAlert ([Alert](#) *alertModel*) [protected],[pure virtual]

Takes in a final Alert Model that needs to be updated in db.

Parameters

<i>alertModel</i>	A valid Alert model
-------------------	-------------------------------------

Returns

An [AlertResponseModel](#) that has the success or error message.

Implemented in [TrackingAlertService](#), [TransitAlertService](#), and [UserAlertService](#).

5.14.3 Member Data Documentation

5.14.3.1 [AlertExecuteStrategyFactory](#) alertExecuteStrategyFactory [package]

{

See Also

[AlertExecuteStrategyFactory](#)}. This is autowired via Spring Framework.

5.14.3.2 [AlertRepository](#) alertRepository [package]

{

See Also

[AlertRepository](#)}. This is autowired via Spring Framework.

5.15 AlertServiceFactory Class Reference

[AlertServiceFactory](#) initializes appropriate alert service depending upon the parameter being passed.

Public Member Functions

- [AlertService](#) [getAlertService](#) ([AlertRequestModel](#) requestModel)
This method returns a concrete implementation of the alert service.

Private Attributes

- [UserAlertService](#) userAlertService
{
- [TrackingAlertService](#) trackingAlertService
{
- [TransitAlertService](#) transitAlertService
{

5.15.1 Detailed Description

[AlertServiceFactory](#) initializes appropriate alert service depending upon the parameter being passed.

Currently, each module is aligned to each service but in the future it could change such that 2 module can use same service.

5.15.2 Member Function Documentation

5.15.2.1 `AlertService` `getAlertService (AlertRequestModel requestModel)`

This method returns a concrete implementation of the alert service.

Parameters

<i>requestModel</i>	A JSON representation of the request from controller.
---------------------	---

Returns

An extension of [AlertService](#).

thrown an error if applicable.

5.15.3 Member Data Documentation

5.15.3.1 `TrackingAlertService` `trackingAlertService` [private]

{

See Also

[TrackingAlertService](#)}. This is autowired via Spring Framework.

5.15.3.2 `TransitAlertService` `transitAlertService` [private]

{

See Also

[TransitAlertService](#)}. This is autowired via Spring Framework.

5.15.3.3 `UserAlertService` `userAlertService` [private]

{

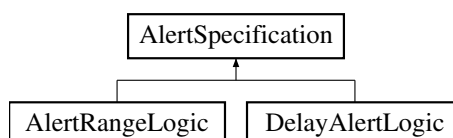
See Also

[UserAlertService](#)}. This is autowired via Spring Framework.

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 [inAlertRange](#) ([GPSLocationObject](#) vehicleLocation)
Compare GPS location and time with alert information to determine if vehicle is within range of a stop and user(s) need to be notified.

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.

5.16.2 Member Function Documentation

5.16.2.1 boolean [inAlertRange](#) ([GPSLocationObject](#) *vehicleLocation*)

Compare GPS location and time with alert information to determine if vehicle is within range of a stop and user(s) need to be notified.

Parameters

<i>lastUpdateTime</i>	- Time GPS information was last updated
<i>vehicleLocation</i>	- Latest GPS coordinates obtained from a vehicle

Returns

true if vehicle is in alert range, false if vehicle is not in alert range

Implemented in [AlertRangeLogic](#), and [DelayAlertLogic](#).

5.17 AlertStatus Enum Reference

Represents the status of an alert depending upon its state in its life cycle.

Public Attributes

- [Active](#)
Represents an alert that is active and is ready to be run when time comes.
- [Deactive](#)
Represents an alert that is in dormant state and wont run even if its time parameter is valid.
- [Running](#)
Represents an alert that is in running state.
- [Expired](#)
Represents an alert that is expired.
- [Error](#)
Represents an alert that is in error state due to technical difficulties.

5.17.1 Detailed Description

Represents the status of an alert depending upon its state in its life cycle.

5.17.2 Member Data Documentation

5.17.2.1 Deactive

Represents an alert that is in dormant state and wont run even if its time parameter is valid.

Usually alert canbe in this status if it is paused.

5.17.2.2 Error

Represents an alert that is in error state due to technical difficulties.

This alert will be run 3 times before it is permanently paused until further action from user or admin.

5.17.2.3 Expired

Represents an alert that is expired.

This type of alert are deleted periodically.

5.18 AlertType Enum Reference

Enumeration of the alert types recognized by bus buddy.

Public Attributes

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

5.18.1 Detailed Description

Enumeration of the alert types recognized by bus buddy.

Bus Buddy can then use a configuration file to tie the alert type to the [AlertSpecification](#) to determine the logic necessary to determine if users registered to a vehicle should be sent alerts.

5.19 AlertUserClient Class Reference

Client layer will handle all the responsibility of sending request to other modules or external sources.

Public Member Functions

- [UserInformation](#) [getUserInformation](#) (String userId, String sessionToken)
Calls User module to get User information for a particular user such as their contact preference via a REST call.

Package Attributes

- String [userModuleURL](#)
A user module url that is used to make a REST call to get users information This information is saved in a config file that is retrieved via Spring's Resource annotation.

5.19.1 Detailed Description

Client layer will handle all the responsibility of sending request to other modules or external sources.

This particular client class will handle request to User module to get necessary user information.

5.19.2 Member Function Documentation

5.19.2.1 `UserInformation` `getUserInformation (String userId, String sessionToken)`

Calls User module to get User information for a particular user such as their contact preference via a REST call.

Parameters

<i>userId</i>	User ID of the user that should receive the alert.
<i>sessionToken</i>	Valid long lived session token that can be used to get information about a particular user.

Returns

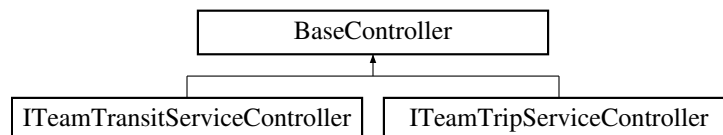
A [UserInformation](#) that contains user data.

gets the user information from the user module via REST call

5.20 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](#)).

5.20.1 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.

5.20.2 Member Function Documentation

5.20.2.1 `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

<i>e</i>	exception which was thrown
----------	----------------------------

Returns

ResponseEntity object

5.20.2.2 `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

<i>e</i>	exception which was thrown
----------	----------------------------

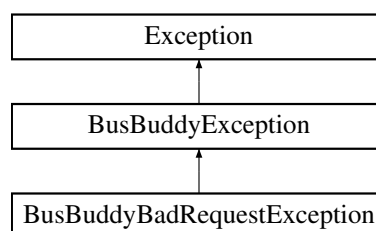
Returns

ResponseEntity object

5.21 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 [getStatusCode](#) ()

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

Static Private Attributes

- static final long **serialVersionUID** = -597422588227245539L

Additional Inherited Members

5.21.1 Detailed Description

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

5.21.2 Member Function Documentation

5.21.2.1 HttpStatus getHttpCode () [virtual]

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

Returns

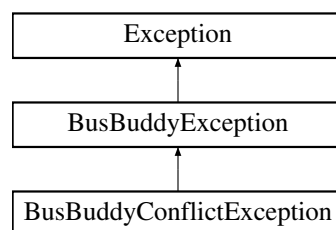
HTTP 400 Bad Request

Implements [BusBuddyException](#).

5.22 BusBuddyConflictException Class Reference

This exception object is thrown when a request would create a conflict which violates constraints set within the system.

Inheritance diagram for BusBuddyConflictException:



Public Member Functions

- **BusBuddyConflictException** (String message)
- **BusBuddyConflictException** (Throwable cause)
- **BusBuddyConflictException** (String message, Throwable cause)
- HttpStatus [getHttpCode](#) ()

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

Static Private Attributes

- static final long **serialVersionUID** = -2044397352042431762L

Additional Inherited Members

5.22.1 Detailed Description

This exception object is thrown when a request would create a conflict which violates constraints set within the system.

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

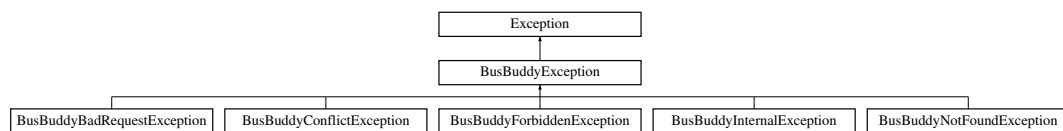
HTTP 409 Conflict

Implements [BusBuddyException](#).

5.23 BusBuddyException Class Reference

This exception object is an abstract base class.

Inheritance diagram for BusBuddyException:



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 exceptions 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

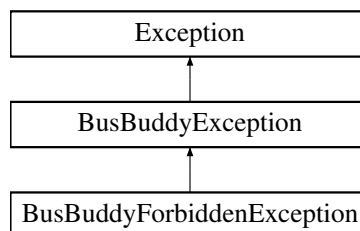
HTTP Status Code object

Implemented in [BusBuddyConflictException](#), [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

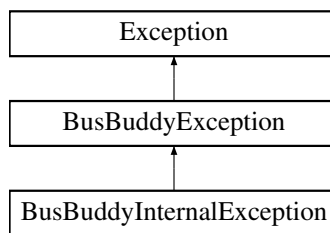
HTTP 403 Forbidden

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

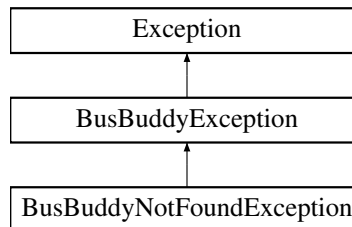
HTTP 500 Internal Server Error

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

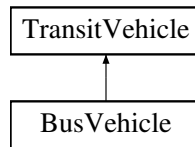
HTTP 404 Not Found

Implements [BusBuddyException](#).

5.27 BusVehicle Class Reference

Bus Vehicle is a concrete implementation of the abstract Transit Vehicle.

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 Detailed Description

Bus Vehicle is a concrete implementation of the abstract Transit Vehicle.
Contains data and functionality specific to buses.

5.27.2 Member Data Documentation

5.27.2.1 `ArrayList<TrackingAlertObserver> alertList` [private]

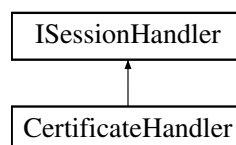
List of alerts registered for this vehicle.

Note alerts may be tracking or delay alerts

5.28 CertificateHandler Class Reference

A concrete strategy implementation of [ISessionHandler](#) that can verify the certificate token being passed.

Inheritance diagram for CertificateHandler:



Public Member Functions

- boolean [verifySessionToken](#) (String sessionToken) throws [BusBuddyForbiddenException](#)
This method takes in an encrypted certificate from Tracking and Transit module in a form of string and then validates for its authenticity.

5.28.1 Detailed Description

A concrete strategy implementation of [ISessionHandler](#) that can verify the certificate token being passed.

5.28.2 Member Function Documentation

5.28.2.1 boolean verifySessionToken (String sessionToken) throws [BusBuddyForbiddenException](#)

This method takes in an encrypted certificate from Tracking and Transit module in a form of string and then validates for its authenticity.

Exceptions

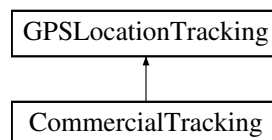
BusBuddyForbiddenException	Throws an exception when the token is not valid.
--	--

Implements [ISessionHandler](#).

5.29 CommercialTracking Class Reference

Implements Subject [GPSLocationTracking](#) 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](#) ()
Instantiates a single Commercial Tracking service to the caller.

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.

5.29.1 Detailed Description

Implements Subject [GPSLocationTracking](#) for retrieving GPS location updates from outside commercial tracking services.

Postcondition

New GPS commercial tracker created or existing one returned.

5.29.2 Constructor & Destructor Documentation

5.29.2.1 [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.

5.29.3 Member Function Documentation

5.29.3.1 static [CommercialTracking getInstance](#) () [static]

Instantiates a single Commercial Tracking service to the caller.

Returns

- [CommercialTracking](#) instance

5.30 [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](#)()

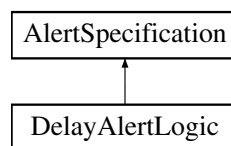
5.30.1 Detailed Description

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

5.31 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 [inAlertRange](#) ([GPSLocationObject](#) vehicleLocation)
Implements Subject [GPSLocationTracking](#) for retrieving GPS location updates from outside commercial tracking services.

5.31.1 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.

5.32 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](#).

5.32.1 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.

5.32.2 Member Data Documentation

5.32.2.1 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.

5.33 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](#).

5.33.1 Detailed Description

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

5.33.2 Member Function Documentation

5.33.2.1 void **setDiscountedFare** (BigDecimal *discountedFare*)

Precondition

discountedFare ≥ 0

5.33.2.2 void **setRegularFare** (BigDecimal *regularFare*)

Precondition

regularFare ≥ 0

5.34 FavoriteTransitService Class Reference

This class is a single transit service in a user's list of favorites.

Public Member Functions

- [FavoriteTransitService](#) (String transitServiceUrl)
This creates a new favorites entry.
- String [getTransitServiceUrl](#) ()
This retrieves the transit service URL associated with the transit service represented by this object.
- boolean [isFavoriteTransitService](#) ()
This retrieves whether this transit service itself is a favorite.
- void [setFavoriteTransitService](#) (boolean favoriteTransitService)
This sets whether this transit service itself is a favorite.
- List< String > [getFavoriteRoutelds](#) ()
This retrieves the ordered list of favorite routes for this transit service.
- void [setFavoriteRoutelds](#) (List< String > favoriteRoutelds)
This sets the ordered list of favorite routes for this transit service.

Private Attributes

- final String **transitServiceUrl**
- boolean **favoriteTransitService**
- List< String > **favoriteRoutelds**

5.34.1 Detailed Description

This class is a single transit service in a user's list of favorites.

If just the routes are favorites, and not the service, then the favoriteTransitService boolean can be set to false. Likewise, if just the transit service is a favorite, and there are no favorite routes, the list of favorite routes can be empty.

5.34.2 Constructor & Destructor Documentation

5.34.2.1 FavoriteTransitService (String transitServiceUrl)

This creates a new favorites entry.

Once created, the service URL cannot be changed.

Parameters

<i>transitServiceUrl</i>	URL to the transit service represented by this object
--------------------------	---

5.34.3 Member Function Documentation

5.34.3.1 List<String> getFavoriteRoutelds ()

This retrieves the ordered list of favorite routes for this transit service.

Returns

ordered list of favorite route IDs

5.34.3.2 String getTransitServiceUrl ()

This retrieves the transit service URL associated with the transit service represented by this object.

Returns

transit service URL

5.34.3.3 boolean isFavoriteTransitService ()

This retrieves whether this transit service itself is a favorite.

Returns

true if it is, false if it is just a container object for favorite routes

5.34.3.4 void setFavoriteRouteIds (List< String > favoriteRouteIds)

This sets the ordered list of favorite routes for this transit service.

Parameters

<i>favoriteRouteIds</i>	ordered list of favorite route IDs
-------------------------	------------------------------------

5.34.3.5 void setFavoriteTransitService (boolean favoriteTransitService)

This sets whether this transit service itself is a favorite.

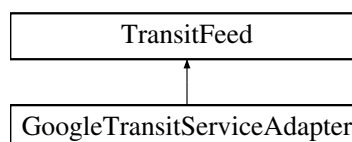
Parameters

<i>favoriteTransit-Service</i>	true if it is, false if it is just a container object for favorite routes
--------------------------------	---

5.35 GoogleTransitServiceAdapter Class Reference

An Adapter Class to allow a [GoogleTransitServiceAPI](#) service to appear as a [TransitService](#).

Inheritance diagram for GoogleTransitServiceAdapter:

**Public Member Functions**

- [GoogleTransitServiceAdapter](#) ([GoogleTransitServiceAPI](#) googleTransitServiceAPI)

Instantiates a new [GoogleTransitServiceAdapter](#) with a [GoogleTransitServiceAPI](#) 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 [GoogleTransitServiceAPI](#) to adapt as a [TransitService](#).

5.35.1 Detailed Description

An Adapter Class to allow a [GoogleTransitServiceAPI](#) service to appear as a [TransitService](#).

5.35.2 Constructor & Destructor Documentation

5.35.2.1 GoogleTransitServiceAdapter ([GoogleTransitServiceAPI](#) *googleTransitServiceAPI*)

Instantiates a new [GoogleTransitServiceAdapter](#) with a [GoogleTransitServiceAPI](#) to delegate calls to.

Parameters

<i>googleTransit-ServiceAPI</i>	the google transit service api
---------------------------------	--------------------------------

5.35.3 Member Function Documentation

5.35.3.1 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

<i>routeId</i>	The unique identifier of the Route
----------------	--

Returns

The matching [Route](#), or null if not found

Implements [TransitFeed](#).

5.35.3.2 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

<i>pickup</i>	The requested pickup Location
<i>dropoff</i>	The requested dropoff Location
<i>distance</i>	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](#).

5.36 GoogleTransitServiceAPI Interface Reference

A client to Google's [Maps API](#).

5.36.1 Detailed Description

A client to Google's [Maps API](#).

5.37 GPSLocationObject Class Reference

GPS Location is a value object used for GPS coordinates and the time of the last update.

Public Member Functions

- double **getLatitude** ()
- void **setLatitude** (double [latitude](#))
- double **getLongitude** ()
- void **setLongitude** (double [longitude](#))
- Date **getLastUpdateTime** ()
- void **setLastUpdateTime** (Date [lastUpdateTime](#))

Private Attributes

- double [latitude](#)
current GPS latitude
- double [longitude](#)
current GPS longitude
- Date [lastUpdateTime](#)
time of last GPS update from device

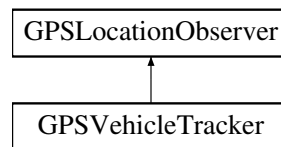
5.37.1 Detailed Description

GPS Location is a value object used for GPS coordinates and the time of the last update.

5.38 GPSLocationObserver Class Reference

Observer Pattern - Observer interface for GPS location tracking.

Inheritance diagram for GPSLocationObserver:



Public Member Functions

- abstract void [gpsUpdate](#) (int [gpsID](#), [GPSLocationObject](#) newLocation)
Observer Pattern update method to update transit vehicle GPS location.
- [GPSLocationObject](#) [getGPSLocation](#) ()
Return current GPS location received from a vehicle.

Protected Member Functions

- void [setGPSLocation](#) ([GPSLocationObject](#) gpsLocation)
Set the current GPS location of a vehicle (state).

Protected Attributes

- [GPSLocationTracking](#) [gpsDevice](#)
Observer Pattern Subject.
- int [gpsID](#)
GPS Device ID being tracked.
- [GPSLocationObject](#) [gpsLocation](#)
Current GPS latitude and longitude from GPS tracker.

5.38.1 Detailed Description

Observer Pattern - Observer interface for GPS location tracking.

5.38.2 Member Function Documentation

5.38.2.1 [GPSLocationObject](#) [getGPSLocation](#) ()

Return current GPS location received from a vehicle.

This is the state of the observer pattern.

Returns

- Location

5.38.2.2 **abstract void** `gpsUpdate (int gpsID, GPSLocationObject newLocation)` `[pure virtual]`

Observer Pattern update method to update transit vehicle GPS location.

Parameters

<i>gpsID</i>	- integer device ID from the GPS unit being tracked
<i>latitude</i>	- double new latitude from GPS device
<i>longitude</i>	- double new longitude from GPS device

Implemented in [GPSVehicleTracker](#).

5.38.2.3 **void** `setGPSLocation (GPSLocationObject gpsLocation)` `[protected]`

Set the current GPS location of a vehicle (state).

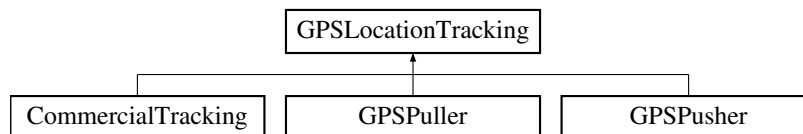
Parameters

<i>gpsLocation</i>	- Location latest latitude and longitude of vehicle
--------------------	---

5.39 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

5.39.1 Detailed Description

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

5.39.2 Member Function Documentation

5.39.2.1 **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](#).

5.39.2.2 `abstract void unregisterGPSDevice (GPSLocationObserver gpsObs) [pure virtual]`

`unregisterGPSDevice` - remove a vehicle from list.

Stop updating vehicle location.

Parameters

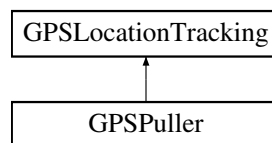
<i>gpsObs</i>	GPSLocationObserver - vehicle to remove from notification list
---------------	--

Implemented in [GPSPusher](#), [CommercialTracking](#), and [GPSPuller](#).

5.40 GPSPuller Class Reference

GPS Puller is a concrete implementation of [GPSLocationTracking](#) 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.

5.40.1 Detailed Description

GPS Puller is a concrete implementation of [GPSLocationTracking](#) 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.

Postcondition

New GPS Puller created if one did not previously exist.

5.40.2 Constructor & Destructor Documentation

5.40.2.1 `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.

5.41 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](#)()

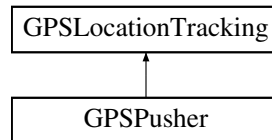
5.41.1 Detailed Description

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

5.42 GPSPusher Class Reference

Implements Subject [GPSLocationTracking](#) 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.

5.42.1 Detailed Description

Implements Subject [GPSLocationTracking](#) 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 Pusher receives the necessary configuration information (e.g. port) from the user interface when the GPS device is registered.

Postcondition

New GPS Listener created if none existed previously.

5.42.2 Constructor & Destructor Documentation

5.42.2.1 GPSPusher() [private]

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

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

5.42.3 Member Function Documentation

5.42.3.1 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.

5.43 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\(\)](#)

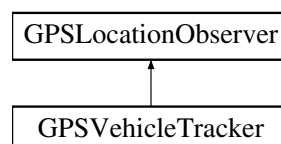
5.43.1 Detailed Description

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

5.44 GPSVehicleTracker Class Reference

Implementation of the Observer, update transit vehicle GPS location and time GPS Vehicle Tracker gets the state as new GPS coordinates and time from GPS Location Tracking and updates the transit vehicle.

Inheritance diagram for GPSVehicleTracker:



Public Member Functions

- [GPSVehicleTracker](#) ([GPSTracking](#) gpsDevice)
Register the Transit Vehicle GPS device with GPS location tracking.
- void [gpsUpdate](#) (int [gpsID](#), [GPSTrackingObject](#) newLocation)

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

Additional Inherited Members

5.44.1 Detailed Description

Implementation of the Observer, update transit vehicle GPS location and time GPS Vehicle Tracker gets the state as new GPS coordinates and time from GPS Location Tracking and updates the transit vehicle.

Postcondition

Transit Vehicle GPS location updated.

5.44.2 Constructor & Destructor Documentation

5.44.2.1 GPSVehicleTracker (GPSLocationTracking *gpsDevice*)

Register the Transit Vehicle GPS device with GPS location tracking.

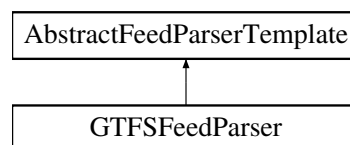
Parameters

<i>gpsDevice</i>	- GPSLocationTracking Subject being observed
------------------	--

5.45 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

5.45.1 Detailed Description

A [AbstractFeedParserTemplate](#) implementation designed to parse [GTFS](#) format ZIP files into [Routes](#).

5.45.2 Member Function Documentation

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

Parse the [GTFS](#) format ZIP files into [Routes](#).

See Also

[AbstractFeedParserTemplate::parseFeed](#)

Implements [AbstractFeedParserTemplate](#).

5.46 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.

5.46.1 Detailed Description

This is a utility class to handle secure hashes.

5.46.2 Member Function Documentation

5.46.2.1 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

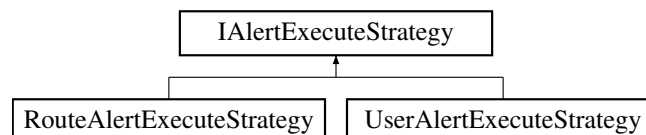
<i>input</i>	
--------------	--

Returns

5.47 IAlertExecuteStrategy Interface Reference

An interface for executing different type of alert based on alert type.

Inheritance diagram for IAlertExecuteStrategy:



Public Member Functions

- boolean [execute](#) ([Alert](#) alertModel)

A method that executes alert passed in based on the type of alert.

5.47.1 Detailed Description

An interface for executing different type of alert based on alert type.

Each implementation of this interface will apply their own execute method that can send alert based on user, route etc.

5.47.2 Member Function Documentation

5.47.2.1 boolean execute ([Alert](#) *alertModel*)

A method that executes alert passed in based on the type of alert.

Parameters

<i>alertModel</i>	An Alert model fetched from database.
-------------------	---

Returns

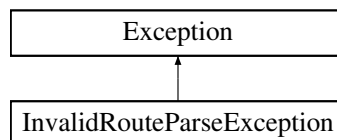
A boolean indicating if overall process complete successfully.

Implemented in [RouteAlertExecuteStrategy](#), and [UserAlertExecuteStrategy](#).

5.48 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.

5.48.1 Detailed Description

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

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

5.48.2 Constructor & Destructor Documentation

5.48.2.1 `InvalidRouteParseException (Set< Route > routeBatch)` `[protected]`

Instantiates a new invalid route parse exception.

Parameters

<code>routeBatch</code>	the route batch
-------------------------	-----------------

5.48.3 Member Data Documentation

5.48.3.1 `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.

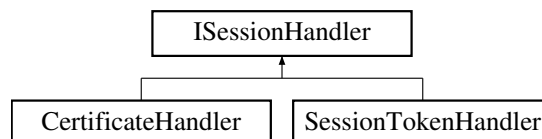
5.48.3.2 `final long serialVersionUID = -4399874766965916500L` `[static], [private]`

The Constant `serialVersionUID`.

5.49 ISessionHandler Interface Reference

An interface to verify the validity of encrypted token being passed.

Inheritance diagram for `ISessionHandler`:



Public Member Functions

- boolean `verifySessionToken` (String encryptedToken) throws `BusBuddyForbiddenException`
This method takes in an encrypted token in a form of string and then validates for its authenticity.

5.49.1 Detailed Description

An interface to verify the validity of encrypted token being passed.

The implementation of this interface must decrypt the session token or certificate and then verify it.

5.49.2 Member Function Documentation

5.49.2.1 boolean verifySessionToken (String *encryptedToken*) throws **BusBuddyForbiddenException**

This method takes in an encrypted token in a form of string and then validates for its authenticity.

Parameters

<i>encryptedToken</i>	<ul style="list-style-type: none"> An encrypted string.
-----------------------	--

Returns

A boolean indicating whether the token was valid or not.

Exceptions

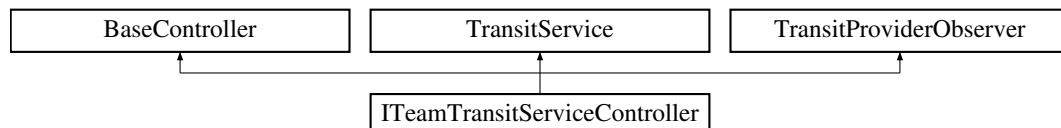
<i>BusBuddyForbiddenException</i>	<ul style="list-style-type: none"> Throws an exception when the token is not valid.
-----------------------------------	--

Implemented in [CertificateHandler](#), and [SessionTokenHandler](#).

5.50 ITeamTransitServiceController Class Reference

The iTeam implementation of the [TransitService](#) that exposes Transit data via a REST Service.

Inheritance diagram for ITeamTransitServiceController:



Public Member Functions

- void [handleRouteDisruptionEvent](#) ([RouteDisruptionEvent](#) routeDisruptionEvent)
After a [RouteDisruptionEvent](#) is received, this class will perform the following:
- [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)
- [AlertRequestController](#) [getAlertRequestController](#) ()
- void [setAlertRequestController](#) ([AlertRequestController](#) alertRequestController)

Private Attributes

- [TransitFeed](#) `transitFeed`
The [TransitFeed](#) used to provide data to this [TransitService](#) implementation.
- [AlertRequestController](#) `alertRequestController`
The inter-module dependency to the Alert Module.

5.50.1 Detailed Description

The iTeam implementation of the [TransitService](#) that exposes Transit data via a REST Service.

5.50.2 Member Function Documentation

5.50.2.1 [Route](#) `getRoute (String routeld)`

Gets a [Route](#) by its unique identifier.

Parameters

<i>routeld</i>	The unique identifier of the Route
----------------	--

Returns

The matching [Route](#), or null if not found

Precondition

routeld is not null or blank.

Postcondition

The [Route](#) is returned if the **routeld** is found, else null.

Implements [TransitService](#).

5.50.2.2 `Set<Route> getRoutes (Location pickup, Location dropoff, int distance)`

Gets all available [Route](#)s that match a **pickup** or **dropoff** [Location](#) by not more than a given **distance**.

Parameters

<i>pickup</i>	The requested dropoff Location
<i>dropoff</i>	the dropoff
<i>distance</i>	The distance (in miles) that each Route can deviate from the requested pickup or dropoff

Returns

The matching [Routes](#)

Precondition

pickup is not null or blank.

dropoff is not null or blank.

distance is non-negative. [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.

Implements [TransitService](#).

5.50.2.3 URL `getServiceURL ()`

The URL that uniquely identifies this [TransitService](#).

In a REST environment, this might be the root of the REST API path.

Returns

The URL of this service

Implements [TransitService](#).

5.50.2.4 [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](#).

5.50.2.5 `void handleRouteDisruptionEvent (RouteDisruptionEvent routeDisruptionEvent)`

After a [RouteDisruptionEvent](#) is received, this class will perform the following:

- Internally register the [RouteDisruptionEvent](#) so that any subsequent requests for affected [Routes](#) will include appropriate [Detour](#) information.
- Notify the Alert module via the [AlertRequestController](#) of the disruption with the updated [Route](#). This updated [Route](#) should include all necessary [Detour](#) information.

Implements [TransitProviderObserver](#).

5.50.3 Member Data Documentation

5.50.3.1 [AlertRequestController](#) `alertRequestController` [private]

The inter-module dependency to the Alert Module.

The [AlertRequestController](#) accepts requests from this class to inform the Alert Module of a [RouteDisruptionAlert](#).

NOTE: This represents a conceptual dependency to the [AlertRequestController](#). During implementation phase, actual communication with the [AlertRequestController](#) will happen via some client object or service. Implementation of the actual client and its link to the [AlertRequestController](#) (to include REST URLs and JSON structure) will be left to the next phase, or as design details for the development team.

5.50.3.2 [TransitFeed](#) `transitFeed` [private]

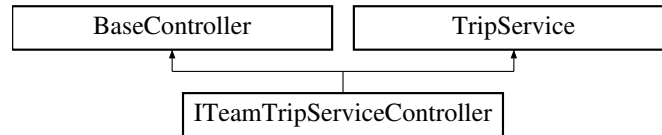
The [TransitFeed](#) used to provide data to this [TransitService](#) implementation.

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

5.51 ITeamTripServiceController Class Reference

An iTeam implementation of the [TripService](#) that exposes [Trip](#) data via a REST Service.

Inheritance diagram for ITeamTripServiceController:



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.

5.51.1 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.

5.51.2 Member Function Documentation

5.51.2.1 [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

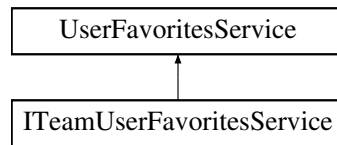
The calculated [Trip](#)

Implements [TripService](#).

5.52 ITeamUserFavoritesService Class Reference

This is the iTeam's implementation of [UserFavoritesService](#).

Inheritance diagram for ITeamUserFavoritesService:



Public Member Functions

- [UserFavoritesList readFavorites](#) (String sessionToken) throws BusBuddyException
- void [saveFavorites](#) (String sessionToken, [UserFavoritesList favorites](#)) throws BusBuddyException

Protected Attributes

- [ITeamUserLoginService userLoginService](#)
- [UserFavoritesRepository userFavoritesRepository](#)

5.52.1 Detailed Description

This is the iTeam's implementation of [UserFavoritesService](#).

5.52.2 Member Function Documentation

5.52.2.1 [UserFavoritesList readFavorites](#) (String *sessionToken*) throws BusBuddyException

See Also

[UserFavoritesService.readFavorites](#)

Implements [UserFavoritesService](#).

5.52.2.2 void [saveFavorites](#) (String *sessionToken*, [UserFavoritesList favorites](#)) throws BusBuddyException

See Also

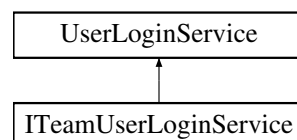
[UserFavoritesService.saveFavorites](#)

Implements [UserFavoritesService](#).

5.53 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) throws `BusBuddyException`
- void [resetPassword](#) (String username, short countryCode, String mobile) throws `BusBuddyException`

Protected Member Functions

- boolean [checkPermissions](#) (String sessionToken, [User](#) user) throws `BusBuddyException`
This method checks to see if the currently logged in user has permissions to modify another given user object.

Protected Attributes

- [SessionRepository](#) **sessionRepository**
- [UserRepository](#) **userRepository**

5.53.1 Detailed Description

This is the iTeam's implementation of [UserLoginService](#).

5.53.2 Member Function Documentation

5.53.2.1 boolean [checkPermissions](#) (String *sessionToken*, [User](#) *user*) throws `BusBuddyException` [protected]

This method checks to see if the currently logged in user has permissions to modify another given user object.

A session can modify a user if it is the currently logged in user that is being modified, or if the currently logged in user is a system administrator.

Precondition

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

Postcondition

The expiration time will be advanced based on this activity against the session.

Parameters

<i>sessionToken</i>	The session token identifying the session of the currently logged in user.
<i>user</i>	The user we are checking to see if the session has permission to modify.

Returns

true if the currently signed in user has permission to modify the user specified in the user parameter, false otherwise

Exceptions

<i>BusBuddyBadRequestException</i>	This exception is thrown if the session token is blank.
<i>BusBuddyForbiddenException</i>	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.
<i>BusBuddyInternalException</i>	This exception is thrown if an internal error prevents processing of the request.

5.53.2.2 String createAlertSession (String sessionToken) throws BusBuddyException

See Also

[UserLoginService.createAlertSession](#)Implements [UserLoginService](#).

5.53.2.3 User getUser (String sessionToken) throws BusBuddyException

See Also

[UserLoginService.getUser](#)Implements [UserLoginService](#).

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

See Also

[UserLoginService.login](#)Implements [UserLoginService](#).

5.53.2.5 void logout (String sessionToken) throws BusBuddyException

See Also

[UserLoginService.logout](#)Implements [UserLoginService](#).

5.53.2.6 void resetPassword (String username, String email) throws BusBuddyException

See Also

[UserLoginService.resetPassword\(String, String\)](#)Implements [UserLoginService](#).

5.53.2.7 void resetPassword (String username, short countryCode, String mobile) throws BusBuddyException

See Also

[UserLoginService.resetPassword\(String, short, String\)](#)Implements [UserLoginService](#).

5.53.2.8 void setUsername (String email) throws BusBuddyException

See Also

[UserLoginService.setUsername\(String\)](#)

Implements [UserLoginService](#).

5.53.2.9 void setUsername (short countryCode, String mobile) throws BusBuddyException

See Also

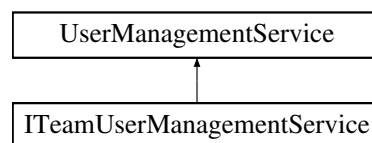
[UserLoginService.setUsername\(short, String\)](#)

Implements [UserLoginService](#).

5.54 ITeamUserManagementService Class Reference

This is the iTeam's implementation of [UserManagementService](#).

Inheritance diagram for ITeamUserManagementService:



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) throws BusBuddyException
- [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

Protected Attributes

- [ITeamUserLoginService](#) **userLoginService**
- [UserRepository](#) **userRepository**

5.54.1 Detailed Description

This is the iTeam's implementation of [UserManagementService](#).

5.54.2 Member Function Documentation

5.54.2.1 User createUser (User userToCreate, String password) throws BusBuddyException

See Also

[UserManagementService.createUser](#)

Implements [UserManagementService](#).

5.54.2.2 void deleteUser (String *sessionToken*, User *userToDelete*) throws BusBuddyException

See Also

[UserManagementService.delete](#)

Implements [UserManagementService](#).

5.54.2.3 User findUserByEmail (String *sessionToken*, String *email*) throws BusBuddyException

See Also

[UserManagementService.findUserByEmail](#)

Implements [UserManagementService](#).

5.54.2.4 User findUserByMobile (String *sessionToken*, short *countryCode*, String *mobile*) throws BusBuddyException

See Also

[UserManagementService.findUserByMobile](#)

Implements [UserManagementService](#).

5.54.2.5 User findUserByUsername (String *sessionToken*, String *username*) throws BusBuddyException

See Also

[UserManagementService.findUserByUsername](#)

Implements [UserManagementService](#).

5.54.2.6 void updateUser (String *sessionToken*, User *newUserData*, String *password*) throws BusBuddyException

See Also

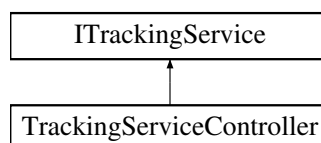
[UserManagementService.updateUser](#)

Implements [UserManagementService](#).

5.55 ITrackingService Interface Reference

Interface for the Tracking Service Controller.

Inheritance diagram for ITrackingService:



Public Member Functions

- void [registerVehicleOnRoute](#) (URL url, int gpsDeviceID)
Interface for registering vehicles on a route from the user interface.
- void [unregisterVehicleFromRoute](#) (String url, int gpsDeviceID)
Interface for removing a registered vehicle from a route when the vehicle goes out of service.
- void [addUserTrackingAlert](#) ([UserTrackingAlertObject](#) utao)
Add a user tracking alert from the user interface.
- void [startTrackingController](#) ()
Logic necessary when the tracking controller is cold started.
- [GPSLocationObject](#) [getTransitVehicleLocation](#) (int gpsDeviceID)
Allows users and modules outside of the tracking package to query for the current location of a registered vehicle.

5.55.1 Detailed Description

Interface for the Tracking Service Controller.

The tracking service purpose is to allow vehicles to register on routes and to establish a connection with the GPS device installed in the vehicle to provide regular vehicle location updates. The tracking service uses the current GPS coordinates to determine when to send alerts to registered users.

5.55.2 Member Function Documentation

5.55.2.1 void [addUserTrackingAlert](#) ([UserTrackingAlertObject](#) utao)

Add a user tracking alert from the user interface.

Parameters

<i>utao</i>	- UserTrackingAlertObject information from user interface necessary to create alert.
-------------	--

Implemented in [TrackingServiceController](#).

5.55.2.2 [GPSLocationObject](#) [getTransitVehicleLocation](#) (int *gpsDeviceID*)

Allows users and modules outside of the tracking package to query for the current location of a registered vehicle.

Parameters

<i>gpsDeviceID</i>	- unique hardware GPS device
--------------------	------------------------------

Returns

- Location current latitude and longitude of vehicle

Implemented in [TrackingServiceController](#).

5.55.2.3 void [registerVehicleOnRoute](#) (URL url, int *gpsDeviceID*)

Interface for registering vehicles on a route from the user interface.

Parameters

<i>url</i>	- Transit company URL
<i>gpsDeviceID</i>	- unique hardware GPS device ID

Implemented in [TrackingServiceController](#).

5.55.2.4 void startTrackingController ()

Logic necessary when the tracking controller is cold started.

Retrieves the saved user alerts from the [IAAlertService](#)

Implemented in [TrackingServiceController](#).

5.55.2.5 void unregisterVehicleFromRoute (String url, int gpsDeviceID)

Interface for removing a registered vehicle from a route when the vehicle goes out of service.

Parameters

<i>url</i>	- URL uniquely identifying a transit company.
<i>gpsDeviceID</i>	- unique hardware GPS id being unregistered on user interface

Implemented in [TrackingServiceController](#).

5.56 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.

5.56.1 Detailed Description

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

5.56.2 Constructor & Destructor Documentation

5.56.2.1 Location (double latitude, double longitude)

Instantiates a new immutable [Location](#) with the given latitude and longitude.

Parameters

<i>latitude</i>	The point latitude
<i>longitude</i>	The point longitude

5.57 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.

5.57.1 Detailed Description

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

5.57.2 Member Function Documentation

5.57.2.1 static void [sendEmail](#) (String to, String from, String subject, String htmlBody) throws [BusBuddyInternalException](#) [static]

This method sends an HTML e-mail.

Parameters

<i>to</i>	recipient address
<i>from</i>	sender address
<i>subject</i>	subject line
<i>htmlBody</i>	HTML body of the message

Exceptions

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

5.57.2.2 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

<i>countryCode</i>	country code for the recipient
<i>mobileNumber</i>	mobile number to send to
<i>message</i>	body of the message to send

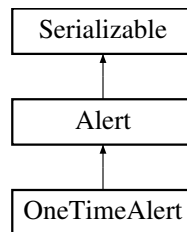
Exceptions

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

5.58 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

5.58.1 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](#).

5.58.2 Member Data Documentation

5.58.2.1 Date `dateExecuted` [private]

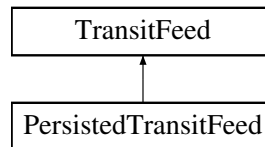
Date when it was executed.

If absent, then the alert hasn't been executed.

5.59 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 routeld)
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.

5.59.1 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](#).

5.59.2 Member Function Documentation

5.59.2.1 [Route](#) getRoute (String routeld)

Gets a [Route](#) by its unique identifier.

Precondition

routeld is not null or blank.

Postcondition

The [Route](#) is returned if the **routeld** is found, else null.

Parameters

<i>routeld</i>	The unique identifier of the Route
----------------	--

Returns

The matching [Route](#), or null if not found

Implements [TransitFeed](#).

5.59.2.2 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

<i>pickup</i>	The requested pickup Location
<i>dropoff</i>	The requested dropoff Location
<i>distance</i>	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

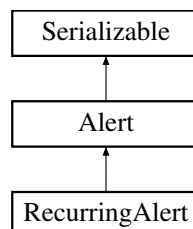
The matching [Routes](#)

Implements [TransitFeed](#).

5.60 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](#))

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

5.60.1 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.

5.60.2 Member Data Documentation

5.60.2.1 [AlertRecurringType](#) [alertRecurringType](#) [private]

Type of recurring alert.

Value is as defined in [AlertRecurringType](#)

5.60.2.2 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.

5.61 RecurringData Class Reference

This model holds the information about the date and time the alert needs to run.

Public Member Functions

- int **getDayOfYear** ()
- void **setDayOfYear** (int [dayOfYear](#))
- int **getDayOfMonth** ()
- void **setDayOfMonth** (int [dayOfMonth](#))
- int **getDayOfWeek** ()
- void **setDayOfWeek** (int [dayOfWeek](#))

- int **getStartHour** ()
- void **setStartHour** (int [startHour](#))
- int **getStartMinute** ()
- void **setStartMinute** (int [startMinute](#))

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.

5.61.1 Detailed Description

This model holds the information about the date and time the alert needs to run.

5.61.2 Member Data Documentation

5.61.2.1 int [dayOfMonth](#) [private]

Day of month that the alert should run.

Valid value is from 1-28.

5.61.2.2 int [dayOfWeek](#) [private]

Day of week that the alert should run.

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

5.61.2.3 int [dayOfYear](#) [private]

Day of year that the alert should run.

Valid value = 1-365

5.61.2.4 int [startHour](#) [private]

The exact hour when the alert should run.

Valid value is from 0 - 23

5.61.2.5 int [startMinute](#) [private]

The exact minute when the alert should run.

Valid value is from 0-59.

5.62 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.

5.62.1 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](#).

5.62.2 Member Data Documentation

5.62.2.1 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](#).

5.62.2.2 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](#).

5.62.2.3 List<Stop> stops [private]

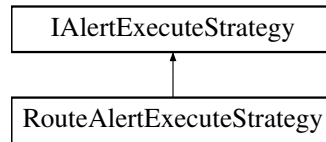
And ordered list of [Stops](#) to be visited in this [Route](#).

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](#).

5.63 RouteAlertExecuteStrategy Class Reference

A concrete implementation of [IAlertExecuteStrategy](#) that handles executing alert related to route changes.

Inheritance diagram for RouteAlertExecuteStrategy:



Public Member Functions

- boolean [execute](#) ([Alert](#) alert)

Package Attributes

- [AlertRepository](#) [alertRepository](#)
An instance of [AlertRepository](#) that is used to fetch alerts that are effected by particular route.
- [AlertUserClient](#) [userClient](#)
A spring autowired instance of [AlertUserClient](#) that can call the User module to get user information.

5.63.1 Detailed Description

A concrete implementation of [IAlertExecuteStrategy](#) that handles executing alert related to route changes.

5.63.2 Member Function Documentation

5.63.2.1 boolean execute ([Alert](#) alert)

See Also

[IAlertExecuteStrategy::execute\(List\)](#) Takes in a list of alerts usually provided from Transit or Tracking module where route disruption information is stored. This method finds all the alerts (that users have created) that are associated with a particular route. Hence, each route disruption alert can [execute](#) multiple other alerts.

Find other alert that is associated with routeId in this alert.

call user module for each and every of these alerts and push the notification.

if success

Implements [IAlertExecuteStrategy](#).

5.63.3 Member Data Documentation

5.63.3.1 [AlertRepository](#) [alertRepository](#) [package]

An instance of [AlertRepository](#) that is used to fetch alerts that are effected by particular route.

This is autowired via Spring Framework.

5.63.3.2 `AlertUserClient` `userClient` [package]

A spring autowired instance of `AlertUserClient` that can call the User module to get user information.

This is autowired via Spring Framework.

5.64 `RouteDisruptionAlert` Class Reference

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

Public Member Functions

- URL `getTransitServiceUrl` ()
- void `setTransitServiceUrl` (URL `transitServiceUrl`)
- String `getRouteld` ()
- void `setRouteld` (String `routeld`)

Private Attributes

- URL `transitServiceUrl`
The URL callback of the originating `TransitService`.
- String `routeld`
The unique identifier of the affected `Route`.

5.64.1 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 `routeld` from the `getRouteld()` 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.

5.64.2 Member Data Documentation

5.64.2.1 `String routeld` [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.

5.64.2.2 `URL transitServiceUrl` [private]

The URL callback of the originating `TransitService`.

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

5.65 `RouteDisruptionEvent` Class Reference

An event indicating a disruption in normal `Route` scheduling or service.

Public Member Functions

- [RouteDisruptionEvent](#) ([Route](#) *disruptedRoute*)
Instantiates a new route disruption event with the affected [Route](#).
- [Route](#) **getDisruptedRoute** ()
- void **setDisruptedRoute** ([Route](#) *disruptedRoute*)

Private Attributes

- [Route](#) *disruptedRoute*
The disrupted [Route](#), complete with any [Detour](#) information that is available.

5.65.1 Detailed Description

An event indicating a disruption in normal [Route](#) scheduling or service.

This event will be initiated by a [TransitProvider](#) in cases of mechanical failure, scheduled maintenance, infrastructure delays such as construction or road closures, etc.

Note that a [RouteDisruptionEvent](#) can signal that a [Route](#) is returning back to normal service after the disruption has cleared. This is done by sending a [Route](#) with no [Detours](#).

5.65.2 Constructor & Destructor Documentation

5.65.2.1 [RouteDisruptionEvent](#) ([Route](#) *disruptedRoute*)

Instantiates a new route disruption event with the affected [Route](#).

Parameters

<i>disruptedRoute</i>	The Disrupted Route
-----------------------	-------------------------------------

5.66 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.

5.66.1 Detailed Description

A Repository Pattern supporting lifecycle operations of [Routes](#), such as Read, Save, Delete, and Query functionality.

5.66.2 Member Function Documentation

5.66.2.1 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

<i>routeId</i>	
----------------	--

5.66.2.2 Collection<[Route](#)> getAll ()

Retrieves all available [Routes](#) in the Repository.

Returns

All available [Routes](#).

5.66.2.3 [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

<i>routeId</i>	The identifier of the requested Route
----------------	---

Returns

The requested [Route](#)

5.66.2.4 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

<i>route</i>	The Route to save.
--------------	------------------------------------

5.66.2.5 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.

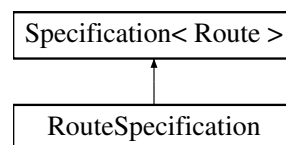
Parameters

<i>route</i>	The Route to save.
--------------	------------------------------------

5.67 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)

5.67.1 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`, `or`, or `not` methods of separate [Specifications](#).

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](#)

5.67.2 Member Function Documentation

5.67.2.1 boolean `isSatisfiedBy (Route candidate)`

Validates the given [Route](#) with the appropriate validation rules of this Specification.

Parameters

<i>candidate</i>	The Route to be validated
------------------	---

Returns

True if a valid [Route](#), False if not

5.68 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**

5.68.1 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.

5.68.2 Constructor & Destructor Documentation

5.68.2.1 [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

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

5.68.3 Member Function Documentation

5.68.3.1 [Calendar](#) getCreationTime () [protected]

This retrieves the time that the session was created.

Returns

session creation time

5.68.3.2 [Calendar](#) getExpirationTime () [protected]

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

Returns

session expiration time

5.68.3.3 String getSessionToken () [protected]

This retrieves the session token.

Returns

session token

5.68.3.4 int getUserId () [protected]

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

Returns

user's ID number

5.68.3.5 boolean isAlertSession () [protected]

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

Returns

true if it is, false otherwise

5.68.3.6 boolean isValid () [protected]

This checks to see if the session is valid.

Returns

true if it is, false otherwise

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

This sets the time that the session should expire.

Parameters

<i>expirationTime</i>	expiration time to set
-----------------------	------------------------

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

This sets whether the session is valid.

Parameters

<i>valid</i>	true if it is, false otherwise
--------------	--------------------------------

5.69 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.

5.69.1 Detailed Description

This class is responsible for handling database access for Sessions, and to construct, persist, and retrieve [Session](#) objects.

5.69.2 Member Function Documentation

5.69.2.1 `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

<i>user</i>	This is the user to create the session for.
<i>isAlertSession</i>	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

<code>BusBuddyInternalException</code>	This exception is thrown when there is a database error.
--	--

5.69.2.2 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

<i>sessionToken</i>	This is the session token that identifies the session.
---------------------	--

Returns

[Session](#) object represented by the session token that was passed in.

Exceptions

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

5.69.2.3 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

<i>sessionToken</i>	This is the session token that identifies the session.
---------------------	--

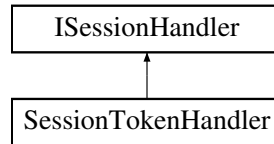
Exceptions

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

5.70 SessionTokenHandler Class Reference

A concrete strategy implementation of [ISessionHandler](#) that can verify the session token being passed.

Inheritance diagram for SessionTokenHandler:



Public Member Functions

- boolean [verifySessionToken](#) (String sessionToken) throws [BusBuddyForbiddenException](#)
This method takes in an encrypted session token in a form of string and then validates for its authenticity.

5.70.1 Detailed Description

A concrete strategy implementation of [ISessionHandler](#) that can verify the session token being passed.

5.70.2 Member Function Documentation

5.70.2.1 boolean verifySessionToken (String sessionToken) throws [BusBuddyForbiddenException](#)

This method takes in an encrypted session token in a form of string and then validates for its authenticity.

Exceptions

BusBuddyForbiddenException	<ul style="list-style-type: none"> • Throws an exception when the token is not valid.
--	--

Implements [ISessionHandler](#).

5.71 SessionVerificationFactory Class Reference

A factory that picks the appropriate strategy [ISessionHandler](#) to verify the encrypted token.

Static Public Member Functions

- static [ISessionHandler getSessionTokenVerificationStrategy](#) ([AlertInitiator](#) alertInitiator)
This method takes in an [AlertInitiator](#) and depending upon the value can invoke different strategy to verify the token.

5.71.1 Detailed Description

A factory that picks the appropriate strategy [ISessionHandler](#) to verify the encrypted token.

5.71.2 Member Function Documentation

5.71.2.1 static `ISessionHandler` getSessionTokenVerificationStrategy (`AlertInitiator alertInitiator`) [static]

This method takes in an `AlertInitiator` and depending upon the value can invoke different strategy to verify the token.

Parameters

<code>alertInitiator</code>	A valid <code>AlertInitiator</code>
-----------------------------	-------------------------------------

Returns

A implementation of `ISessionHandler`

5.72 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.

5.72.1 Detailed Description

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

Parameters

<code><T></code>	The candidate Type accepted by the Specification.
------------------------	---

5.72.2 Member Function Documentation

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

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

Parameters

<code>specification</code>	The Specification to apply the 'AND' operation to.
----------------------------	--

Returns

The 'AND' Specification

5.72.2.2 boolean isSatisfiedBy (T *candidate*)

Checks if the given candidate satisfies the specification.

Parameters

<i>candidate</i>	The candidate
------------------	---------------

Returns

true, if is satisfied by the candidate

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

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

Parameters

<i>specification</i>	The Specification to apply the 'NOT' operation to.
----------------------	--

Returns

The 'NOT' Specification

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

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

Parameters

<i>specification</i>	The Specification to apply the 'OR' operation to.
----------------------	---

Returns

The 'OR' Specification

5.73 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](#)
A short text-based description of describing the [Stop](#) and its location.
- [Location](#) [location](#)
The physical location of the [Stop](#).

5.73.1 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.

5.73.2 Member Function Documentation

5.73.2.1 `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

<i>begin</i>	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.
<i>end</i>	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.

5.73.3 Member Data Documentation

5.73.3.1 `String description [private]`

A short text-based description of describing the [Stop](#) and its location.

This could be an intersection:

- "18th Ave and 58th St." or a landmark/park/attraction:

"Como Zoo North Entrance" or other identifying text.

This description should be sufficient enough to allow a user to find the given [Stop](#) without necessarily needing the [Location](#) information.

5.74 TrackingAlertFactory Class Reference

The Alert Factory handles the creation of a user alert.

Public Member Functions

- [TrackingAlertObserver](#) `createAlertObserver` ([TransitVehicle](#) vehicle)

5.74.1 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.

Postcondition

New user alert registered with a transit vehicle in the array list.

5.74.2 Member Function Documentation

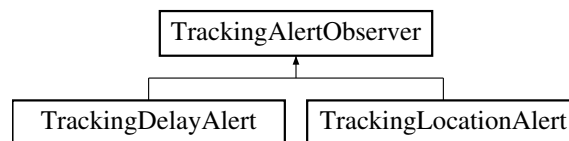
5.74.2.1 TrackingAlertObserver createAlertObserver (TransitVehicle vehicle)

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.

5.75 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 range.
- abstract void [updateAlert](#) ()
The observer pattern update method called from the subject [TransitVehicle](#) when a vehicle is determined to be in an alert range and a user needs to be notified.

Protected Member Functions

- void [setSpec](#) ([AlertSpecification](#) spec)
Set the alert specification.

Private Attributes

- [UserTrackingAlertObject](#) [userAlertTrackingObject](#)
Value Object containing the items necessary for an alert.
- [AlertSpecification](#) [specification](#)
The business logic specification of how to determine if an alert needs to be sent for a vehicle.
- [AlertRequestController](#) [arc](#) = new [AlertRequestController](#)()
Alert Module Controller called via a REST API [processUserAlertRequest\(\)](#)

5.75.1 Detailed Description

Abstract class defining the methods for the tracking alert observer.

Precondition

Vehicle is determined by [AlertSpecification](#) to within range

Postcondition

Alert request is sent to the [AlertService](#) Abstract class defining the methods for the tracking alert observer. This class calls alert module's alert controller via REST call to fetch necessary information.

5.75.2 Member Function Documentation

5.75.2.1 [AlertSpecification](#) getSpec ()

Return the specification to use to determine if a vehicle is in an alert range.

Return values

AlertSpecification	
------------------------------------	--

5.75.2.2 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.
------	---

5.75.2.3 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 range and a user needs to be notified.

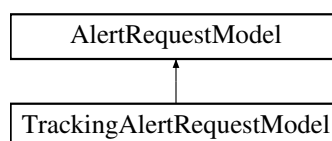
Alert notifications are actually sent using the [IAlertService](#)

Implemented in [TrackingLocationAlert](#), and [TrackingDelayAlert](#).

5.76 TrackingAlertRequestModel Class Reference

This model is a JSON representation of the request from Tracking module.

Inheritance diagram for TrackingAlertRequestModel:



Additional Inherited Members

5.76.1 Detailed Description

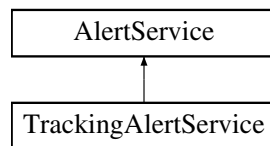
This model is a JSON representation of the request from Tracking module.

During implementation, any additional data that is needed can be added.

5.77 TrackingAlertService Class Reference

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

Inheritance diagram for TrackingAlertService:



Public Member Functions

- [AlertResponseModel](#) [createAlert](#) ([AlertRequestModel](#) requestModel)
- {
- [AlertResponseModel](#) [updateAlert](#) ([Alert](#) alertModel)
- {
- boolean [sendAlert](#) ()
- {

Additional Inherited Members

5.77.1 Detailed Description

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

This handles all the alerts initiated by Tracking module.

5.77.2 Member Function Documentation

5.77.2.1 [AlertResponseModel](#) [createAlert](#) ([AlertRequestModel](#) requestModel) [virtual]

{

See Also

[AlertService::createAlert\(AlertRequestModel\)](#)

Implements [AlertService](#).

5.77.2.2 boolean [sendAlert](#) () [virtual]

{

See Also

[AlertService::sendAlert\(\)](#)

Implements [AlertService](#).

5.77.2.3 AlertResponseModel updateAlert (Alert *alertModel*) [virtual]

{

See Also

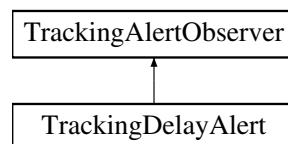
[AlertService::updateAlert\(Alert\)](#)

Implements [AlertService](#).

5.78 TrackingDelayAlert Class Reference

Tracking Alert Observer implements the abstract tracking alert observer and provides the method to actually send an alert to a registered user that their bus is approaching their stop.

Inheritance diagram for TrackingDelayAlert:



Public Member Functions

- void [updateAlert](#) ()

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

Additional Inherited Members

5.78.1 Detailed Description

Tracking Alert Observer implements the abstract tracking alert observer and provides the method to actually send an alert to a registered user that their bus is approaching their stop.

Precondition

Vehicle is determined by [AlertSpecification](#) to within range

Postcondition

Alert request is sent to the [AlertService](#)

5.78.2 Member Function Documentation

5.78.2.1 void updateAlert () [virtual]

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

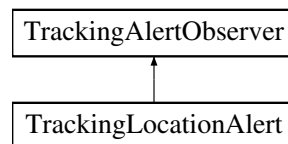
Use the `AlertService` to contact the registered user.

Implements [TrackingAlertObserver](#).

5.79 TrackingLocationAlert Class Reference

Concrete implementation of the tracking alert observer.

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, calls Alert Service.

Additional Inherited Members

5.79.1 Detailed Description

Concrete implementation of the tracking alert observer.

The subject calls the update alert for registered vehicles.

5.79.2 Constructor & Destructor Documentation

5.79.2.1 TrackingLocationAlert ([TransitVehicle](#) vehicle)

Tracking Location Alert constructor.

Associates user information with vehicle to monitor.

Parameters

<i>vehicle</i>	TransitVehicle Vehicle to add alert monitoring
<i>alert</i>	- TrackingLocationAlert User contact details and vehicle location indicating when user wants to receive alert.

5.80 TrackingResponseModel Class Reference

This is a basic tracking response model that is returned for every tracking related request.

Public Member Functions

- String [formatJSONResponse](#) ()

Formats output responses and requests from the tracking module.

- [VehicleObject convertJSONVehicleInput \(\)](#)

Converts input JSON formatted vehicle registration request to internal vehicle object.

- [UserTrackingAlertObject convertJSONAlertInput \(\)](#)

Convert input JSON formatted alert request from user interface to internal user alert object.

Private Attributes

- String [status](#)

Status message for the tracking request.

- String [errorMessage](#)

Any error message if the tracking request fails.

5.80.1 Detailed Description

This is a basic tracking response model that is returned for every tracking related request.

Additional parameters can be added as needed during implementation phase. This model is responsible for verify that all data received is within range before returning local object.

5.80.2 Member Function Documentation

5.80.2.1 [UserTrackingAlertObject convertJSONAlertInput \(\)](#)

Convert input JSON formatted alert request from user interface to internal user alert object.

Returns

[UserTrackingAlertObject](#) values

5.80.2.2 [VehicleObject convertJSONVehicleInput \(\)](#)

Converts input JSON formatted vehicle registration request to internal vehicle object.

Returns

[VehicleObject](#) created from user registration.

5.80.2.3 [String formatJSONResponse \(\)](#)

Formats output responses and requests from the tracking module.

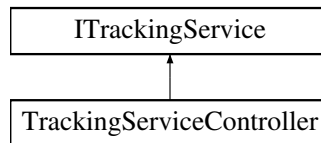
Returns

String containing response

5.81 TrackingServiceController Class Reference

Tracking service controller is the concrete implementation of the tracking service interface.

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.
- [GPSLocationObject](#) [getTransitVehicleLocation](#) (int gpsDeviceID)
Find locations of the specified GPS device ID.

Package Attributes

- [TransitVehicleFactory](#) [transitFactory](#) = new [TransitVehicleFactory](#)()
Logic for creating transit vehicles.
- [TrackingAlertFactory](#) [alertFactory](#) = new [TrackingAlertFactory](#)()
Logic for creating new user alerts.

5.81.1 Detailed Description

Tracking service controller is the concrete implementation of the tracking service interface.

Provides the tracking functionality to other Bus Buddy modules and ties vehicle location to registered user alerts. The tracking service controller accepts REST requests from the User Interface module, the Alert Module

5.81.2 Member Function Documentation

5.81.2.1 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 route 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 range.
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](#).

5.81.2.2 void registerVehicleOnRoute (URL *url*, int *gpsDeviceID*)

Create a vehicle when a user registers a vehicle on a route through the user interface.

Accepts a [VehicleObject](#) JSON

Implements [ITrackingService](#).

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

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

Parameters

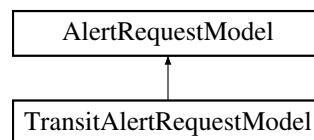
<i>url</i>	- URL transit company and route information for this vehicle
<i>gpsDeviceID</i>	- integer GPS device ID installed in vehicle, must match the ID the vehicle registered with.

Implements [ITrackingService](#).

5.82 TransitAlertRequestModel Class Reference

This model is a JSON representation of the request from Transit module.

Inheritance diagram for TransitAlertRequestModel:



Additional Inherited Members

5.82.1 Detailed Description

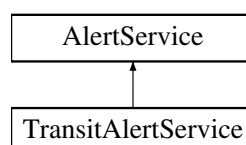
This model is a JSON representation of the request from Transit module.

During implementation, any additional data that is needed can be added.

5.83 TransitAlertService Class Reference

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

Inheritance diagram for TransitAlertService:



Public Member Functions

- [AlertResponseModel](#) [createAlert](#) ([AlertRequestModel](#) requestModel)
{
- [AlertResponseModel](#) [updateAlert](#) ([Alert](#) alertModel)
{
- boolean [sendAlert](#) ()
{

Additional Inherited Members

5.83.1 Detailed Description

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

This handles all the alerts initiated by Transit module.

5.83.2 Member Function Documentation

5.83.2.1 [AlertResponseModel](#) [createAlert](#) ([AlertRequestModel](#) *requestModel*) [virtual]

{

See Also

[AlertService::createAlert\(AlertRequestModel\)](#)

Implements [AlertService](#).

5.83.2.2 boolean [sendAlert](#) () [virtual]

{

See Also

[AlertService::sendAlert\(\)](#)

Implements [AlertService](#).

5.83.2.3 [AlertResponseModel](#) [updateAlert](#) ([Alert](#) *alertModel*) [virtual]

{

See Also

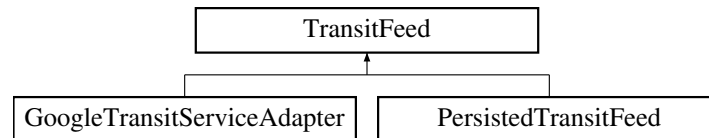
[AlertService::updateAlert\(Alert\)](#)

Implements [AlertService](#).

5.84 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**.*

5.84.1 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 [TransitService#getTransitInfo](#) 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](#).

5.84.2 Member Function Documentation

5.84.2.1 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

<i>routeId</i>	The unique identifier of the Route
----------------	--

Returns

The matching [Route](#), or null if not found

Implemented in [GoogleTransitServiceAdapter](#), and [PersistedTransitFeed](#).

5.84.2.2 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

<i>pickup</i>	The requested pickup Location
<i>dropoff</i>	The requested dropoff Location
<i>distance</i>	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](#).

5.85 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.

5.85.1 Detailed Description

An immutable Value Object describing metadata about a [TransitService](#).

Each [TransitService](#) is required to supply the following information.

5.85.2 Member Data Documentation

5.85.2.1 `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.

5.85.2.2 `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](#).

5.85.2.3 `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.

5.86 TransitProvider Class Reference

A [TransitProvider](#) is a description of a company or organization that is the producer of public transportation services.

Private Attributes

- `String providerId`
A unique identifier that globally identifies this [TransitProvider](#).
- `Set< TransitProviderObserver > transitProviderObserver`
An Observer Pattern mechanism to alertlink [TransitProviderObserver](#)s of changes in [Route](#) service;.
- `String name`
A text description of the [TransitProvider](#).
- `abstract void fireRouteDistruptionEvent ()`
This method is called internally by the [TransitProvider](#) to fire a [RouteDisruptionEvent](#).
- `String getProviderId ()`
- `void setProviderId (String providerId)`
- `String getName ()`
- `void setName (String name)`
- `void registerObserver (TransitProviderObserver transitProviderObserver)`
Register the provided [TransitProviderObserver](#) as an observer to this class.
- `void unregisterObserver (TransitProviderObserver transitProviderObserver)`
Unregister the provided [TransitProviderObserver](#) as an observer to this class.

5.86.1 Detailed Description

A [TransitProvider](#) is a description of a company or organization that is the producer of public transportation services.

Observers may subscribe to a [TransitProvider](#) to receive updates on [Route](#) disruptions, such as changes in service availability or schedule.

5.86.2 Member Function Documentation

5.86.2.1 `abstract void fireRouteDisruptionEvent () [protected],[pure virtual]`

This method is called internally by the [TransitProvider](#) to fire a [RouteDisruptionEvent](#).

[TransitProvider](#) subclasses will determine when these Events are fired – for example, if a [TransitProvider](#) has scheduled maintenance days, or known outages due to mechanical breakdown.

After a [RouteDisruptionEvent](#) is fired, this class will perform the following:

- Notify all [TransitProviderObservers](#) of the disruption with the updated [Route](#). This updated [Route](#) should include all necessary [Detour](#) information.

5.86.2.2 `void registerObserver (TransitProviderObserver transitProviderObserver)`

Register the provided [TransitProviderObserver](#) as an observer to this class.

Parameters

<i>transitProvider-Observer</i>	The TransitProviderObserver to unregister.
---------------------------------	--

5.86.2.3 `void unregisterObserver (TransitProviderObserver transitProviderObserver)`

Unregister the provided [TransitProviderObserver](#) as an observer to this class.

Parameters

<i>transitProvider-Observer</i>	The TransitProviderObserver to unregister.
---------------------------------	--

5.86.3 Member Data Documentation

5.86.3.1 `String name [private]`

A text description of the [TransitProvider](#).

This is the text that will be displayed on guides, [Route](#) maps, and advertisements.

5.86.3.2 `String providerId [private]`

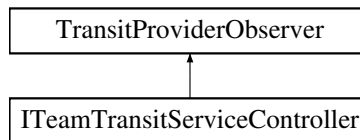
A unique identifier that globally identifies this [TransitProvider](#).

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.

5.87 TransitProviderObserver Interface Reference

An asynchronous update interface for receiving notifications about [TransitProvider Route](#) disruptions.

Inheritance diagram for TransitProviderObserver:



Public Member Functions

- void [handleRouteDisruptionEvent](#) ([RouteDisruptionEvent](#) routeDisruptionEvent)
This method is called when a [TransitProvider](#) fires a [RouteDisruptionEvent](#) indicating a disruption in normal services and/or schedules.

5.87.1 Detailed Description

An asynchronous update interface for receiving notifications about [TransitProvider Route](#) disruptions.

5.87.2 Member Function Documentation

5.87.2.1 void handleRouteDisruptionEvent ([RouteDisruptionEvent](#) routeDisruptionEvent)

This method is called when a [TransitProvider](#) fires a [RouteDisruptionEvent](#) indicating a disruption in normal services and/or schedules.

Parameters

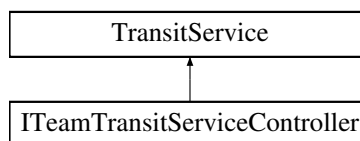
<i>routeDisruption-Event</i>	The Route Disruption Event
------------------------------	--

Implemented in [ITeamTransitServiceController](#).

5.88 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](#).

5.88.1 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.

5.88.2 Member Function Documentation

5.88.2.1 [Route](#) `getRoute (String routeId)`

Gets a [Route](#) by its unique identifier.

Parameters

<i>routeId</i>	The unique identifier of the Route
----------------	--

Returns

The matching [Route](#), or null if not found

Precondition

routeId is not null or blank.

Postcondition

The [Route](#) is returned if the **routeId** is found, else null.

Implemented in [ITeamTransitServiceController](#).

5.88.2.2 `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**.

Parameters

<i>pickup</i>	The requested dropoff Location
<i>dropoff</i>	the dropoff
<i>distance</i>	The distance (in miles) that each Route can deviate from the requested pickup or dropoff

Returns

The matching [Routes](#)

Precondition

pickup is not null or blank.

dropoff is not null or blank.

distance is non-negative. [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.

Implemented in [ITeamTransitServiceController](#).

5.88.2.3 `URL getServiceURL ()`

The URL that uniquely identifies this [TransitService](#).

In a REST environment, this might be the root of the REST API path.

Returns

The URL of this service

Implemented in [ITeamTransitServiceController](#).

5.88.2.4 TransitInfo getTransitInfo ()

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

Returns

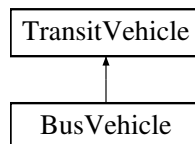
The [TransitInfo](#) of the Transit Authority of this service.

Implemented in [ITeamTransitServiceController](#).

5.89 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 [removeAlertSpecification](#) ()
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 range.

5.89.1 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.

5.89.2 Member Function Documentation

5.89.2.1 void addAlertSpecification ()

Add an alert specification [AlertSpecification](#) to this vehicle.

A vehicle may have these alerts:

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

5.89.2.2 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.

5.90 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.

5.90.1 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.

5.90.2 Member Function Documentation

5.90.2.1 `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 [GPSLocation-Tracking](#) service.

5.90.2.2 `int getGPSTypeFromURL (URL url) [private]`

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

Parameters

<code>url</code>	- URL from User Interface, contains GPS connection information.
------------------	---

Returns

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

5.90.2.3 `int getVehicleGPSDeviceID (URL url) [protected]`

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

Parameters

<code>url</code>	- URL identifying the transit company
------------------	---------------------------------------

Returns

integer GPS Device ID

5.91 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](#).

5.91.1 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.

5.92 TripInformation Class Reference

This model stores the information about a trip as a value object.

Public Member Functions

- List< String > [getRouteIds](#) ()
Provides a list of routeId in the [Trip](#) model.
- [Trip](#) [getTripData](#) ()
- void [setTripData](#) ([Trip](#) tripData)
- Date [getLastModifiedDate](#) ()
- void [setLastModifiedDate](#) (Date [lastModifiedDate](#))
- Date [getCreatedDate](#) ()
- void [setCreatedDate](#) (Date [createdDate](#))

Private Attributes

- [Trip](#) tripData
Necessary data about a trip.
- Date [lastModifiedDate](#)
Last date when the trip Information was modified or updated.
- Date [createdDate](#)
Date when the actual trip was created.

5.92.1 Detailed Description

This model stores the information about a trip as a value object.

Hence, we have created and last modified date to track the freshness of data. Currently, the [Trip](#) is referring to the model in Trip module. But we expect this to be stored as a value object and during implementation we can create a copy of its model for alert module.

5.92.2 Member Function Documentation

5.92.2.1 List<String> getRouteIds ()

Provides a list of routeId in the [Trip](#) model.

Returns

5.92.3 Member Data Documentation

5.92.3.1 Trip tripData [private]

Necessary data about a trip.

This contains a collection of routes. {

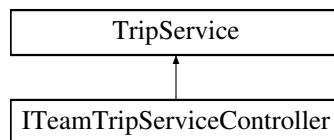
See Also

Trip}

5.93 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](#).

5.93.1 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](#).

5.93.2 Member Function Documentation

5.93.2.1 Trip calculateTrip (Location start, Location end)

Calculate an optimal [Trip](#) given a **start** [Location](#) and an **end** [Location](#).

Parameters

<i>start</i>	The requested start Location of the Trip .
<i>end</i>	The requested end Location of the Trip .

Returns

The calculated [Trip](#)

Implemented in [ITeamTripServiceController](#).

5.94 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

- `User (int userId, String username)`
This constructs a new `User` object.
- `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.

Private Attributes

- final int **userId**
- final String **username**
- String **password**
- String **passwordHash**
- boolean **forcePasswordChange**
- String **firstName**
- String **email**
- Short **countryCode**
- String **mobile**
- [UserType](#) **userType**

5.94.1 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.

5.94.2 Constructor & Destructor Documentation

5.94.2.1 `User (int userId, String username)` [protected]

This constructs a new [User](#) object.

It is not visible to clients, as [User](#) objects should only be constructed through the [UserRepository](#).

Parameters

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

5.94.3 Member Function Documentation

5.94.3.1 `Short getCountryCode ()`

This gets the user's country code.

Returns

user's country code

5.94.3.2 `String getEmail ()`

This retrieves the user's e-mail.

Returns

user's e-mail

5.94.3.3 `String getFirstName ()`

This retrieves the user's first name.

Returns

user's first name

5.94.3.4 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

5.94.3.5 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

5.94.3.6 int getUserId ()

This retrieves the user identifier.

Returns

user identifier

5.94.3.7 String getUsername ()

This retrieves the user's username.

Returns

username

5.94.3.8 UserType getUserType ()

This retrieves the type of the current user.

Returns

user type

5.94.3.9 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

5.94.3.10 void setCountryCode (Short *countryCode*)

This sets the user's country code.

Parameters

<i>countryCode</i>	user's country code
--------------------	---------------------

5.94.3.11 void setEmail (String *email*)

This sets the user's e-mail.

Parameters

<i>email</i>	user's e-mail
--------------	---------------

5.94.3.12 void setFirstName (String *firstName*)

This sets the user's first name.

Parameters

<i>firstName</i>	user's first name
------------------	-------------------

5.94.3.13 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

<i>forcePassword-Change</i>	true if it should be set, false if it should be cleared
-----------------------------	---

5.94.3.14 void setMobile (String *mobile*)

This sets the user's mobile phone number.

Precondition

The mobile parameter should be a String containing only digits.

Parameters

<i>mobile</i>	user's mobile phone number
---------------	----------------------------

5.94.3.15 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

<i>passwordHash</i>	hash of the user's password
---------------------	-----------------------------

5.94.3.16 void setUserType (UserType *userType*)

This sets the type of the current user.

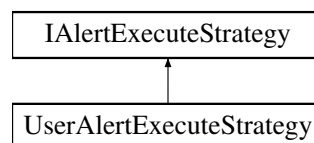
Parameters

<i>userType</i>	user type
-----------------	-----------

5.95 UserAlertExecuteStrategy Class Reference

A concrete implementation of [IAlertExecuteStrategy](#) that handles executing alert related to user.

Inheritance diagram for UserAlertExecuteStrategy:



Public Member Functions

- boolean [execute](#) ([Alert](#) alertModel)

For each passed in alert, it pushes notification to the user informing about their upcoming trip and routes (as they have scheduled in the first place)

Package Attributes

- [AlertRepository](#) [alertRepository](#)

An instance of [AlertRepository](#) that is used to fetch alerts that are effected by particular route.

- [AlertUserClient](#) [userClient](#)

A spring autowired instance of [AlertUserClient](#) that can call the User module to get user information.

5.95.1 Detailed Description

A concrete implementation of [IAlertExecuteStrategy](#) that handles executing alert related to user.

5.95.2 Member Data Documentation

5.95.2.1 `AlertRepository alertRepository` [package]

An instance of `AlertRepository` that is used to fetch alerts that are effected by particular route.

This is autowired via Spring Framework.

5.95.2.2 `AlertUserClient userClient` [package]

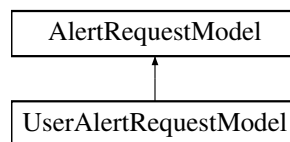
A spring autowired instance of `AlertUserClient` that can call the User module to get user information.

This is autowired via Spring Framework.

5.96 `UserAlertRequestModel` Class Reference

This model is a JSON representation of the request from User module.

Inheritance diagram for `UserAlertRequestModel`:



Additional Inherited Members

5.96.1 Detailed Description

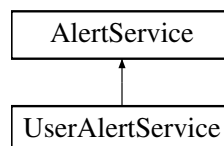
This model is a JSON representation of the request from User module.

During implementation, any additional data that is needed can be added.

5.97 `UserAlertService` Class Reference

Extends `AlertService` and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

Inheritance diagram for `UserAlertService`:



Public Member Functions

- `AlertResponseModel createAlert` (`AlertRequestModel` requestModel)
{
- `AlertResponseModel updateAlert` (`Alert` alertModel)
{

- boolean [sendAlert](#) ()
{

Additional Inherited Members

5.97.1 Detailed Description

Extends [AlertService](#) and provides CRUD methods to operate on alerts as well as sends alerts to execute via appropriate strategy.

This handles all the alerts initiated by User module.

5.97.2 Member Function Documentation

5.97.2.1 `AlertResponseModel createAlert (AlertRequestModel requestModel)` [virtual]

{

See Also

[AlertService::createAlert\(AlertRequestModel\)](#)}

Implements [AlertService](#).

5.97.2.2 `boolean sendAlert ()` [virtual]

{

See Also

[AlertService::sendAlert\(\)](#)}

Implements [AlertService](#).

5.97.2.3 `AlertResponseModel updateAlert (Alert alertModel)` [virtual]

{

See Also

[AlertService::updateAlert\(Alert\)](#)}

Implements [AlertService](#).

5.98 UserFavoritesList Class Reference

This class ties a userId to that user's ordered list of favorites.

Public Member Functions

- [UserFavoritesList](#) (int userId)
This creates a new [UserFavoritesList](#) object.
- int [getUserId](#) ()

This gets the `userId` associated with this favorites list.

- List< [FavoriteTransitService](#) > [getFavoriteTransitServices](#) ()

This retrieves the list of favorite transit services for this user.

- void [setFavoriteTransitServices](#) (List< [FavoriteTransitService](#) > `favoriteTransitServices`)

This sets the ordered list of favorite transit services for this user.

Private Attributes

- final int **`userId`**
- List< [FavoriteTransitService](#) > **`favoriteTransitServices`**

5.98.1 Detailed Description

This class ties a `userId` to that user's ordered list of favorites.

5.98.2 Constructor & Destructor Documentation

5.98.2.1 [UserFavoritesList](#) (int *userId*)

This creates a new [UserFavoritesList](#) object.

Once created, the `userId` cannot be modified.

Parameters

<i>userId</i>	<code>userId</code> to link this favorites object to
---------------	--

5.98.3 Member Function Documentation

5.98.3.1 List<[FavoriteTransitService](#)> [getFavoriteTransitServices](#) ()

This retrieves the list of favorite transit services for this user.

Returns

list of favorite transit services

5.98.3.2 int [getUserId](#) ()

This gets the `userId` associated with this favorites list.

Returns

`userId`

5.98.3.3 void [setFavoriteTransitServices](#) (List< [FavoriteTransitService](#) > *favoriteTransitServices*)

This sets the ordered list of favorite transit services for this user.

Parameters

<i>favoriteTransitServices</i>	list to use
--------------------------------	-------------

5.99 UserFavoritesRepository Class Reference

This class is responsible for handling database access for favorites, and to persist and retrieve [UserFavoritesList](#) objects.

Protected Member Functions

- [UserFavoritesList](#) `getFavorites` (int `userId`) throws `BusBuddyInternalException`, `BusBuddyNotFoundException`
This method retrieves the [UserFavoritesList](#) object for a given user.
- void `updateFavorites` (int `userId`, [UserFavoritesList](#) `favorites`) throws `BusBuddyInternalException`, `BusBuddyNotFoundException`
This method updates the [UserFavoritesList](#) object for a given user.

5.99.1 Detailed Description

This class is responsible for handling database access for favorites, and to persist and retrieve [UserFavoritesList](#) objects.

5.99.2 Member Function Documentation

5.99.2.1 `UserFavoritesList getFavorites (int userId) throws BusBuddyInternalException, BusBuddyNotFoundException [protected]`

This method retrieves the [UserFavoritesList](#) object for a given user.

Precondition

The `userId` passed in must have already saved favorites.

Parameters

<code>userId</code>	User to retrieve favorites for.
---------------------	---

Returns

Favorites object for the `userId` that was passed in.

Exceptions

<code>BusBuddyInternalException</code>	This exception is thrown when there is a database error.
<code>BusBuddyNotFoundException</code>	This exception is thrown if no data has been saved yet, or no such user exists.

5.99.2.2 `void updateFavorites (int userId, UserFavoritesList favorites) throws BusBuddyInternalException, BusBuddyNotFoundException [protected]`

This method updates the [UserFavoritesList](#) object for a given user.

It creates it if it doesn't exist, and overwrites it if it does.

Precondition

The `userId` must be valid.

Parameters

<code>userId</code>	User to set favorites for.
<code>favorties</code>	Favorites to set.

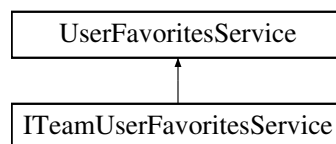
Exceptions

<code>BusBuddyInternalException</code>	This exception is thrown when there is a database error.
<code>BusBuddyNotFoundException</code>	This exception is thrown if no such user exists.

5.100 UserFavoritesService Interface Reference

This is the generic BusBuddy [UserFavoritesService](#) interface.

Inheritance diagram for UserFavoritesService:

**Public Member Functions**

- [UserFavoritesList](#) `readFavorites` (String `sessionToken`) throws `BusBuddyException`
This method retrieves the [UserFavoritesList](#) object for the current user of a given session.
- void `saveFavorites` (String `sessionToken`, [UserFavoritesList](#) `favorites`) throws `BusBuddyException`
This method updates the [UserFavoritesList](#) object for a given user.

5.100.1 Detailed Description

This is the generic BusBuddy [UserFavoritesService](#) interface.

This interface contains methods dealing with user favorites. 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.

Every method call here will (besides `createUser`) will result in the session's expiration time being updated due to activity on the session.

5.100.2 Member Function Documentation

5.100.2.1 [UserFavoritesList](#) `readFavorites` (String *sessionToken*) throws `BusBuddyException`

This method retrieves the [UserFavoritesList](#) object for the current user of a given session.

Precondition

The `userId` linked to the session must have already saved favorites.

Parameters

<i>sessionToken</i>	Session whose user favorites will be retrieved for.
---------------------	---

Returns

Favorites object for the `userId` that was passed in.

Exceptions

<i>BusBuddyInternalException</i>	This exception is thrown when there is an internal error.
<i>BusBuddyForbiddenException</i>	This exception is thrown when the current session is invalidated.
<i>BusBuddyNotFoundException</i>	This exception is thrown if no data has been saved yet.
<i>BusBuddyBadRequestException</i>	This exception is thrown when the <code>sessionToken</code> is blank or empty.

Implemented in [ITeamUserFavoritesService](#).

5.100.2.2 void saveFavorites (String sessionToken, UserFavoritesList favorites) throws BusBuddyException

This method updates the [UserFavoritesList](#) object for a given user.

It creates it if it doesn't exist, and overwrites it if it does.

Parameters

<i>sessionToken</i>	Session whose user the favorites will be retrieved for.
<i>favorites</i>	Favorites to set.

Exceptions

<i>BusBuddyInternalException</i>	This exception is thrown when there is an internal error.
<i>BusBuddyForbiddenException</i>	This exception is thrown when the current session is invalidated.
<i>BusBuddyBadRequestException</i>	This exception is thrown when the <code>sessionToken</code> is blank or empty, or the <code>userId</code> on the UserFavoritesList object passed in doesn't match the session.

Implemented in [ITeamUserFavoritesService](#).

5.101 UserInformation Class Reference

[UserInformation](#) contains the user related data that we get from User Module.

Public Member Functions

- int **getUserId** ()
- void **setUserId** (int userId)
- String **getUsername** ()
- void **setUsername** (String username)
- String **getPassword** ()
- void **setPassword** (String password)
- String **getPasswordHash** ()

- void **setPasswordHash** (String passwordHash)
- boolean **isForcePasswordChange** ()
- void **setForcePasswordChange** (boolean forcePasswordChange)
- String **getFirstName** ()
- void **setFirstName** (String firstName)
- String **getEmail** ()
- void **setEmail** (String email)
- Short **getCountryCode** ()
- void **setCountryCode** (Short countryCode)
- String **getMobile** ()
- void **setMobile** (String mobile)
- String **getUserType** ()
- void **setUserType** (String userType)

Private Attributes

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

5.101.1 Detailed Description

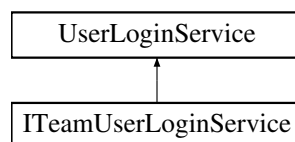
[UserInformation](#) contains the user related data that we get from User Module.

This is more or less an exact copy of the [User](#) class.

5.102 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

This method resets a user's password and sends them their new password via e-mail.

- void [resetPassword](#) (String username, short countryCode, String mobile) throws BusBuddyException

This method resets a user's password and sends them their new password via SMS.

5.102.1 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.

5.102.2 Member Function Documentation

5.102.2.1 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

<i>sessionToken</i>	The session token identifying the session that is creating the new alert session.
---------------------	---

Returns

[Session](#) token representing the new alert session.

Exceptions

<i>BusBuddyBadRequestException</i>	This exception is thrown if the session token is blank.
<i>BusBuddyForbiddenException</i>	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.
<i>BusBuddyInternalException</i>	This exception is thrown if an internal error prevents processing of the request.

Implemented in [ITeamUserLoginService](#).

5.102.2.2 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

<i>sessionToken</i>	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

<i>BusBuddyBadRequestException</i>	This exception is thrown if the session token is blank.
<i>BusBuddyForbiddenException</i>	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.
<i>BusBuddyInternalException</i>	This exception is thrown if an internal error prevents processing of the request.

Implemented in [ITeamUserLoginService](#).

5.102.2.3 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

<i>username</i>	Username of the user to login as.
<i>password</i>	Password of the user to login as.

Returns

session token of the new session

Exceptions

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

Implemented in [ITeamUserLoginService](#).

5.102.2.4 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

<i>sessionToken</i>	This is the session token that identifies the session.
---------------------	--

Exceptions

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

Implemented in [ITeamUserLoginService](#).

5.102.2.5 void resetPassword (String username, String email) throws BusBuddyException

This method resets a user's password and sends them their new password via e-mail.

Precondition

The username and e-mail address provided both link to the same user in the database.

Postcondition

The user's password is reset and sent to the user via e-mail.

Parameters

<i>username</i>	Username of the User to reset the password for.
<i>email</i>	E-mail address of the User to reset the password for.

Exceptions

<i>BusBuddyBadRequestException</i>	This exception is thrown if the username or e-mail address is blank.
<i>BusBuddyForbiddenException</i>	This exception is thrown if the username and e-mail address combination don't link to a valid user. Also thrown if the account is suspended or deleted.
<i>BusBuddyInternalException</i>	This exception is thrown if an internal error prevents processing of the request.

Implemented in [ITeamUserLoginService](#).

5.102.2.6 void resetPassword (String username, short countryCode, String mobile) throws BusBuddyException

This method resets a user's password and sends them their new password via SMS.

Precondition

The username and mobile information provided both link to the same user in the database.

Postcondition

The user's password is reset and sent to the user via SMS.

Parameters

<i>username</i>	Username of the User to reset the password for.
<i>countryCode</i>	country code of the user's mobile phone number
<i>mobile</i>	String representing the user's mobile phone number (String should consist entirely of digits)

Exceptions

<i>BusBuddyBadRequestException</i>	This exception is thrown if the username or mobile number is blank (or non numeric).
<i>BusBuddyForbiddenException</i>	This exception is thrown if the username and mobile information combination don't link to a valid user. Also thrown if the account is suspended or deleted.
<i>BusBuddyInternalException</i>	This exception is thrown if an internal error prevents processing of the request.

Implemented in [ITeamUserLoginService](#).

5.102.2.7 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

<i>email</i>	E-mail address of the account to send to.
--------------	---

Exceptions

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

Implemented in [ITeamUserLoginService](#).

5.102.2.8 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 SMS has been sent to the user, containing the user's username.

Parameters

<i>email</i>	E-mail address of the account to send to.
--------------	---

Exceptions

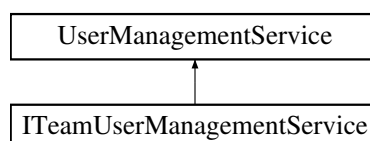
<i>BusBuddyBadRequestException</i>	This exception is thrown if the mobile is blank or non-numeric.
<i>BusBuddyForbiddenException</i>	This exception is thrown if the mobile information is linked to an account that is suspended or deleted.
<i>BusBuddyNotFoundException</i>	This exception is thrown if the mobile information doesn't link to a valid user.
<i>BusBuddyInternalException</i>	This exception is thrown if an internal error prevents processing of the request.

Implemented in [ITeamUserLoginService](#).

5.103 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`
This method creates a user in the database.
- **User** `findUserByUsername` (String sessionToken, String username) throws `BusBuddyException`
This method attempts to retrieve a user by username.
- **User** `findUserByEmail` (String sessionToken, String email) throws `BusBuddyException`
This method attempts to retrieve a user by e-mail address.
- **User** `findUserByMobile` (String sessionToken, short countryCode, String mobile) throws `BusBuddyException`
This method attempts to retrieve a user by mobile phone number.
- void `updateUser` (String sessionToken, **User** newUserData, String password) throws `BusBuddyException`
This method updates a user in the database.
- void `deleteUser` (String sessionToken, **User** userToDelete) throws `BusBuddyException`
This method deletes a user from the database.

5.103.1 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.

Every method call here will (besides `createUser`) will result in the session's expiration time being updated due to activity on the session.

5.103.2 Member Function Documentation

5.103.2.1 `User` `createUser` (`User` userToCreate, String password) throws `BusBuddyException`

This method creates a user in the database.

Precondition

No other user with this username, e-mail address, or mobile phone exists.

Postcondition

`User` is created with the given user data.

Parameters

<i>user</i>	<code>User</code> data of the user to create (the ID will be ignored).
<i>password</i>	Password of the user to create.

Returns

new user object

Exceptions

<i><code>BusBuddyInternalException</code></i>	This exception is thrown when an internal error prevents creation of the user.
<i><code>BusBuddyConflictException</code></i>	This exception is thrown when the requested user record would create a duplicate username, e-mail address, or mobile phone data in the database.

Implemented in [ITeamUserManagementService](#).

5.103.2.2 void deleteUser (String sessionToken, User userToDelete) throws BusBuddyException

This method deletes a user from the database.

It will delete the user with the same ID as the user passed in as a parameter.

Precondition

A user with the specified user ID on the [User](#) object must already exist.

Postcondition

[User](#) object in database will be deleted.

Parameters

<i>sessionToken</i>	session token for the currently logged in user
<i>userToDelete</i>	This user object should have the same ID as the user to be deleted.

Exceptions

<i>BusBuddyInternalException</i>	This exception is thrown when there is an internal error preventing execution of the request.
<i>BusBuddyForbidden-Exception</i>	This exception is thrown if the session token is invalid, linked to an expired session, or the user does not have permission to make this change.
<i>BusBuddyNotFound-Exception</i>	This exception is thrown if the targeted user is not found in the database.

Implemented in [ITeamUserManagementService](#).

5.103.2.3 User findUserByEmail (String sessionToken, String email) throws BusBuddyException

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

<i>email</i>	This is the e-mail address to look up.
--------------	--

Returns

The user with the given e-mail address.

Exceptions

<i>BusBuddyInternalException</i>	An internal error prevents execution of the request.
<i>BusBuddyForbiddenException</i>	The currently logged in user does not have permission to view the result of this search.
<i>BusBuddyNotFoundException</i>	This exception is thrown when the requested user record could not be found.

Implemented in [ITeamUserManagementService](#).

5.103.2.4 User findUserByMobile (String *sessionToken*, short *countryCode*, String *mobile*) throws BusBuddyException

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

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

Returns

The user with the given mobile phone details.

Exceptions

<i>BusBuddyInternalException</i>	An internal error prevents execution of the request.
<i>BusBuddyForbiddenException</i>	The currently logged in user does not have permission to view the result of this search.
<i>BusBuddyNotFoundException</i>	This exception is thrown when the requested user record could not be found.

Implemented in [ITeamUserManagementService](#).

5.103.2.5 User findUserByUsername (String *sessionToken*, String *username*) throws BusBuddyException

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

<i>username</i>	This is the username to look up.
-----------------	----------------------------------

Returns

The user with the given username.

Exceptions

<i>BusBuddyInternalException</i>	An internal error prevents execution of the request.
<i>BusBuddyForbiddenException</i>	The currently logged in user does not have permission to view the result of this search.
<i>BusBuddyNotFoundException</i>	This exception is thrown when the requested user record could not be found.

Implemented in [ITeamUserManagementService](#).

5.103.2.6 void updateUser (String sessionToken, User newUserData, String password) throws BusBuddyException

This method updates a user in the database.

It will update the user with the same ID as the user passed in as a parameter. The username will not be updated, but all other fields will be.

Precondition

A user with the specified user ID on the [User](#) object must already exist.

Postcondition

[User](#) object in database will be updated with the data from the parameter [User](#) object.

Parameters

<i>sessionToken</i>	session token for the currently logged in user
<i>newUserData</i>	User object with the new user data on it
<i>password</i>	New password to set on the user.

Exceptions

<i>BusBuddyInternalException</i>	This exception is thrown when there is an internal error preventing execution of the request.
<i>BusBuddyForbiddenException</i>	This exception is thrown if the session token is invalid, linked to an expired session, or the user does not have permission to make this change.
<i>BusBuddyNotFoundException</i>	This exception is thrown when the targeted user to receive the update does not exist.
<i>BusBuddyConflictException</i>	This exception is thrown when the requested change would create a duplicate mobile phone or e-mail address in the database.

Implemented in [ITeamUserManagementService](#).

5.104 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](#) ([User](#) user, String password) throws [BusBuddyConflictException](#), [BusBuddyInternalException](#)
This method creates a user in the database.
- [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) throws [BusBuddyInternalException](#), [BusBuddyNotFoundException](#), - [BusBuddyConflictException](#)
This method updates a user in the database.
- void [deleteUser](#) ([User](#) userToDelete) throws [BusBuddyInternalException](#), [BusBuddyNotFoundException](#)
This method deletes a user from the database.

5.104.1 Detailed Description

This class is responsible for handling database access for [User](#) objects, and to construct, persist, and retrieve [User](#) objects.

5.104.2 Member Function Documentation

5.104.2.1 [User](#) [createUser](#) ([User](#) user, String password) throws [BusBuddyConflictException](#), [BusBuddyInternalException](#) [package]

This method creates a user in the database.

Precondition

No other user with this username, e-mail address, or mobile phone exists.

Postcondition

[User](#) is created with the given user data.

Parameters

<i>user</i>	User data of the user to create (the ID will be ignored).
<i>password</i>	Password of the user to create.

Returns

new user object

Exceptions

<i>BusBuddyInternalException</i>	This exception is thrown when there is a database error.
<i>BusBuddyConflictException</i>	This exception is thrown when the requested user record would create a duplicate username, e-mail address, or mobile phone data in the database.

5.104.2.2 void deleteUser (User userToDelete) throws BusBuddyInternalException, BusBuddyNotFoundException [package]

This method deletes a user from the database.

It will delete the user with the same ID as the user passed in as a parameter.

Precondition

A user with the specified user ID on the [User](#) object must already exist.

Postcondition

[User](#) object in database will be deleted.

Exceptions

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

5.104.2.3 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

<i>email</i>	This is the e-mail address to look up.
--------------	--

Returns

The user with the given e-mail address.

Exceptions

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

5.104.2.4 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

<i>userId</i>	This is the user ID to look up.
---------------	---------------------------------

Returns

The user with the given ID.

Exceptions

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

5.104.2.5 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

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

Returns

The user with the given mobile phone details.

Exceptions

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

5.104.2.6 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

<i>username</i>	This is the username to look up.
-----------------	----------------------------------

Returns

The user with the given username.

Exceptions

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

5.104.2.7 void `updateUser (User newUserData)` throws `BusBuddyInternalException`, `BusBuddyNotFoundException`, `BusBuddyConflictException` [package]

This method updates a user in the database.

It will update the user with the same ID as the user passed in as a parameter. The username will not be updated, but all other fields will be.

Precondition

A user with the specified user ID on the `User` object must already exist.

Postcondition

`User` object in database will be updated with the data from the parameter `User` object.

Parameters

<i>newUserData</i>	User object with the new user data on it
--------------------	--

Exceptions

<i>BusBuddyInternalException</i>	This exception is thrown when there is a database error.
<i>BusBuddyNotFoundException</i>	This exception is thrown when the requested user record could not be found.
<i>BusBuddyConflictException</i>	This exception is thrown when the requested user record would create a duplicate mobile phone or e-mail address in the database.

5.105 UserSessionInformation Class Reference

A model that stores all the information needed to call user module about user information.

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.

5.105.1 Detailed Description

A model that stores all the information needed to call user module about user information.

5.105.2 Member Data Documentation

5.105.2.1 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.

5.106 UserTrackingAlertObject Class Reference

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

Public Member Functions

- URL **getTransitColInfo** ()
- int **getRouteID** ()

Private Attributes

- URL [transitColInfo](#)
URL uniquely identifies transit company information.
- int [routeID](#)
Route number user is watching for vehicles.
- Location [stopLocation](#)
GPS coordinates of vehicle stop closest to user.
- Date [scheduledTime](#)
Time vehicle is scheduled to be at closest stop.
- Date [alertTime](#)
Amount of lead time user needs to get to the vehicle stop.
- String [userContactInfo](#)
How to contact a user with an alert.
- [AlertType](#) type
Type of alert the user is registered.

5.106.1 Detailed Description

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

Primarily, this meta data will come from the user interface and stored in user module.

5.106.2 Member Data Documentation

5.106.2.1 Date [alertTime](#) [private]

Amount of lead time user needs to get to the vehicle stop.

Obtained from the user interface when the user signs up for an alert.

5.106.2.2 int [routeID](#) [private]

Route number user is watching for vehicles.

Obtained by translating user interface route description drop down to transit information route id.

5.106.2.3 Date [scheduledTime](#) [private]

Time vehicle is scheduled to be at closest stop.

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

5.106.2.4 Location [stopLocation](#) [private]

GPS coordinates of vehicle stop closest to user.

Obtained from information uploaded to transit module by a transit company.

5.106.2.5 URL [transitColInfo](#) [private]

URL uniquely identifies transit company information.

The transit company URL information to distinguish this vehicle from other vehicles in different cities with similar route numbers. Obtained from the user interface when the user registers for an alert, selected from a drop down derived from metadata {}.

5.106.2.6 `AlertType` type [private]

Type of alert the user is registered.

A user may sign up for tracking alerts when a vehicle is approaching their stop. Transit companies may sign up for delay alerts when their vehicle(s) are running behind schedule.

5.106.2.7 `String` `userContactInfo` [private]

How to contact a user with an alert.

User contact information (phone number or URL) where to send tracking alerts. Obtained from user interface when user signs up for an alert. On Bus Buddy system start, obtained from alert repository.

5.107 `UserType` Enum Reference

This is an enumeration of the different statuses that a user can be assigned.

Public Attributes

- `NORMAL_USER`
This is a standard user.
- `SYSTEM_ADMINISTRATOR`
This is a system administrator, who can read and write to other users' data.
- `SUSPENDED_USER`
A user account marked as suspended cannot create or use sessions.

5.107.1 Detailed Description

This is an enumeration of the different statuses that a user can be assigned.

5.108 `VehicleObject` Class Reference

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

Public Member Functions

- `String` `getGPSDeviceInfo` ()
- `URL` `getTransitCoURL` ()
- `int` `getGPSDeviceID` ()
- `int` `getCurrentRoute` ()

Private Attributes

- `int` `gpsDeviceID`
GPS hardware device ID.
- `String` `gpsDeviceInfo`
GPS device contact information, commercial web URL, GPS wireless connection or port number.
- `URL` `transitCoURL`
Transit company operating this vehicle.

- `int currentRoute`
Current route number.

5.108.1 Detailed Description

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

5.108.2 Member Data Documentation

5.108.2.1 `int currentRoute` `[private]`

Current route number.

Obtain and updated by the user interface.

5.108.2.2 `int gpsDeviceID` `[private]`

GPS hardware device ID.

Hardware GPS device ID, uniquely identifies a GPS unit. Obtained from user interface when a vehicle registers for tracking.

5.108.2.3 `String gpsDeviceInfo` `[private]`

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

Obtained from user interface when a vehicle is registered for tracking.

5.108.2.4 `URL transitCoURL` `[private]`

Transit company operating this vehicle.

Obtained from the information uploaded to TransitInfo by the transit company. User selects transit company name from a drop down on the user interface when registering a vehicle.

5.109 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.

5.109.1 Detailed Description

Repository for information on vehicles registered on a route.

5.109.2 Member Function Documentation

5.109.2.1 [TransitVehicle](#) findVehicle (int *gpsDeviceID*)

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

Parameters

<i>gpsDeviceID</i>	- integer GPS device ID
--------------------	-------------------------

Returns

[VehicleObject](#) matching vehicle or null if no matching vehicle found.

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

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

Parameters

<i>transitCoURL</i>	URL or the transit company
<i>routeID</i>	- 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.

5.109.2.3 void removeVehicle (int *gpsDeviceID*)

Remove a vehicle from the repository.

Parameters

<i>gpsDeviceID</i>	- integer the GPS id if the vehicle to remove.
--------------------	--

5.109.2.4 void updateVehicle (TransitVehicle *vehicle*)

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

Parameters

<i>vehicle</i>	- VehicleObject new vehicle information from driver through UI
----------------	--

Index

- AbstractFeedParserTemplate, [23](#)
- addAlertSpecification
 - tracking::TransitVehicle, [117](#)
- addUserTrackingAlert
 - tracking::ITrackingService, [78](#)
 - tracking::TrackingServiceController, [107](#)
- Alert, [25](#)
- alert.client, [13](#)
- alert.client.model, [13](#)
- alert.controller, [14](#)
- alert.controller.model, [14](#)
- alert.domain, [15](#)
- alert.domain.model, [15](#)
- alert.enums, [16](#)
- alert.service, [16](#)
- alert::client::AlertUserClient
 - getUserInformation, [43](#)
- alert::controller::AlertRequestController
 - processTrackingAlertRequest, [34](#)
 - processTransitAlertRequest, [34](#)
 - processUserAlertRequest, [34](#)
 - verifySession, [35](#)
- alert::controller::CertificateHandler
 - verifySessionToken, [51](#)
- alert::controller::ISessionHandler
 - verifySessionToken, [69](#)
- alert::controller::SessionTokenHandler
 - verifySessionToken, [97](#)
- alert::controller::SessionVerificationFactory
 - getSessionTokenVerificationStrategy, [97](#)
- alert::controller::model::AlertRequestModel
 - alertInitiator, [36](#)
- alert::domain::AlertFactory
 - createAlert, [28](#)
- alert::domain::AlertRepository
 - deleteAlert, [31](#)
 - getAlertByDateTime, [31](#)
 - getAlertByRoute, [32](#)
 - getAlertByUserId, [32](#)
 - saveAlert, [32](#)
 - updateAlert, [32](#)
- alert::domain::model::Alert
 - alertInitiator, [27](#)
 - alertRunType, [27](#)
 - alertType, [27](#)
 - Status, [27](#)
 - userInformation, [27](#)
- alert::domain::model::OneTimeAlert
 - dateExecuted, [81](#)
- alert::domain::model::RecurringAlert
 - alertRecurringType, [84](#)
 - repeatEvery, [84](#)
- alert::domain::model::RecurringData
 - dayOfMonth, [85](#)
 - dayOfWeek, [85](#)
 - dayOfYear, [85](#)
 - startHour, [85](#)
 - startMinute, [85](#)
- alert::domain::model::TripInformation
 - getRouteIds, [119](#)
 - tripData, [120](#)
- alert::domain::model::UserSessionInformation
 - userSessionToken, [146](#)
- alert::enums::AlertNotificationType
 - PlannedDisruption, [29](#)
 - ScheduleInformation, [29](#)
 - UnplannedDisruption, [30](#)
- alert::enums::AlertStatus
 - Deactive, [42](#)
 - Error, [42](#)
 - Expired, [42](#)
- alert::service::AlertExecuteStrategyFactory
 - getAlertService, [28](#)
- alert::service::AlertService
 - alertExecuteStrategyFactory, [39](#)
 - alertRepository, [39](#)
 - createAlert, [37](#)
 - deleteAlert, [38](#)
 - saveAlert, [38](#)
 - sendAlert, [38](#)
 - updateAlert, [38](#)
- alert::service::AlertServiceFactory
 - getAlertService, [40](#)
 - trackingAlertService, [40](#)
 - transitAlertService, [40](#)
 - userAlertService, [40](#)
- alert::service::IAlertExecuteStrategy
 - execute, [67](#)
- alert::service::RouteAlertExecuteStrategy
 - alertRepository, [87](#)
 - execute, [87](#)
 - userClient, [87](#)
- alert::service::TrackingAlertService
 - createAlert, [103](#)
 - sendAlert, [103](#)
 - updateAlert, [104](#)
- alert::service::TransitAlertService
 - createAlert, [109](#)

- sendAlert, 109
 - updateAlert, 109
- alert::service::UserAlertExecuteStrategy
 - alertRepository, 126
 - userClient, 126
- alert::service::UserAlertService
 - createAlert, 127
 - sendAlert, 127
 - updateAlert, 127
- AlertExecuteStrategyFactory, 27
- alertExecuteStrategyFactory
 - alert::service::AlertService, 39
- AlertFactory, 28
- AlertInitiator, 29
- alertInitiator
 - alert::controller::model::AlertRequestModel, 36
 - alert::domain::model::Alert, 27
- alertList
 - tracking::BusVehicle, 50
- AlertNotificationType, 29
- AlertRangeLogic, 30
- AlertRecurringType, 30
- alertRecurringType
 - alert::domain::model::RecurringAlert, 84
- AlertRepository, 31
- alertRepository
 - alert::service::AlertService, 39
 - alert::service::RouteAlertExecuteStrategy, 87
 - alert::service::UserAlertExecuteStrategy, 126
- AlertRequestController, 33
- alertRequestController
 - transit::ITeamTransitServiceController, 71
- AlertRequestModel, 35
- AlertResponseModel, 36
- AlertRunType, 36
- alertRunType
 - alert::domain::model::Alert, 27
- AlertService, 37
- AlertServiceFactory, 39
- AlertSpecification, 40
- AlertStatus, 41
- alertTime
 - tracking::UserTrackingAlertObject, 147
- AlertType, 42
- alertType
 - alert::domain::model::Alert, 27
- AlertUserClient, 42
- and
 - common::Specification< T >, 98
- BaseController, 43
- BusBuddyBadRequestException, 44
- BusBuddyConflictException, 45
- BusBuddyException, 46
- BusBuddyForbiddenException, 47
- BusBuddyInternalException, 48
- BusBuddyNotFoundException, 49
- BusVehicle, 49
- calculateTrip
 - transit::ITeamTripServiceController, 72
 - transit::TripService, 120
- cause
 - transit::Detour, 54
- CertificateHandler, 50
- checkForAlerts
 - tracking::TransitVehicle, 117
- checkPermissions
 - user::ITeamUserLoginService, 74
- CommercialTracking, 51
 - tracking::CommercialTracking, 52
- CommercialTracking.CommercialTrackingHolder, 52
- common, 17
- common::BaseController
 - handleBusBuddyException, 43
 - handleGenericException, 44
- common::BusBuddyBadRequestException
 - getHttpCode, 45
- common::BusBuddyConflictException
 - getHttpCode, 46
- common::BusBuddyException
 - getHttpCode, 47
- common::BusBuddyForbiddenException
 - getHttpCode, 47
- common::BusBuddyInternalException
 - getHttpCode, 48
- common::BusBuddyNotFoundException
 - getHttpCode, 49
- common::HashUtility
 - hash, 66
- common::MessageDeliveryUtility
 - sendEmail, 80
 - sendSms, 80
- common::Specification< T >
 - and, 98
 - isSatisfiedBy, 98
 - not, 99
 - or, 99
- convertJSONAlertInput
 - tracking::TrackingResponseModel, 106
- convertJSONVehicleInput
 - tracking::TrackingResponseModel, 106
- createAlert
 - alert::domain::AlertFactory, 28
 - alert::service::AlertService, 37
 - alert::service::TrackingAlertService, 103
 - alert::service::TransitAlertService, 109
 - alert::service::UserAlertService, 127
- createAlertObserver
 - tracking::TrackingAlertFactory, 101
- createAlertSession
 - user::ITeamUserLoginService, 75
 - user::UserLoginService, 133
- createSession
 - user::SessionRepository, 95
- createTransitVehicle
 - tracking::TransitVehicleFactory, 118

- createUser
 - user::ITeamUserManagementService, 76
 - user::UserManagementService, 138
 - user::UserRepository, 142
- currentRoute
 - tracking::VehicleObject, 149
- dateExecuted
 - alert::domain::model::OneTimeAlert, 81
- dayOfMonth
 - alert::domain::model::RecurringData, 85
- dayOfWeek
 - alert::domain::model::RecurringData, 85
- dayOfYear
 - alert::domain::model::RecurringData, 85
- Deactive
 - alert::enums::AlertStatus, 42
- DelayAlertLogic, 53
- delete
 - transit::RouteRepository, 90
- deleteAlert
 - alert::domain::AlertRepository, 31
 - alert::service::AlertService, 38
- deleteUser
 - user::ITeamUserManagementService, 76
 - user::UserManagementService, 139
 - user::UserRepository, 143
- description
 - transit::Stop, 100
- Detour, 53
- detours
 - transit::Route, 86
- Error
 - alert::enums::AlertStatus, 42
- execute
 - alert::service::AlertExecuteStrategy, 67
 - alert::service::RouteAlertExecuteStrategy, 87
- Expired
 - alert::enums::AlertStatus, 42
- Fare, 54
- FavoriteTransitService, 55
 - user::FavoriteTransitService, 55
- findUserByEmail
 - user::ITeamUserManagementService, 77
 - user::UserManagementService, 139
- findUserByMobile
 - user::ITeamUserManagementService, 77
 - user::UserManagementService, 140
- findUserByUsername
 - user::ITeamUserManagementService, 77
 - user::UserManagementService, 140
- findVehicle
 - tracking::VehicleRepository, 150
- findVehiclesByRoute
 - tracking::VehicleRepository, 150
- fireRouteDistruptionEvent
 - transit::TransitProvider, 113
- formatJSONResponse
 - tracking::TrackingResponseModel, 106
- GPSLocationObject, 58
- GPSLocationObserver, 59
- GPSLocationTracking, 60
- GPSPuller, 61
 - tracking::GPSPuller, 62
- GPSPuller.GPSPullerHolder, 62
- GPSPusher, 62
 - tracking::GPSPusher, 64
- GPSPusher.GPSPusherHolder, 64
- GPSVehicleTracker, 64
 - tracking::GPSVehicleTracker, 65
- GTFSFeedParser, 65
- getAlertByDateTime
 - alert::domain::AlertRepository, 31
- getAlertByRoute
 - alert::domain::AlertRepository, 32
- getAlertByUserId
 - alert::domain::AlertRepository, 32
- getAlertService
 - alert::service::AlertExecuteStrategyFactory, 28
 - alert::service::AlertServiceFactory, 40
- getAll
 - transit::RouteRepository, 90
- getCountryCode
 - user::User, 122
- getCreationTime
 - user::Session, 93
- getEmail
 - user::User, 122
- getExpirationTime
 - user::Session, 93
- getFavoriteRouteIds
 - user::FavoriteTransitService, 55
- getFavoriteTransitServices
 - user::UserFavoritesList, 128
- getFavorites
 - user::UserFavoritesRepository, 129
- getFirstName
 - user::User, 122
- getGPSLocation
 - tracking::GPSLocationObserver, 59
- getGPSTypeFromURL
 - tracking::TransitVehicleFactory, 118
- getHttpCode
 - common::BusBuddyBadRequestException, 45
 - common::BusBuddyConflictException, 46
 - common::BusBuddyException, 47
 - common::BusBuddyForbiddenException, 47
 - common::BusBuddyInternalException, 48
 - common::BusBuddyNotFoundException, 49
- getInstance
 - tracking::CommercialTracking, 52
 - tracking::GPSPusher, 64
- getMobile
 - user::User, 123
- getPasswordHash

- user::User, 123
- getRoute
 - transit::GoogleTransitServiceAdapter, 57
 - transit::ITeamTransitServiceController, 70
 - transit::PersistedTransitFeed, 82
 - transit::TransitFeed, 110
 - transit::TransitService, 115
- getRouteIds
 - alert::domain::model::TripInformation, 119
- getRoutes
 - transit::GoogleTransitServiceAdapter, 57
 - transit::ITeamTransitServiceController, 70
 - transit::PersistedTransitFeed, 82
 - transit::TransitFeed, 110
 - transit::TransitService, 115
- getServiceURL
 - transit::ITeamTransitServiceController, 71
 - transit::TransitService, 115
- getSession
 - user::SessionRepository, 95
- getSessionToken
 - user::Session, 94
- getSessionTokenVerificationStrategy
 - alert::controller::SessionVerificationFactory, 97
- getSpec
 - tracking::TrackingAlertObserver, 102
- getStopTimes
 - transit::Stop, 100
- getTransitServiceUrl
 - user::FavoriteTransitService, 56
- getTransitVehicleLocation
 - tracking::ITrackingService, 78
- getTransitInfo
 - transit::ITeamTransitServiceController, 71
 - transit::TransitService, 116
- getUser
 - user::ITeamUserLoginService, 75
 - user::UserLoginService, 134
- getUserByEmail
 - user::UserRepository, 143
- getUserById
 - user::UserRepository, 144
- getUserByMobile
 - user::UserRepository, 144
- getUserByUsername
 - user::UserRepository, 145
- getUserId
 - user::Session, 94
 - user::User, 123
 - user::UserFavoritesList, 128
- getUserType
 - user::User, 123
- getUsername
 - user::User, 123
- getVehicleGPSDeviceID
 - tracking::TransitVehicleFactory, 118
- getUserInformation
 - alert::client::AlertUserClient, 43
- GoogleTransitServiceAPI, 58
- GoogleTransitServiceAdapter, 56
 - transit::GoogleTransitServiceAdapter, 57
- gpsDeviceID
 - tracking::VehicleObject, 149
- gpsDeviceInfo
 - tracking::VehicleObject, 149
- gpsUpdate
 - tracking::GPSLocationObserver, 59
- handleBusBuddyException
 - common::BaseController, 43
- handleGenericException
 - common::BaseController, 44
- handleRouteDisruptionEvent
 - transit::ITeamTransitServiceController, 71
 - transit::TransitProviderObserver, 114
- hash
 - common::HashUtility, 66
- HashUtility, 66
- IAAlertExecuteStrategy, 66
- ISessionHandler, 68
- ITeamTransitServiceController, 69
- ITeamTripServiceController, 72
- ITeamUserFavoritesService, 72
- ITeamUserLoginService, 73
- ITeamUserManagementService, 76
- ITrackingService, 77
- inAlertRange
 - tracking::AlertSpecification, 41
- InvalidRouteParseException, 67
 - transit::InvalidRouteParseException, 68
- isAlertSession
 - user::Session, 94
- isFavoriteTransitService
 - user::FavoriteTransitService, 56
- isForcePasswordChange
 - user::User, 123
- isSatisfiedBy
 - common::Specification< T >, 98
 - transit::RouteSpecification, 92
- isValid
 - user::Session, 94
- killSession
 - user::SessionRepository, 96
- loadFeed
 - transit::AbstractFeedParserTemplate, 24
- Location, 79
 - transit::Location, 79
- login
 - user::ITeamUserLoginService, 75
 - user::UserLoginService, 134
- logo
 - transit::TransitInfo, 112
- logout
 - user::ITeamUserLoginService, 75

- user::UserLoginService, 135
- MessageDeliveryUtility, 80
- name
 - transit::TransitProvider, 113
- not
 - common::Specification< T >, 99
- OneTimeAlert, 81
- or
 - common::Specification< T >, 99
- parseFeed
 - transit::AbstractFeedParserTemplate, 24
 - transit::GTFSFeedParser, 65
- PersistedTransitFeed, 81
- PlannedDisruption
 - alert::enums::AlertNotificationType, 29
- processTrackingAlertRequest
 - alert::controller::AlertRequestController, 34
- processTransitAlertRequest
 - alert::controller::AlertRequestController, 34
- processUserAlertRequest
 - alert::controller::AlertRequestController, 34
- providerId
 - transit::TransitProvider, 113
- read
 - transit::RouteRepository, 90
- readFavorites
 - user::ITeamUserFavoritesService, 73
 - user::UserFavoritesService, 130
- RecurringAlert, 83
- RecurringData, 84
- registerGPSDevice
 - tracking::GPSLocationTracking, 60
- registerObserver
 - transit::TransitProvider, 113
- registerVehicleOnRoute
 - tracking::ITrackingService, 78
 - tracking::TrackingServiceController, 107
- removeVehicle
 - tracking::VehicleRepository, 150
- repeatEvery
 - alert::domain::model::RecurringAlert, 84
- resetPassword
 - user::ITeamUserLoginService, 75
 - user::UserLoginService, 135, 136
- Route, 86
- RouteAlertExecuteStrategy, 87
- routeBatch
 - transit::InvalidRouteParseException, 68
- RouteDisruptionAlert, 88
- RouteDisruptionEvent, 88
 - transit::RouteDisruptionEvent, 89
- routeId
 - tracking::UserTrackingAlertObject, 147
- routeId
 - transit::RouteDisruptionAlert, 88
- routeName
 - transit::Route, 86
- RouteRepository, 89
- RouteSpecification, 91
- save
 - transit::RouteRepository, 90, 91
- saveAlert
 - alert::domain::AlertRepository, 32
 - alert::service::AlertService, 38
- saveFavorites
 - user::ITeamUserFavoritesService, 73
 - user::UserFavoritesService, 131
- saveRoutes
 - transit::AbstractFeedParserTemplate, 24
- ScheduleInformation
 - alert::enums::AlertNotificationType, 29
- scheduledTime
 - tracking::UserTrackingAlertObject, 147
- sendAlert
 - alert::service::AlertService, 38
 - alert::service::TrackingAlertService, 103
 - alert::service::TransitAlertService, 109
 - alert::service::UserAlertService, 127
- sendEmail
 - common::MessageDeliveryUtility, 80
- sendSms
 - common::MessageDeliveryUtility, 80
- sendUsername
 - user::ITeamUserLoginService, 75, 76
 - user::UserLoginService, 136, 137
- serialVersionUID
 - transit::InvalidRouteParseException, 68
- Session, 92
 - user::Session, 93
- SessionRepository, 95
- SessionTokenHandler, 97
- SessionVerificationFactory, 97
- setCountryCode
 - user::User, 124
- setDiscountedFare
 - transit::Fare, 54
- setEmail
 - user::User, 124
- setExpirationTime
 - user::Session, 94
- setFavoriteRouteIds
 - user::FavoriteTransitService, 56
- setFavoriteTransitService
 - user::FavoriteTransitService, 56
- setFavoriteTransitServices
 - user::UserFavoritesList, 128
- setFirstName
 - user::User, 124
- setForcePasswordChange
 - user::User, 124
- setGPSLocation
 - tracking::GPSLocationObserver, 60

- setMobile
 - user::User, 124
- setPasswordHash
 - user::User, 125
- setRegularFare
 - transit::Fare, 54
- setSpec
 - tracking::TrackingAlertObserver, 102
- setUserType
 - user::User, 125
- setValid
 - user::Session, 94
- Specification< T >, 98
- start
 - transit::AbstractFeedParserTemplate, 25
- startHour
 - alert::domain::model::RecurringData, 85
- startMinute
 - alert::domain::model::RecurringData, 85
- startTrackingController
 - tracking::ITrackingService, 79
- Status
 - alert::domain::model::Alert, 27
- Stop, 99
- stopLocation
 - tracking::UserTrackingAlertObject, 147
- stops
 - transit::Route, 86
- tracking, 18
- tracking::AlertSpecification
 - inAlertRange, 41
- tracking::BusVehicle
 - alertList, 50
- tracking::CommercialTracking
 - CommercialTracking, 52
 - getInstance, 52
- tracking::GPSLocationObserver
 - getGPSLocation, 59
 - gpsUpdate, 59
 - setGPSLocation, 60
- tracking::GPSLocationTracking
 - registerGPSDevice, 60
 - unregisterGPSDevice, 61
- tracking::GPSPuller
 - GPSPuller, 62
- tracking::GPSPusher
 - GPSPusher, 64
 - getInstance, 64
- tracking::GPSVehicleTracker
 - GPSVehicleTracker, 65
- tracking::ITrackingService
 - addUserTrackingAlert, 78
 - getTransitVehicleLocation, 78
 - registerVehicleOnRoute, 78
 - startTrackingController, 79
 - unregisterVehicleFromRoute, 79
- tracking::TrackingAlertFactory
 - createAlertObserver, 101
- tracking::TrackingAlertObserver
 - getSpec, 102
 - setSpec, 102
 - updateAlert, 102
- tracking::TrackingDelayAlert
 - updateAlert, 104
- tracking::TrackingLocationAlert
 - TrackingLocationAlert, 105
- tracking::TrackingResponseModel
 - convertJSONAlertInput, 106
 - convertJSONVehicleInput, 106
 - formatJSONResponse, 106
- tracking::TrackingServiceController
 - addUserTrackingAlert, 107
 - registerVehicleOnRoute, 107
 - unregisterVehicleFromRoute, 108
- tracking::TransitVehicle
 - addAlertSpecification, 117
 - checkForAlerts, 117
- tracking::TransitVehicleFactory
 - createTransitVehicle, 118
 - getGPSTypeFromURL, 118
 - getVehicleGPSDeviceID, 118
- tracking::UserTrackingAlertObject
 - alertTime, 147
 - routeID, 147
 - scheduledTime, 147
 - stopLocation, 147
 - transitColInfo, 147
 - type, 147
 - userContactInfo, 148
- tracking::VehicleObject
 - currentRoute, 149
 - gpsDeviceID, 149
 - gpsDeviceInfo, 149
 - transitCoURL, 149
- tracking::VehicleRepository
 - findVehicle, 150
 - findVehiclesByRoute, 150
 - removeVehicle, 150
 - updateVehicle, 150
- TrackingAlertFactory, 100
- TrackingAlertObserver, 101
- TrackingAlertRequestModel, 102
- TrackingAlertService, 103
- trackingAlertService
 - alert::service::AlertServiceFactory, 40
- TrackingDelayAlert, 104
- TrackingLocationAlert, 105
 - tracking::TrackingLocationAlert, 105
- TrackingResponseModel, 105
- TrackingServiceController, 106
- transit, 19
- transit::AbstractFeedParserTemplate
 - loadFeed, 24
 - parseFeed, 24
 - saveRoutes, 24
 - start, 25

- validate, 25
- transit::Detour
 - cause, 54
- transit::Fare
 - setDiscountedFare, 54
 - setRegularFare, 54
- transit::GTFSFeedParser
 - parseFeed, 65
- transit::GoogleTransitServiceAdapter
 - getRoute, 57
 - getRoutes, 57
 - GoogleTransitServiceAdapter, 57
- transit::ITeamTransitServiceController
 - alertRequestController, 71
 - getRoute, 70
 - getRoutes, 70
 - getServiceURL, 71
 - getTransitInfo, 71
 - handleRouteDisruptionEvent, 71
 - transitFeed, 71
- transit::ITeamTripServiceController
 - calculateTrip, 72
- transit::InvalidRouteParseException
 - InvalidRouteParseException, 68
 - routeBatch, 68
 - serialVersionUID, 68
- transit::Location
 - Location, 79
- transit::PersistedTransitFeed
 - getRoute, 82
 - getRoutes, 82
- transit::Route
 - detours, 86
 - routeName, 86
 - stops, 86
- transit::RouteDisruptionAlert
 - routeId, 88
 - transitServiceUrl, 88
- transit::RouteDisruptionEvent
 - RouteDisruptionEvent, 89
- transit::RouteRepository
 - delete, 90
 - getAll, 90
 - read, 90
 - save, 90, 91
- transit::RouteSpecification
 - isSatisfiedBy, 92
- transit::Stop
 - description, 100
 - getStopTimes, 100
- transit::TransitFeed
 - getRoute, 110
 - getRoutes, 110
- transit::TransitInfo
 - logo, 112
 - transitAuthorityName, 112
 - website, 112
- transit::TransitProvider
 - fireRouteDisruptionEvent, 113
 - name, 113
 - providerId, 113
 - registerObserver, 113
 - unregisterObserver, 113
- transit::TransitProviderObserver
 - handleRouteDisruptionEvent, 114
- transit::TransitService
 - getRoute, 115
 - getRoutes, 115
 - getServiceURL, 115
 - getTransitInfo, 116
- transit::TripService
 - calculateTrip, 120
- TransitAlertRequestModel, 108
- TransitAlertService, 108
- transitAlertService
 - alert::service::AlertServiceFactory, 40
- transitAuthorityName
 - transit::TransitInfo, 112
- transitColInfo
 - tracking::UserTrackingAlertObject, 147
- transitCoURL
 - tracking::VehicleObject, 149
- TransitFeed, 110
- transitFeed
 - transit::ITeamTransitServiceController, 71
- TransitInfo, 111
- TransitProvider, 112
- TransitProviderObserver, 113
- TransitService, 114
- transitServiceUrl
 - transit::RouteDisruptionAlert, 88
- TransitVehicle, 116
- TransitVehicleFactory, 117
- Trip, 118
- tripData
 - alert::domain::model::TripInformation, 120
- TripInformation, 119
- TripService, 120
- type
 - tracking::UserTrackingAlertObject, 147
- UnplannedDisruption
 - alert::enums::AlertNotificationType, 30
- unregisterGPSDevice
 - tracking::GPSLocationTracking, 61
- unregisterObserver
 - transit::TransitProvider, 113
- unregisterVehicleFromRoute
 - tracking::ITrackingService, 79
 - tracking::TrackingServiceController, 108
- updateAlert
 - alert::domain::AlertRepository, 32
 - alert::service::AlertService, 38
 - alert::service::TrackingAlertService, 104
 - alert::service::TransitAlertService, 109
 - alert::service::UserAlertService, 127
 - tracking::TrackingAlertObserver, 102

- tracking::TrackingDelayAlert, 104
- updateFavorites
 - user::UserFavoritesRepository, 129
- updateUser
 - user::ITeamUserManagementService, 77
 - user::UserManagementService, 141
 - user::UserRepository, 145
- updateVehicle
 - tracking::VehicleRepository, 150
- User, 121
 - user::User, 122
- user, 20
- user::FavoriteTransitService
 - FavoriteTransitService, 55
 - getFavoriteRouteIds, 55
 - getTransitServiceUrl, 56
 - isFavoriteTransitService, 56
 - setFavoriteRouteIds, 56
 - setFavoriteTransitService, 56
- user::ITeamUserFavoritesService
 - readFavorites, 73
 - saveFavorites, 73
- user::ITeamUserLoginService
 - checkPermissions, 74
 - createAlertSession, 75
 - getUser, 75
 - login, 75
 - logout, 75
 - resetPassword, 75
 - sendUsername, 75, 76
- user::ITeamUserManagementService
 - createUser, 76
 - deleteUser, 76
 - findUserByEmail, 77
 - findUserByMobile, 77
 - findUserByUsername, 77
 - updateUser, 77
- user::Session
 - getCreationTime, 93
 - getExpirationTime, 93
 - getSessionToken, 94
 - getUserId, 94
 - isAlertSession, 94
 - isValid, 94
 - Session, 93
 - setExpirationTime, 94
 - setValid, 94
- user::SessionRepository
 - createSession, 95
 - getSession, 95
 - killSession, 96
- user::User
 - getCountryCode, 122
 - getEmail, 122
 - getFirstName, 122
 - getMobile, 123
 - getPasswordHash, 123
 - getUserId, 123
 - getUserType, 123
 - getUsername, 123
 - isForcePasswordChange, 123
 - setCountryCode, 124
 - setEmail, 124
 - setFirstName, 124
 - setForcePasswordChange, 124
 - setMobile, 124
 - setPasswordHash, 125
 - setUserType, 125
 - User, 122
- user::UserFavoritesList
 - getFavoriteTransitServices, 128
 - getUserId, 128
 - setFavoriteTransitServices, 128
 - UserFavoritesList, 128
- user::UserFavoritesRepository
 - getFavorites, 129
 - updateFavorites, 129
- user::UserFavoritesService
 - readFavorites, 130
 - saveFavorites, 131
- user::UserLoginService
 - createAlertSession, 133
 - getUser, 134
 - login, 134
 - logout, 135
 - resetPassword, 135, 136
 - sendUsername, 136, 137
- user::UserManagementService
 - createUser, 138
 - deleteUser, 139
 - findUserByEmail, 139
 - findUserByMobile, 140
 - findUserByUsername, 140
 - updateUser, 141
- user::UserRepository
 - createUser, 142
 - deleteUser, 143
 - getUserByEmail, 143
 - getUserById, 144
 - getUserByMobile, 144
 - getUserByUsername, 145
 - updateUser, 145
- UserAlertExecuteStrategy, 125
- UserAlertRequestModel, 126
- UserAlertService, 126
- userAlertService
 - alert::service::AlertServiceFactory, 40
- userClient
 - alert::service::RouteAlertExecuteStrategy, 87
 - alert::service::UserAlertExecuteStrategy, 126
- userContactInfo
 - tracking::UserTrackingAlertObject, 148
- UserFavoritesList, 127
 - user::UserFavoritesList, 128
- UserFavoritesRepository, 129
- UserFavoritesService, 130

- UserInformation, [131](#)
- userInformation
 - alert::domain::model::Alert, [27](#)
- UserLoginService, [132](#)
- UserManagementService, [137](#)
- UserRepository, [142](#)
- UserSessionInformation, [146](#)
- userSessionToken
 - alert::domain::model::UserSessionInformation, [146](#)
- UserTrackingAlertObject, [146](#)
- UserType, [148](#)
- validate
 - transit::AbstractFeedParserTemplate, [25](#)
- VehicleObject, [148](#)
- VehicleRepository, [149](#)
- verifySession
 - alert::controller::AlertRequestController, [35](#)
- verifySessionToken
 - alert::controller::CertificateHandler, [51](#)
 - alert::controller::ISessionHandler, [69](#)
 - alert::controller::SessionTokenHandler, [97](#)
- website
 - transit::TransitInfo, [112](#)