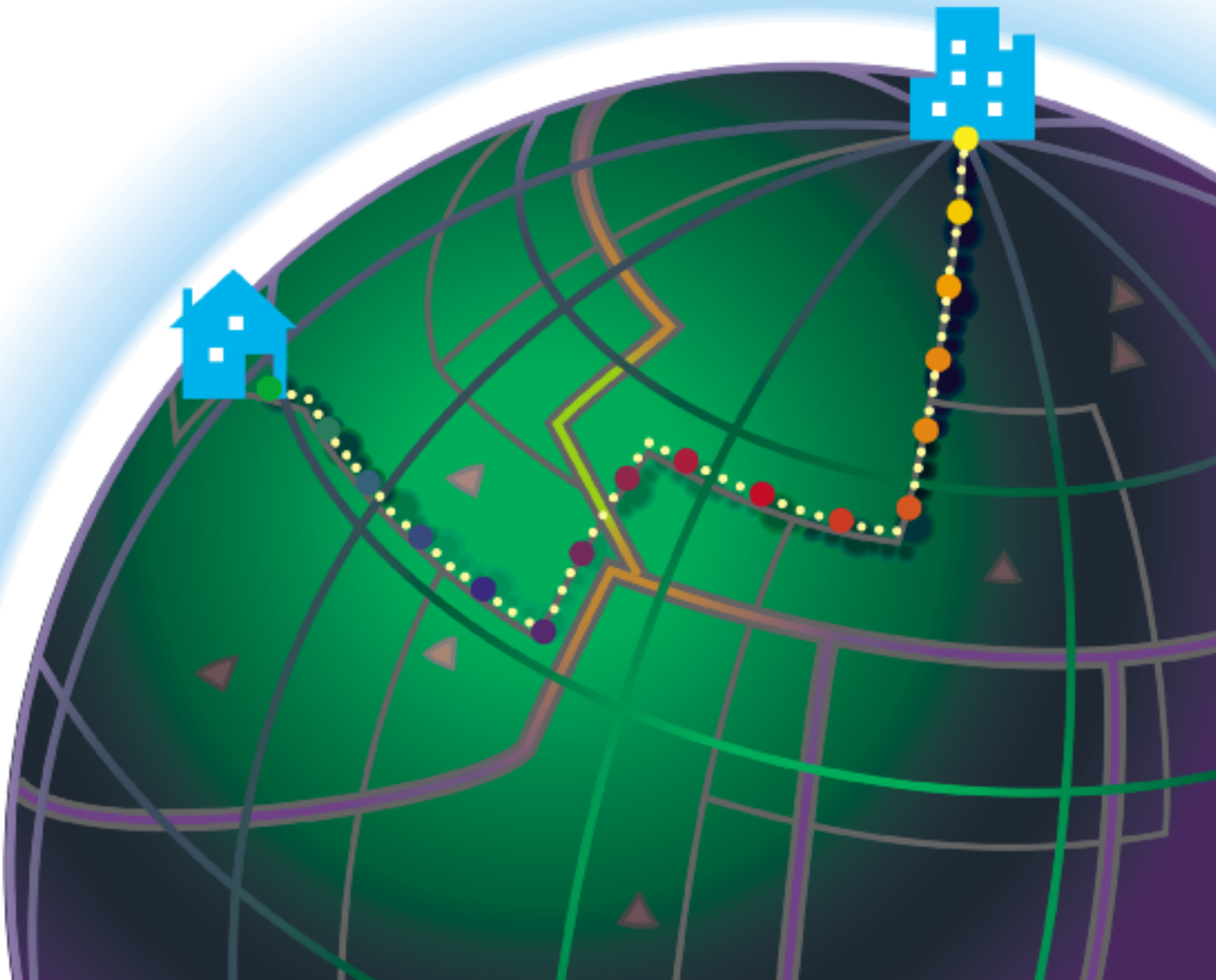
MapInfo® Routing J Server



MapInfo COM Client for Routing J Server

Developer's Guide

MapInfo

Contact MapInfo Corporation on the Internet at: http://www.mapinfo.com.

MapInfo Corporate Headquarters:	**MapInfo Europe Headquarters:**	**Germany:**
Voice: (518) 285-6000	Voice: +44 (0)1753 848 200	Voice: +49 (0)6142-203-400
Fax: (518) 285-6060	Fax: +44 (0)1753 621 140	Fax: +49 (0)6142-203-444
Sales Info Hotline: (800) 327-8627	email: uk@mapinfo.com	email: germany@mapinfo.com
Federal Sales: (800) 619-2333		
Technical Support Hotline: (518) 285-7283 Technical Support Fax: (518) 285-6080	Toll-free telephone support is available in the U.S. and Canada. Contact your MapInfo sales representative for details. For international customers, please use the Technical Support Fax number.	

MapInfo Com Client 2.5 for Routing J Server
August 2002

Table of Contents

Introduction

The MapInfo Routing COM Client provides access to Windows users without requiring Java on the client side. The client can be used with any programming language that supports Microsoft COM, including Visual Basic, Delphi, ASP, and C++.

MapInfo Routing COM Client consists of two Windows Dlls: MIRoute.dll and MIRouteRes.dll.

MapInfo Routing COM Client is usable by any COM container that can create and use a COM component.

MapInfo Routing COM Client can be driven by any application that supports hosting of COM components.

1 Chapter

Overview

MapInfo Routing COM Client provides you with:

- a broad set of objects and methods to use the route information returned from a Routing J Server, including driving directions and access to the individual streets and intersections returned as part of the route,
- isochrones which return a boundary representing an area that can be reached from a starting point in a given amount of time or by driving a given distance,
- the ability to set preference-values to specify time and distance units, and request shortest distance, driving directions, endpoints, shape points.

MapInfo Routing COM Client and Routing J Server

The MapInfo Routing COM Client component is modeled after the Java client that ships with Routing J Server. Use MapInfo Routing COM Client objects and methods to make a request and then display, analyze, and manipulate the returned information.

Contact Information for MapInfo Corporate Headquarters

MapInfo MapInfo Routing COM Client is produced by MapInfo Corporation. MapInfo creates and sells a full line of mapping software and data products. If you need information on other mapping solutions or data products, contact us at the phone and fax numbers listed below, or visit our Web sites.

If you need assistance working with MapInfo MapInfo Routing COM Client, our technical support specialists can help. Technical support for MapInfo Routing COM Client includes referrals to documentation, assistance with error messages and suggestions for causes of error messages. Telephone technical support for MapInfo Routing COM Client is limited to customers who have purchased MapInfo Routing COM Client. You can arrange for MapInfo Routing COM Client training, or obtain customized assistance from MapInfo's Developer Services.

- MapInfo Web Site: **http://www.mapinfo.com**
- Main Phone: **518.285.6000**
- Main Fax: **518.285.6060**
- Sales Information Order Hotline: **800.327.8627**
- Federal Sales: **800.619.2333**
- Sales Fax: **518.285.6070**
- Technical Support Hotline: **518.285.7283**
- Technical Support Fax: **518.285.6080**
- Training e-mail: **training@mapinfo.com**
- Training Phone: **800.552.2511**
- Developer Services e-mail: **devservices@mapinfo.com**
- Developer Services Phone: **518.285.7175**

Getting Started

This section describes how to install and uninstall the MapInfo Routing COM Client.

2 Chapter

- System Requirements
- Program Files
- Installing MapInfo Routing COM Client
- Removing the MapInfo Routing COM Client

System Requirements

- Windows 98; NT; 2000
- Disk space: 400KB for the DLLs

Program Files

MIRoute.dll: contains all the COM objects including MIRouteClient, MIRouteRequest, MIRouteIsoRequest, etc.

MIRouteRes.dll: contains only string resources.

MIRoute.i and MIRoute.iid: files for the C++ programmer. They contain the interface and class GUIDs (unique identifiers), enumerated constants such as RoadTypes, and the methods and properties on each interface.

Installing MapInfo Routing COM Client

To install MapInfo Routing COM Client:

1. Place the MapInfo Routing J Server CD in the CD drive.
2. Open the Routing COM Client folder and double-click on the Setup.exe.

   or

   Type Drive:\ RoutingCOMClient\setup.exe at the Run Command from the Start menu.

   The Install Shield Wizard displays. Choose Next.
3. The Destination Folder screen displays:

   Specify the directory where MapInfo Routing COM Client will be installed. The default location is: C:\Program Files\Common Files\MapInfo Shared\Routing COM Client.

   Specify the directory where MapInfo Routing COM Client Samples will be installed. The default location is: C:\Program Files\MapInfo\Routingjserver-2.5.0\examples_us\COM Client\.

   To designate a different location, choose the Change button, and specify the destination. Choose Next to continue the installation process. Choose Back to return to the previous screen; choose Cancel to stop the Installation process.
4. The Ready to Install the Program screen displays; choose Install to install MapInfo Routing COM Client.
5. The InstallShield Wizard Completed screen displays.

If You Have a Previous Version of MapInfo Routing COM Client

If you already have a previous version of MapInfo Routing COM CLient installed do one of the following:

- Uninstall the previous version by using the Add/Remove or Setup.exe.

  or

- From the command line type:

  Drive ....\setup.exe /v"REINSTALLMODE=voums REINSTALL=ALL"

  Note: The only spaces should follow Setup.exe and voums.

Removing the MapInfo Routing COM Client

To uninstall the MapInfo Routing COM Client:

1. Either insert the MapInfo Routing J Server CD in the CD drive and run setup.exe from the RoutingCOMClient directory

   or

   use the Control Panel to access the Add/Remove Programs option.

If running Setup.exe perform the following steps:

1. The Install Shield Wizard screen displays. Choose Next to continue the uninstall process.
2. The Program Maintenance screen displays allowing you to modify, repair or remove MapInfo Routing COM Client.
3. The Remove the Program screen displays; choose Remove to uninstall MapInfo Routing COM Client.
4. The Uninstalling MapInfo Routing COM Client screen displays. After the uninstall process is complete; choose Finish to exit the InstallWizard.

MIRouting Basics

This section describes many of the basics for the MapInfo Routing COM client.

3
Chapter

- **Using the MapInfo Routing COM Client Component**
- **Connecting to the Routing J Server**
- **Visual Basic and C++ Routing COM Client Sample Applications**

Using the MapInfo Routing COM Client Component

Use the MapInfo Routing COM Client to get directions:

- Create the necessary MapInfo Routing COM Client objects
- Create a MapInfo Routing COM Route Request object
- Specify the Start and End Points.
- Specify to return directions by setting the ReturnDirections property.
- Create a MapInfo Routing COM Client Client object.
- Set the URL to the server.
- Use the Find Route method, passing the Route Request object.
- Use MapInfo Routing COM Client API to manipulate the route result.

Connecting to the Routing J Server

To connect to a Routing J Server you only need the URL of a running Routing J Server. Enter the URL to a compatible, running version of Routing J Server servlet.

Visual Basic and C++ Routing COM Client Sample Applications

The Routing COM Client sample applications are found at C:\Program Files\MapInfo\Routingjserver25\Samples\Routing COM Client.

These samples demonstrate many of the features and methods available with the Routing COM Client.

Required

- Routing J Server v2.5 - Server with data installed and configured properly.
- MapMarker NT RPC Server/OCX. Evaluation available from http://testdrive.mapinfo.com/mapmarker.
- MapInfo MapX v5.0. Evaluation available from http://testdrive.mapinfo.com/mapx.
- Routing COM Client v2.5 - properly installed and registered.
- C++ applications require MFC42D.DLL, MSRCRTD.DLL, MFCO42D.DLL

Procedure

Prior to starting the application, start the Routing J server and the MapMarker server.

1. Upon starting the application, set the Routing Server and MapMarker Server in the Settings Dialog, if they are other than the local host. Choose Settings to get this dialog.
2. Enter a valid start address and end address.
3. For each address, click the Geocode Button. Address matches will be returned.
4. Select the address most accurate for the desired location. The Latitude and Longitude values will display.
5. Once the start and end addresses are properly geocoded, click on the Driving Directions button to retrieve the route and directions.
6. To enter new address values and recalculate directions, repeat steps 2 through 5. Click the Reset button to clear all previously designated items on the original form.

   You do not have to clear the address before changing it.
7. Use the settings dialog to change other preferences used when processing the route.

The MIRoute API (Reference Guide)

This section contains a list of methods and properties as well as reference entries for each of MIRoute's methods and properties. Each entry includes: Purpose and Syntax line/table.

4 Chapter

- Object Overview
- MIRoute Objects

Object Overview

The MapInfo Routing COM Client provides you with a variety of objects designed to meet all of your needs. Each object is described in detail beginning on the following pages.

MIRoute Objects

The following is a list of all of the MIRoute objects available to you.

MIRouteBoundary Object

The MIRouteBoundary object is returned in a MultiPolygon result to an IsoChrone or IsoDistance request where the Response Type is set to ISO_RESPONSE_TYPE_GEOMETRY The boundary provides a list of points that draw an iso, island or hole.

Object Properties

- gid
- srsName

Object Method

- GetPoints

MIRouteBoundary.GetPoints method

Purpose:

Returns an MIRouteCollection of MIRoutePoint objects.

Syntax

MIRouteBoundary.GetPoints() MIRouteCollection

MIRouteBoundary.gid property

Purpose

Used internally by the system.

MIRouteBoundary.srsName property

Purpose

Used internally by the system.

MIRouteClient Object

The MIRouteClient object is used to send route requests to the Routing J Server, and makes the results available to the programmer.

Creating the MIRouteClient object does not initiate the connection to the server. The connection is made, then terminated, with each FindRoute() or FindIso() call. There is no way to specify the communication protocol.

Methods

- FindIso
- FindMultiPoint
- FindRoute
- GetClientID
- GetURL
- SetClientID
- SetURL

MIRouteClient.FindIso method

Purpose

Sends an IsoDistance or IsoChrone request (contained in the MIRouteIsoRequest object) to the Routing J Server and receives the response back in an MIRouteIsoResult object

Syntax

MIRouteClient.FindIso(MIRouteIsoRequest as object) as MIRouteIsoResponse

For example:

[MIRouteIsoResponse object =] **MIRouteClient.FindIso**(MIRouteIsoRequest object)

MIRouteClient.FindMultiPoint method

Purpose

Sends a route request to the server, and fills in an MIRouteMultiPointRequest object on return. This method will return an error if there is a problem in connecting to the RoutingJServer. (For example, the URL to connect to the server is incorrect.). This method determines the order of the intermediate points on the final route and returns a route from the start to the endpoint, passing through the now ordered intermediate points. Returns route or an error inside the MIRouteMultiPointRequest.

Syntax

MIRouteClient.FindMultiPoint (MIRouteMultiPointRequest as object) As MIRouteMultiPointResponse

MIRouteClient.FindRoute method

Purpose

Sends a route request to the server, and fills in an MIRouteResult object on return. This method will return an error if there is a problem in connecting to the RoutingJServer. (For example, the URL to connect to the server is incorrect.). This method returns routes from the start to the end point. Returns route or an error inside the MIRouteResult object.

There is only one FindRoute method, and it takes an MIRouteRequest object as its parameter.

Syntax

MIRouteClient.FindRoute(MIRouteRequest as object) As MIRouteResult

For example:

[MIRouteResult object =] **MIRouteClient.FindRoute**(MIRouteRequest object)

MIRouteClient.GetClientID method

Purpose

Returns the client ID that has been set on the MIRouteClient object.

Syntax

MIRouteClient.GetClientID() As String

MIRouteClient.GetURL method

Purpose

Returns the URL that has been set on the MIRouteClient object for the FindRoute() and FindIso() methods to use when connecting to the Routing J Server.

Syntax

MIRouteClient.GetURL() As String

MIRouteClient.SetClientID method

Purpose

Sets the client ID. This ID that will be written to the log file if user request logging is enabled.

Syntax

MIRouteClient.**SetClientID**(*strID* As String)

strID - the client ID

MIRouteClient.SetURL method

Purpose

Tells the MIRouteClient object what URL the FindRoute() and FindIso() methods will use to connect to the RoutingJ Server.

Syntax

MIRouteClient.SetURL(*URL* As String)

MIRouteCollection Object

The MIRouteCollection object is designed to hold a collection or group of other Routing COM Client objects -- points, streets, boundaries, etc. It is compatible with the *for...each* syntax in Visual Basic.

The MIRouteCollection is typically used as a return value from a method such as MIRouteResult.GetStreets(). Refer to the following example

```
Dim res as new MIRouteResult
' call to FindRoute, etc.
dim StreetsCollection as MIRouteCollection
StreetsCollection = res.GetStreets()
```

Methods

- Add
- Count
- Item
- Remove

MIRouteCollection.Add method

Purpose

Adds the specified object to the collection.

Syntax

MIRouteCollection.Add(*pdisp* As Object)

MIRouteCollection.Count method

Purpose

Returns the number of objects in the collection.

Syntax

MIRouteCollection.Count() As Long

MIRouteCollection.Item method

Purpose

Return the specified object from the collection

Syntax

MIRouteCollection.Item(index As Long) As Object

MIRouteCollection.Remove method

Purpose

Removes the specified object from the collection.

Syntax

MIRouteCollection.Remove(*pdisp* As Object)

MIRouteDistance Object

Encapsulates a linear unit and a linear value that represents a distance such as 10 miles. Generally, this object is a parameter to a method or a property.

Properties

- LinearUnit
- LinearValue

MIRouteDistance.LinearUnit property

Purpose

Identifies the unit of measure used such as miles or kilometers.

Syntax

MIRouteDistance.LinearUnit As DistanceUnits

MIRouteDistance.LinearValue property

Purpose

Identifies the number of units represented such as 10.

Syntax

MIRouteDistance.LinearValue As Double

MIRouteIsoRequest Object

Encapsulates all information for an IsoChrone or IsoDistance route request, including required information and preferences.

Properties

- BandingStyle
- DefaultAmbientSpeed
- DefaultPropagationFactor
- IncludeActualConstraints
- IncludeHoles
- IncludeIsland
- LinearUnit
- MajorRoadsOnly
- MaxOffRoadDistance
- PropagationFactor
- ResponseType
- SimplificationFactor
- StartPoint
- TimeUnit
- Units

Methods

- AddAmbientSpeedOverride
- AddCost
- AddPropagationOverride
- GetCost
- Reset

MIRouteIsoRequest.AddAmbientSpeedOverride method

Purpose

Overrides the default ambient speed for a specific road type. This property only applies to isochrones, i.e., an iso with a time unit.

Syntax

MIRouteIsoRequest.AddAmbientSpeedOverride(*nRoadType* As RoadTypes, *pVelocity* As MIRouteVelocity)

MIRouteIsoRequest.AddCost method

Purpose

Adds a cost with an associated tag string to this request. For example, to define a 15 minute isochrone you would create an MIRouteIsoRequest with a unit of TIME_UNIT_MINUTES, a cost of 15 and a tag of "15 minutes".

The tag allows the results to be matched with the original request. By adding multiple costs to one request, that request can generate multiple iso results such as 1,3 and 5 mile isodistances from the same point.

Syntax

MIRouteIsoRequest.AddCost(*Tag* As String, *nVal* As Long)

MIRouteIsoRequest.AddPropagationOverride method

Purpose:

Overrides the propagation factor to be used in the calculation of the iso for a given road type. Propagation factor applies only to isodistances, i.e. an iso with a distance unit.

Syntax

MIRouteIsoRequest.AddPropagationFactorOverride(*nRoadType* As RoadTypes, *dSpeed* As Double)

MIRouteIsoRequest.BandingStyle property

Purpose:

The banding style is used to format the result of an iso. This applies only if multiple costs are specified in the iso definition and geometry is the requested result type.

BANDING_STYLE_ENCOMPASSING - Each cost will generate a result that starts at the starting point. BANDING_STYLE_DONUT - Each cost will generate a result that starts where the previous cost ended.

The default setting for this property is BANDING_STYLE_ENCOMPASSING.

Syntax

MIRouteIsoRequest.BandingStyle As IsoBandingStyles

MIRouteIsoRequest.DefaultAmbientSpeed property

Purpose:

The default ambient speed is used for the calculation of the iso. Default ambient speed is the ambient speed that is used for all road types that do not have specific ambient speed overrides. Ambient speed is the speed at which travel is allowed off network roads. This property only applies to isochrones, i.e. an iso with a time unit.

The default setting for this property is 15 mph.

It can be overridden by road type with the AddAmbientSpeedOverride method.

Syntax

MIRouteIsoRequest.DefaultAmbientSpeed As MIRouteVelocity

MIRouteIsoRequest.DefaultPropagationFactor property

Purpose

The default propagation factor is used for the calculation of the iso. The default propagation factor is the propagation factor that is used for all road types that do not have specific propagation factor overrides. A propagation factor is the percentage of the remaining cost for which off network travel is allowed. This property only applies to isodistances, i.e. an iso with a distance unit.

The default setting for this property is 0.16.

It can be overridden by road type with the AddPropagationFactorOverride method.

Syntax

MIRouteIsoRequest.DefaultPropagationFactor As Double

MIRouteIsoRequest.GetCost method

Purpose

Returns the value associated with a specified tag.

Syntax

MIRouteIsoRequest.GetCost(Tag As String) As Double

MIRouteIsoRequest.IncludeActualConstraintsproperty

Purpose

This setting indicates if the actual constraints (preferences) the server used to generate the route will be included in the response.

A value of True will return the constrains.

The default setting is False.

Syntax

MIRouteIsoRequest.IncludeActualConstraints As Boolean

MIRouteIsoRequest.IncludeHolesproperty

Purpose

This setting indicates if holes will be returned in the result. Applies only if geometry is the result type.

If true, holes are allowed in the result. The default setting for this property is true.

Syntax

MIRouteISoRequest.IncludeHoles As Boolean

MIRouteIsoRequest.IncludeIslands property

Purpose

This setting indicates if islands will be returned. Applies only if geometry is the result type.

If true, islands are allowed in the result. The default setting for this property is true.

Syntax

MIRouteIsoRequest.IncludeIslands As Boolean

MIRouteIsoRequest.LinearUnit property

Purpose

The unit of measure for the cost associated with an isodistance.

For example, a request with Linear Unit = DIST_UNIT_MILES and a Cost = 5 will return a 5 mile isodistance.

The default is null (TimeUnit is set as the default property).

Syntax

MIRouteIsoRequest.LinearUnit As DistanceUnits

MIRouteIsoRequest.MajorRoadsOnly property

Purpose

This setting indicates if only the major roads will be used in calculating an iso or if secondary roads will also be considered.

The default value is false.

Syntax

MIRouteIsoRequest.MajorRoadsOnly As Boolean

MIRouteIsoRequest.MaxOffRoadDistance property

Purpose

This setting indicates the maximum distance ambient travel will be allowed to go off roads.

The default setting for this property is no limit.

Syntax

MIRouteIsoRequest.MaxOffRoadDistance As MIRouteDistance

MIRouteIsoRequest.Reset method

Purpose:

This method resets the MIRouteIsoRequest to a default state with a startpoint = 0, all costs cleared and all preferences set to defaults.

Syntax

MIRouteIsoRequest.Reset()

MIRouteIsoRequest.ResponseType property

Purpose

This setting indicates the type of response to be provided.

The default setting is ISO_RESPONSE_TYPE_GEOMETRY.

Syntax

MIRouteISoRequest.ResponseType As IsoResponseTypes

MIRouteIsoRequest.SimplificationFactor property

Purpose:

This setting indicates the percentage of original points in the geometry to be returned.

This process reduces the complexity of the result geometry and improves the appearance. Applies only if geometry is the result type.

The default setting for this property is 0.05.

Syntax

MIRouteIsoRequest.SimplificationFactor As Double

MIRouteIsoRequest.StartPoint property

Purpose:

This setting indicates the point from which the iso will be calculated.

For example, in a 3 mile iso, the 3 mile distance is calculated from this starting point. For an isochrone, the latitude and longitude of the point must be set. The level property should be 0.

Syntax

MIRouteIsoRequest.StartPoint As MIRoutePoint

MIRouteIsoRequest.TimeUnit property

Purpose

The unit of measure for the cost associated with an isochrone. For example, a request with a time unit of TIME_UNIT_MINUTES (default) and a cost of 5 would generate a 5 minute isochrone.

Syntax

MIRouteIsoRequest.TimeUnit As TimeUnits

MIRouteIsoRequest.Units property

Purpose

This setting indicates whether the TimeUnit or LinearUnit property is currently populated and in effect (Read only).

Syntax

MIRouteIsoRequest.Units As IsoUnits

MIRouteIsoResponse Object

This object encapsulates the response to the MIRouteIsoRequest and is generated by a call to MIRouteClient.FindIso(MIRouteIsoRequest). The MIRouteIsoResponse object has an MIRouteCollection of MIRouteIsoResult objects, one MIRouteIsoResult for each "Cost" in the MIRouteIsoRequest. The MIRouteIsoResult objects can be accessed via:

- the collection (the GetIsoResults method returns the collection);
- by the tag associated with a particular Cost using the GetCost(String tagName) method.

The properties that correspond to preferences set in the MIRouteIsoRequest will only be populated if the property MIRouteIsoRequest IncludeActualContraints was set to True. They are read-only.

Properties

- BandingStyle
- DefaultAmbientSpeed
- DefaultPropagationFactor
- IncludeActualConstraints
- IncludeHoles
- IncludeIslands
- MajorRoadsOnly
- MaxOffRoadDistance
- RequestConstraintsOverridden
- ResponseType
- SimplificationFactor
- Success

Methods

- GetAmbientOverriddenRoadTypes
- GetAmbientSpeedOverride
- GetCost
- GetErrorCode
- GetErrorMessage
- GetIsoResults
- GetPropagationFactorOverriddenRoadTypes
- GetPropagationFactorOverride

MIRouteIsoResponse.BandingStyle property

Purpose

This setting indicates the format used for the MIRouteIsoResult.

Syntax

MIRouteIsoResponse.BandingStyle As IsoBandingStyles

MIRouteIsoResponse.DefaultAmbientSpeed property

Purpose

This setting indicates the default ambient speed used in calculating the route.

Syntax

MIRouteIsoResponse.DefaultAmbientSpeed As MIRouteVelocity

MIRouteIsoResponse.DefaultPropagationFactor property

Purpose

This setting indicates the default propagation factor used in calculating the route.

Syntax

MIRouteIsoResponse.DefaultPropagationFactor As Double

MIRouteIsoResponse.GetAmbientOverriddenRoadTypes method

Purpose

Returns an array of RoadTypes whose ambient speeds were overridden on the MIRouteIsoRequest or set on the server. The array is 1-based.

Syntax

MIRouteIsoResponse.GetAmbientOverriddenRoadTypes() As RoadTypes()

MIRouteIsoResponse.GetAmbientSpeedOverride method

Purpose

Returns the default speed override for a specific road type returned on the array of road types by GetAmbientOverriddenRoadTypes.

Syntax

MIRouteIsoResponse.GetAmbientSpeedOverride(*nRoadType* As RoadTypes) As MIRouteVelocity

MIRouteIsoResponse.GetCost method

Purpose

Returns the MIRouteIsoResult associated with a specific tag

Syntax

MIRouteIsoResponse.GetCost(*Tag* As String) MIRouteIsoResult

MIRouteIsoResponse.GetErrorCode method

Purpose

Returns an error code in the event of a problem. Available only when Success is False. In the case of an unspecified error, the error code will be –1, meaning, "an error occurred, but the server did not return an error code."

Syntax

MIRouteIsoResponse.GetErrorCode() As Long

MIRouteIsoResponse.GetErrorMessage method

Purpose

Returns a text description explaining the error code returned in the GetErrorCode method. Available only when the Success property is false

Syntax

MIRouteIsoResponse.GetErrorMessage() As String

MIRouteIsoResponse.GetIsoResults method

Purpose

Returns a collection of MIRouteIsoResult objects, one for each cost in the original request.

Syntax

MIRouteIsoResponse.GetIsoResults() As MIRouteIsoRequest

MIRouteIsoResponse.GetPropagationFactorOverriddenRoadTypes method

Purpose

Returns an array of RoadTypes whose propagation factors were overridden on the MIRouteIsoRequest or set on the server. The array is 1-based.

Syntax

MIRouteIsoResponse.GetPropagationFactorOverriddenRoadTypes() As RoadTypes()

MIRouteIsoResponse.GetPropagationFactorOverride method

Purpose

Returns the propagation factor for a specific road type on the array returned by GetPropagationOverriddenRoadTypes.

Syntax

MIRouteIsoResponse.GetPropagationFactorOverride(*nRoadType* As RoadTypes) As Double

MIRouteIsoResponse.IncludeActualConstraints property

Purpose

This setting indicates if IncludeActualConstraints was set on the original request.

Syntax

MIRouteIsoResponse.IncludeActualConstraints As Boolean

MIRouteIsoResponse.IncludeHoles property

Purpose

This setting indicates if holes are returned on the response.

Syntax

MIRouteIsoResponse.IncludeHoles As Boolean

MIRouteIsoResponse.IncludeIslands property

Purpose

This setting indicates if islands are returned on the response.

Syntax

MIRouteIsoResponse.IncludeIslands As Boolean

MIRouteIsoResponse.MajorRoadsOnly property

Purpose

This setting indicates if only major roads were considered in the calculation of the iso (true) or if the secondary roads were used (false).

Syntax

MIRouteIsoResponse.MajorRoadsOnly As Boolean

MIRouteIsoResponse.MaxOffRoadDistance property

Purpose

This setting indicates the maximum distance ambient travel was allowed to go off roads.

Syntax

MIRouteIsoResponse.MaxOffRoadDistance As MIRouteDistance

MIRouteIsoResponse.RequestConstraintsOverridden property

Purpose

This setting indicates that one or more constraints were overridden by server settings instead of using the requested constraint. These settings are read-only.

Syntax

MIRouteIsoResponse.RequestConstraintsOverridden As Boolean

MIRouteIsoResponse.ResponseType property

Purpose

This setting indicates the type of response provided.

Syntax

MIRouteIsoResponse.ResponseType As IsoResponseTypes

MIRouteIsoResponse.SimplificationFactor property

Purpose

This setting indicates the percentage of original points returned.

Syntax

MIRouteIsoResponse.SimplificationFactor As Double

MIRouteIsoResponse.Success property

Purpose

This setting indicates if the calculation was successful (true) or an error occurred (false).

Syntax

MIRouteIsoResponse.Success As Boolean

MIRouteIsoResult Object

The MIRouteIsoResult object corresponds to a "Cost" specified in the MIRouteIsoRequest. The MIRouteIsoResponse will contain one for each cost on the original request.

Properties

- ResultType
- Tag

Methods

- GetMultipoint
- GetMultiPolygon

MIRouteIsoResult.GetMultipoint method

Purpose

Returns an MIRouteMultiPoint object that contains an MIRouteCollection of MIRoutePoints, when the Result Type is ISO_RESULT_TYPE_NODE. This occurs when the request specified a response type of ISO_RESPONSE_TYPE.ACCESSIBLENODES or ISO_RESPONSE_TYPE_STARTNODES.

Syntax

MIRouteIsoResult.GetMultiPoint() As MIRouteCollection

MIRouteIsoResult.GetMultiPolygon method

Purpose

Returns an MIRouteMultiPolygon object, which contains a collection of MIRoutePolygon objects, when the Result Type is ISO_RESULT_TYPE_POLYGON. This occurs when the request is for a response type of geometry.

Syntax

MIRouteIsoResult.GetMultiPolygon() As MIRouteMultiPolygon

MIRouteIsoResult.ResultType property

Purpose

This setting indicates the type of information returned in the response: points or polygons.

Syntax

MIRouteIsoResult.ResultType As IsoResultTypes

MIRouteIsoResult.Tag property

Purpose

This setting contains the string associated with a particular cost in the request. It allows results to be distinguished from each other if matched to the original request.

Syntax

MIRouteIsoResult.Tag As String

MIRouteMultiPoint Object

This object is an MIRouteIsoResult that holds a collection of points that represents either the starting point for the iso or the set of points to represent each node or segment within the iso's parameter.

Properties

- gid
- srsName

Methods

- GetPoints

MIRouteMultiPoint.GetPoints method

Purpose

Returns a MIRouteCollection of MIRouteMultiPoint objects.

Syntax

MIRouteMultiPoint.GetPoints() As MIRouteCollection

MIRouteMultiPoint.Gid property

Purpose

Used internally by the system.

MIRouteMultiPoint.srsName property

Purpose

Used internally by the system.

MIRouteMultiPointRequest Object

The COM Client supports sending multipoint routing requests to the Routing J Server, and provides objects and methods to help use the returned results.

A MIRouteMultiPointRequest object is created and passed to the MIRouteClient.FindMultiPoint method. The MIRouteMultiPointResponse object is the returned. Preferences can be set by adding MIRouteMultiPointPreferences and MIRouteMultiPointResponsePreferences objects to the request object.

To send a MultiPoint request to the server:

- Create an MIRouteMultiPointRequest object.
- Fill in the StartPoint and EndPoint properties.
- Add at least one intermediate point. The maximum number of points is limited by system capabilities.
- (Optional) Create MIRouteMultiPointPreferences and/or MIRouteMultiPointResponsePreferences objects and fill in the corresponding properties on the MultiPoint request object.
- Call MIRouteClient.FindMultiPoint, passing the MultiPoint request object as an argument.
- (Optional) Get the MultiPointSummary object by accessing the property on the MultiPoint Response or get info by using the methods on the MultiPointSummary.

Note: Intermediate points do not have to be put in the desired order of result that will be determined by processing.

Properties

- Preferences
- ResponsePreferences
- StartPoint
- EndPoint

Methods

- AddIntermediatePoints
- ClearIntermediatePoints

MIRouteMultiPointRequest.AddIntermediatePoint method

Purpose

Adds a single intermediate point.

Syntax

MIRouteMultiPointRequest.AddIntermediatePoint(MIRoutePoint point**)**

MIRouteMultiPointRequest.AddIntermediatePoints method

Purpose

Add a collection of MIRoutePoints as intermediate points.

Syntax

MIRouteMultiPointRequest.AddIntermediatePoints(MIRouteCollection points)

MIRouteMultiPointRequest.ClearIntermediatePoints method

Purpose

Clears all intermediate points from the request.

Syntax

MIRouteMultiPointRequest.ClearIntermediatePoints()

MIRouteMultiPointRequest.EndPoint property

Purpose

Sets the endpoint of a MultiRoute. Initially set to 0., i.e. x=0, y=0.

Syntax

MIRouteMultiPointRequest.EndPoint As MIRoutePoint

MIRouteMultiPointRequest.Preferences property

Purpose

Sets the preferences for a multipoint request. Can be set to nothing if default preferences are to be use.

Syntax

MIRouteMultiPointRequest.PreferencesAs MIRouteMultiPointPreferences

MIRouteMultiPointRequest.ResponsePreferences property

Purpose

Sets the response preferences for a MultiRoute. Can be set to nothing if default preferences are to be used.

Syntax

MIRouteMultiPointRequest.ResponsePreferences As MIRouteMultiPointResponsePreferences

MIRouteMultiPointRequest.StartPoint property

Purpose

Sets the start point of a MultiRoute. Initially set to 0., i.e. x=0, y=0, z=0.

Syntax

MIRouteMultiPointRequest.StartPoint As MIRoutePoint

MIRouteMultiPointResponse Object

This object packages the returned values associated with a MultiPointRequest. It contains the match results and route information. The client application can use its methods to interrogate the results of the routing.

The Preferences and Response Preferences objects on the MIRouteMultiPointResponse will only be populated if IncludeActualConstrains was set to TRUE on the Response Preferences on the request. These properties are read-only on this object.

Properties

- Preferences
- RequestConstraintsOverridden
- ResponsePreferences
- RouteSectionCount
- RouteSummary
- Success

Methods

- GetDirections()
- GetDirectionsList()
- GetErrorCode()
- GetErrorMessage()
- GetRouteSection
- GetRouteSections()
- GetSectionDirections()
- GetSectionsDirectionsList()

MIRouteMultiPointResponse.GetDirections method

Purpose

Returns the directions for the route.

Syntax

MIRouteMultiPointResponse.GetDirections() As String

MIRouteMultiPointResponse.GetDirectionsList method

Purpose

Returns an array of strings, one string per route section, of driving directions for the route.

Syntax

MIRouteMultiPointResponse.GetDirectionsList() As String()

MIRouteMultiPointResponse.GetErrorCode method

Purpose

Returns an error code in the event of a problem. Available only when Success is False. In the case of an unspecified error, the error code will be –1, meaning, "an error occurred, but the server did not return an error code."

Syntax

MIRouteMultiPointResponse.GetErrorCode() As Long

MIRouteMultiPointResponse.GetErrorMessage method

Purpose

Returns a text description explaining the error code returned in the GetErrorCode method. Available only when the Success property is False.

Syntax

MIRouteMultiPointResponse.GetErrorMessage() As String

MIRouteMultiPointResponse.GetRouteSection method

Purpose

Returns an MIRouteMultiPointRouteSection object, or error if index is out of range. (index is 1-based.) Error is 1017

Syntax

MIRouteMultiPointResponse.GetRouteSection(long index) As MIRouteMultiPointRouteSection

MIRouteMultiPointResponse.GetRouteSections method

Purpose

Returns collection of MIRouteMultiPointRouteSection objects. If there are no RouteSections, error 1084 (IDS_ERR_NO_ROUTESECTIONS) is thrown.

Syntax

MIRouteMultiPointResponse.GetRouteSections( ) As MIRouteCollection

MIRouteMultiPointResponse.GetSectionDirections method

Purpose

Returns the directions, from the server, for the specified RouteSection

Syntax

MIRouteMultiPointResponse.GetSectionDirections(long index) As String

MIRouteMultiPointResponse.GetSectionDirectionsList method

Purpose

Returns an array of strings, one string per street, of driving directions for each street in the section

Syntax

MIRouteMultiPointResponse.GetSectionDirectionsList (long index) As String( )

MIRouteMultiPointResponse.Preferences property

Purpose

Returns Preferences actually used. Can be set to nothing if default preferences are to be used. Will only be filled in if IncludeActualConstraints was requested. (Read-only)

Syntax

MIRouteMultiPointResponse.Preferences As MIRouteMultiPointPreferences

MIRouteMultiPointResponse.RequestConstraintsOverridden property

Purpose

Returns True if preferences were overridden by the server and submitted with the request.

Syntax

MIRouteMultiPointResponse.RequestConstraintsOverridden As Boolean

MIRouteMultiPointResponse.ResponsePreferences property

Purpose

Returns Response Preferences. Can be set to nothing if default preferences are to be used. Will only be filled in if IncludeActualConstraints was requested. (Read-only)

Syntax

MIRouteMultiPointResponse.ResponsePreferences As MIRouteMultiPointResponsePreferences

MIRouteMultiPointResponse.RouteSectionCount property

Purpose

Returns the number of RouteSections. (Read-only)

Syntax

MIRouteMultiPointResponse.RouteSectionCount As Long

MIRouteMultiPointResponse.RouteSummary property

Purpose

Returns a route summary object.

Syntax

MIRouteMultiPointResponse.RouteSummary As MIRoutePointSummary

MIRouteMultiPointResponse.Success property

Purpose

Returns True if a route was returned for the request

Syntax

MIRouteMultiPointResponse.Success As Boolean

MIRouteMultiPointResponsePreferences Object

The following properties and method allow the user to set or retrieve the MultiPoint Response Preferences.

Properties

- DistanceUnit
- IncludeActualConstraints
- PrimaryNameOnly
- ReturnDirections
- ReturnSegmentData
- ReturnSegmentGeometry
- TimeUnit
- VelocityUnit

Method

- Reset()

MIRouteMultiPointResponsePreferences.DistanceUnit property

Purpose

This setting indicates the unit of measure to be used when returning the length of the route, segments and streets.

The default value is DIST_UNIT_MILES.

Syntax

MIRouteMultiPointResponsePreferences.DistanceUnit As DistanceUnits

MIRouteMultiPointResponsePreferences.IncludeActualConstraints property

Purpose

This setting indicates if the actual constraints (preferences) the server used to generate the route will be included in the response.

The default setting is false.

Syntax

MIRouteMultiPointResponsePreferences.IncludeActualConstraints As Boolean

MIRouteMultiPointResponsePreferences.PrimaryNameOnly property

Purpose

Flag indicating whether only the Primary Street Name name should be included in driving directions and alternate names ignored

The default is false. Requires ReturnSegmentData to be set to True to take effect.

Syntax

MIRouteMultiPointResponsePreferences.PrimaryNameOnly As Boolean

MIRouteMultiPointResponsePreference.Reset() method

Purpose

Resets all properties to defaults, and clears all overrides.

Syntax

MIRouteMultiPointResponsePreferences.Reset( )

MIRouteMultiPointResponsePreferences.ReturnDirections property

Purpose

This setting indicates if driving directions should be returned by the server.

The default is true.

Syntax

MIRouteMultiPointResponsePreferences.ReturnDirections As Boolean

MIRouteMultiPointResponsePreferences.ReturnSegmentData property

Purpose

This setting indicates if the segment data is returned in the result.

Syntax

MIRouteMultiPointResponsePreferences.ReturnSegmentData As Boolean

MIRouteMultiPointResponsePreferences.ReturnSegmentGeometry property

Purpose

Sets how to return segment geometry. Default is all points.

Syntax

MIRouteMultiPointResponsePreferences.ReturnSegmentGeometry As Segment Geometry

MIRouteMultiPointResponsePreferences.TimeUnit property

Purpose

The unit of measure to use in presenting time in the result such as total time required of 18.3 minutes.

The default = TIME_UNIT_MINUTES.

Syntax

MIRouteMultiPointResponsePreferences.TimeUnit As TimeUnits

MIRouteMultiPointResponsePreferences.VelocityUnit property

Purpose

Indicates the unit of measure to be used when calculating the route result. The default is mile per hour (mph).

Syntax

MIRouteMultiPointResponsePreferences.Velocity As VelocityUnit

MIRouteMultipointRouteSection Object

This object packages the returned values associated with a single section of a MultiPoint Response. A section is the portion of the result between 2 intermediate points, a start point and the first intermediate point or the last intermediate point and the endpoint.

Property

- StartPoint()

Methods

- GetDirections()
- GetDirectionsList()
- GetDirectionsSummary()
- GetDistances()
- GetDistanceUnits()
- GetDistanceUnitString()
- GetDistanceValue()
- GetEndPoint()
- GetIntersections()
- GetPointCount()
- GetPoints()
- GetRouteSectionStreet()
- GetSegmentCount()
- GetStreetCount()
- GetStreets().
- GetTimeUnits()
- GetTimeUnitString()
- GetTimeValue()

MIRouteMultiPointRouteSection.GetDirections method

Purpose

Returns the directions for the this section of the route.

Syntax

MIRouteMultiPointRouteSection.GetDirections() As String

MIRouteMultiPointRouteSection.GetDirectionsList method

Purpose

Returns an array of strings, one string for each street in the section.

Syntax

MIRouteMultiPointRouteSection.GetDirectionsList As String[( )

MIRouteMultiPointRouteSection.GetDirectionsSummary method

Purpose

Returns total time and distance for this section. The string returned by this method is provided by the server.

Syntax

MIRouteMultiPointRouteSection.GetDirectionsSummary As String

MIRouteMultiPointRouteSection.GetDistances method

Purpose

Returns array of doubles, one distance for each street in the section

Syntax

MIRouteMultiPointRouteSection.GetDistances( ) as Double (**)**

MIRouteMultiPointRouteSection.GetDistanceUnits method

Purpose

Returns the unit of measure used in calculating the length of the sections as a code.

Syntax

MIRouteMultiPointRouteSection.GetDistanceUnits( ) As DistanceUnits

MIRouteMultiPointRouteSection.GetDistanceUnitString method

Purpose

Returns the unit of measure used in calculating the length of the route as text such as meter.

Syntax

MIRouteMultiPointRouteSection.GetDistanceUnitString( ) As String

MIRouteMultiPointRouteSection.GetDistanceValue method

Purpose

Returns the number of distance units or length of the section such as 10.3 in 10.3 miles.

Syntax

MIRouteMultiPointRouteSection.GetDistanceValue( ) As Double

MIRouteMultiPointRouteSection.GetEndpoint method

Purpose

Returns the section endpoint.

Syntax

MIRouteMultiPointRouteSection.GetEndPoint ( ) as MIRoutePoint

MIRouteMultiPointRouteSection.GetIntersections method

Purpose

Returns a collection of MIRoutePoints which represent the intersections on this section.

Syntax

MIRouteMultiPointRouteSection.GetIntersections( ) As MIRouteCollection

MIRouteMultiPointRouteSection.GetPointCount method

Purpose

Returns the number of MIRoutePoints in this section

Syntax

MIRouteMultiPointRouteSection.GetPointCount( ) As Long

MIRouteMultiPointRouteSection.GetPoints method

Purpose

Returns collection of all MIRoutePoints in this section

Syntax

MIRouteMultiPointRouteSection.GetPoints( ) As MiRouteCollection

MIRouteMultiPointRouteSection.GetRouteSectionStreet method

Purpose

Returns a street from the street list.

Syntax

MIRouteMultiPointRouteSection.GetRouteSectionStreet (long index) As MIRouteStreet

MIRouteMultiPointRouteSection.GetSegmentCount method

Purpose

Returns the count of all the segments in all the streets in the section.

Syntax

MIRouteMultiPointRouteSection.GetSegmentCount( ) As Long

MIRouteMultiPointRouteSection.GetStreetCount method

Purpose

Returns the count of all the streets in all the streets in the section.

Syntax

MIRouteMultiPointRouteSection.GetStreetCount( ) As Long

MIRouteMultiPointRouteSection.GetStreets method

Purpose

Returns collection of all the streets within this section.

Syntax

MIRouteMultiPointRouteSection.GetStreets( ) As MIRouteCollection

MIRouteMultiPointRouteSection.GetTimeUnits method

Purpose

Returns the unit of measure used in calculating the travel time for this section as a code.

Syntax

MIRouteMultiPointRouteSection.GetTimeUnits( ) As TimeUnits

MIRouteMultiPointRouteSection.GetTimeUnitString method

Purpose

Returns the unit of measure used in calculating the travel time for this section as text such as minutes or hours.

Syntax

MIRouteMultiPointRouteSection.GetTimeUnitString( ) As String

MIRouteMultiPointRouteSection.GetTimeValue method

Purpose

Returns the number of time units or the amount of travel time such as 32.25 in 32.25 minutes.

Syntax

MIRouteMultiPointRouteSection.GetTimeValue( ) As Double

MIRouteMultiPointRouteSection.StartPoint property

Purpose

Returns the StartPoint for this section.

Syntax

MIRouteMultiPointRouteSection.StartPoint As MIRoutePoint

MIRouteMultiPointPreferences Object

The following properties and method allow the user to set or retrieve the preferences.

Properties

- FindShortestDistance
- MajorRoadsOnly
- StopThreshold
- TimeOut

Method

- Reset()

MIRouteMultiPointPreferences.FindShortestDistance property

Purpose

This setting indicates the route should be calculated to be the one with the shortest distance (true) or the shortest time (false).

Syntax

MIRouteMultiPointPreferences.FindShortestDistance As Boolean

MIRouteMultiPointPreferences.MajorRoadsOnly property

Purpose

This setting indicates if only major roads were considered in the calculation of the iso(true) or if the secondary roads were used (false). Initially set to False.

Syntax

MIRouteMultiPointPreferences.MajorRoadsOnly As Boolean

MIRouteMultiPointPreferences.Reset method

Purpose

Resets the MIRouteMultiPointPreferences to a default state.

Syntax

MIRouteMultiPointPreferences.Reset ()

MIRouteMultiPointPreferences.StopThreshold property

Purpose

Sets the route stop threshold. Initially set to.01.

Syntax

MIRouteMultiPointPreferenes.StopThreshold As Double

MIRouteMultiPointPreferences.TimeOut property

Purpose

Sets when time out will occur in milliseconds. The default is 0 which disables time out.

Syntax

MIRouteMultiPointPreferenes.TimeOut As Long

MIRouteMultipointSummary Object

This object packages key summary data for the MultiPoint Response

Methods

- GetDirectionsSummary()
- GetDistanceUnits()
- GetDistanceUnitString()
- GetDistanceValue()
- GetIntermediatePoints()
- GetIntermediatePointsCount()
- GetRouteDistanceString()
- GetRouteTimeString()
- GetTimeUnits()
- GetTimeUnitString()
- GetTimeValue()

MIRouteMultiPointSummary.GetDirectionsSummary method

Purpose

Returns total time and distance for the MultiPoint route. The string returned by this method is provided by the server.

Syntax

MIRouteMultiPointSummary.GetDirectionsSummary As String

MIRouteMultiPointSummary.GetDistanceUnit method

Purpose

Returns a unit of measure used in calculating the length of the MultiPoint route as a code.

Syntax

MIRouteMultiPointSummary.GetDistanceUnits As DistanceUnits

MIRouteMultiPointSummary.GetDistanceUnitString method

Purpose

Returns the unit of measure used in calculating the length of the route as text such as meter.

Syntax

MIRouteMultiPointSummary.GetDistanceUnitString( ) As String

MIRouteMultiPointSummary.GetDistanceValue method

Purpose

Returns the number of distance units or length of the section such as 10.3 in 10.3 miles.

Syntax

MIRouteMultiPointSummary.GetDistanceValue( ) As Double

MIRouteMultiPointSummary.GetIntermediatePoints method

Purpose

Returns the collection of the ordered intermediate points for the route.

Syntax

MIRouteMultiPointSummary.GetIntermediatePoints( ) As MIRouteCollection

MIRouteMultiPointSummary.GetIntermediatePointsCount method

Purpose

Returns the number of Intermediate Points in the MultiPoint Route.

Syntax

MIRouteMultiPointRouteSection.GetIntermediatePointCount( ) As Long

MIRouteMultiPointSummary.GetRouteDistanceString method

Purpose

Returns a string that contains the value and units for the length of the resulting route. For example, "38.8 miles".

Syntax

MIRouteMultiPointSummary.GetRouteDistanceString() As String

MIRouteMultiPointSummary.GetRouteTimeString method

Purpose

Returns a string that contains the value and units for the travel time of the resulting route. For example, 2.4 hours.

Syntax

MIRouteMultiPointSummary.GetRouteTimeString() As String

MIRouteMultiPointSummary.GetTimeUnits method

Purpose

Returns the unit of measure used in calculating the travel time for the MultiPoint route as a code.

Syntax

MIRouteMultiPointSummary.GetTimeUnits( ) As TimeUnits

MIRouteMultiPointSummary.GetTimeUnitString method

Purpose

Returns the unit of measure used in calculating the travel time for the MultiPoint route as text such as minutes or hours.

Syntax

MIRouteMultiPointSummary.GetTimeUnitString( ) As String

MIRouteMultiPointSummary.GetTimeValue method

Purpose

Returns the number of time units or the amount of travel time such as 32.25 in 32.25 minutes.

Syntax

MIRouteMultiPointSummary.GetTimeUnitValue( ) As Double

MIRouteMultiPolygon Object

This object is a collection of polygons that make up a single iso result. It may contain the main polygon, islands and holes.

Property

- gid
- srsName

Method

- GetPolygons

MIRouteMultiPolygon.GetPolygons method

Purpose

Returns a MIRouteCollection of MIRoutePolygon objects.

Syntax

MIRouteMultiPoint.GetPolygons() As MIRoutePolygon

MIRouteMultiPolygon.gid property

Purpose

Used internally by the system.

MIRouteMultiPolygon.srsName property

Purpose

Used internally by the system.

MIRoutePoint Object

The MIRoutePoint object stores longitude, latitude and level values of the d latitude values of the points of a route. Points can be input values or can be return values specifying the end points and/or shape points of the route returned from the server.

Properties

- Latitude
- Level
- Longitude
- gid
- srsName

Methods

- Equals
- SetPoint
- ToString

MIRoutePoint.Equals method

Purpose

Compares the latitude, longitude and level of two points to 15 significant digits.

Returns True if the points match.

Syntax

MIRoutePoint.Equals(*RtPnt* As Object) As Boolean

MIRoutePoint.gid property

Purpose

Used internally by the system.

MIRoutePoint.Latitude property

Purpose

This property holds the latitude or Y-axis value of a point.

Syntax

MIRoutePoint.Latitude As Double

MIRoutePoint.Level property

Purpose

This property holds the level or relative altitude indicator for a point.

Syntax

MIRoutePoint. Level As Long

MIRoutePoint.Longitude property

Purpose

This property holds the longitude or X-axis value of a point.

Syntax

MIRoutePoint.Longitude As Double

MIRoutePoint.SetPoint method

Purpose

This method sets the longitude, latitude and level values from the user's input.

Syntax

MIRoutePoint.SetPoint(*Longitude* As Double, *Latitude* As Double, *Level* As Long)

MIRoutePoint.srsName property

Purpose

Used internally by the system.

MIRoutePoint.ToString method

Purpose

Returns the values of a point in the format (latitude, longitude, level).

Syntax

MIRoutePoint.ToString() As String

MIRoutePolygon Object

The MIRoutePolygon object contains one MIRouteBoundary object representing the outer boundary of the IsoResult. Additionally, the MIRoutePolygon object can contain one or more MIRouteBoundary objects that represent the inner boundaries of the IsoResult also known as holes.

Property

- gid
- srsName

Method

- GetInnerBoundaries
- GetOuterBoundaries

MIRoutePolygon.GetInnerBoundaries method

Purpose

Returns an MIRouteCollection of MIRouteBoundary objects, or an error indication of no Inner Boundary objects.

Syntax

MIRoutePolygon.GetInnerBoundaries() As MIRouteCollection

MIRoutePolygon.GetOuterBoundary method

Purpose

Returns an MIRouteBoundary object. Each MIRouteMultiPolygon will have one or more MIRoutePolygons. Each MIRoutePolygon will have one and only one OuterBoundary.

Syntax

MIRoutePolygon.GetOuterBoundary() As MIRouteBoundary

MIRoutePolygon.gid property

Purpose

Used internally by the system.

MIRoutePolygon.srsName property

Purpose

Used internally by the system.

MIRouteRequest Object

This class holds a request including preferences, for a route. The preferences may be overridden by server preferences if the server preferences are more restrictive. For example, if the user requests driving directions and the server has them disabled, driving directions would not be returned.

Properties

- DistanceUnit
- EndPoint
- FindShortestDistance
- IncludeActualConstraints
- PrimaryNamesOnly
- ReturnDirections
- ReturnSegmentData
- ReturnSegmentGeometry
- SpeedUnit
- StartPoint
- TimeUnit

Methods

- Copy
- GetDistanceUnitString
- GetTimeUnitString
- Reset

MIRouteRequest.Copy method

Purpose

Copy this MIRouteRequest object to a new MIRouteRequest object.

Syntax

MIRouteRequest.Copy() As MIRouteRequest

MIRouteRequest.DistanceUnit property

Purpose

This setting indicates the unit of measure to be used when returning the length of the route, segments and streets.

The default value is DIST_UNIT_MILES.

Syntax

MIRouteRequest.DistanceUnit As DistanceUnits

MIRouteRequest.EndPoint property

Purpose

MIRoutePoint containing the latitude, longitude and level of the end of the route.

Syntax

MIRouteRequest.EndPoint As MIRoutePoint

MIRouteRequest.FindShortestDistance property

Purpose

This setting indicates the route should be calculated to be the one with the shortest distance (true) or the shortest time (false).

Syntax

MIRouteRequest.FindShortestDistance As Boolean

MIRouteRequest.GetDistanceUnitString method

Purpose

This method returns a string that describes the distance unit type: feet, yards, miles, meters or kilometers.

Syntax

MIRouteRequest.GetDistanceUnitString() As String

MIRouteRequest.GetTimeUnitString method

Purpose

This method returns a string describing the time unit type: seconds, minutes, or hours.

Syntax

MIRouteRequest.GetTimeUnitString() As String

MIRouteRequest.IncludeActualConstraints property

Purpose

This setting indicates if the actual constraints (preferences) the server used to generate the route will be included in the response.

The default setting is false.

Syntax

MIRouteRequest.IncludeActualConstraints As Boolean

MIRouteRequest.PrimaryNamesOnly property

Purpose

Flag indicating whether alternate street name should be included in driving directions.

The default is false.

Syntax

MIRouteRequest.PrimaryNameOnly As Boolean

MIRouteRequest.Reset method

Purpose

Resets the MIRouteRequest object to a default state with start and end points = 0 and all preferences set to the defaults.

Syntax

MIRouteRequest.Reset()

MIRouteRequest.ReturnDirections property

Purpose

This setting indicates if driving directions should be returned by the server.

The default is true.

Syntax

MIRouteRequest.ReturnDirections As Boolean

MIRouteRequest.ReturnSegmentData property

Purpose

This setting indicates if all data for each segment should be returned such as alternate names, road types, etc.

The default is false.

Syntax

MIRouteRequest.ReturnSegmentData As Boolean

MIRouteRequest.ReturnSegmentGeometry property

Purpose

This setting indicates whether no points, end points, or all points should be returned.

The default = SEQMENT_GEOMETRY_ALL_POINTS.

Syntax

MIRouteRequest.ReturnSegmentGeometry(As SegmentGeometry

MIRouteRequest.SpeedUnit property

Purpose

Indicates the unit of measure to be used when calculating the route result.

The default = SPEED_UNIT_MPH.

Syntax

MIRouteRequest.SpeedUnit As SpeedUnits

MIRouteRequest.StartPoint property

Purpose

MIRoutePoint containing the latitude, longitude and level where the route should start.

Syntax

MIRouteRequest.StartPoints As MIRoutePoint

MIRouteRequest.TimeUnit property

Purpose

The unit of measure to use in presenting time in the result such as total time required of 18.3 minutes.

The default = TIME_UNIT_SECONDS.

Syntax

MIRouteRequest**.TimeUnit** As TimeUnits

MIRouteResult Object

This object packages the returned values associated with a single routing request. It contains the match results and route information. It is returned by the RoutingClient FindRoute. The client application can use its methods to interrogate the results of the routing.

The properties that corresponds to preferences available on the MIRouteRequest object will only be populated if IncludeActualConstrains was set to TRUE on the request. These properties are read-only on this object.

Properties

- DistanceUnit
- IncludeActualConstraints
- RequestConstraintsOverridden
- ReturnSegment Data
- ReturnSegment Geometry
- ReturnStreetDirections
- ShortestTime
- Success
- TimeUnit
- VelocityUnit

Methods

- GetDirections
- GetDistances
- GetErrorCode
- GetErrorMessage
- GetIntersections
- GetIntersectionsCount
- GetPointCount
- GetPoints
- GetRequest
- GetRouteDistance
- GetRouteDistanceString
- GetRouteStreet
- GetRouteSummary
- GetRouteTime

- GetRouteTimeString
- GetSegmentCount
- GetStartPoint
- GetStreetCount
- GetStreets
- ResetResult

MIRouteResult.DistanceUnit

Purpose

The unit of measure used when returning the length of the route, segments and streets.Distance unit used in the route.

Syntax

MIRouteResult.DistanceUnit As DistanceUnits

MIRouteResult.GetDirections method

Purpose

Returns the driving directions for the entire route.

Syntax

MIRouteResult.GetDirections() As String

MIRouteResult.GetDistances method

Purpose

This method returns an array of all the distances in the route

Syntax

MIRouteResult.GetDistances() As Double ()

MIRouteResult.GetErrorCode method

Purpose

Returns an error code in the event of a problem. This method is only meaningful if Success property is false. Note that a return of –1 indicates an error, but no error code was set by the server.

The value of 0 indicates no error occurred.

Syntax

MIRouteResult.GetErrorCode() As Long

MIRouteResult.GetErrorMessage method

Purpose

Returns a text description that explains the error code returned in GetErrorCode method.

Available only when the Success property is false.

Syntax

MIRouteResult.GetErrorMessage() As String

MIRouteResult.GetIntersections method

Purpose

Returns an MIRouteCollection of MIRoutePoint objects. Each point is the first point of each street in the route (excluding the first street.)

Syntax

MIRouteResult.GetIntersections() As MIRouteCollection

MIRouteResult.GetIntersectionsCount method

Purpose

This method returns the number of intersections in the route.

Syntax

MIRouteResult.GetIntersectionsCount() As Long

MIRouteResult.GetPointCount method

Purpose

This method returns the total count of points in the route.

Syntax

GetPointCount() As Long

MIRouteResult.GetPoints method

Purpose

Returns an MIRouteCollection of MIRoutePoints.

This method returns a collection of all the point objects in the route. It is a convenience function for drawing maps without providing driving directions.

Syntax

MIRouteResult.GetPoints() As MIRouteCollection

MIRouteResult.GetRequest method

Purpose

Returns the original MIRouteRequest object.

Syntax

MIRouteResult.GetRequest() As MIRouteRequest

MIRouteResult.GetRouteDistance method

Purpose

Returns the value of the distance of the route such as 38.8. It does not include the unit of measure such as miles.

Syntax

MIRouteResult.GetRouteDistance() As Double

MIRouteResult.GetRouteDistanceString method

Purpose

Returns a string that contains the value and units for the length of the resulting route. For example, "38.8 miles".

Syntax

MIRouteResult.GetRouteDistanceString() As String

MIRouteResult.GetRouteStreet method

Purpose

This method gets the specified street from the result.

Syntax

MIRouteResult.GetRouteStreet(*index* As Long) As MIRouteStreet

MIRouteResult.GetRouteSummary method

Purpose

Returns the total time and distance for the route, formatted to a text string such as "Route totaltime = 6664.1 minutes, distance = 739.8 miles".

Syntax

MIRouteResult.GetRouteSummary() As String

MIRouteResult.GetRouteTime method

Purpose

This method returns the time of the complete route using the unit of measure specified in the TimeUnit property.

Syntax

MIRouteResult.GetRouteTime() As Double

MIRouteResult.GetRouteTimeString method

Purpose

Return a string formatted with the requested time units. For example, "2.4 hours".

Syntax

MIRouteResult.GetRouteTimeString() As String

MIRouteResult.GetSegmentCount method

Purpose

Returns the total number of segments in the route.

Syntax

MIRouteResult.GetSegmentCount() As Long

MIRouteResult.GetStartPoint method

Purpose

This method returns the starting point for the route

Syntax

MIRouteResult.GetStartPoint() As MIRoutePoint

MIRouteResult.GetStreetCount method

Purpose

Returns number of streets included in the result.

Syntax

MIRouteResult.GetStreetCount() As Long

MIRouteResult.GetStreets method

Purpose

Returns an MIRouteCollection of MIRouteStreet objects.

Syntax

MIRouteResult. GetStreets() As MIRouteCollection

MIRouteResult.IncludeActualConstraints

Purpose

This setting indicates if the IncludeActualConstraints property was set on the original request.

Syntax

MIRouteResult. IncludeActualConstraints As Boolean

MIRouteResult.RequestConstraintsOverridden property

Purpose

This setting indicates that one or more constraints were overridden by the server settings instead of using the requested constraint.

Syntax

MIRouteResult. RequestConstraintsOverridden As Boolean

MIRouteResult.ResetResult method

Purpose

Resets the object to an empty, uninitialized state. The MIRouteResult object is only properly filled in by a call to MIRouteClient.FindRoute().

Syntax

MIRouteResult. ResetResult()

MIRouteResult.ReturnSegmentData property

Purpose

This setting indicates if the segment data was returned in the result.

Syntax

MIRouteResult. ReturnSegmentData As Boolean

MIRouteResult.ReturnSegmentGeometry property

Purpose

This setting indicates whether no points were returned, end points only, or all points on each segment.

Syntax

MIRouteResult. ReturnSegmentGeometry As SegmentGeometry

MIRouteResult.ReturnStreetDirections property

Purpose

This setting indicates if driving directions were returned from the server.

Syntax

MIRouteResult. ReturnStreetDirections As Boolean

MIRouteResult.ShortestTime property

Purpose

Indicates if the request is for the shortest-time route, instead of the shortest distance.

Syntax

MIRouteResult. ShortestTime As Boolean

MIRouteResult.Success property

Purpose

This setting indicates whether a route was returned. Note that it is possible for the request to go to the server, but the server us unable find a route for the given request. In this case, the server "responds" successfully (i. e., it receives and evaluates the request), but it is unable to find a route. Therefore, the Success flag would be set to false.

When the Success flag is False, the GetErrorMessage method will return the error message provided by the server

Syntax

MIRouteResult. Success As Boolean

MIRouteResult.TimeUnit property

Purpose

This setting indicates the unit of measure used in presenting time in the result.

Syntax

MIRouteResult. TimeUnit As TimeUnits

MIRouteResult.VelocityUnit property

Purpose

“Speed unit” used in the route.

Syntax

MIRouteResult. VelocityUnit As MIRouteVelocity

MIRouteSegment Object

The MIRouteSegment object stores the information for the segments that make up a street in a route.

When a route is returned from the server, it consists of a series of segments. These segments are the smallest unit of information returned for a street making up a route. The segment can contain its road type (rural route or highway), distance, time to traverse, etc. depending on the information available about the segment.

A segment may also contain a of a series of points which define its geometry. The segment structure contains a count of the points and an array of them. However, if the user did not request point information, no points will be returned. The user can also specify that the server only return the starting and ending points of the segment. In that case, the segment would only have one point which would be the end point of the segment. The ending of the previous segment would be the new starting point for the next segment.

Properties

- CompassDirection
- Distance
- gid
- srsName
- OneWay
- RoutingSegmentID
- Time
- TollRoad
- TurnAngle
- Type

Methods

- GetAltName
- GetCompassDirectionString
- GetPoint
- GetPointCount
- GetPoints
- GetTurnString

MIRouteSegment.CompassDirection property

Purpose

Returns an integer indicating the compass direction of this segment:

north = 1, northeast = 2, east = 3, southeast = 4, south = 5, southwest = 6, west = 7, northwest = 8

Syntax

MIRouteSegment.CompassDirection As Long

MIRouteSegment.Distance property

Purpose

Returns the distance that the route runs along this segment.

Syntax

MIRouteSegment.Distance As Double

MIRouteSegment.GetAltName method

Purpose

Returns the alternate name specified by index, or an empty string if there is no alternate name.

Syntax

MIRouteSegment.GetAltName(index As Long) As String

MIRouteSegment.GetCompassDirectionString method

Purpose

Returns the compass direction as text such as North, West, Southeast, etc.

Syntax

MIRouteSegment.GetCompassDirectionString() As String

MIRouteSegment.GetPoint method

Purpose

Returns the MIRoutePoint object specified by index.

Syntax

MIRouteSegment.GetPoint(index As Long) As MIRoutePoint

MIRouteSegment.GetPointCount method

Purpose

Returns the number of point objects on this segment.

Syntax

MIRouteSegment.**GetPointCount()** As Long

MIRouteSegment.GetPoints method

Purpose

Returns an MIRouteCollection of MIRoutePoint objects.

Syntax

MIRouteSegment.GetPoints() As MIRouteCollection

MIRouteSegment.GetTurnString method

Purpose

Returns a string based on the turn angle: turn left, make a sharp right, continue, and so on.

Syntax

MIRouteSegment.GetTurnString() As String

MIRouteSegment.OneWay property

Purpose

Integer indicating the direction this segment can be traveled.

ONEWAY_BIDIRECTIONAL = 1, ONEWAY_FROM_TO = 2, ONEWAY_TO_FROM = 3, ONEWAY_NON_TRAVERSABLE = 4.

Syntax

MIRouteSegment.OneWay As Long

MIRouteSegment.RoutingSegmentId property

Purpose

This setting indicates a unique string segment id used internally by the server for segment identification. Note: segment ids for the same segments may differ from version to version.

Syntax

MIRouteSegment.RoutingSegmentID As String

MIRouteSegment.Time property

Purpose

This setting returns the amount of time the route runs along this segment.

Syntax

MIRouteSegment.Time As Double

MIRouteSegment.TollRoad method

Purpose

Indicates if this segment is in a toll road.

Syntax

MIRouteSegment.TollRoad As Boolean

MIRouteSegment.TurnAngle property

Purpose

This setting indicates the turn angle (-180 <= turnAngle <= 180). A negative turn angle implies a right turn, a positive turn angle is a left turn.

Syntax

MIRouteSegment.TurnAngle As Double

MIRouteSegment.Type property

Purpose

This setting indicates the road type for this segment.

Syntax

MIRouteSegment.Type As RoadType

MIRouteStreet Object

The MIRouteStreet object stores the information for the streets that make up a route. When a route is returned from the server, it consists of a series of segments. These segments are the smallest unit of information returned for the route. A segment can contain its street name, speed, distance, time to traverse, etc., depending on the information available about the segment. The MIRoute COM client combines the segments of the same name into streets to reduce the information passed back as driving directions.

Methods

- GetCompassString
- GetDirections
- GetDistance
- GetSegment
- GetSegmentCount
- GetSegments
- GetStreetName
- GetStreetNamePlusAlts
- GetTime

MIRouteStreet.GetCompassString method

Purpose

Returns a string indicating the compass direction of the street.

Syntax

MIRouteStreet.GetCompassString() As String

MIRouteStreet.GetDirections method

Purpose

Returns the driving directions for the route on this street.

Syntax

MIRouteStreet.GetDirections() As String

MIRouteStreet.GetDistance method

Purpose

Get the distance of the route on this street.

Syntax

MIRouteStreet.GetDistance() As Double

MIRouteStreet.GetSegment method

Purpose

Returns the MIRouteSegment object specified by index.

Syntax

MIRouteStreet.GetSegment(index As Long) As MIRouteSegment

MIRouteStreet.GetSegmentCount method

Purpose

Return the number of segment objects that make up this street.

Syntax

MIRouteStreet.GetSegmentCount() As Long

MIRouteStreet.GetSegments method

Purpose

Returns a collection of all segments from a street.

Syntax

MIRouteStreet.GetSegments() As MIRouteCollection

MIRouteStreet.GetStreetName method

Purpose

Returns the street name.

Syntax

MIRouteStreet.GetStreetName() As String

MIRouteStreet.GetStreetNamePlusAlts method

Purpose

Returns a string consisting of the street name plus the alternate name concatenated with "/" (e. g., "Hwy 1/Bleeker St.")

Syntax

MIRouteStreet.GetStreetNamePlusAlts() As String

MIRouteStreet.GetTime method

Purpose

Returns the time that the route runs along this street.

Syntax

MIRouteStreet.GetTime() As Double

MIRouteVelocity Object

The MIRouteVelocity object represents a speed unit. It has a LinearUnits property (for example, DIST_UNIT_MILES), and a TimeUnits property (e.g., TIME_UNIT_HOURS). Taken together: "miles per hour".

Additionally, the VelocityValue property can be filled in to represent an actual speed measurement: "15 mph". The MIRouteIsoResponse.GetAmbientSPeedOverride method returns just such a MIRouteVelocity object.

Properties

- LinearUnits
- TimeUnits
- VelocityValue

MIRouteVelocity.LinearUnit property

Purpose

The unit of measure for the distance used in stating the rate of travel such as miles in 25 mph.

Syntax

MIRouteVelocity.LinearUnit As DistanceUnits

MIRouteVelocity.TimeUnits property

Purpose

The unit of measure for the time used in stating the rate of travel such as hours in 25 mph.

Syntax

MIRouteVelocity.TimeUnit As TimeUnits

MIRouteVelocity.VelocityValue property

Purpose

The number of units used in stating the rate of travel such as 25 in 25 mph.

Syntax

MIRouteVelocity.VelocityValue As Double

Appendix A: Enumerated Constants

DistanceUnits

- DIST_UNIT_FEET
- DIST_UNIT_YARDS
- DIST_UNIT_MILES
- DIST_UNIT_METERS
- DIST_UNIT_KILOMETERS

IsoBandingStyles

- ISO_BANDING_ENCOMPASSING
- ISO_BANDING_DONUT

IsoResponseTypes

- ISO_RESPONSE_TYPE_GEOMETRY
- ISO_RESPONSE_TYPE_STARTNODES,
- ISO_RESPONSE_TYPE_ACCESSIBLENODES

IsoResultTypes

- ISO_RESULT_TYPE_POLYGON
- ISO_RESULT_TYPE_NODE

IsoUnits

- ISO_UNIT_DISTANCE
- ISO_UNIT_TIME

RoadTypes

- LIMITED_ACCESS_DENSE_URBAN
- PRIMARY_HIGHWAY_DENSE_URBAN,
- SECONDARY_HIGHWAY_DENSE_URBAN,
- MAJOR_ROAD_DENSE_URBAN,
- NORMAL_ROAD_DENSE_URBAN,
- MAJOR_LOCAL_ROAD_DENSE_URBAN,
- LOCAL_ROAD_DENSE_URBAN,
- MINOR_LOCAL_ROAD_DENSE_URBAN,
- RAMP_DENSE_URBAN,
- LIMITED_ACCESS_URBAN,
- PRIMARY_HIGHWAY_URBAN,
- SECONDARY_HIGHWAY_URBAN,
- MAJOR_ROAD_URBAN,
- NORMAL_ROAD_URBAN,
- MAJOR_LOCAL_ROAD_URBAN,

- LOCAL_ROAD_URBAN,
- MINOR_LOCAL_ROAD_URBAN,
- RAMP_URBAN,
- LIMITED_ACCESS_SUBURBAN,
- PRIMARY_HIGHWAY_SUBURBAN,
- SECONDARY_HIGHWAY_SUBURBAN,
- MAJOR_ROAD_SUBURBAN,
- NORMAL_ROAD_SUBURBAN,
- MAJOR_LOCAL_ROAD_SUBURBAN,
- LOCAL_ROAD_SUBURBAN,
- MINOR_LOCAL_ROAD_SUBURBAN,
- RAMP_SUBURBAN,
- LIMITED_ACCESS_RURAL,
- PRIMARY_HIGHWAY_RURAL,
- SECONDARY_HIGHWAY_RURAL,
- MAJOR_ROAD_RURAL,
- NORMAL_ROAD_RURAL,
- MAJOR_LOCAL_ROAD_RURAL,
- LOCAL_ROAD_RURAL,
- MINOR_LOCAL_ROAD_RURAL,
- RAMP_RURAL,
- RAMP_LIMITED_ACCESS,
- RAMP_MAJOR_ROAD,
- RAMP_PRIMARY_HIGHWAY,
- RAMP_SECONDARY_HIGHWAY,
- FOOTPATH,
- FERRY,
- BACK_ROAD

SegmentGeometry

- SEGMENT_GEOMETRY_ALL_POINTS
- SEGMENT_GEOMETRY_END_POINTS
- SEGMENT_GEOMETRY_NONE

SpeedUnits

- SPEED_UNIT_MPH
- SPEED_UNIT_KMPH

TimeUnits

- TIME_UNIT_SECONDS
- TIME_UNIT_MINUTES
- TIME_UNIT_HOURS