

# AUTODESK® LOCATIONLOGIC

## XML Web Services Developer's Guide

LL 6.6/WSG 3.6, July 27, 2007

**Autodesk®**  
Location Services



This publication, or parts thereof, may not be reproduced in any form, by any method, for any purpose.

**AUTODESK, INC. MAKES NO WARRANTY, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, REGARDING THESE MATERIALS AND MAKES SUCH MATERIALS AVAILABLE SOLELY ON AN "AS-IS" BASIS.**

**IN NO EVENT SHALL AUTODESK, INC. BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH OR ARISING OUT OF PURCHASE OR USE OF THESE MATERIALS. THE SOLE AND EXCLUSIVE LIABILITY TO AUTODESK, INC., REGARDLESS OF THE FORM OF ACTION, SHALL NOT EXCEED THE PURCHASE PRICE OF THE MATERIALS DESCRIBED HEREIN.**

Autodesk, Inc. reserves the right to revise and improve its products as it sees fit. This publication describes the state of this product at the time of its publication, and may not reflect the product at all times in the future.

#### GOVERNMENT USE

Use, duplication, or disclosure by the U. S. Government is subject to restrictions as set forth in FAR 12.212 (Commercial Computer Software-Restricted Rights) and DFAR 267.7202 (Rights in Technical Data and Computer Software), as applicable.

# Contents

<b>About This Guide</b>	<b>vii</b>
Audience and Purpose	viii
How This Guide Is Organized	viii
Conventions Used in This Guide	viii
Text Conventions	viii
Code and Syntax Conventions	ix
Before You Begin	x
Programming and Markup Languages	x
GIS Concepts	x
Where to Get More Information	x
 <b>Chapter 1</b>	 <b>Getting Started with the LocationLogic XML Web Services</b>
About LocationLogic XML Web Services	2
XML Web Services Architecture	3
Services Provided	3
Comparison with the OpenLS Schema	4
LocationLogic XML Web Services Schema Files	5
Working with the Web Services Gateway	6
Content	6
Authentication and Authorization	7
Error Handling	7
Firewall Considerations	8
Secure Communications	9
Encoding XML Documents	11
Request Validation	12
Working with Geographical Data	17
Spatial Reference Systems (SRS)	17
Coordinates	17
Geometry	18
Using Precision Elements	19
 <b>Chapter 2</b>	 <b>About the XML Web Services</b>
Provisioning Services	22
Client Provisioning	22
Location Privacy Provisioning	23
Location Gateway Services	25
Mobile Devices	25

Locations . . . . .	25
Locating Mobile Devices . . . . .	25
Geoservices . . . . .	26
Location Utility (Geocoding/Reverse Geocoding) Service . .	26
Directory Service . . . . .	26
Presentation Service . . . . .	27
Route Service . . . . .	30
Alert Service . . . . .	31
Auxiliary Services. . . . .	33
Transaction Logging Service . . . . .	33
Service and Deployment Test URLs . . . . .	34

### **Chapter 3 Making LocationLogic XML Web Services Requests . . . . . 35**

XML Web Services Requests . . . . .	36
LocationLogic and OpenLS (XLS) Services . . . . .	36
OpenLS (XLS) Document Structure . . . . .	36
Client Provisioning Requests . . . . .	39
Adding a Client . . . . .	39
Retrieving a Client . . . . .	41
Updating a Client. . . . .	45
Removing Client Properties or Roles. . . . .	46
Removing a Client . . . . .	47
Location Privacy Provisioning Requests . . . . .	48
Creating Privacy Profiles . . . . .	48
Retrieving a Privacy Profile. . . . .	49
Updating a Privacy Profile . . . . .	51
Removing Part of a Privacy Profile . . . . .	52
Removing a Privacy Profile. . . . .	53
Location Gateway Requests . . . . .	55
Location Request . . . . .	55
Location Request with Quality of Position . . . . .	56
Enhanced Support of MLP Elements. . . . .	58
Geocoding/Reverse Geocoding Requests . . . . .	60
Geocoding an Address . . . . .	60
Geocoding a Fully-Qualified Address. . . . .	60
Geocoding a Partial Address . . . . .	62
Geocoding a Free-Form Address . . . . .	63
Geocoding a Street Intersection . . . . .	65
Geocoding Search Modes . . . . .	67
Reverse Geocoding a Coordinate . . . . .	68
Interpolation Mode . . . . .	70
Intelligent Reverse Geocoding. . . . .	70
Directory Requests . . . . .	71
Finding the Nearest Places . . . . .	71

Retrieving Alternate Names for POIs . . . . .	73
Accessing Extended Attributes of a POI . . . . .	75
Finding Places Within a Circular Boundary . . . . .	76
Finding Places Along a Route (Within Distance of a Linestring). . . . .	77
Finding Places With SQL WHERE Queries. . . . .	79
Find POIs with Specified Name Segments. . . . .	80
Find POIs with a Specific Name . . . . .	80
Find POIs of a Specific Type . . . . .	81
Find POIs with Specific Address Criteria . . . . .	81
Fuzzy POI Name Search . . . . .	81
Optimizing Directory Requests . . . . .	84
Using DirectoryUpdateRequest . . . . .	84
Creating a New POI. . . . .	84
Creating a New POI with Extended Attributes . . . . .	85
Editing POI Attributes . . . . .	87
Deleting a POI . . . . .	88
Presentation Requests . . . . .	89
Generating a Map URL for a Specified Area . . . . .	89
Generating a Map URL Surrounding a Point . . . . .	91
Generating a Binary Map . . . . .	94
Specifying a Logical Map Name . . . . .	95
Adding a Point to a Map . . . . .	96
Map Symbols. . . . .	97
Adding a Route to a Map . . . . .	99
Adding Points Along a Route. . . . .	101
Adding a Route to a Map Using a Route Handle. . . . .	104
Displaying Routes with Different Line Styles. . . . .	104
Displaying Positions with Precision . . . . .	106
Drawing with Position Overlays. . . . .	109
Hiding the Position Overlay Symbol . . . . .	113
Drawing a CircularArc . . . . .	114
Drawing Nested Polygons. . . . .	115
Adding Labels to Map Overlays . . . . .	117
Creating a Map with a City Boundary . . . . .	118
Adding an Image Map Hotspot to a Route . . . . .	121
Converting from Points to Pixels . . . . .	123
Converting from Pixels to Points . . . . .	124
Using a Map URL for Coordinate Conversions . . . . .	126
Route Requests . . . . .	127
Generating Maps . . . . .	127
Calculating a Route Between Two Addresses . . . . .	127
Calculating a Route Between Two Points . . . . .	130
Requesting Maps with Route Requests. . . . .	132
Retrieving Previously Determined Routes with Route Handles . . . . .	139
Specifying Features to Avoid in Route Requests . . . . .	142

Time Zone Support in Route Requests . . . . .	142
Time-Dependent and Time-Independent Route Service Responses . . . . .	142
Alert Requests . . . . .	144
Listing Subscriptions . . . . .	144
Creating a Subscription for Traffic Along a Route . . . . .	145
Editing a Subscription . . . . .	146
Activating a Subscription . . . . .	147
Deactivating a Subscription . . . . .	148
Deleting a Subscription . . . . .	148
Receiving Alert Messages . . . . .	149
<b>Appendix A Error Codes . . . . .</b>	<b>151</b>
Geoservice Error Codes . . . . .	152
Geoservice Error Codes . . . . .	152
Geoservice Implementation Error Codes . . . . .	153
Geoservice Validation Error Codes . . . . .	156
Geoservice Unsupported Feature and Format Error Codes . . . . .	159
Geoservice Generic Error Codes . . . . .	159
Location Provisioning Result Codes . . . . .	160
Client Provisioning Result Codes . . . . .	162
Auxiliary Result Codes . . . . .	164
<b>Index . . . . .</b>	<b>167</b>

# About This Guide

Autodesk® LocationLogic XML Web Services provide simple and powerful building blocks for rapid development of location-based applications on the Autodesk LocationLogic platform.

This guide contains instructions and guidelines for developing applications using LocationLogic XML Web Services.

This chapter explains what's in the guide and how it's organized.

## In this chapter

- Audience and purpose
- How this guide is organized
- Conventions used in this guide
- Before you begin
- Where to get more information

# Audience and Purpose

This guide is intended for LocationLogic application programmers. It introduces LocationLogic XML Web Services and provides guidelines for developing your applications.

## How This Guide Is Organized

This guide is organized as follows:

- Chapter 1, “Getting Started with the LocationLogic XML Web Services,” provides guidelines for developing applications with LocationLogic XML Web Services.
- Chapter 2, “About the XML Web Services,” presents a conceptual overview of each group of the XML Web Services and provides a list of service and deployment test URLs.
- Chapter 3, “Making LocationLogic XML Web Services Requests,” explains what your application can do with XML Web Services, and provides XML examples of common service requests and responses.
- Appendix A, “Error Codes,” lists the error codes provided by the Web Services Gateway (WSG).

## Conventions Used in This Guide

This section describes the following conventions used in this guide:

- Text conventions
- Code and syntax conventions

### Text Conventions

This guide uses the following text conventions:

- *Italic* is used to introduce new terms. Italic is also used for database column names, file and folder names, and book titles.
- **Bold** is used for any text you must enter, such as at a command line prompt or in a dialog box.



- A `monospace` font is used for all code elements (variable names, data values, method names, and so forth), command lines, scripts, and source code listings.
- ***Bold italic monospace*** is used for replaceable elements and placeholders within code listings.

## Code and Syntax Conventions

Indentation has been added to make examples more legible.

The following table describes code and syntax conventions used in this manual:

These symbols...	Indicate this...
	<p>The vertical-bar or pipe symbol separates alternative items that may be optional or required. You may choose exactly one of the given items. Do not type the vertical bar. For example, the text:</p> <pre>A   B   C</pre> <p>indicates that you should choose only one item—A or B or C.</p>
[ ]	<p>Square brackets enclose one or more optional items. Do not type the brackets. For example, the text:</p> <pre>[A   B   C]</pre> <p>indicates that you can choose no items or a single item—A or B or C.</p> <p>While the text:</p> <pre>[D]</pre> <p>indicates that you can choose no items or item D.</p>
{ }	<p>Braces enclose one or more required items. Do not type the braces. For example, the text:</p> <pre>{A   B   C}</pre> <p>indicates that you must choose a single item—A or B or C.</p>
...	<p>Ellipses mean that the preceding item(s) may be repeated any number of times.</p>

# Before You Begin

To benefit from this guide, it helps to be familiar with the following:

- Programming and markup languages
- GIS concepts

## Programming and Markup Languages

You should also have the following technical knowledge:

- XML tagging
- Web application architecture

LocationLogic XML Web Services is vendor- and language-independent.

## GIS Concepts

An understanding of GIS (Geographic Information Systems) concepts such as coordinate systems, geocoding, and reverse geocoding is helpful in using the LocationLogic routing functions. For more information, see “Working with Geographical Data” on page 17.

# Where to Get More Information

LocationLogic XML Web Services adhere to several industry standards. The following websites may be helpful:

- ISO 639 two-letter language code standard:  
<http://lcweb.loc.gov/standards/iso639-2/iso639jac.html>
- Open GIS Consortium (OGC) specifications:  
<http://www.opengis.org/techno/specs.htm>
- Open Location Services - OpenLS™  
<http://www.opengeospatial.org/standards/olscore>
- XML (Extensible Markup Language):  
<http://www.w3.org/XML/>

# Getting Started with the LocationLogic XML Web Services

This chapter introduces you to LocationLogic XML Web Services and provides guidelines for developing location-based applications.

# 1

## In this chapter

- About LocationLogic XML Web Services
- Working with the Web Services Gateway
- Working with geographical data

# About LocationLogic XML Web Services

Autodesk LocationLogic XML Web Services provide a language- and vendor-independent solution for remotely developing and running Location-Based Services (LBS) applications.

The Autodesk LocationLogic platform implements building blocks for geoservices based on the Open Location Services (OpenLS) standard. LocationLogic XML Web Services are aligned with the OpenLS XML for Location Services (XLS) specification. Autodesk is a principal member of the Open GIS Consortium and a co-author of XLS.

LocationLogic XML Web Services enable you to develop LBS applications for multiple delivery channels, such as SMS and MMS. You can remotely build and deploy a wide variety of applications using development frameworks such as J2SE™, J2EE, JavaME, .NET, BREW®, and Windows Mobile®.

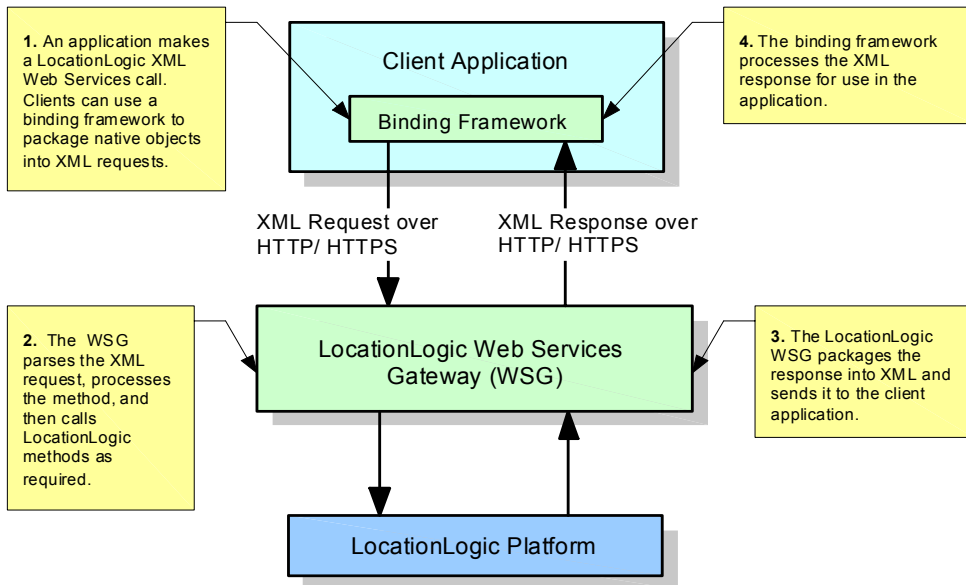
Applications using LocationLogic XML Web Services interact with the LocationLogic platform via the *XML Web Services Gateway* (WSG)—a programmatic interface to LocationLogic, accessible across a network such as the internet.

This section covers the following topics:

- “XML Web Services Architecture” on page 3
- “Services Provided” on page 3
- “Comparison with the OpenLS Schema” on page 4
- “LocationLogic XML Web Services Schema Files” on page 5

# XML Web Services Architecture

The following diagram shows the functional components of the LocationLogic XML Web Services architecture.



## LocationLogic XML Web Services Architecture

**Note** While a binding framework can simplify the creation and handling of XML messages, there is no requirement that a binding framework be used by a client application when accessing the WSG.

## Services Provided

LocationLogic XML Web Services are represented as XML schemas. The following services are described in more detail in Chapter 2, “About the XML Web Services.”

## Provisioning Services

- **Client Provisioning service**—Establishes client application identifiers, passwords, and roles for authorized use of specific location services and geoservices.
- **Location Privacy Provisioning service**—Defines default and specific levels of subscriber location privacy in relation to client applications.

## Location Gateway Services

- **Determination of device location**—Determines the location of mobile devices.

## Geoservices

- **Geocoding/Reverse Geocoding (Location Utility) service**—Supplies geocoded and reverse geocoded data.
- **Directory service**—Provides a requested specific or nearest place, product, or service.
- **Presentation service**—Provides requested maps, including maps with overlay information, which applications can display on a variety of output devices.
- **Route service**—Provides requested routes for various start- and end-point specifications.
- **Alert service**—Provides subscriptions to events, such as traffic incidents along a route, and sends XML message alerts when a subscription is triggered.

## Auxiliary Services

- **Transaction Logging service**—Clients can log transactions through the Auxiliary Services. The logging service includes the ability to record fine-grained information. To enable billing for premium feature categories, the feature category queried in Directory requests is recorded in the transaction log.

For more information about the Auxiliary Services, contact your LocationLogic administrator.

## Comparison with the OpenLS Schema

Although the LocationLogic XML Web Services wire (data transmission) format is identical to that of the OpenLS Specification schema, the corresponding LocationLogic XML schema has been simplified for the benefit of application developers, as follows:

- The LocationLogic XML schema omits features (such as substitution groups with abstract head elements) that are not adequately supported by third-party XML tools.

- To reduce the learning curve for developers, the LocationLogic XML schema omits the complex abstract inheritance hierarchy of the OpenLS schema.
- Because the LocationLogic XML schema uses the `anyType` type for unsupported OpenLS elements, the WSG can validate any XML document that can be validated by the OpenLS schema.
- The LocationLogic XML Web Services schema extends the OpenLS schema for several services. In the Directory service, for example, applications can filter results by using any `POIProperty` name, not just the pre-defined enumerated values allowed by the OpenLS schema. In addition, applications using the Directory service can search for a specific set of POI properties. Another schema extension example appears in the Route service, where applications can request routes that avoid ferries, bridges, and tunnels.
- The LocationLogic XML Web Services schema supports additional services not included in the OpenLS standard. These include client provisioning and location privacy provisioning.

**Note** Autodesk is working to have these extensions incorporated into future revisions of the OpenLS specification.

## LocationLogic XML Web Services Schema Files

The LocationLogic XML Web Services schema is defined in the following files:

- *xls2.xsd*—Types and definitions for all elements within the `xls` namespace. In the OpenLS Specification, these types and definitions are distributed in more than ten separate files.
- *gml4xls2.xsd*—Types and definitions for all elements within the `gml` (Geographic Markup Language) namespace. In the OpenLS Specification, these types and definitions are in the *gml4xls2.xsd* file.
- *clientProv.xsd*—Types and definitions for all the elements for client provisioning requests and responses. These are extensions to the OpenLS specification.
- *locProv.xsd*—Types and definitions for all the elements for location provisioning requests and responses. These are extensions to the OpenLS specification.
- *llaux.xsd*—Types and definitions for various auxiliary functions. These are extensions to the OpenLS specification.

For complete descriptions of the LocationLogic XML Web Services schemas, refer to the *Autodesk LocationLogic XML Web Services Schema Reference*.

# Working with the Web Services Gateway

The Web Services Gateway (WSG) is the server-side component of LocationLogic XML Web Services, and is deployed with the LocationLogic platform. To work with the WSG, you need to understand the following:

- “Content” on page 6
- “Authentication and Authorization” on page 7
- “Error Handling” on page 7
- “Firewall Considerations” on page 8
- “Secure Communications” on page 9
- “Encoding XML Documents” on page 11
- “Request Validation” on page 12

## Content

LocationLogic XML Web Services client applications work with content deployed on the LocationLogic platform, including:

- Street network data
- POIs
- Basemaps

When making XML requests, you will need to know the following about the content deployed with the LocationLogic platform:

- Feature category hierarchy
- Content area (for example, US, Europe, Italy)
- Location of maps and their logical names
- Highlight styles
- Map symbols

For detailed information about the LocationLogic feature category hierarchy, and content and map configuration, contact your LocationLogic administrator.

## Related Topic

- For a list of map symbols that you can use with LocationLogic maps, see “Map Symbols” on page 97.



## Authentication and Authorization

The authentication services of the WSG control which client applications can access LocationLogic. Application developers should provide authentication information in the `RequestHeader` element of each XML request.

Once the WSG has authenticated a client application, the WSG authorization services control which LocationLogic services will be available to the client application. For details on client application authentication and authorization, see “Client Provisioning” on page 22.

## Error Handling

You should be aware of the following when you develop the error handling portion of your application:

- **Ignored elements**—If the LocationLogic XML Web Services Gateway encounters an element or attribute that it ignores, and which would not cause erroneous results, no warning message or error is generated.
- **Multiple service requests in a single XML document**—If an application sends a single XML document that contains multiple service requests, each request is validated independently. Since it is possible that some requests will succeed while others will fail, the LocationLogic XML Web Services Gateway returns error codes for the failed requests, and processes the successful requests independently.
- **LocationLogic errors**—When the LocationLogic XML Web Services Gateway reports LocationLogic errors, the `message` attribute of the `Error` element is used. The string begins with “Lbs error code:”, and the LocationLogic error code itself is included.  
**Note** For assistance with resolving Lbs errors, contact your LocationLogic administrator.
- **Stack trace information**—The LocationLogic XML Web Services Gateway reports stack traces when an XML parsing error or validation error occurs. If you need to view stack trace information for implementation errors, contact your LocationLogic administrator.
- **XML comments**—XML comments are not retained by the LocationLogic XML Web Services Gateway during processing. Therefore, if you receive an error response that includes the line number (as is the case when the LocationLogic XML Web Services Gateway detects an incorrectly formed document during XML parsing), the line number will not be accurate if your request document includes comments.

## Support for OpenLS Error Attributes

The following table describes the OpenLS Specification error parameters and how they are used and supported by LocationLogic XML Web Services:

### WSG Support of OpenLS Error Attributes

OpenLS Attribute	Required	Description	WSG Support
errorCode	Yes	Enumeration that categorizes the error	Supported.  For a complete list of error codes, see “Error Codes” on page 151.
severity	No	Indicates Warning or Error	Error supported.
locationID	No	Refers to the element that caused an error	Unsupported.
locationPath	No	In a well-formed XML document, this attribute contains the path of the XML element or attribute associated with the error	Supported for syntactic validation, as described in “Deserialization,” on page 13.
message	No	Provides a human-readable explanation of the error	Supported.  Includes error codes for both WSG and LocationLogic errors.  LocationLogic errors use the prefix, “Lbs error code:”  WSG errors use the prefix, “WSG error code:”

## Firewall Considerations

For LocationLogic XML Web Services applications to successfully communicate with the WSG, the WSG must be visible to clients outside the firewall. Similarly, for applications to successfully access map URLs created in response to Presentation service requests, the map URL must also be visible to clients outside the firewall. If you cannot access such URLs, contact your LocationLogic administrator.

## Secure Communications

The WSG allows clients to use secure communications to ensure the privacy of exchanged data.

The SSL (*Secure Sockets Layer*) is a security protocol that allows data to be encrypted before it is sent across the wire between a client and a server. To enable SSL communication, the client and server must undergo a handshaking process where each party has the option of authenticating the other, determining which encryption algorithm to use for the remainder of the session, and using a combination of public and private keys to perform the actual data encryption and decryption. By convention, URLs that require an SSL-enabled channel begin with *https*: instead of *http*:

For information about the Java implementation of the SSL handshaking process, refer to the following website: <http://java.sun.com/j2se/1.5.0/docs/guide/security/jsse/JSSERefGuide.html>

## Configuring Java Clients for SSL

To enable your application to communicate with the WSG by using SSL, perform the steps outlined in the following table:

### Process Overview: Configuring Java Clients for SSL

Task	Description	Refer To
1	Use the JDK classes and JSSE	"Task 1: Using the JDK Classes and JSSE" on page 9
2	Register the WSG digital certificate	"Task 2: Registering the WSG Digital Certificate" on page 10
3	Configure SSL properties	"Task 3: Configuring SSL Properties" on page 11

### Task 1: Using the JDK Classes and JSSE

For a typical HTTP connection, Java clients can use the `HttpURLConnection` class that is part of the JDK (Java Development Kit). To make HTTPS connections, Java clients use the `HttpsURLConnection` class.

Java applications that require secure connections to the WSG server should use the JSSE (Java Secure Socket Extension) library. Depending on which version of the JDK you use, you might need to download JSSE from the Java website:

- JDK1.2.x or 1.3.x—Download the JSSE library from the Java website: <http://java.sun.com/products/jsse/index-103.html>
- JDK1.4 and later—The JSSE library is included in the development kit.

### **Related Topic**

- For more information and configuration details about JSSE, refer to the following website: <http://java.sun.com/products/jsse/>

### **Task 2: Registering the WSG Digital Certificate**

In addition to installing and configuring the JSSE library, your application must also register the digital certificate provided by the WSG.

You may obtain a digital certificate from any reputable certificate authority, including:

- VeriSign—<http://www.verisign.com/>
- Entrust.net—<http://entrust.net/>
- thawte—<http://www.thawte.com/>

For more information about digital certificates and how SSL works, refer to the following website: <http://java.sun.com/j2se/1.5.0/docs/guide/security/jsse/JSSERefGuide.html>

### Task 3: Configuring SSL Properties

To run Java code that makes HTTPS connections, you must configure a number of SSL properties. The following table lists some of the properties that need to be set. For a full list of SSL-related properties, refer to the following website: <http://java.sun.com/j2se/1.5.0/docs/guide/security/jsse/JSSERefGuide.html>

#### Required SSL Properties for Java Client Applications

Property Name	Description	Sample Value
<code>java.protocol.handler.pkgs</code>	Class for handling URLs that specify the HTTPS protocol	<code>com.sun.net.ssl.internal.www.protocol</code>
<code>security.provider.n</code>	Provider implementing one or more engine classes for specific cryptographic algorithms.  Note that this property must be configured in the <i>java.security</i> file located in the directory, <code>&lt;java_home&gt;/lib/security</code> .	<code>com.sun.net.ssl.internal.ssl.Provider</code>
<code>javax.net.ssl.trustStore</code>	Location of the trust store file that contains the web server's certificate.  Note that if this property is not set, Java will attempt to find the trust store file in a set of predefined locations.	<code>C:/wsg/core/testing/config/cacerts</code>

## Encoding XML Documents

The LocationLogic WSG supports a wide range of XML document character encodings. You can find online descriptions of common international character encodings at the following websites:

- <http://www.iana.org/assignments/character-sets>
- <http://www.w3.org/TR/2006/REC-xml-20060816/#charencoding>

The WSG looks for encoding information in the XML document's header. If the information is in the header, the WSG uses it to read (parse) the document.

If the document's header did not contain the encoding information, the WSG checks the HTTP headers (transport headers). If the HTTP headers contain encoding information, the WSG uses it accordingly. Otherwise, the WSG interprets the XML document as a UTF-8 encoded document.

**Note** String literals must be specially encoded for HTTP before they are sent to the WSG. Use standard HTML character entity references: "&lt;" represents the < sign; "&gt;" represents the > sign; "&amp;" represents the & sign; and "&quot;" represents the " mark. Use two single quotes in a row to signify an escape sequence from the normal interpretation of the single quote character. Encode the string 'BEN & JERRY'S ICE CREAM' as 'BEN &amp; JERRY' 'S ICE CREAM', for example.

## Request Validation

The following table describes the XML request validation sequence performed by the Web Services Gateway (WSG):

### XML Request Validation Process

Step	Description	Refer To
1	The WSG deserializes the XML request.	"Deserialization" on page 13
2	The WSG validates the XML document against the LocationLogic XML Web Services schema (defined in <i>XL52.xsd</i> , <i>gml4xls2.xsd</i> , <i>clientProv.xsd</i> , <i>locProv.xsd</i> , and <i>llaux.xsd</i> ).	"Schema Validation" on page 13
3	The WSG checks for unsupported elements and attributes.	"Unsupported Elements and Attributes Check" on page 13
4	The WSG validates the XML content.	"Content Validation" on page 15
5	The WSG validates the data against the LocationLogic platform data requirements.	"LocationLogic Platform Code Data Validation" on page 16

## Deserialization

During the deserialization process, the WSG converts the XML request into the LocationLogic native object representation. If a geoservice XML request document is found to be syntactically invalid, the WSG returns a response (via the `errorCode` attribute of the `Error` element) indicating that an `Unknown` error has occurred, and halts request processing. In the case of provisioning and privacy requests, the response comes via `resultID`.

## Schema Validation

The LocationLogic XML Web Services schema enforce valid input, specifying which elements and attributes are valid, which tags can appear inside other tags, how many occurrences of a tag can appear, whether the tag may be omitted, and so on.

When the WSG parses an XML document it receives, it performs validation against the schema. If any errors are found, the WSG stops processing the message, and returns an `Unknown` error to the client application. If no errors are found, then the WSG proceeds to the next step in the XML request validation process.

## Unsupported Elements and Attributes Check

There may be cases where an XML document is sent to the WSG that contains elements that pass the content validation phase, but that the WSG does not support.

Therefore, the WSG checks the XML document for unsupported elements and attributes, performing additional checks beyond simple schema and content validation.

**Note** The WSG does ignore some elements or attributes that do not affect the response returned; no error is returned in this case.

The WSG performs the following checks, in sequence:

- 1 The WSG checks the main `XLS`, `clientProv`, or `locProv` element for unsupported attributes. (See the “WSG Checks for Unsupported Elements and Attributes” table for the list of specific validation checks.) If any errors are found, the WSG returns a `NotSupported` error to the client without processing any requests in the XML document being checked.
- 2 The WSG checks the `RequestHeader` or `ReqHdr` element. If any errors are found, the WSG returns a `NotSupported` error to the client without processing any requests in the XML document being checked.

- 3 The WSG checks the `Request` elements. If any unsupported elements or attributes are found that would cause an invalid response to the request, the WSG returns a `NotSupported` error to the client for that request.

**Note** Whether or not any unsupported elements or attributes are found in this step, responses that were generated in previous request processing will be retained, and the WSG will continue to process new requests.

- 4 The WSG checks elements and attributes contained by the `Request` element as the elements are converted to their associated `LocationLogic` Java API objects. If any invalid data or the use of unsupported features is found during this processing, the WSG returns a `NotSupported` error to the client for that request.

**Note** Whether or not any errors occur during this step, responses that were generated in previous request processing are retained, and the WSG continues to process new requests.

The following table describes the checks for unsupported elements and attributes that the WSG performs:

**WSG Checks for Unsupported Elements and Attributes**

Element	Child	WSG Check
Address	<code>freeFormAddress</code>	Verifies that this element is not used; use <code>streetAddress</code> or <code>streetIntersection</code> instead
ReverseGeocode Preference		Verifies that the <code>StreetAddress</code> enum is used; other enums are unsupported
POIProperties	<code>directoryType</code>	Verifies that the value contains a valid <code>LocationLogic</code> feature category path
POILocation, Nearest	<code>nearest Criterion</code>	Verifies that neither the <code>Proximity</code> nor <code>Easiest</code> enum values are used

For more information about which elements and attributes the WSG ignores, refer to the `LocationLogic` XML Web Services schema annotations in the schema files themselves (*XLS2.xsd* and *gml4xls2.xsd*, *clientProv.xsd*, *locProv.xsd*, and *llaux.xsd*).



## Content Validation

Although the WSG uses schema validation, there are some cases where the schema cannot be used to specify constraints on specific elements. The WSG validates the XML document content before it begins processing the request, validating up to, and including, the `Request` element, as described in the following table:

### XLS Content Validation Checks

Element	Child	WSG Validation
XLS	lang	Verifies that the value is a valid language format (such as <code>en</code> or <code>en-US</code> )
RequestHeader	clientName	Verifies the following: <ul style="list-style-type: none"><li>• The <code>clientName</code> is non-null</li><li>• If both the application name and user ID tokens are sent, there must be a single delimiter, <code>"\"</code>, separating the tokens</li></ul>
Request	methodName	Verifies that the value is the same as the child of the <code>Request</code> element

The WSG performs the content validation checks for elements included in the `Request` element as the elements are converted into their corresponding `LocationLogic` objects, as described in the following table:

### WSG Content Validation Checks, LocationLogic Object Elements

Element	Child	WSG Validation
Address		Verifies that the value provided is not a <code>freeFormAddress</code>
Point	Pos	Verifies that the value, which is a <code>string</code> , correctly represents a list of <code>double</code> values

If the XML document fails the content validation, the WSG does one of the following, depending on which element failed the content validation:

- The `XLS` element or the `RequestHeader` element—The WSG stops processing the current XML document, and returns an error (as described in the “XLS Error Codes for Content Validation Failures” table) to the client application.
- The `Request` element or its children—The WSG adds an error code (as described in the “XLS Error Codes for Content Validation Failures” table) to the `ErrorList` element of the corresponding `Response` element, and continues processing subsequent requests.

The following table describes the errors that the WSG reports for content validation failures:

#### XLS Error Codes for Content Validation Failures

Error Code	Validation Failure Description
<code>ValueNotRecognized</code>	Although the XML request document is valid with respect to the LocationLogic XML Web Services schema, it contains an inappropriate value.
<code>Inconsistent</code>	The XML request document contains semantic errors, such as lack of a request header, or no <code>Request</code> element child name match for the <code>Request</code> element’s <code>methodName</code> attribute. Also used for content validation errors.

If the XML document passes the content validation, then the WSG proceeds to the next step in the XML request validation process.

## LocationLogic Platform Code Data Validation

The WSG does not duplicate any portion of the LocationLogic platform data validation. Instead, the WSG just passes through the corresponding data.

If the LocationLogic platform encounters any invalid values, it reports them back to the WSG, which in turn sends a `NotSupported` error to the client application.

# Working with Geographical Data

The LocationLogic XML Web Services schema (as defined in *gml4xls2.xsd*) contains elements defined in GML (Geographic Markup Language). When developing LBS client applications that use LocationLogic XML Web Services, it is helpful to understand the following concepts:

- “Spatial Reference Systems (SRS)” on page 17
- “Coordinates” on page 17
- “Geometry” on page 18
- “Using Precision Elements” on page 19

## Spatial Reference Systems (SRS)

A *spatial reference system (SRS)* is a coordinate system that is used to locate places on the Earth's surface. The SRS establishes a point of origin, orientation of reference axes, and geometric meaning of measurements. Every geometry has an associated SRS that it references to establish coordinates. The WSG assumes that all coordinates are provided in the WGS84 (World Geodetic System 1984) coordinate system, which describes positions on the Earth as degrees of latitude and longitude.

If you specify the `requestedSrsName` attribute in a Gateway Service request, the WSG (and LocationLogic) ignores the value, assumes that all supplied values are in the WGS84 spatial reference system (SRS), and returns results as WGS84-based values.

## Coordinates

A coordinate is an ordered set of elements that takes numeric values in each position, or *ordinate*. In LocationLogic, a coordinate is a two-dimensional (latitude, longitude or *x, y*) pair. Coordinates may represent positions on the Earth's surface with respect to an SRS.

In the *latitude-longitude* coordinate system (defined by the WGS84 SRS), the first ordinate is longitude and the second ordinate is latitude. *Latitude* measures how far north or south of the equator a place is located. The equator is situated at 0°, the North Pole at 90° (a positive latitude implies north), and the South Pole at -90°. Latitude measurements range from -90° to +90°.

*Longitude* measures how far east or west of the prime meridian a place is located. The prime meridian runs through Greenwich, England. Longitude is measured in terms of east (a positive number) or west (a negative number). Longitude measurements range from -180° to +180°.

In LocationLogic XML Web Services requests and responses, the coordinate system order is latitude, longitude for all services.

## Geometry

A *geometry model* uses primitive geometric figures to represent real-world and abstract places on the Earth's surface. For example, a point can represent a hotel, a polygon can represent a park, and a rectangle can represent a route's bounding box. Each geometry must have an associated spatial reference system (SRS) that defines the coordinate system.

The following geometry types are commonly used in location-based services:

- Linestring
- Point
- Polygon
- Rectangle

A *linestring* represents a segmented linear path defined by a list of coordinates that are connected by straight line segments. Two or more coordinates define a linestring. A closed path has coincident first and last coordinates. A linestring geometry must specify the SRS in which its coordinates are measured.

A *point* is defined by a single coordinate and is the fundamental tool of geometric representation (points can be strung together to form lines, lines can be grouped to make polygons, and so on).

A *polygon* is a planar surface defined by one exterior boundary and zero or more interior boundaries (each interior boundary defines a hole in the polygon). Boundaries are closed paths that cannot cross themselves or other paths defining the polygon. They are specified with a list of coordinates connected by straight line segments. To be closed, a boundary's first and last coordinates must be coincident.

A *rectangle* is a four-sided polygon whose opposite sides are parallel and form four right angles. In LocationLogic, rectangles are defined as extents, so they must be aligned with their SRS and cannot be rotated. Rectangles are usually used to represent bounding boxes, city blocks, and other (roughly) square or rectangular features.

## Using Precision Elements

*Precision elements* (such as `Circle`, `CircularArcArea`, and `Polygon`) help describe the accuracy of a mobile device's geographic location, as returned from a device locator. A device is often located in relationship to a *cell sector*—a region covered by one or more cell towers.

A typical use of precision elements to describe location accuracy would be to specify:

- The center of a cell sector
- The distance (or radius) of the user from the center of that sector (defined as a range)
- The direction (or angle) from the center of the sector (defined as a range)



# About the XML Web Services

# 2

This chapter introduces you to the LocationLogic XML Web Services: Provisioning services, Location Gateway services, Geoservices, and Auxiliary services.

## In this chapter

- Provisioning services
- Location Gateway services
- Geoservices
- Auxiliary services
- Service and deployment test URLs

# Provisioning Services

The Web Services Gateway (WSG) offers services for both *client provisioning* and *location provisioning*. Client provisioning services control which applications can access LocationLogic, and which services a client application is authorized to use. Location provisioning services manage subscriber privacy profiles, which control the level of access that clients and requestors have to the subscribers' locations.

## Client Provisioning

Before a client application can actually access a service, a LocationLogic client account must be provisioned. LocationLogic XML Web Services provides provisioning services to create accounts for client applications.

Once an account has been created, the following information is maintained by LocationLogic:

- **Client name**—Designation determined by the application developer
- **Client ID**—Determined by LocationLogic or specified by the application developer
- **Password**—Determined by the application developer
- **Description**—Determined by the application developer
- **Type**—Determined by the application developer
- **Properties**—Determined by the application developer
- **Roles**—Assigned to client applications by the LocationLogic administrator according to the XML Web services they are allowed to use. When a WSG role is assigned, the dependent LBSAPI roles are granted as well. Roles are available for services as follows:

WSG Service Roles (1 of 2)

Service	WSG Role	Dependent LBSAPI Roles
Alerts	WSG_ALERTS	ALERTS
Content Management	WSG_OPENLS_DIRECTORY, WSG_DIRECTORYUPDATE	CONTENT_MANAGEMENT, GEOCODING, ROUTING
Client Provisioning	WSG_CLIENT_PROVISIONING	CLIENT_PROVISIONING



## WSG Service Roles (2 of 2)

Service	WSG Role	Dependent LBSAPI Roles
Geocoding	WSG_OPENLS_GEOCODING, WSG_OPENLS_REVERSEGEOCODING	GEOCODING
Location	WSG_OPENLS_GATEWAY, WSG_MLP_LOCATION	LOCATION
Location Privacy Management	WSG_LOCATION_PROVISIONING, WSG_LOCATION_PROVISIONING_ MSCLIENT_SELF	LOCATION_PRIVACY_ MANAGEMENT, CLIENT_PROVISIONING
Mapping	WSG_OPENLS_PORTRAYMAP, WSG_OPENLS_COORDINATECONVERSION	MAPPING
Routing	WSG_OPENLS_DETERMINEROUTE	ROUTING, MAPPING, GEOCODING

## Location Privacy Provisioning

Location Privacy Provisioning provides the requisite processes for establishing and maintaining privacy for application clients and requestors. Mobile subscribers are provisioned through updates to the Privacy Profile Registry. Subscriber privacy profiles contain default, client specific, and requestor specific rules for allowing others to obtain information about the subscriber's location. Default profiles apply when the subscriber's privacy profile does not have an entry for a specific client application. Location Privacy Provisioning includes the following functions for mobile subscriber privacy profiles:

- Add, update, and remove default privacy profiles
- Set a mobile subscriber's privacy settings for a specific client application
- Perform client application authentication and logging via the `MSID` and `MSID_TYPE` attributes.
- Authorize client applications to locate a mobile subscriber

The following table describes the subscriber privacy setting levels.

#### Subscriber Privacy Levels

Level	Description
Not authorized	Client application is never authorized to determine the subscriber's location.
Authorized	Client application is always authorized to determine the subscriber's location.
Authorized with notification	Client application is always authorized to determine the subscriber's location, but must notify the subscriber.
Authorized with notification and ask (default response is yes)	Client application must notify the subscriber and ask for authorization to determine the subscriber's location. If the subscriber does not respond, the default answer of 'yes' gives the client authorization.
Authorized with notification and ask (default response is no)	Client application must notify the subscriber and ask for authorization to determine the subscriber's location. If the subscriber does not respond, the default answer of 'no' denies the client authorization.

# Location Gateway Services

Autodesk LocationLogic XML Web Services uses an MLP-based (Mobile Location Protocol) API to provide privacy checking of requests for the mobile devices' location.

## Mobile Devices

A *mobile device* is any cell phone or other non-stationary device that can be located. Every mobile device has a device ID (MSID) that uniquely identifies the device to the carrier. For cell phones, this is typically the phone number of the mobile device, often including the international prefix. However, the format of the device identifier is dependent on the location determination technology (LDT) provider, so other values may be used.

## Locations

A *location* is a place on the Earth's surface. Location can be determined for POIs (points of interest) as well as for mobile devices.

## Locating Mobile Devices

Working with Location Privacy Provisioning, the Location Gateway Service allows authorized client applications to determine:

- The location of one or more mobile devices
- A list of probable locations for a single mobile device
- The location of a single mobile device

The carrier cannot guarantee the exact location of a mobile device, and therefore associates a *precision* with the returned location. This precision describes the area that the LDT provider can confidently say contains the device.

# Geoservices

Autodesk LocationLogic XML Web Services support the following geoservices:

- “Location Utility (Geocoding/Reverse Geocoding) Service” on page 26
- “Directory Service” on page 26
- “Presentation Service” on page 27
- “Route Service” on page 30
- “Alert Service” on page 31

## Location Utility (Geocoding/Reverse Geocoding) Service

*Forward geocoding*, often simply called *geocoding*, standardizes an address and translates it to a point. For example, the address “3835 Scott St, San Francisco, California, 94123, US” geocodes as (37.80450, -122.44205) in the latitude-longitude coordinate system. You can geocode a full address, a free-form address string, or a street intersection.

*Reverse geocoding* translates a point to an address. For example, the previous location, (37.80450, -122.44205), might reverse geocode as “[3000-3999] Scott St, San Francisco, California, 94123, US”. With reverse geocoding, you can display addresses to users instead of displaying coordinates. In standard reverse geocoding, a range of street numbers is returned. When using interpolation, the reverse geocoding service returns a specific street or civic address.

## Directory Service

The LocationLogic XML Web Services Directory service uses the LocationLogic platform’s extensive database of *features*, such as points of interest (POIs), businesses, and facilities. Features are stored in a hierarchical *feature category* structure.

## Feature Categories

Applications access features by navigating through a hierarchy of feature categories, such as Restaurants or Banks. The content hierarchy that defines the feature categories must be set up by your LocationLogic administrator before you can access the feature category table.

The table names and top-level feature category path are specified in the LocationLogic configuration properties. Feature category paths are specified by a string of dot-separated fields, such as `POIs.Restaurants.Chinese`.

For information about the installed feature category hierarchy, contact your LocationLogic administrator.

## Features and Feature Properties

Features, such as points of interest (POIs), have properties associated with them, including:

- Name
- Location
- Phone number
- Address
- Specific properties (for example, food type for restaurants)

Features that belong to the same feature type will have the same set of properties.

## Presentation Service

The Presentation Service provides mapping functionality, allowing you to retrieve either a map image as raw data or as a URL referencing a map image.

You can retrieve maps of different sizes and formats, choose types of information for inclusion with a map, and select the way in which to display it.

Server locations, generated URL locations, and other map format options are preset in the LocationLogic configuration properties. Contact your LocationLogic administrator to modify these settings.

## Map Characteristics

The LocationLogic XML Presentation Service provides a number of options for defining maps. The following characteristics are described in this section.

- **Name**—A logical map name.
- **Context**—The area of the earth being mapped and method of display.
- **Size and Format**—Height and width of the map in pixels displayed in a standard MIME format type.
- **Overlays**—POIs, positions, and routes superimposed on a map.
- **Hotspots**—Bounding box information returned for POI and position overlays on a map.

### Name

The LocationLogic server can provide multiple map configurations for a given geographical area. Each map configuration is assigned a logical map name and represents a particular set of map properties, such as map server location, default overlay stylization, and default pixels per inch. Logical maps can be configured on a global or per-client basis. Autodesk provides the following maps:

- NAVTEQ Europe
- NAVTEQ North America
- Tele Atlas Europe
- Tele Atlas North America

Use the `Basemap` element to specify a logical map name in your XML requests. If you do not provide a map name, the default map name specified in the LocationLogic configuration properties will be used. Contact your LocationLogic administrator for a list of valid map names.

### Context

The area being mapped can be portrayed in any of the following ways:

- Within a bounding box, with the `BBoxContext` output type.
- From a center point, using radius and azimuth with the `CenterContext` output type.
- From a center point, using display scale, DPI, and azimuth, with the `CenterContext` output type.

**Tip** Using azimuth allows you to specify degrees of map rotation.

In addition, you can obtain georegistration information about the returned map, such as the positions (lat/lon coordinates) of the four corners of the map, as well as its display scale.

### Size and Format

The height and width of a map must be 16 to 2048 pixels. Set the width and height to correspond to a client device.

You can choose any of the following raster formats:

- 1-bit BMP—Black-and-white bitmap
- 1-bit WBMP—Black-and-white wireless bitmap
- 8-bit GIF—Graphics Interchange Format (256 colors)
- 2-bit PNG—Grayscale Portable Network Graphics
- 8-bit PNG—Color Portable Network Graphics (256 colors)
- 24-bit PNG—Color Portable Network Graphics (16,777,216 colors)
- 2-bit JPEG—Grayscale Joint Photographic Experts Group
- 8-bit JPEG—Color Joint Photographic Experts Group (256 colors)
- 24-bit JPEG—Color Joint Photographic Experts Group (16,777,216 colors)

### Overlays

For all map formats requested using a `PortrayMapRequest`, you can specify POIs, positions, and routes that are overlaid on the map as symbols or polylines. You can specify which symbols and linestyles are used.

Route maps are provided by the LocationLogic XML Web Services Route service. When your application uses `DetermineRouteRequest` to calculate a route, or requests a route handle for a cached route, you can request that a map be returned along with it. The route will be highlighted on the map, and will contain symbols for the start and end points, as well as for any waypoints.

**Note** To modify symbol and linestyle settings, contact your LocationLogic administrator.

**Tip** When requesting a map with the Route service, you can get a map that displays the entire route by simply omitting a context (specifying neither bounding box nor center point). In this case, the WSG will derive a bounding box from the route. (Note that unlike `DetermineRouteRequest`, `PortrayMapRequest` requires you to specify a context.)

## Hotspots

You can leverage hotspot information in the user interface of your application. For example, you can use hotspots to construct image maps for the support of mouseovers.

Hotspots can be obtained for POI and position overlays and the associated overlay labels. Hotspots do not apply to routes or other linestring geometry.

You request a hotspot by including a `HotSpotRequest` element in your map request and setting the `ID` attributes of the `Overlay` elements. The WSG response will include bounding box information for those overlays for which you provided an `ID`. This information does not change the appearance of a map.

**Tip** Because LocationLogic automatically creates IDs for symbols contained in the route maps that it generates, you can use `DetermineRouteRequest` to get hotspot information for a route map without explicitly specifying an overlay or its ID.

## Coordinate Conversion

Coordinate conversion services translate between real-world points and pixels within a map. *Point-to-pixel* conversions are useful when you want to draw something corresponding to a real-world object on top of a previously generated map. *Pixel-to-point* conversions are useful in situations where you want to respond to mouse clicks within a map image.

## Route Service

The Route service calculates routes and directions that meet specific requirements for travel from one place to another.

### Routes

A *route* is a trip calculated by connecting a start location with one or more destination locations.

Using the Route service, applications can request *complex* routes, which consist of one or more intermediate stops (*waypoints* or *viapoints*).



## Route Preferences

*Route preferences* determine how a route is traversed. Preferences include:

- Shortest (auto)
- Fastest (auto)
- Pedestrian

LocationLogic XML Web Services also offer support for travel restrictions, such as avoiding toll roads, bridges, ferries, and tunnels. In addition, applications using LocationLogic XML Web Services can avoid specific POIs, positions, addresses, or areas of interest when calculating a route.

## Street Networks

LocationLogic uses a *street network*, also called a road system or road network, to calculate travel directions or draw a map onscreen. A street network is an abstract representation of real-world road system data such as:

- Coordinates of the street center line (a *polyline*)
- Name or label of the center line (for example, “Baltimore Avenue”)
- Address ranges on either side of a road
- Topological connections (whether adjacent links are accessible)
- Navigation attributes such as speed limits, turn restrictions, and one-way streets

For more information about the data provided with LocationLogic XML Web Services, contact your LocationLogic administrator.

## Alert Service

The Alert Service enables applications to subscribe to different types of events in order to be alerted by an XML message when the subscription is triggered.

Applications access and use the subscribed event information either from the alert message (when the subscription is triggered), or from the LocationLogic platform’s database of features (where the event information is stored).

The Alert Service supports the following types of alerts:

- Traffic alerts—sent as a result of the occurrence of events along a route
- Mobile to mobile alerts—sent when one mobile device moves within a buffer zone that surrounds a second device
- Mobile to stationary alerts—sent when one device enters or leaves a buffer zone that surrounds a fixed location

## How Subscriptions Are Triggered

When creating a subscription, you supply a temporal filter to specify criteria for when and how the subscription should be triggered. When the criteria are met, the LocationLogic platform notifies the WSG, which in turn sends an alert message to the client application, which processes the data and sends one or more alerts to the end user.

A *temporal filter* represents a time-based event, which can be any of the following:

- Single instant in time (for example, 8:00AM exactly, for one day only)
- Range of time (for example, from 5:00PM to 6:30PM, for one day only)
- Recurrence (for example, 8:00AM every day, or 5:00PM to 6:30PM every Monday, Wednesday, and Friday)

## Alert Conditions

After you create a subscription, an alert is sent out at the beginning of the specified subscription period if there are any traffic events along the route. During the subscription period, as additional traffic events occur, additional alerts are sent out according to the `AlertUpdateMode` element's value:

- **Periodic**—Traffic events that occur along the route are accumulated, and the WSG periodically sends out batched alerts. The frequency is a configurable property, with a default of 10 minutes. To change this property, contact your LocationLogic administrator.
- **Continuous**—The WSG sends out alert updates as soon as new traffic events occur along the route.

## Time Zone Considerations

To correctly adjust your subscription time specifications for daylight saving time (DST), you must specify time zones by using long format time zone identifiers, such as “Africa/Lagos”. If you specify a time zone using any form of a Greenwich Mean Time (GMT) offset, such as “PST” or “GMT-8”, the DST adjustment may be incorrect depending on the desired country and/or country subdivision for which the subscription is applicable.

See Appendix C, “Table of Time Zones” on page 173, for a list of representative time zone identifiers. To get a list of all the time zones supported in Java, refer to the following website:

<http://java.sun.com/j2se/1.4.2/docs/api/java/util/TimeZone.html>

# Auxiliary Services

## Transaction Logging Service

Client applications can log transactions in the `API_CALL_LOG` table through the Auxiliary Services. The logging service can record fine-grained information by populating the Operation Subtype fields (`OPSUBTYPE1` and `OBSUBTYPE2`). Additionally, the logging service records the feature category queried in Directory requests, enabling custom billing for premium feature categories.

Contact your LocationLogic administrator for more information about transaction logging.

# Service and Deployment Test URLs

As a developer, you need two basic pieces of information to use any of the LocationLogic XML Web Services:

- The XML schema for content to send or receive. Each schema is documented in the *Autodesk LocationLogic XML Web Services Schema Reference*.
- The URL to which to send a service request. The URLs for testing deployments and invoking services are listed in the following table.

## XML Web Service URLs and Deployment Test URLs

Service	Service URL	Deployment Test URL
Location Privacy Provisioning	http://<server>:<port>/llwsg/locprov	http://<server>:<port>/llwsg/status/ test_locprov.jsp
Client Provisioning	http://<server>:<port>/llwsg/clientprov	http://<server>:<port>/llwsg/status/ test_clientprov.jsp
Geoservices (OpenLS)	http://<server>:<port>/llwsg/openls	http://<server>:<port>/llwsg/status/ test.jsp
Auxiliary Services	http://<server>:<port>/llwsg/llaux	http://<server>:<port>/llwsg/status/ test_llaux.jsp

# Making LocationLogic XML Web Services Requests

# 3

This chapter provides examples of common XML web services requests. This chapter also offers guidelines for developing your client applications. For a complete description of a LocationLogic XML Web Services schema, refer to its corresponding *Autodesk LocationLogic XML Web Services Schema Reference*.

## In this chapter

- XML Web Services Requests
- Client Provisioning Requests
- Location Privacy Provisioning Requests
- Location Gateway Requests
- Geocoding/Reverse Geocoding Requests
- Directory Requests
- Presentation Requests
- Route Requests
- Alert Requests

# XML Web Services Requests

The sample documents in this chapter show how to make common Web Services requests via LocationLogic XML Web Services.

## LocationLogic and OpenLS (XLS) Services

The XML Web Services OpenLS services are aligned with the Open Geospatial Consortium XML for Location Services (XLS) Implementation Specification. The LocationLogic Provisioning Services are not included in the OpenLS standard, and use a custom document structure. All other LocationLogic Services use the OpenLS (XLS) document structure.

The following sections provide examples of LocationLogic service requests:

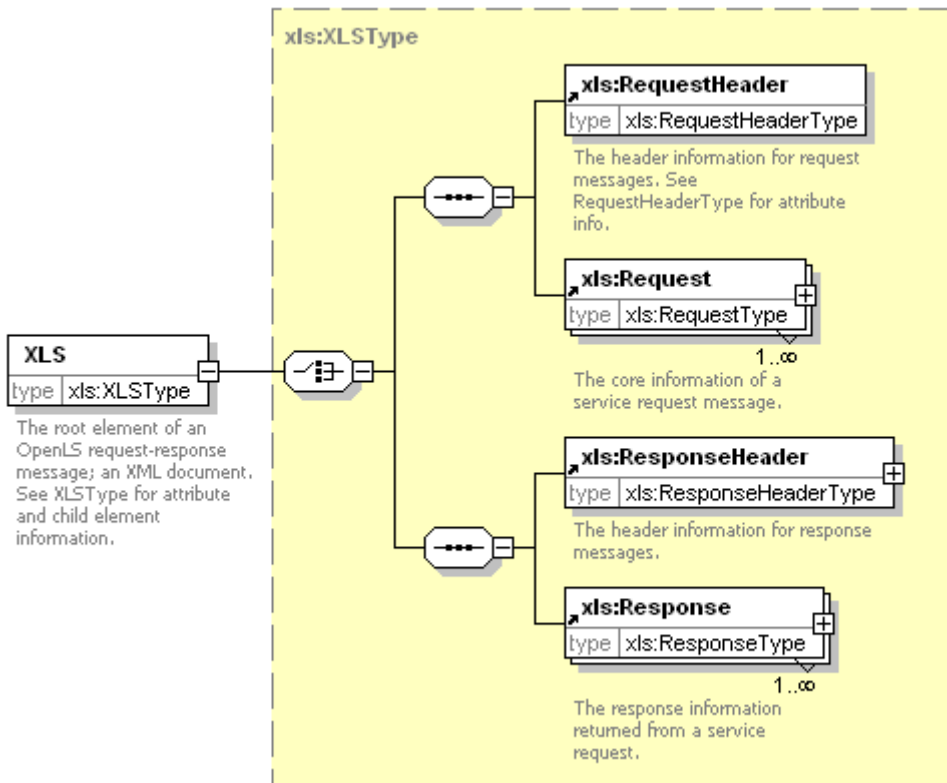
- “Client Provisioning Requests” on page 39
- “Location Privacy Provisioning Requests” on page 48
- “Location Gateway Requests” on page 55
- “Geocoding/Reverse Geocoding Requests” on page 60
- “Directory Requests” on page 71
- “Presentation Requests” on page 89
- “Route Requests” on page 127
- “Alert Requests” on page 144

## OpenLS (XLS) Document Structure

The top-level element in an OpenLS XML request or response is called the *XLS envelope*.

A valid request document consists of an XLS envelope containing one request header element and one or more request body elements. The LocationLogic Web Services Gateway response documents contain an XLS envelope with one response header and one or more response body elements.

The following diagram shows the structure of a LocationLogic OpenLS XML document.



#### LocationLogic XML Web Services support for the XLS envelope element

The `RequestHeader` element contains header information for a request message. The attribute values that you specify depend on your LocationLogic configuration and vary by system and user.

The `clientName` and `clientPassword` attributes, used for authentication, specify the name and password for the client application. (The `clientName` attribute is required for WSG requests even though the XLS schema designates it as optional.) WSG ignores the value of `deviceType`, if specified. The `sessionID` attribute is a client-defined unique session identifier, which is simply echoed by the server in the response header.

The `Request` element contains the actual body of the request. The `methodName` attribute specifies the name of the method to be invoked by the service, and the `version` attribute specifies the version level of the request parameters supported by the client. A document can contain one or more request elements and the request elements may be of differing request types. Only one request header is required in each document.



# Client Provisioning Requests

The XML Web Services Client Provisioning service establishes client application identifiers, passwords, and roles for authorized use of specific location gateway services and geoservices.

## Adding a Client

You can use the client provisioning service to add support for a new client application within LocationLogic. The `addClientReq` request creates a new client application identifier with the roles and properties that are specified in the request.

The following example shows how to create a client with the `WSG_OPENLS_DIRECTORY` role, including a single property named `Property1`.

### Add Client Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Provisions a new client, returning the resulting client ID. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <addClientReq requestID="ID1">
    <clientName>TestClient</clientName>
    <password>password</password>
    <description>Client added in example code</description>
    <type>Testing</type>
    <propertyList>
      <property name="Property1">Value of Property1</property>
    </propertyList>
    <roleList>
      <role>WSG_OPENLS_DIRECTORY</role>
    </roleList>
  </addClientReq>
</clientProv>
```

The following response includes a result (showing that the request executed as expected, without errors) and the client identification for the new client.

### Add Client Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <addClientResp requestID="ID1">
    <result resultId="0">OK</result>
    <clientId>121</clientId>
  </addClientResp>
</clientProv>
```

The next example shows an alternative way of adding a client, where the ClientId is set in the request.

### Add Client and Set ClientID Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <addClientReq requestID="ID1">
    <clientId>ID_1</clientId>
    <clientName>WSG client provisioning</clientName>
    <password>password</password>
    <description>Client added from WSH test_clientprov.jsp</description>
    <type>Testing</type>
    <propertyList>
      <property name="Property1">Value of Property1</property>
    </propertyList>
    <roleList>
      <role>WSG_OPENLS_DIRECTORY</role>
    </roleList>
  </addClientReq>
</clientProv>
```

The response in this alternative example differs from the first response in that it contains the `ClientID` set in the request.

### Add Client and Set ClientID Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <addClientResp requestID="ID1">
    <result resultId="0">OK</result>
    <clientId>ID_1</clientId>
  </addClientResp>
</clientProv>
```

## Retrieving a Client

The client provisioning service can also be used to retrieve a client. The following example shows how to retrieve the client created by either of the previous examples.

### Retrieve Client Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Returns the client. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <retrieveClientReq requestID="ID1">
    <clientId>121</clientId>
  </retrieveClientReq>
</clientProv>
```

The following response contains the same information that was used to create the client.

### Retrieve Client Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <retrieveClientResp requestID="ID1">
    <result resultId="0">OK</result>
    <client>
      <clientId>121</clientId>
      <clientName>TestClient</clientName>
      <description>Client added in example code</description>
      <type>Testing</type>
      <propertyList>
        <property name="Property1">Value of Property1</property>
      </propertyList>
      <roleList>
        <role>WSG_OPENLS_DIRECTORY</role>
      </roleList>
    </client>
  </retrieveClientResp>
</clientProv>
```

The next example shows how to retrieve all provisioned clients.

### Retrieve All Provisioned Clients Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Returns all provisioned client. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <retrieveClientReq requestID="ID1"/>
</clientProv>
```

The response contains information for every provisioned client.

### Retrieve All Provisioned Clients Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <retrieveClientResp requestID="ID1">
    <result resultId="0">OK</result>
    <client>
      <clientId>ID_1</clientId>
      <description>Client added from WSH test_clientprov.jsp</description>
      <type>Testing</type>
      <propertyList>
        <property name="Property1">Value of Property1</property>
      </propertyList>
      <roleList>
        <role>WSG_OPENLS_DIRECTORY</role>
      </roleList>
    </client>
    <client>
      <clientId>ClientGen</clientId>
      <clientName>ClientGen</clientName>
      <description>Client Administration Client</description>
      <type>LLADMIN</type>
    </client>
  </retrieveClientResp>
</clientProv>
```

The next example shows a request to retrieve all clients, filtering them by the property, "Property1", used while provisioning the client.

### Retrieve All Provisioned Clients with Property Filter Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Returns the client. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <retrieveClientReq requestID="ID1">
    <queryPropertyList comparisonType="All">
      <property name="Property1">Value of Property1</property>
    </queryPropertyList>
  </retrieveClientReq>
</clientProv>
```

The following response shows the result of the filtering.

### Retrieve All Provisioned Clients with Property Filter Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <retrieveClientResp requestID="ID1">
    <result resultId="0">OK</result>
    <client>
      <clientId>ID_1</clientId>
      <clientName>WSG client provisioning</clientName>
      <description>Client added from WSH test_clientprov.jsp</description>
      <type>Testing</type>
      <propertyList>
        <property name="Property1">Value of Property1</property>
      </propertyList>
      <roleList>
        <role>WSG_OPENLS_DIRECTORY</role>
      </roleList>
    </client>
    <client>
      <clientId>15964045</clientId>
      <clientName>WSG client provisioning</clientName>
      <description>Client added from WSH test_clientprov.jsp</description>
      <type>Testing</type>
      <propertyList>
        <property name="Property1">Value of Property1</property>
      </propertyList>
      <roleList>
        <role>WSG_OPENLS_DIRECTORY</role>
      </roleList>
    </client>
  </retrieveClientResp>
</clientProv>
```

## Updating a Client

You can use the client provisioning service to update a client. You can add client roles or change the value of any property. The following example shows how to change the value of a property for the client created in the earlier example.

### Update Client Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Updates a property. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <updateClientReq requestID="ID1">
    <clientId>121</clientId>
    <propertyList>
      <property name="Property1">Updated Value of property1</property>
    </propertyList>
  </updateClientReq>
</clientProv>
```

The following response contains a result showing that the request executed as expected, with no errors. (To confirm that the property value change actually did take place, use `retrieveClientReq`.)

### Update Client Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <updateClientResp requestID="ID1">
    <result resultId="0">OK</result>
  </updateClientResp>
</clientProv>
```

## Removing Client Properties or Roles

The following example shows how to remove a property from the client created earlier.

### Remove Client Properties or Roles Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Removes the property from the client. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <removeClientReq requestID="ID1">
    <clientId>121</clientId>
    <propertyList>
      <property name="Property1"/>
    </propertyList>
  </removeClientReq>
</clientProv>
```

The following response contains a result showing that the request executed as expected, with no errors.

### Remove Client Properties or Roles Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <removeClientResp requestID="ID1">
    <result resultId="0">OK</result>
  </removeClientResp>
</clientProv>
```



## Removing a Client

The client provisioning service can also be used to remove any provisioned client. This is done by using the remove client command, specifying only the client identification. Specifying any other properties will cause those properties to be removed from the client, but the client itself will not be removed.

### Remove Client Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Removes the client. -->
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <removeClientReq requestID="ID1">
    <clientId>121</clientId>
  </removeClientReq>
</clientProv>
```

The following response contains a result showing that the request executed as expected, with no errors.

### Remove Client Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<clientProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/clientProv">
  <respHdr ver="1.0.1" />
  <removeClientResp requestID="ID1">
    <result resultId="0">OK</result>
  </removeClientResp>
</clientProv>
```

# Location Privacy Provisioning Requests

The XML Web Services Location Privacy Provisioning service defines default and specific levels of subscriber location privacy in relation to client applications.

## Creating Privacy Profiles

You can use the location privacy provisioning services to give an existing client a privacy profile. The following example shows the provisioning for the client created in the earlier client provisioning example. Both a default profile and a client profile are created.

### Create Privacy Profiles Request (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Adds a privacy profile for an MSID. -->
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <provisionPrivacyReq requestID="ID123">
    <provisionMSIDReq>
      <msid type="MSISDN">121</msid>
      <defaultPrivacy>
        <authorizationType action="NotAuthorized"/>
        <activationPeriod>
          <Start tzname="US/Central">2003-10-19T03:30:00.000</Start>
          <End tzname="US/Central">2004-11-28T19:30:00.000</End>
          <Reoccurs>
            <ByDay>SU MO TU WE TH FR SA</ByDay>
          </Reoccurs>
        </activationPeriod>
        <requestorRequired>true</requestorRequired>
      </defaultPrivacy>
      <clientPrivacy>
        <clientId>clientname</clientId>
        <authorizationType action="AuthorizedNotifyAskYes"/>
        <activationPeriod>
          <Start tzname="US/Central">2003-10-19T03:30:00.000</Start>
          <End tzname="US/Central">2004-11-28T19:30:00.000</End>
          <Reoccurs>
            <ByDay>MO TU WE TH FR</ByDay>
          </Reoccurs>
        </activationPeriod>
      </clientPrivacy>
    </provisionMSIDReq>
  </provisionPrivacyReq>
</locProv>
```

## Create Privacy Profiles Request (2 of 2)

---

```
</activationPeriod>
<requestorRequired>true</requestorRequired>
</clientPrivacy>
</provisionMSIDReq>
</provisionPrivacyReq>
</locProv>
```

The following response contains a result showing that the request executed as expected, with no errors.

## Create Privacy Profiles Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <respHdr ver="1.0.1"/>
  <provisionPrivacyResp requestID="ID123">
    <provisionMSIDResp>
      <result resultId="0">OK</result>
    </provisionMSIDResp>
  </provisionPrivacyResp>
</locProv>
```

## Retrieving a Privacy Profile

The following request shows how to retrieve the profile that was created in the previous example.

## Retrieve Privacy Profile Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Returns the MSID's privacy profile. -->
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <provisionPrivacyReq requestID="ID123">
    <retrieveMSIDReq>
      <msid type="MSISDN">121</msid>
    </retrieveMSIDReq>
  </provisionPrivacyReq>
</locProv>
```

The following response contains all the information that was used to create the profile for the client.

## Retrieve Privacy Profile Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <respHdr ver="1.0.1">
    <provisionPrivacyResp requestID="ID123">
      <retrieveMSIDResp xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
        <result resultId="0">OK</result>
        <msid type="MSISDN">122</msid>
        <defaultPrivacy>
          <authorizationType action="NotAuthorized" />
          <activationPeriod>
            <Start tzname="US/Central">2003-10-19T03:30:00.000</Start>
            <End tzname="US/Central">2004-11-28T19:30:00.000</End>
            <Reoccurs>
              <ByDay>SU MO TU WE TH FR SA</ByDay>
              <ByHour>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23</ByHour>
              <ByMinute>0 5 10 15 20 25 30 35 40 45 50 55</ByMinute>
              <Duration>PT0S</Duration>
            </Reoccurs>
          </activationPeriod>
          <qopLimit>0</qopLimit>
          <requestorRequired>true</requestorRequired>
        </defaultPrivacy>
        <clientPrivacy>
          <clientId>clientname</clientId>
          <authorizationType action="AuthorizedNotifyAskYes" />
          <activationPeriod>
            <Start tzname="US/Central">2003-10-19T03:30:00.000</Start>
            <End tzname="US/Central">2004-11-28T19:30:00.000</End>
            <Reoccurs>
              <ByDay>MO TU WE TH FR</ByDay>
              <ByHour>0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23</ByHour>
              <ByMinute>0 5 10 15 20 25 30 35 40 45 50 55</ByMinute>
              <Duration>PT0S</Duration>
            </Reoccurs>
          </activationPeriod>
          <qopLimit>0</qopLimit>
          <requestorRequired>true</requestorRequired>
        </clientPrivacy>
      </retrieveMSIDResp>
    </provisionPrivacyResp>
  </locProv>
```

## Updating a Privacy Profile

The following request shows an update to the default privacy profile that involves a change to the `reoccurs` element.

### Update Privacy Profile Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Updates the default privacy in the MSID's privacy profile. -->
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <reqHdr ver="1.0.0">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <provisionPrivacyReq requestID="ID123">
    <updateMSIDReq>
      <msid type="MSISDN">121</msid>
      <defaultPrivacy>
        <authorizationType action="NotAuthorized"/>
        <activationPeriod>
          <Start tzname="US/Central">2003-10-19T03:30:00.000</Start>
          <End tzname="US/Central">2004-11-28T19:30:00.000</End>
          <Reoccurs>
            <ByDay>WE</ByDay>
            <ByHour>2</ByHour>
            <ByMinute>15</ByMinute>
            <Duration>PT4M</Duration>
          </Reoccurs>
        </activationPeriod>
        <requestorRequired>true</requestorRequired>
      </defaultPrivacy>
    </updateMSIDReq>
  </provisionPrivacyReq>
</locProv>
```

The following response contains a result showing that the request executed as expected, with no errors. (To ensure that the privacy profile was actually updated, use `retrieveMSIDReq`.)

### Update Privacy Profile Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <respHdr ver="1.0.1"/>
  <provisionPrivacyResp requestID="ID123">
    <updateMSIDResp>
      <result resultId="0">OK</result>
    </updateMSIDResp>
  </provisionPrivacyResp>
</locProv>
```

## Removing Part of a Privacy Profile

The following example shows how to remove the client privacy from the privacy profile.

### Remove Part of a Privacy Profile Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Removes the client privacy from the MSID's privacy profile. -->
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <provisionPrivacyReq requestID="ID123">
    <removeMSIDReq>
      <msid type="MSISDN">121</msid>
      <clientPrivacy>
        <clientId>clientname</clientId>
      </clientPrivacy>
    </removeMSIDReq>
  </provisionPrivacyReq>
</locProv>
```

The following response contains a result showing that the request executed as expected, with no errors.

### Remove Part of a Privacy Profile Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <respHdr />
  <provisionPrivacyResp requestID="ID123">
    <removeMSIDResp>
      <result resultId="0">OK</result>
    </removeMSIDResp>
  </provisionPrivacyResp>
</locProv>
```

## Removing a Privacy Profile

The following example show hows to remove a privacy profile. When removing a privacy profile, specify only the MSID. Specifying any other elements will cause those elements to be removed from the profile, but the profile itself will not be removed.

### Remove Privacy Profile Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Removes the privacy profile from the client. -->
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <reqHdr ver="1.0.1">
    <client>
      <id>clientname</id>
      <pwd>password</pwd>
    </client>
  </reqHdr>
  <provisionPrivacyReq requestID="ID123">
    <removeMSIDReq>
      <msid type="MSISDN">121</msid>
    </removeMSIDReq>
  </provisionPrivacyReq>
</locProv>
```

The following response contains a result showing that the request executed as expected, with no errors.

### Remove Privacy Profile Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<locProv xmlns="http://www.autodesk.com/lbs/wsg/schemas/locProv">
  <respHdr ver="1.0.1"/>
  <provisionPrivacyResp requestID="ID123">
    <removeMSIDResp>
      <result resultId="0">OK</result>
    </removeMSIDResp>
  </provisionPrivacyResp>
</locProv>
```



# Location Gateway Requests

The Location Gateway service enables LocationLogic client applications to retrieve the position of a known mobile device from the network.

## Location Request

The following request uses the SLIR (Standard Location Immediate Request) service; the `InputMSInformation` element's `msIDValue` attribute specifies a known, valid device identifier of a mobile subscriber.

### Location Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password">
    <Request methodName="SLIR" version="1.0.1" requestID="1">
      <SLIR>
        <InputGatewayParameters>
          <InputMSIDS>
            <InputMSInformation msIDValue="46708123456789"/>
          </InputMSIDS>
        </InputGatewayParameters>
      </SLIR>
    </Request>
  </XLS>
```

The device's location information is returned in the following response, which includes a point and a polygon that measures the point's precision.

### Location Response (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <SLIA>
      <OutputGatewayParameters locationType="CURRENT"
        ver="1.0" priority="HIGH">
        <OutputMSIDS>
          <OutputMSInformation msIDType="MSISDN"
            msIDValue="46708123456789" encryption="ASC">
            <Position>
              <gml:Point>
                <gml:pos>37.756 -122.5111</gml:pos>
```

## Location Response (2 of 2)

---

```
</gml:Point>
<gml:Polygon>
  <gml:exterior>
    <gml:pos>37.756 -122.51</gml:pos>
    <gml:pos>37.756776 -122.51032</gml:pos>
    <gml:pos>37.757097 -122.5111</gml:pos>
    <gml:pos>37.756776 -122.51188</gml:pos>
    <gml:pos>37.756 -122.5122</gml:pos>
    <gml:pos>37.755224 -122.51188</gml:pos>
    <gml:pos>37.754903 -122.5111</gml:pos>
    <gml:pos>37.755224 -122.51032</gml:pos>
    <gml:pos>37.756 -122.51</gml:pos>
  </gml:exterior>
</gml:Polygon>
</Position>
</OutputMSInformation>
</OutputMSIDS>
</OutputGatewayParameters>
</SLIA>
</Response>
</XLS>
```

## Location Request with Quality of Position

If a network provides more than one location determination technology (for example, both cell-ID-based and GPS-based LDT), the LocationLogic XML Web Services allow an application to request different qualities of position (QoP) under different circumstances. For example, if the subscriber requires a pinpointed location, such as a street address, a higher QoP should be included in the Location Gateway service request. If, however, the user needs to search within a 20-mile radius, the application can request a lower QoP.

In the following example, the RequestedQoP element specifies a QoP with a horizontal accuracy of 100 meters and a vertical (altitude) accuracy of 50 meters.

## Location Request with Quality of Position (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientName" clientPassword="password">
    <Request methodName="SLIR" version="1.0.1" requestID="1">
      <SLIR>
        <InputGatewayParameters>
          <InputMSIDS>
            <InputMSInformation msIDValue="46708123456790"/>
          </InputMSIDS>
        </InputGatewayParameters>
      </SLIR>
    </Request>
  </RequestHeader>
</XLS>
```

## Location Request with Quality of Position (2 of 2)

---

```
</InputMSIDS>
<RequestedQoP>
  <HorizontalAcc>
    <Distance value="100.0"/>
  </HorizontalAcc>
  <VerticalAcc>
    <Distance value="50.0"/>
  </VerticalAcc>
</RequestedQoP>
</InputGatewayParameters>
</SLIR>
</Request>
</XLS>
```

The following response contains the location, with the requested quality of position:

## Location Response with Quality of Position

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <SLIA>
      <OutputGatewayParameters priority="HIGH">
        <OutputMSIDS>
          <OutputMSInformation msIDValue="46708123456790" encryption="ASC">
            <Position>
              <gml:Point>
                <gml:pos>37.75 -122.35</gml:pos>
              </gml:Point>
              <gml:CircleByCenterPoint
                interpolation="circularArcCenterPointWithRadius" numArc="1">
                <gml:pos>37.75 -122.35</gml:pos>
                <gml:radius uom="M">3.14</gml:radius>
              </gml:CircleByCenterPoint>
              <Time begin="2003-12-09T23:10:27.000Z" />
              <Speed value="45.67" uom="KPH" />
              <Direction value="60" uom="DecimalDegrees" />
            </Position>
          </OutputMSInformation>
        </OutputMSIDS>
      </OutputGatewayParameters>
    </SLIA>
  </Response>
</XLS>
```

To create a map displaying a position, along with a circle representing the quality of that position, see “Displaying Positions with Precision,” on page 106.

## Enhanced Support of MLP Elements

LocationLogic XML Web Services offers enhanced support for the following MLP (Mobile Location Protocol) elements:

### LocationLogic XML Web Services Enhanced MLP Element Support (1 of 2)

MLP Element	Equivalent LocationLogic XML Web Services Element or Attribute	Description
<code>max_loc_age</code>	<code>maxLocAge</code>	Maximum allowable age, in seconds, of a position fix sent as a response to a location request. This location information may have been cached somewhere in the system from a previous location update.
<code>loc_type</code>	<code>locationType</code>	Type of location requested. Type values are used to qualify whether to return a cached location at the LDT, the current location of the subscriber, or the initial location fix attained for the subscriber. Possible values are <code>current</code> , <code>last</code> , and <code>initial</code> . In LocationLogic XML Web Services, this value is an attribute of the <code>InputGatewayParameters</code> element.
<code>prio</code>	<code>priority</code>	Priority of a location request ( <code>normal</code> , <code>high</code> ). In LocationLogic XML Web Services, this value is an attribute of the <code>InputGatewayParameters</code> element.
<code>eqop</code>	<code>RequestedQoP</code>	Desired quality of position. In LocationLogic XML Web Services, this value is a child element of the <code>InputGatewayParameters</code> element.
<code>hor_acc</code>	<code>HorizontalAcc</code>	Minimum required horizontal radius to be used for determining the requested quality of position. In LocationLogic XML Web Services, this value is an attribute of the <code>RequestedQoP</code> element.

## LocationLogic XML Web Services Enhanced MLP Element Support (2 of 2)

MLP Element	Equivalent LocationLogic XML Web Services Element or Attribute	Description
alt_acc	VerticalAcc	Minimum required vertical range used for determining the requested quality of position. In LocationLogic XML Web Services, this value is an attribute of the RequestedQoP element.
type	msIDType	Type of mobile station device ID: MSISDN, MIN, IMSI, IMEI, MDN, EMEMSID, IPV4, IPV6. In LocationLogic XML Web Services, this value is an attribute of the InputMSInformation element.
enc	encryption	Type of encryption used for position returned (ASC, B64, CRP). Default is ASC. In LocationLogic XML Web Services, this value is an attribute of the InputMSInformation element.

# Geocoding/Reverse Geocoding Requests

The Geocoding/Reverse Geocoding (Location Utility) service enables LocationLogic client applications to request and receive geocoded and reverse-geocoded locations.

**Note** The WSG supports only the WGS84 geographic coordinate system (also known as LL-84 or EPSG:4326). Coordinates are given in latitude-longitude order.

## Geocoding an Address

You can use the Geocoding service to geocode a fully-qualified address (one that contains the street address, municipality, country subdivision, and country code), a partial address, or a free-form address (an address contained in a single comma-delimited string). See the following sections for examples:

- “Geocoding a Fully-Qualified Address” on page 60
- “Geocoding a Partial Address” on page 62
- “Geocoding a Free-Form Address” on page 63
- “Geocoding a Street Intersection” on page 65
- “Geocoding Search Modes” on page 67
- “Reverse Geocoding a Coordinate” on page 68
- “Intelligent Reverse Geocoding” on page 70

## Geocoding a Fully-Qualified Address

The following request geocodes a San Francisco location given a street address, municipality, country subdivision, and country.

### Geocode Address Request (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
<RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="GeocodeRequest"
    version="1.0.1" requestID="1">
    <GeocodeRequest>
      <Address countryCode="US">
        <StreetAddress>
          <Street>2230 Francisco St</Street>
        </StreetAddress>
        <Place type="Municipality">San Francisco</Place>
        <Place type="CountrySubdivision">CA</Place>
      </Address>
```

## Geocode Address Request (2 of 2)

---

```
</GeocodeRequest>
</Request>
</XLS>
```

To geocode a list of addresses, include a sequence of separate `Address` elements into the `GeocodeRequest` element. The geocode responses will be returned in the same order used in the request.

The following response is a list that contains one exact match. The `GeocodedAddress` element includes the geocoded address (a coordinate) and expanded address information (the postal code is returned, for example, even though it was omitted from the request).

## Geocode Response Containing One Exact Match

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response version="1.0.1" requestID="1">
    <GeocodeResponse>
      <GeocodeResponseList numberOfGeocodedAddresses="1">
        <GeocodedAddress>
          <gml:Point>
            <gml:pos>37.801082 -122.44184</gml:pos>
          </gml:Point>
          <Address countryCode="us">
            <StreetAddress>
              <Street>2230 FRANCISCO ST</Street>
            </StreetAddress>
            <Place type="Municipality">SAN FRANCISCO</Place>
            <Place type="CountrySubdivision">CA</Place>
            <PostalCode>94123</PostalCode>
          </Address>
          <GeocodeMatchCode accuracy="0.4085" matchType="Exact" />
        </GeocodedAddress>
      </GeocodeResponseList>
    </GeocodeResponse>
  </Response>
</XLS>
```

## Geocoding a Partial Address

An incomplete or ambiguous address typically results in a geocode response list with multiple matches. In the preceding request, replacing “2230 Francisco St” with simply “Francisco St”, for example, generates the following response with 28 matches (indicated by the `GeocodeResponseList` element’s `numberOfAddresses` attribute).

### Geocode Response Containing Multiple Matches (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response version="1.0.1" requestID="1">
    <GeocodeResponse>
      <GeocodeResponseList numberOfGeocodedAddresses="28">
        <GeocodedAddress>
          <gml:Point>
            <gml:pos>37.803955 -122.41918</gml:pos>
          </gml:Point>
          <Address countryCode="US">
            <StreetAddress>
              <Street>[800-899] FRANCISCO ST</Street>
            </StreetAddress>
            <Place type="Municipality">SAN FRANCISCO</Place>
            <Place type="CountrySubdivision">CA</Place>
            <PostalCode>94109</PostalCode>
          </Address>
          <GeocodeMatchCode accuracy="0.36650002"
            matchType="StreetNumberRange" />
        </GeocodedAddress>
        <GeocodedAddress>
          <gml:Point>
            <gml:pos>37.803624 -122.42196</gml:pos>
          </gml:Point>
          <Address countryCode="us">
            <StreetAddress>
              <Street>[1000-1030] FRANCISCO ST</Street>
            </StreetAddress>
            <Place type="Municipality">SAN FRANCISCO</Place>
            <Place type="CountrySubdivision">CA</Place>
            <PostalCode>94109</PostalCode>
          </Address>
          <GeocodeMatchCode accuracy="0.36650002"
            matchType="StreetNumberRange" />
        </GeocodedAddress>
        ...25 addresses omitted for brevity...
```



## Geocode Response Containing Multiple Matches (2 of 2)

---

```
<GeocodedAddress>
  <gml:Point>
    <gml:pos>37.804135 -122.41784</gml:pos>
  </gml:Point>
  <Address countryCode="us">
    <StreetAddress>
      <Street>[742-799] FRANCISCO ST</Street>
    </StreetAddress>
    <Place type="Municipality">SAN FRANCISCO</Place>
    <Place type="CountrySubdivision">CA</Place>
    <PostalCode>94133</PostalCode>
  </Address>
  <GeocodeMatchCode accuracy="0.36650002"
    matchType="StreetNumberRange" />
</GeocodedAddress>
</GeocodeResponseList>
</GeocodeResponse>
</Response>
</XLS>
```

## Geocoding a Free-Form Address

A free-form geocode request uses a single comma-delimited string to represent an entire address that can include both street address and place name information.

A valid free-form address string includes street address and place name information, presented in order and delimited by a comma. For example:

```
111 McInnis Parkway, San Rafael CA 94903
```

The street address portion of the string can contain some or all of the following information:

- house number
- street name
- street type
- street direction

The place name portion of the string can contain some or all of the following information:

- municipality
- country subdivision
- postal code

Elements in the string must be provided in order, and the string must contain a comma that separates the house and street information from the municipality, country subdivision, and postal code. If the string contains two or more commas, the first comma is used as the delimiter.

If the house and street information is omitted, either a city centroid or postal code centroid is returned, depending on the value of the `geocode.engines.changecentroidorder` property. In this case, the comma delimiter is still required.

The following strings are all valid free-form addresses:

```
111 McInnis Parkway, San Rafael CA 94903
111 McInnis Parkway, San Rafael CA 94903
111 McInnis Parkway, San Rafael, CA 94903
111 McInnis Parkway, San Rafael, CA 94903
111 McInnis Parkway, San Rafael, CA 94903
111 McInnis Parkway, San Rafael CA
McInnis Parkway, San Rafael CA
111 McInnis Parkway, San Rafael
111 McInnis Parkway, 94903
, San Rafael CA
, San Rafael
, 94903
```

To geocode a free-form address, specify the address by using the `freeFormAddress` element instead of `StreetAddress`, as shown in the following request.

### Geocode Free-Form Street Address Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
<RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="GeocodeRequest"
    version="1.0.1" requestID="1">
    <GeocodeRequest>
      <Address countryCode="US">
        <freeFormAddress>
          20000 68th Ave W, Lynnwood WA 98036
        </freeFormAddress>
      </Address>
    </GeocodeRequest>
  </Request>
</XLS>
```

A free-form address geocode response is identical in format to a standard street address response. The following response is a list that contains one exact match.

### Geocode Response Containing One Exact Match

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response version="1.0.1" requestID="1">
    <GeocodeResponse>
      <GeocodeResponseList numberOfGeocodedAddresses="1">
        <GeocodedAddress>
          <gml:Point>
            <gml:pos>47.81749 -122.32489</gml:pos>
          </gml:Point>
          <Address countryCode="US">
            <StreetAddress>
              <Street>20000 68TH AVE W</Street>
            </StreetAddress>
            <Place type="Municipality">LYNNWOOD</Place>
            <Place type="CountrySubdivision">CA</Place>
            <PostalCode>98036</PostalCode>
          </Address>
          <GeocodeMatchCode accuracy="0.4761" matchType="Exact" />
        </GeocodedAddress>
      </GeocodeResponseList>
    </GeocodeResponse>
  </Response>
</XLS>
```

## Geocoding a Street Intersection

To geocode an intersection instead of an exact street address, specify the address by using the `StreetIntersection` element instead of `StreetAddress`, as shown in the following request.

### Geocode Street Intersection Request (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="GeocodeRequest"
    version="1.0.1" requestID="1">
    <GeocodeRequest>
      <Address countryCode="US">
```

## Geocode Street Intersection Request (2 of 2)

---

```
<StreetIntersection>
  <Street>Francisco St</Street>
  <IntersectingStreet>Scott St</IntersectingStreet>
</StreetIntersection>
<Place type="Municipality">San Francisco</Place>
<Place type="CountrySubdivision">CA</Place>
</Address>
</GeocodeRequest>
</Request>
</XLS>
```

**Note** You can use `StreetAddress` to specify an intersection by separating street names by an escaped ampersand (&amp;#x26;#x26;), but `StreetIntersection` is preferred in the WSG. To write a request that is equivalent to the preceding one, for example, replace the `StreetIntersection` element with the following `StreetAddress` element.

```
<StreetAddress>
  <Street>Francisco St &#x26;#x26; Scott St</Street>
</StreetAddress>
```

The following response is similar to that of an exact street address, but note that the match type is no longer exact, but is a street-number range, and the match accuracy has dropped (refer to the `GeocodeMatchCode` element).

## Geocode Response Containing Range of Street Numbers (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <GeocodeResponse>
      <GeocodeResponseList numberOfGeocodedAddresses="1">
        <GeocodedAddress>
          <gml:Point>
            <gml:pos>37.80115 -122.44133</gml:pos>
          </gml:Point>
          <Address countryCode="us">
            <StreetAddress>
              <Street>FRANCISCO ST &#x26;#x26; SCOTT ST</Street>
            </StreetAddress>
            <Place type="Municipality">SAN FRANCISCO</Place>
            <Place type="CountrySubdivision">CA</Place>
            <PostalCode>94123</PostalCode>
          </Address>
        </GeocodedAddress>
      </GeocodeResponseList>
    </GeocodeResponse>
  </Response>
</XLS>
```

## Geocode Response Containing Range of Street Numbers (2 of 2)

---

```
<GeocodeMatchCode accuracy="0.3657"
  matchType="StreetNumberRange" />
</GeocodedAddress>
</GeocodeResponseList>
</GeocodeResponse>
</Response>
</XLS>
```

## Geocoding Search Modes

Developers can specify a geocoding search mode to control how LocationLogic matches the words of the address in a GeocodeRequest to candidate addresses in the LocationLogic database. The `searchMode` property of the GeocodePreferences element can have one of the following values:

- **Auto**—Tries exact matching. If this fails, then Token matching is used.
- **Stemming**—Matches addresses where words in the candidate address fields (base street name and/or municipality) begin with the letters in the corresponding fields of the input address. Also known as partial matching.
- **Token**—Matches addresses where most of the words in the candidate address fields (base street name and/or municipality) match the corresponding words in the base name or municipality fields of the input address.

If a GeocodeRequest does not contain a GeocodePreferences element, Auto will be used as the default `searchMode` value.

Developers might want to use Stemming for the `searchMode` value in situations where subscribers only want to type in the first few characters of an address. The following example uses Stemming for the `searchMode` value.

## Geocode Request Using Search Mode(1 of 2)

---

```
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="GeocodeRequest" version="1.0.1" requestID="1">
    <GeocodeRequest>
      <Address countryCode="US">
        <StreetAddress>
          <Street>4445 Admi</Street>
        </StreetAddress>
        <Place type="Municipality">Marina Del Rey</Place>
        <Place type="CountrySubdivision">CA</Place>
      </Address>
    </GeocodeRequest>
  </Request>
</XLS>
```

## Geocode Request Using Search Mode(2 of 2)

---

```
</Address>
<GeocodePreferences searchMode="Stemming"/>
</GeocodeRequest>
</Request>
</XLS>
```

In the following response, the request's use of `Stemming` for the `searchMode` value has caused `LocationLogic` to find the complete street address.

## Geocode Response Using Search Mode

---

```
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml">
  <ResponseHeader/>
  <Response requestID="1" version="1.0.1">
    <GeocodeResponse>
      <GeocodeResponseList numberOfGeocodedAddresses="1">
        <GeocodedAddress>
          <gml:Point>
            <gml:pos>33.98415 -118.448383</gml:pos>
          </gml:Point>
          <Address countryCode="US">
            <StreetAddress>
              <Street>4445 ADMIRALTY WAY</Street>
            </StreetAddress>
            <Place type="Municipality">MARINA DEL REY</Place>
            <Place type="CountrySubdivision">CA</Place>
            <PostalCode>90292</PostalCode>
          </Address>
          <GeocodeMatchCode matchType="Exact" accuracy="0.69075817"/>
        </GeocodedAddress>
      </GeocodeResponseList>
    </GeocodeResponse>
  </Response>
</XLS>
```

## Reverse Geocoding a Coordinate

The following request reverse geocodes a coordinate in the United States.

## Reverse Geocode Request (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml">
```

## Reverse Geocode Request (2 of 2)

---

```
<RequestHeader clientName="clientname" clientPassword="password" />
<Request methodName="ReverseGeocodeRequest"
  version="1.0.1" requestID="1">
  <ReverseGeocodeRequest>
    <Position>
      <gml:Point>
        <gml:pos>37.801082 -122.44184</gml:pos>
      </gml:Point>
    </Position>
  </ReverseGeocodeRequest>
</Request>
</XLS>
```

The reverse geocoding service returns expanded location information for a position in a list of zero or more matches, sorted by match accuracy (given by the SearchCentreDistance element). The following response returns a single match. A client application can extract the location information consistent with the context (street address or place name, for example).

## Reverse Geocode Response Containing One Match

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <ReverseGeocodeResponse>
      <ReverseGeocodedLocation>
        <gml:Point>
          <gml:pos>37.80093 -122.44301</gml:pos>
        </gml:Point>
        <Address countryCode="US">
          <StreetAddress>
            <Street>[2200-2299] FRANCISCO ST</Street>
          </StreetAddress>
          <Place type="Municipality">SAN FRANCISCO</Place>
          <Place type="CountrySubdivision">CA</Place>
          <PostalCode>94123</PostalCode>
        </Address>
        <SearchCentreDistance value="104" uom="M" />
      </ReverseGeocodedLocation>
    </ReverseGeocodeResponse>
  </Response>
</XLS>
```

## Interpolation Mode

The Reverse Geocoding service determines whether to perform interpolation or standard reverse geocoding by calculating the area of the precision polygon (if specified) that accompanies a point location.

If the area falls below a configurable threshold value (default is the area of a circle with a 20-meter radius), the point is interpolated to a street or civic address. If the area is greater than or equal to the threshold, or if no polygon is specified, standard reverse geocoding is performed.

To configure the threshold value for interpolation mode (`wsg.reversegeocodingareathreshold`), contact your LocationLogic administrator.

## Intelligent Reverse Geocoding

The following reverse geocode request uses the `Prominence` value for the `sortOrder` option to return results which are potentially more relevant.

### Intelligent Reverse Geocode Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="ReverseGeocodeRequest" maximumResponses="10"
    version="1.0.1" requestID="1">
    <ReverseGeocodeRequest>
      <Position>
        <gml:Point>
          <gml:pos>37.76143 -122.43499</gml:pos>
        </gml:Point>
        <gml:CircleByCenterPoint numArc="1">
          <gml:pos>37.76143 -122.43499</gml:pos>
          <gml:radius uom="M">4</gml:radius>
        </gml:CircleByCenterPoint>
      </Position>
      <ReverseGeocodePreference
        sortOrder="Prominence">StreetAddress</ReverseGeocodePreference>
    </ReverseGeocodeRequest>
  </Request>
</XLS>
```



# Directory Requests

The Directory service queries an online directory for a specific or nearest place, product, or service. Through a LocationLogic client application, a user enters a name, type, category, keyword, phone number, or some other common identifier that distinguishes the desired place.

The request must include a position when the user seeks the nearest place, a place within a specific area, or a place at a specific location. The position can be the mobile device's current position, as determined through the Gateway service, or a remote position determined in some other way.

The directory type (shops, restaurants, or yellow pages, for example) is specified as a LocationLogic feature category path. Given the request, the Directory service searches the appropriate online directory for places of interest (POIs) that fulfill the search criteria. The service returns a list of zero or more POIs (including location names and descriptions, depending on directory content), with locations ranked in order based on the search criteria.

For a list of feature category paths, contact your LocationLogic administrator.

## Finding the Nearest Places

The following request finds POIs in a restaurant directory closest to a specified San Francisco location. The Request element's maximumResponses attribute limits the number of POIs returned. The Nearest element's nearestCriterion attribute defines the spatial filter which selects the POI. The POIProperties element's directoryType attribute specifies the feature category path.

### Directory Request for Nearest POIs (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="DirectoryRequest"
    version="1.0.1" requestID="1" maximumResponses="15">
    <DirectoryRequest>
      <POILocation>
        <Nearest nearestCriterion="Proximity">
          <Position>
            <gml:Point>
              <gml:pos>37.801082 -122.44184</gml:pos>
            </gml:Point>
          </Position>
        </Nearest>
      </POILocation>
    </DirectoryRequest>
  </Request>
</XLS>
```

## Directory Request for Nearest POIs (2 of 2)

---

```
</Position>
</Nearest>
</POILocation>
<POIProperties directoryType="POIS.NT_SF_POI.RESTAURANT"/>
</DirectoryRequest>
</Request>
</XLS>
```

The following response includes several points of interest within the `DirectoryResponse` list, each enclosed in `POIContext` elements. Each POI includes a unique identifier, name, point geometry, address, distance, and possibly other information, such as phone number.

## Directory Response with List of Nearest POIs (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response version="1.0.1" requestID="1">
    <DirectoryResponse>
      <POIContext>
        <POI ID="17605912" POIName="BECHELLI'S RESTAURANT">
          <gml:Point>
            <gml:pos>37.800212 -122.44193</gml:pos>
          </gml:Point>
          <Address countryCode="US">
            <StreetAddress>
              <Street>2346 CHESTNUT ST</Street>
            </StreetAddress>
            <Place type="Municipality">SAN FRANCISCO</Place>
            <Place type="CountrySubdivision">CA</Place>
          </Address>
        </POI>
        <Distance value="97.0767" uom="M" />
      </POIContext>
      <POIContext>
        <POI ID="17566575" POIName="CAFE MARIMBA">
          <gml:Point>
            <gml:pos>37.800083 -122.44139</gml:pos>
          </gml:Point>
          <Address countryCode="US">
            <StreetAddress>
              <Street>2317 CHESTNUT ST</Street>
            </StreetAddress>
            <Place type="Municipality">SAN FRANCISCO</Place>
            <Place type="CountrySubdivision">CA</Place>
          </Address>
        </POI>
      </POIContext>
    </DirectoryResponse>
  </Response>
</XLS>
```

## Directory Response with List of Nearest POIs (2 of 2)

---

```
</Address>
</POI>
<Distance value="117.881" uom="M" />
</POIContext>
...12 points of interest omitted for brevity...
<POIContext>
  <POI ID="19419348" POIName="ANDIAMO RISTORANTE">
    <gml:Point>
      <gml:pos>37.79965 -122.44092</gml:pos>
    </gml:Point>
    <Address countryCode="USA">
      <StreetAddress>
        <Street>3242 SCOTT ST</Street>
      </StreetAddress>
      <Place type="Municipality">SAN FRANCISCO</Place>
      <Place type="CountrySubdivision">CA</Place>
    </Address>
  </POI>
  <Distance value="178.653" uom="M" />
</POIContext>
</DirectoryResponse>
</Response>
</XLS>
```

## Retrieving Alternate Names for POIs

POIs can have multiple alternate names. You specify which alternate name properties to retrieve in the `ReturnProperties` element's `PropertyName` child element. The following request finds the alternate names, alternate name types, and alternate name language codes for a specific POI whose ID is known.

## Directory Request for a Specific POI's Alternate Names (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DirectoryRequest" version="1.0.1" requestID="2"
maximumResponses="1">
    <DirectoryRequest>
      <POILocation>
        <Nearest>
          <POI ID="50927427">
            <gml:Point>
              <gml:pos>45.43774 12.31784</gml:pos>
            </gml:Point>
```

## Directory Request for a Specific POI's Alternate Names (2 of 2)

---

```
</POI>
</Nearest>
</POILocation>
<POIProperties directoryType="POIS.NT_IT_POI.NAMED_PLACE" />
<ReturnProperties>
  <PropertyName>NAME</PropertyName>
  <PropertyName>FID</PropertyName>
  <PropertyName>ALTERNATENAME</PropertyName>
  <PropertyName>ALTERNATENAMETYPE</PropertyName>
  <PropertyName>ALTERNATENAMELANGCODE</PropertyName>
</ReturnProperties>
</DirectoryRequest>
</Request>
</XLS>
```

The following response includes the POI's alternate names, along with their corresponding name types and language codes.

## Directory Response with List of Alternate Names (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls">
<ResponseHeader />
<Response requestID="2" version="1.0.1">
  <DirectoryResponse>
    <POIContext>
      <POI ID="50927427">
        <Property valueType="string" value="VENEZIA" name="NAME" />
        <Property valueType="long" value="50927427" name="FID" />
        <Property valueType="List" name="ALTERNATENAME">
          <PropertyList valueType="string">
            <ListElement value="VENEZIA" />
            <ListElement value="VENEDIG" />
            <ListElement value="VENEDIG" />
            <ListElement value="VENEZIA" />
            <ListElement value="VENISE" />
            <ListElement value="VENICE" />
            <ListElement value="VENETIË" />
          </PropertyList>
        </Property>
        <Property valueType="List" name="ALTERNATENAMETYPE">
          <PropertyList valueType="string">
            <ListElement value="E" />
            <ListElement value="E" />
            <ListElement value="E" />
            <ListElement value="B" />
            <ListElement value="E" />
          </PropertyList>
        </Property>
      </POI>
    </POIContext>
  </DirectoryResponse>
</Response>
</XLS>
```

## Directory Response with List of Alternate Names (2 of 2)

---

```
        <ListElement value="E" />
        <ListElement value="E" />
    </PropertyList>
</Property>
<Property valueType="List" name="ALTERNATENAMELANGCODE">
    <PropertyList valueType="string">
        <ListElement value="ES" />
        <ListElement value="SV" />
        <ListElement value="DE" />
        <ListElement value="IT" />
        <ListElement value="FR" />
        <ListElement value="EN" />
        <ListElement value="NL" />
    </PropertyList>
</Property>
</POI>
<Distance uom="M" value="2.5914" />
</POIContext>
</DirectoryResponse>
</Response>
</XLS>
```

## Accessing Extended Attributes of a POI

LocationLogic provides the ability to extend the stored set of POI information beyond the base set of attributes specified by OpenLS. To access extended attribute data, specify the name of the attribute in the `ReturnProperties` element's `PropertyName` child element. For an example of `PropertyName` child element usage, see "Retrieving Alternate Names for POIs," on page 73.

For more information about the extended attributes available in your installation of LocationLogic, contact your LocationLogic administrator.

## Finding Places Within a Circular Boundary

The following request finds POIs in a restaurant directory within a 100-meter circular boundary centered at a San Francisco location. The `WithinBoundary` and `AOI` (area of interest) elements define the search area. The `gml` elements specify the circle's center and radius. (GML also allows bounding boxes and polygons.)

### Directory Request for Places Within a Circular Boundary

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DirectoryRequest"
    version="1.0.1" requestID="1">
    <DirectoryRequest>
      <POILocation>
        <WithinBoundary>
          <AOI>
            <gml:CircleByCenterPoint numArc="1">
              <gml:pos>37.801082 -122.44184</gml:pos>
              <gml:radius uom="M">100</gml:radius>
            </gml:CircleByCenterPoint>
          </AOI>
        </WithinBoundary>
      </POILocation>
      <POIProperties directoryType="POIS.NT_SF_POI.RESTAURANT" />
    </DirectoryRequest>
  </Request>
</XLS>
```

The following response contains two POIs that satisfy the previous request, including the distance to the source location.

### Directory Response with POIs Within a Circular Boundary (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <DirectoryResponse>
      <POIContext>
        <POI ID="17605912" POIName="BECHELLI'S RESTAURANT">
          <gml:Point>
            <gml:pos>37.800212 -122.44193</gml:pos>
```

```
</gml:Point>
<Address countryCode="USA">
  <StreetAddress>
    <Street>2346 CHESTNUT ST</Street>
  </StreetAddress>
  <Place type="Municipality">SAN FRANCISCO</Place>
  <Place type="CountrySubdivision">CA</Place>
</Address>
</POI>
<Distance value="79.6973" uom="M" />
</POIContext>
<POIContext>
  <POI ID="17566575" POIName="CAFE MARIMBA">
    <gml:Point>
      <gml:pos>37.800083 -122.44139</gml:pos>
    </gml:Point>
    <Address countryCode="USA">
      <StreetAddress>
        <Street>2317 CHESTNUT ST</Street>
      </StreetAddress>
      <Place type="Municipality">SAN FRANCISCO</Place>
      <Place type="CountrySubdivision">CA</Place>
    </Address>
  </POI>
  <Distance value="99.7765" uom="M" />
</POIContext>
</DirectoryResponse>
</Response>
</XLS>
```

## Finding Places Along a Route (Within Distance of a Linestring)

With LocationLogic XML Web Services, you can create requests for POIs along a route which are the same as requests for POIs within a specified distance of a linestring. You can specify a `MaximumDistance` value, a `MinimumDistance` value, or both.

**Note** If you do not include a `MaximumDistance` element, the maximum distance will be specified by the `wsg.directory.defaultmaxdistance` property value. To modify this property's value, contact your LocationLogic administrator. The default value for this property is 1000 meters.

The following request finds police stations within a maximum distance of 4 KM (4000 meters) from a specified route in San Francisco. The `gml:pos` elements define the coordinates of the route's `LineString` geometry.

### Directory Request for Places within Distance of a LineString

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1" lang="en-US"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="DirectoryRequest" version="1.0.1" requestID="1">
    <DirectoryRequest>
      <POILocation>
        <WithinDistance>
          <gml:LineString>
            <gml:pos>37.733829 -122.44287</gml:pos>
            <gml:pos>37.733829 -122.44202</gml:pos>
            <gml:pos>37.73383 -122.44202</gml:pos>
            <gml:pos>37.73305 -122.44203</gml:pos>
          </gml:LineString>
          <MaximumDistance value="4000"/>
        </WithinDistance>
      </POILocation>
      <POIProperties directoryType="POIS.NT_SF_POI.POLICE STATION"/>
    </DirectoryRequest>
  </Request>
</XLS>
```

The following response contains four POIs that satisfy the request, including the distance to the source location.

### Directory Response with Places within Distance of a LineString (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <DirectoryResponse>
      <POIContext>
        <POI ID="17628470" POIName="SAN FRANCISCO POLICE DEPT-INGLESIDE">
          <gml:Point>
            <gml:pos>37.72496 -122.44307</gml:pos>
          </gml:Point>
          <Address countryCode="USA">
            <StreetAddress>
              <Street>2 SGT JOHN V YOUNG LN</Street>
            </StreetAddress>
```



## Directory Response with Places within Distance of a LineString (2 of 2)

---

```
        <Place type="Municipality">SAN FRANCISCO</Place>
        <Place type="CountrySubdivision">CA</Place>
      </Address>
    </POI>
    <Distance value="810.53" uom="M" />
  </POIContext>
</POIContext>
  <POI ID="17556084" POIName="SAN FRANCISCO POLICE DEPT-MISSION">
    <gml:Point>
      <gml:pos>37.762782 -122.42175</gml:pos>
    </gml:Point>
    <Address countryCode="USA">
      <StreetAddress>
        <Street>630 VALENCIA ST</Street>
      </StreetAddress>
      <Place type="Municipality">SAN FRANCISCO</Place>
      <Place type="CountrySubdivision">CA</Place>
    </Address>
  </POI>
  <Distance value="3512.03" uom="M" />
</POIContext>
  ...2 POIs omitted for brevity...
</DirectoryResponse>
</Response>
</XLS>
```

## Finding Places With SQL WHERE Queries

To specify the parameters of a directory service request, use SQL WHERE queries. The following sections show just a few of the ways you can constrain directory service requests:

- “Find POIs with Specified Name Segments” on page 80
- “Find POIs with a Specific Name” on page 80
- “Find POIs of a Specific Type” on page 81
- “Find POIs with Specific Address Criteria” on page 81

The example requests in these sections use `POIProperty` elements to find POIs in a restaurant directory within 1000 meters of a San Francisco location. Each `POIProperty` element has a `name` attribute, which specifies a physical column name or logical property name, and a `value` attribute, which specifies a SQL WHERE clause (without the WHERE keyword) to filter column values. WHERE clauses in multiple `POIProperty` elements are joined internally by AND logic. (The responses to the directory requests are omitted; see the preceding sections for example responses.)

**Note** String literals must be specially encoded for HTTP before they are sent to the WSG. Use standard HTML character entity references: "&lt;" represents the < sign; "&gt;" represents the > sign; "&amp;" represents the & sign; and "&quot;" represents the " mark. Use two single quotes in a row to signify an escape sequence from the normal interpretation of the single quote character. Encode the string 'BEN & JERRY'S ICE CREAM' as 'BEN &amp; JERRY''S ICE CREAM', for example.

## Find POIs with Specified Name Segments

The following request searches for POI names that contain the string "CAFE".

### Directory Request for POI Names Containing "CAFE"

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DirectoryRequest" version="1.0.1"
    requestID="1" maximumResponses="15">
    <DirectoryRequest>
      <POILocation>
        <WithinDistance>
          <Position>
            <gml:Point>
              <gml:pos>37.801082 -122.44184</gml:pos>
            </gml:Point>
          </Position>
          <MaximumDistance value="1000" />
        </WithinDistance>
      </POILocation>
      <POIProperties directoryType="POIS.NT_SF_POI.RESTAURANT">
        <POIProperty name="NAME" value="LIKE '%CAFE%'" />
      </POIProperties>
    </DirectoryRequest>
  </Request>
</XLS>
```

## Find POIs with a Specific Name

The following criteria search for POIs with a specific name.

```
<POIProperties directoryType="POIS.NT_SF_POI.RESTAURANT">
  <POIProperty name="NAME"
    value="'BEN &amp; JERRY''S ICE CREAM'" />
</POIProperties>
```

## Find POIs of a Specific Type

The following criteria search for Japanese restaurants (FOODTYPE=10) that have a phone number starting with 510 (PHONENUM LIKE '510%'), are not in San Francisco (CITY <> 'SAN FRANCISCO'), and are on the right side of the street (STREETSIDE='R').

```
<POIProperties directoryType="POIS.NT_SF_POI.RESTAURANT">
  <POIProperty name="FOODTYPE" value="10" />
  <POIProperty name="PHONENUM" value="LIKE '510%'" />
  <POIProperty name="CITY" value="&lt;&gt; 'SAN FRANCISCO'" />
  <POIProperty name="STREETSIDE" value=" 'R'" />
</POIProperties>
```

## Find POIs with Specific Address Criteria

The following criteria search for POIs on Market Street (ADDRESS\_STREETNAME LIKE '% MARKET ST'), in California (ADDRESS\_STATE='CA'), with feature identifiers greater than 23600000 (L\_FID>23600000). The ROWNUM criterion (ROWNUM<=3) overrides the Request method's maximumResponses attribute.

```
<POIProperties directoryType="POIS.NT_SF_POI.RESTAURANT">
  <POIProperty name="L_FID" value=">23600000" />
  <POIProperty name="ADDRESS_STREETNAME"
    value="LIKE '% MARKET ST'" />
  <POIProperty name="ADDRESS_STATE" value=" 'CA'" />
  <POIProperty name="ROWNUM" value="&lt;=3" />
</POIProperties>
```

## Fuzzy POI Name Search

Entering a POI name on a handheld device can be difficult, so a successful user experience depends on quickly finding the right POI based on a partial search string.

The LocationLogic AdskFuzzyNameSearch function will rapidly search POI names for partial matches and matches in non-consecutive words. For example, entering JOHN KEN will find “John Fitzgerald Kennedy Airport”.

**Note** The AdskFuzzyNameSearch function is not enabled in the default LocationLogic installation. Contact your LocationLogic administrator to enable and configure the AdskFuzzyNameSearch function.

The following request illustrates the use of the `AdskFuzzyNameSearch` function.

### Fuzzy POI Name Search Request

---

```
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml" version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="DirectoryRequest" version="1.0.1"
    requestID="1" maximumResponses="12">
    <DirectoryRequest>
      <POILocation>
        <Nearest>
          <Position>
            <gml:Point>
              <gml:pos>32.729454 -117.195613</gml:pos>
            </gml:Point>
          </Position>
        </Nearest>
      </POILocation>
      <POIProperties directoryType="LL.TEST.POIS.NT_NA_POI">
        <POIProperty name="AdskFuzzyNameSearch('HYATT', 0.75)" value=""/>
      </POIProperties>
      <ReturnProperties>
        <PropertyName>NAME</PropertyName>
        <PropertyName>ADDRESS</PropertyName>
        <PropertyName>CITY</PropertyName>
        <PropertyName>STATE</PropertyName>
      </ReturnProperties>
    </DirectoryRequest>
  </Request>
</XLS>
```

The following response shows the results of the `AdskFuzzyNameSearch` function.

### Fuzzy POI Name Search Response (1 of 2)

---

```
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response requestID="1" version="1.0.1">
    <DirectoryResponse>
      <POIContext>
        <POI ID="37130439">
          <Property valueType="string"
            value="MANCHESTER GRAND HYATT" name="NAME"/>
          <Property valueType="string"
            value="1 MARKET PL" name="ADDRESS"/>
        </POI>
      </POIContext>
    </DirectoryResponse>
  </Response>
</XLS>
```

## Fuzzy POI Name Search Response (2 of 2)

---

```
        <Property valueType="string" value="SAN DIEGO" name="CITY"/>
        <Property valueType="string" value="CA" name="STATE"/>
    </POI>
    <Distance uom="M" value="3278.43"/>
</POIContext>
<POIContext>
    <POI ID="34327784">
        <Property valueType="string"
            value="MANCHESTER GRAND HYATT-SAN DIEGO" name="NAME"/>
        <Property valueType="string"
            value="1 MARKET PL" name="ADDRESS"/>
        <Property valueType="string" value="SAN DIEGO" name="CITY"/>
        <Property valueType="string" value="CA" name="STATE"/>
    </POI>
    <Distance uom="M" value="3278.43"/>
</POIContext>
<POIContext>
    <POI ID="17544483">
        <Property valueType="string"
            value="HYATT REGENCY-ISLANDIA" name="NAME"/>
        <Property valueType="string"
            value="1441 QUIVIRA RD" name="ADDRESS"/>
        <Property valueType="string" value="SAN DIEGO" name="CITY"/>
        <Property valueType="string" value="CA" name="STATE"/>
    </POI>
    <Distance uom="M" value="5821.84"/>
</POIContext>
<POIContext>
    <POI ID="37130759">
        <Property valueType="string"
            value="HYATT REGENCY-LA JOLLA AT AVENTINE" name="NAME"/>
        <Property valueType="string"
            value="3777 LA JOLLA VILLAGE DR" name="ADDRESS"/>
        <Property valueType="string" value="LA JOLLA" name="CITY"/>
        <Property valueType="string" value="CA" name="STATE"/>
    </POI>
    <Distance uom="M" value="15967.1"/>
</POIContext>
</DirectoryResponse>
</Response>
</XLS>
```

## Optimizing Directory Requests

When designing directory requests for your application, it is important to optimize your queries to efficiently access the LocationLogic database. LocationLogic currently has data for more than 12 million POIs, so your queries should be designed to scan the smallest number of records possible to correctly accomplish your task.

Directory requests that scan an excessive number of records, return too many results, or require a large amount of database processing time may be automatically cancelled by the LocationLogic server.

Applications using the LocationLogic platform in a commercial environment will be subject to a certification process before each product is allowed to be launched.

Contact your LocationLogic administrator for assistance in designing efficient directory requests and for details on query certification.

## Using DirectoryUpdateRequest

The Web Services Gateway provides a mechanism to create, modify and delete POI elements in the LocationLogic database. These tasks are accomplished using the `DirectoryUpdateRequest`. The following sections describe the use of `DirectoryUpdateRequest`:

- “Creating a New POI” on page 84
- “Creating a New POI with Extended Attributes” on page 85
- “Editing POI Attributes” on page 87
- “Deleting a POI” on page 88

## Creating a New POI

You may add a new POI to the database using the `CreatePOI` child element of the `DirectoryUpdateRequest`. The following request illustrates the creation of a POI with the base set of attributes.

### Create POI Request (1 of 2)

---

```
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="XML_TEST" clientPassword="XML_TEST"/>
  <Request methodName="DirectoryUpdateRequest" version="1.0.1" requestID="1">
    <DirectoryUpdateRequest directoryType="System.UnitTestSystemFeature" >
      <CreatePOI>
        <POI POIName="PoiFeature" ID=""
```

## Create POI Request (2 of 2)

---

```
description="POI with all attributes set" phoneNumber="(415) 507-5000" >
  <gml:Point>
    <gml:pos>37.6 -122.5</gml:pos>
  </gml:Point>
  <Address countryCode="US">
    <StreetAddress>
      <Street>4329 1ST ST</Street>
    </StreetAddress>
    <Place type="Municipality">LIVERMORE</Place>
    <Place type="CountrySubdivision">CA</Place>
    <PostalCode>94551</PostalCode>
  </Address>
</POI>
</CreatePOI>
</DirectoryUpdateRequest>
</Request>
</XLS>
```

The following example shows a response to a `DirectoryUpdateRequest`.

## Create POI Response

---

```
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response requestID="1" version="1.0.1">
    <DirectoryUpdateResponse>
      <POISummary ID="61"/>
    </DirectoryUpdateResponse>
  </Response>
</XLS>
```

## Creating a New POI with Extended Attributes

You may add a new POI with extended attributes to the database using the `CreatePOI` child element of the `DirectoryUpdateRequest`. You must use the `Property` element to specify the names and values of attributes to be set. The following request illustrates the creation of a POI using `Property` elements.

## Create POI Request with Extended Attributes (1 of 3)

---

```
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml" version="1.0.1" lang="en-US">
  <RequestHeader clientName="XML_TEST" clientPassword="XML_TEST"/>
  <Request methodName="DirectoryUpdateRequest" version="1.0.1" requestID="1">
```

## Create POI Request with Extended Attributes (2 of 3)

---

```
<DirectoryUpdateRequest directoryType="System.UnitTestSystemFeature" >
  <CreatePOI>
    <POI ID="">
      <Property valueType="string" value="Name" name="NAME"/>
      <Property valueType="string"
        value="POI specified via Property elements" name="DESCRIPTION"/>
      <Property valueType="string" value="666-6666" name="PHONENUM"/>
      <Property valueType="int" value="42" name="TYPE" />
      <Property valueType="date"
        value="2003-10-19T03:30:00.000-06:00" name="INSERT_DATE"/>
      <Property valueType="Address" name="_ADDRESS">
        <Address countryCode="US">
          <StreetAddress>
            <Street>4329 1ST ST</Street>
          </StreetAddress>
          <Place type="Municipality">LIVERMORE</Place>
          <Place type="CountrySubdivision">CA</Place>
          <PostalCode>94551</PostalCode>
        </Address>
      </Property>
      <Property valueType="Point" name="GEOM">
        <gml:Point>
          <gml:pos>37.7 -122.4</gml:pos>
        </gml:Point>
      </Property>
      <Property valueType="LineString" name="GEOM1">
        <gml:LineString>
          <gml:pos>37.7955 -122.4266</gml:pos>
          <gml:pos>37.7944 -122.4346</gml:pos>
        </gml:LineString>
      </Property>
      <Property valueType="Polygon" name="GEOM2">
        <gml:Polygon srsName="TestCRS">
          <gml:exterior srsName="WGS84">
            <gml:pos>37.55 -121.6</gml:pos>
            <gml:pos>37.2 -121.3</gml:pos>
            <gml:pos>37.9 -121.3</gml:pos>
            <gml:pos>37.9 -121.9</gml:pos>
            <gml:pos>37.2 -121.9</gml:pos>
          </gml:exterior>
        </gml:Polygon>
      </Property>
      <Property valueType="blob" name="BLOB_DATA">
        <Data>
          iVBORw0KGgoAAAANSUhEUgAAABQAAAAUCAMAAAC6V+0/AAAAlBMVEX/////AADAAADgAAD/ICD/
          QED/UFD/YGD/gID/kJD/oKD/sLD/OND/oID/xLD/3NDgcAD/gAD/mDD/qFD/sGD/wID/OKD/2LD/
          6ND/7LD7+O3Q/+jZvrPw8PB/rKmXAAAAnElEQVR42m3Q2xKCMawE0G2XquAlrWiN9v9/08AIOJRM
          n85sk0zwFkjRqcIBYICK9mlGRwV3hoDM6An6IzTdIa8JSXsnZEGRvKD13CM9+r/vgW3jaBCjzHUj
```



## Create POI Request with Extended Attributes (3 of 3)

---

```
aX0HyaLQYQ9kOjSEXsuYiekZTenRGf4qP8YuNkkXnMcX1Oi2kh51smNXJ5Vnbnw4ssLgAzdWQ1jj
J7Tfr9Hu2fH8Be0qHUxulWgjAAAAAE1FTkSuQmCCAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
    </Data>
  </Property>
</POI>
</CreatePOI>
</DirectoryUpdateRequest>
</Request>
</XLS>
```

## Editing POI Attributes

You may change attributes of a POI using the `EditPOI` child element of the `DirectoryUpdateRequest`. The following request shows how to change attributes of an existing POI.

### Edit POI Request

---

```
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="XML_TEST" clientPassword="XML_TEST"/>
  <Request methodName="DirectoryUpdateRequest" version="1.0.1" requestID="1">
    <DirectoryUpdateRequest directoryType="System.UnitTestSystemFeature" >
      <EditPOI>
        <POI ID="11">
          <Property valueType="string" value="NewName" name="NAME"/>
          <Property valueType="string" value="666-6666" name="PHONENUM"/>
          <Property valueType="int" value="666" name="TYPE" />
          <Property valueType="Address" name="_ADDRESS">
            <Address countryCode="US">
              <StreetAddress>
                <Street>69 Grande Paseo</Street>
              </StreetAddress>
              <Place type="Municipality">San Rafael</Place>
              <Place type="CountrySubdivision">CA</Place>
              <PostalCode>94903</PostalCode>
            </Address>
          </Property>
        </POI>
      </EditPOI>
    </DirectoryUpdateRequest>
  </Request>
</XLS>
```

## Deleting a POI

You may delete a POI from the database using the `DeletePOI` child element of the `DirectoryUpdateRequest`. The following request illustrates the removal of a POI from the `LocationLogic` database.

### Delete POI Request

---

```
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="XML_TEST" clientPassword="XML_TEST"/>
  <Request methodName="DirectoryUpdateRequest" version="1.0.1" requestID="1">
    <DirectoryUpdateRequest directoryType="System.UnitTestSystemFeature" >
      <DeletePOI>
        <POI ID="61" />
      </DeletePOI>
    </DirectoryUpdateRequest>
  </Request>
</XLS>
```

# Presentation Requests

The Presentation service lets LocationLogic client applications request and receive maps and routing instructions for display on a variety of output devices, such as monitors and cell phones.

The `PortrayMapRequest` element contains the content of the presentation request. The `Output` element specifies how the generated map is portrayed, including its width and height (in pixels), MIME type for encoding, content type (URL or base64-encoded data), and georegistration data.

## Generating a Map URL for a Specified Area

The following request/response example generates a URL for a 640x480 PNG format map of downtown San Francisco. It includes a request for `GeoRegContext` information. The `BBoxContext` element defines the extent of a map's bounding box.

### Map URL Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
      xmlns:gml="http://www.opengis.net/gml"
      version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="PortrayMapRequest"
            version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480" format="image/png"
              bitsPerPixel="24" content="URL" geoRegContext="true">
        <BBoxContext>
          <gml:pos>37.819 -122.460</gml:pos>
          <gml:pos>37.742 -122.364</gml:pos>
        </BBoxContext>
      </Output>
    </PortrayMapRequest>
  </Request>
</XLS>
```

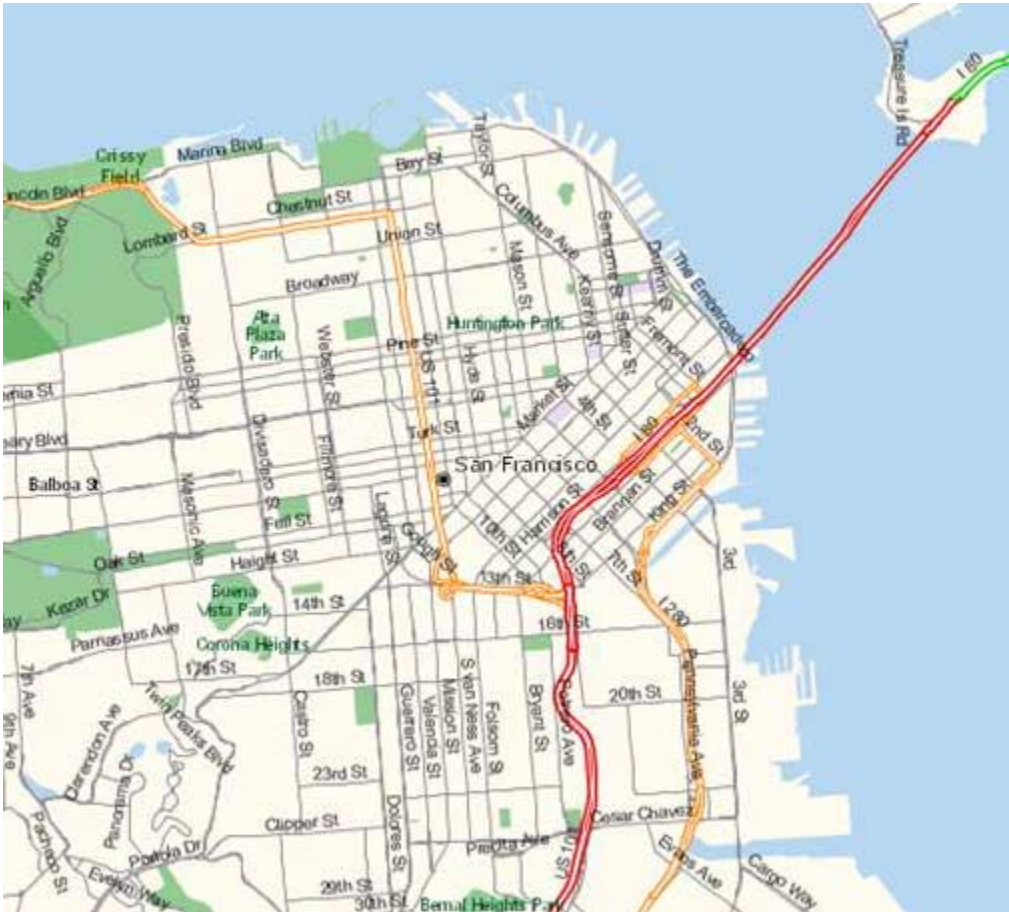
In the following response, the `PortrayMapResponse` element describes the returned map, which can be found at the indicated URL. The returned `GeoRegContext` information provides the positions of the four corners of the map, as well as its display scale. The image following the code shows the map as viewed in a browser.

## Map URL Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response version="1.0.1" requestID="1">
    <PortrayMapResponse>
      <Map>
        <Content format="image/png" width="640" height="480">
          <URL>http://l1stadmin01:8001/geomap/getMap?
            LAYERS=sfcolor.mwf&BBOX=-122.46,37.742,
            -122.364,37.819&FORMAT=PNG&WIDTH=640&HEIGHT=480&
            HLS=1&SRS=ADSK:LL84&REQUEST=map&UNITS=M&
            USER_ID=clientName&APP_SESSION_ID=null&
            LOGIN_SESSION_ID=b12d0345906faae00033d6f1f980e8cb&
            APP_NAME=TEST
          </URL>
        </Content>
        <BBoxContext>
          <gml:pos>37.819 -122.36067</gml:pos>
          <gml:pos>37.742 -122.46333</gml:pos>
        </BBoxContext>
        <GeoRegContext>
          <UpperLeft>
            <gml:pos>37.819 -122.476677</gml:pos>
          </UpperLeft>
          <UpperRight>
            <gml:pos>37.819 -122.347323</gml:pos>
          </UpperRight>
          <LowerLeft>
            <gml:pos>37.742 -122.476677</gml:pos>
          </LowerLeft>
          <LowerRight>
            <gml:pos>37.742 -122.347323</gml:pos>
          </LowerRight>
          <DisplayScale>63777.62415365521</DisplayScale>
        </GeoRegContext>
      </Map>
    </PortrayMapResponse>
  </Response>
</XLS>
```

The map appears as follows.



Map URL with bounding box

## Generating a Map URL Surrounding a Point

You also can define a map area by specifying a circle instead of a bounding box. The following request/response example generates a URL for a San Francisco map centered at the point given by the `CenterPoint` element. The example also shows how to pass an azimuth value to in order to enable map rotation. The `Radius` element sets the radius to 2000 meters; other units of measurement, such as kilometers (KM), miles (MI), yards (YD), and feet (FT), are also valid.

**Note** The `CenterContext` element's `SRS` attribute must define a spatial reference system.

### Center Point Map URL Request with Azimuth

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="PortrayMapRequest"
    version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480"
        format="image/png" bitsPerPixel="24" content="URL">
        <CenterContext SRS="WGS84" azimuth="45">
          <CenterPoint>
            <gml:pos>37.7805 -122.412</gml:pos>
          </CenterPoint>
          <Radius unit="M">2000</Radius>
        </CenterContext>
      </Output>
    </PortrayMapRequest>
  </Request>
</XLS>
```

### Center Point Map URL Response with Azimuth (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response requestID="1" version="1.0.1">
    <PortrayMapResponse>
      <Map>
        <Content height="480" width="640" format="image/png">
          <URL>
            <![CDATA[http://127.0.0.1:8001/geomap/
getMap?LAYERS=sfcolor.mwf&REQUEST=map&CENTER=-
122.412000,37.780500&RADIUS=2000.0&AZIMUTH=45.0&SRS=ADSK:LL84&WIDTH=640&HEIGHT
=480&FORMAT=PNG&PIY=72&HLS=1&CLIENTID=clientname&LOGIN_SESSION_ID=12ac00b1906
f09c500bf4ed364f74f8f]]>
          </URL>
        </Content>
        <CenterContext SRS="WGS84" azimuth="45"
          xmlns:gml="http://www.opengis.net/gml"
          xmlns="http://www.opengis.net/xls">
          <CenterPoint>
```

## Center Point Map URL Response with Azimuth (2 of 2)

```
<gml:pos>37.7805 -122.412</gml:pos>
</CenterPoint>
<Radius unit="M">2000</Radius>
</CenterContext>
</Map>
</PortrayMapResponse>
</Response>
</XLS>
```



Center point map with azimuth

## Generating a Binary Map

If the target output device does not support URLs (or supports only limited-length URLs), you can change the `Output` element's `content` attribute from `content="URL"` to `content="Data"` to embed base64-encoded binary map data in the XML response. The following response returns a binary San Francisco map.

### Binary Map Response

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader/>
  <Response version="1.0.1" requestID="1">
    <PortrayMapResponse>
      <Map>
        <Content format="image/png" width="640" height="480">
          <Data>iVBORw0KGgoAAAANSUhEUgAAoAAAAHgCAMAAACDyzWAAA
            DAFBMVEX///AwMCAGIAAAD/AAD//wAA/wAA//8AAP//AP+AAAC
            ...Binary data omitted for brevity...
            jSlzNiMCpDlJLYpJjMAQ05cWKzETstIe0fNGJZH8IzpnkpJpecD
            RZnWsJe3SwVsL8aIM/D82c0Mb20hOxQAAAABJRU5ErkJggg==</
          </Data>
        </Content>
        <BBoxContext>
          <gml:pos>37.819 -122.36067 </gml:pos>
          <gml:pos>37.742 -122.46333</gml:pos>
        </BBoxContext>
      </Map>
    </PortrayMapResponse>
  </Response>
</XLS>
```



## Specifying a Logical Map Name

Use the Basemap element to specify a logical map name in your XML requests. If you do not provide a map name, the default map name specified in the LocationLogic configuration properties will be used. Contact your LocationLogic administrator for a list of valid map names.

### Map URL Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
      xmlns:gml="http://www.opengis.net/gml"
      version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest"
            version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480"
              format="image/png" content="URL" geoRegContext="true">
        <BoundingBox>
          <gml:pos>37.819 -122.460</gml:pos>
          <gml:pos>37.742 -122.364</gml:pos>
        </BoundingBox>
      </Output>
      <Basemap>
        <Layer name="logicalMapName">
      </Basemap>
    </PortrayMapRequest>
  </Request>
</XLS>
```

## Adding a Point to a Map

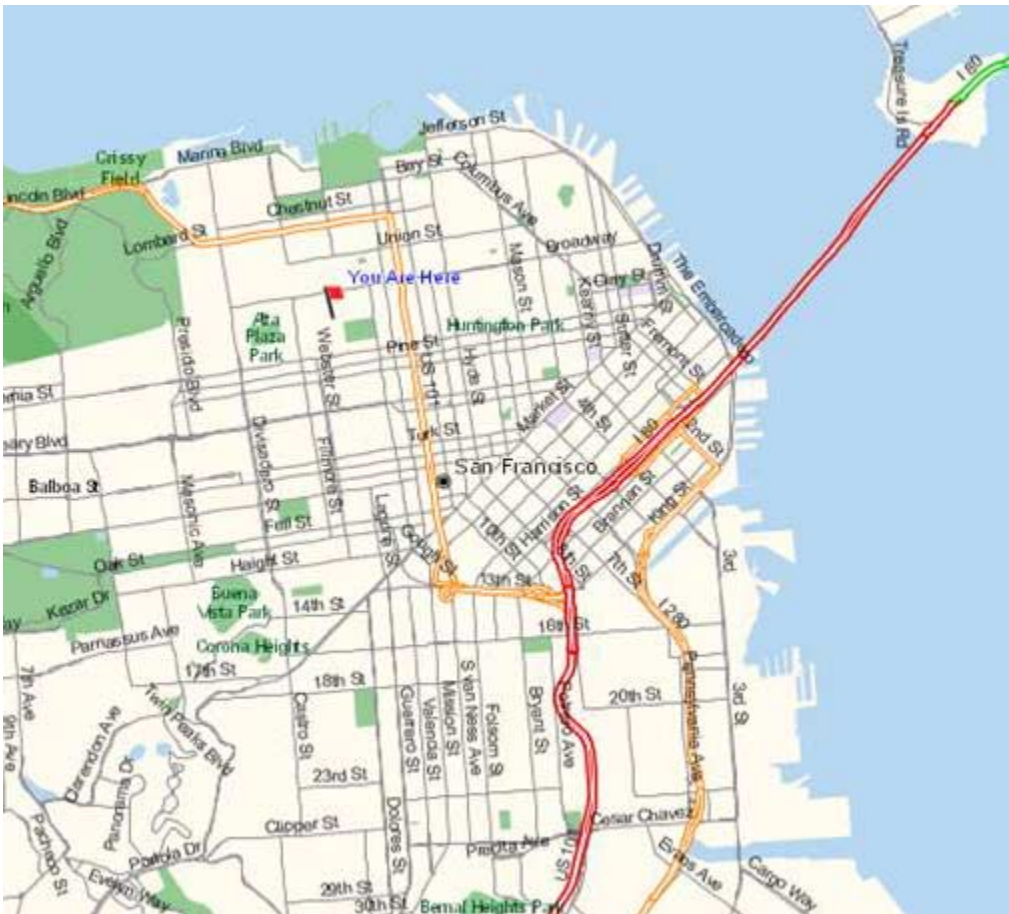
The following request uses an `Overlay` element to place a labeled point on the San Francisco map generated in the preceding example. The `Overlay` element contains a `POI` element and a `Style` element. The `POI` element specifies the position and label for the point. The `Style` element specifies a symbol for the point. (For a list of available map symbols, see “Map Symbols” on page 97.) To display more than one point, extend the `PortrayMapRequest` element to include a separate `Overlay` element for each point to be displayed.

### Map with Point Overlay Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
    xmlns:gml="http://www.opengis.net/gml"
    version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest"
    version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480"
        format="image/png" bitsPerPixel="24" content="URL">
        <BoundingBox>
          <gml:pos>37.819 -122.460</gml:pos>
          <gml:pos>37.742 -122.364</gml:pos>
        </BoundingBox>
      </Output>
      <Overlay>
        <POI ID="364789343" POIName="You Are Here">
          <gml:Point>
            <gml:pos>37.79241 -122.4305</gml:pos>
          </gml:Point>
        </POI>
        <Style>
          <Name>LL - Black_Red Flag</Name>
        </Style>
      </Overlay>
    </PortrayMapRequest>
  </Request>
</XLS>
```

The point appears on the map as follows.



Map with point overlay

## Map Symbols

LocationLogic XML Web Services provide several symbols you can use as map overlays. You specify which symbol to use by including a `Style` element with a `name` attribute in your request, as follows:












```
<Style>
  <Name>LL - Airport</Name>
</Style>
```

## Notes








- If you do not specify an overlay style name, a small box symbol will be used.
- To change the default symbol size or use symbols that are not in the default list, contact your LocationLogic administrator.

The following table describes the default set of symbols provided with LocationLogic maps:

### Symbols for LocationLogic Maps (1 of 2)

Type	Symbol	Name
Friend		LL - Red Friend LL - Black Friend LL - Green Friend LL - Blue Friend LL - Orange Friend
Golf		LL - Golf
Recreation (color)		LL - Recreation
Recreation (black-and-white)		LL - Recreation - bw
Shopping (color)		LL - Shopping
Shopping (black-and-white)		LL - shopping - bw
Historical POI (color)		LL - Historical
Historical POI (black-and-white)		LL - Historical - bw
Hospital (color)		LL - Hospital - Cross - Red
Hospital (black-and-white)		LL - Hospital - Cross
Airport		LL - Airport

## Symbols for LocationLogic Maps (2 of 2)

Type	Symbol	Name
Traffic incident		LL - Incident - bw LL - Incident - Red LL - Incident - Orange LL - Incident - Green
Start point (route)		LL - Start
End point (route)		LL - End
Location marker (you are here)		LL - Black_Red Flag
Step markers and POI markers (1-99)		LL - 1 ... LL - 99
Red flag		LL - Red Flag
Black flag		LL - Black Flag
Invisible (no symbol will be displayed)		LL - INVISIBLE

## Adding a Route to a Map

The following request uses the `Overlay` element to place a `LineString` on a San Francisco map. The `RouteGeometry` element contains the route information from “Calculating a Route Between Two Addresses” on page 127. The `Style` element describes how the map’s route layer should be portrayed.

### Route Map Overlay Request (1 of 2)

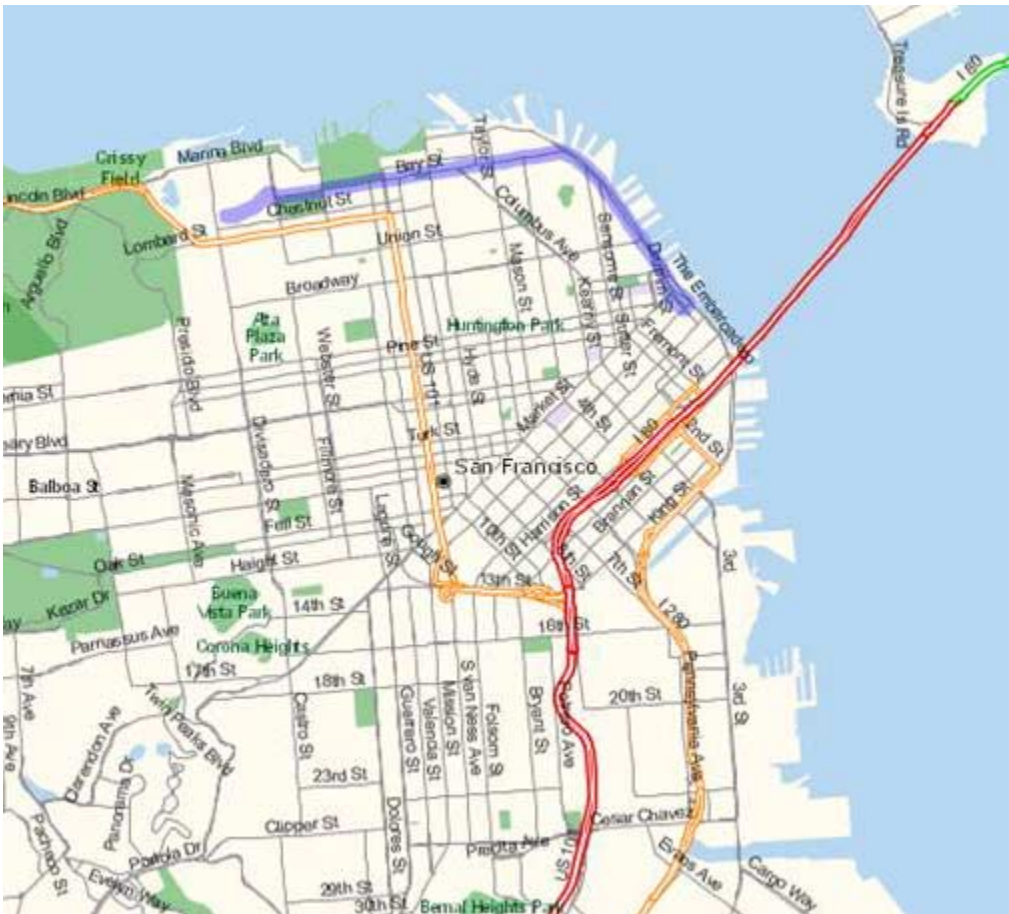
```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest"
    version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480">
```

## Route Map Overlay Request (2 of 2)

---

```
    format="image/png" bitsPerPixel="24" content="URL">
    <BoundingBox>
      <gml:pos>37.819 -122.460</gml:pos>
      <gml:pos>37.742 -122.364</gml:pos>
    </BoundingBox>
  </Output>
  <Overlay>
    <RouteGeometry>
      <gml:LineString>
        <gml:pos>37.801083 -122.44185</gml:pos>
        <gml:pos>37.801151 -122.44133</gml:pos>
        ...49 linestring points omitted for brevity...
        <gml:pos>37.79361 -122.3936</gml:pos>
        <gml:pos>37.79449 -122.39473</gml:pos>
      </gml:LineString>
    </RouteGeometry>
    <Style>
      <Name>2</Name>
    </Style>
  </Overlay>
</PortrayMapRequest>
</Request>
</XLS>
```

The route appears on the map as follows.



Route map overlay

## Adding Points Along a Route

You can combine different overlay elements to mark points along a route. The following request adds start and end symbols to the route.

### Route Map Overlay Request with Start and End Points (1 of 3)

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
      xmlns:gml="http://www.opengis.net/gml">
```

## Route Map Overlay Request with Start and End Points (2 of 3)

---

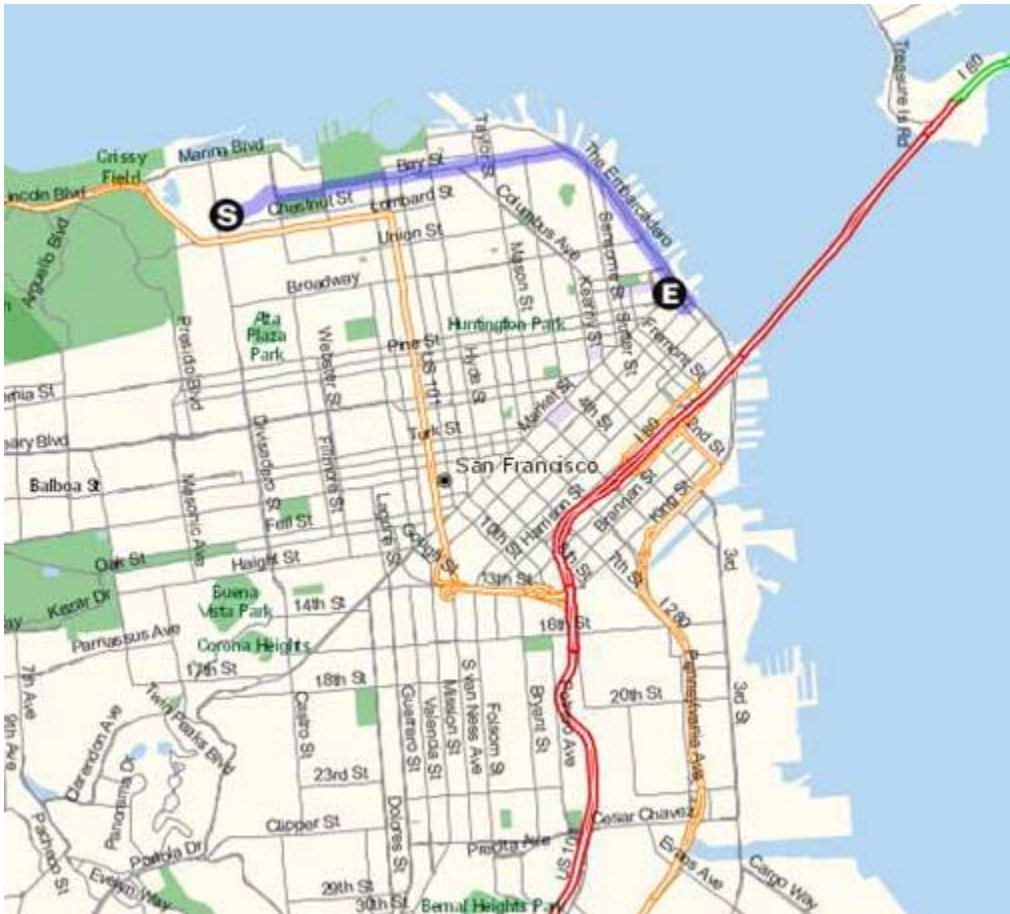
```
version="1.0.1" lang="en-US">
<RequestHeader clientName="clientname" clientPassword="password"/>
<Request methodName="PortrayMapRequest"
  version="1.0.1" requestID="1">
  <PortrayMapRequest>
    <Output width="640" height="480"
      format="image/png" bitsPerPixel="24" content="URL">
      <BoundingBox>
        <gml:pos>37.819 -122.460</gml:pos>
        <gml:pos>37.742 -122.364</gml:pos>
      </BoundingBox>
    </Output>
    <Overlay>
      <RouteGeometry>
        <gml:LineString>
          <gml:pos>37.801083 -122.44185</gml:pos>
          <gml:pos>37.801151 -122.44133</gml:pos>
          ...49 linestring points omitted for brevity...
          <gml:pos>37.79361 -122.3936</gml:pos>
          <gml:pos>37.79449 -122.39473</gml:pos>
        </gml:LineString>
      </RouteGeometry>
      <Style>
        <Name>2</Name>
      </Style>
    </Overlay>
    <Overlay>
      <POI ID="364789343">
        <gml:Point>
          <gml:pos>37.801083 -122.44185</gml:pos>
        </gml:Point>
      </POI>
      <Style>
        <Name>LL - Start</Name>
      </Style>
    </Overlay>
    <Overlay>
      <POI ID="364789343">
        <gml:Point>
          <gml:pos>37.79449 -122.39473</gml:pos>
        </gml:Point>
      </POI>
      <Style>
        <Name>LL - End</Name>
      </Style>
    </Overlay>
  </PortrayMapRequest>
</Request>
</>
```



### Route Map Overlay Request with Start and End Points (3 of 3)

```
</PortrayMapRequest>  
</Request>  
</XLS>
```

The map appears as follows.



Route map with start and end overlays

## Adding a Route to a Map Using a Route Handle

The following request uses a `RouteHandle` element in the `Overlay` element to generate a map displaying the route. For more information about creating route handles, refer to “Retrieving Previously Determined Routes with Route Handles” on page 139.

### Route Map Overlay Request Using Route Handle

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest"
    version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480"
        format="image/png" bitsPerPixel="24" content="URL">
        <BoundingBox>
          <gml:pos>37.819 -122.460</gml:pos>
          <gml:pos>37.742 -122.364</gml:pos>
        </BoundingBox>
      </Output>
      <Overlay>
        <RouteHandle routeID="65201c13-7f00-0001-00fe-8b0e79ebfaf1"
          serviceID="ADSKWSG"/>
        <Style>
          <Name>2</Name>
        </Style>
      </Overlay>
    </PortrayMapRequest>
  </Request>
</XLS>
```

## Displaying Routes with Different Line Styles

The following request uses different line styles to display alternate routes on the same map. For information about available line styles, contact your LocationLogic administrator.

### Map Request Using Different Line Styles (1 of 2)

---

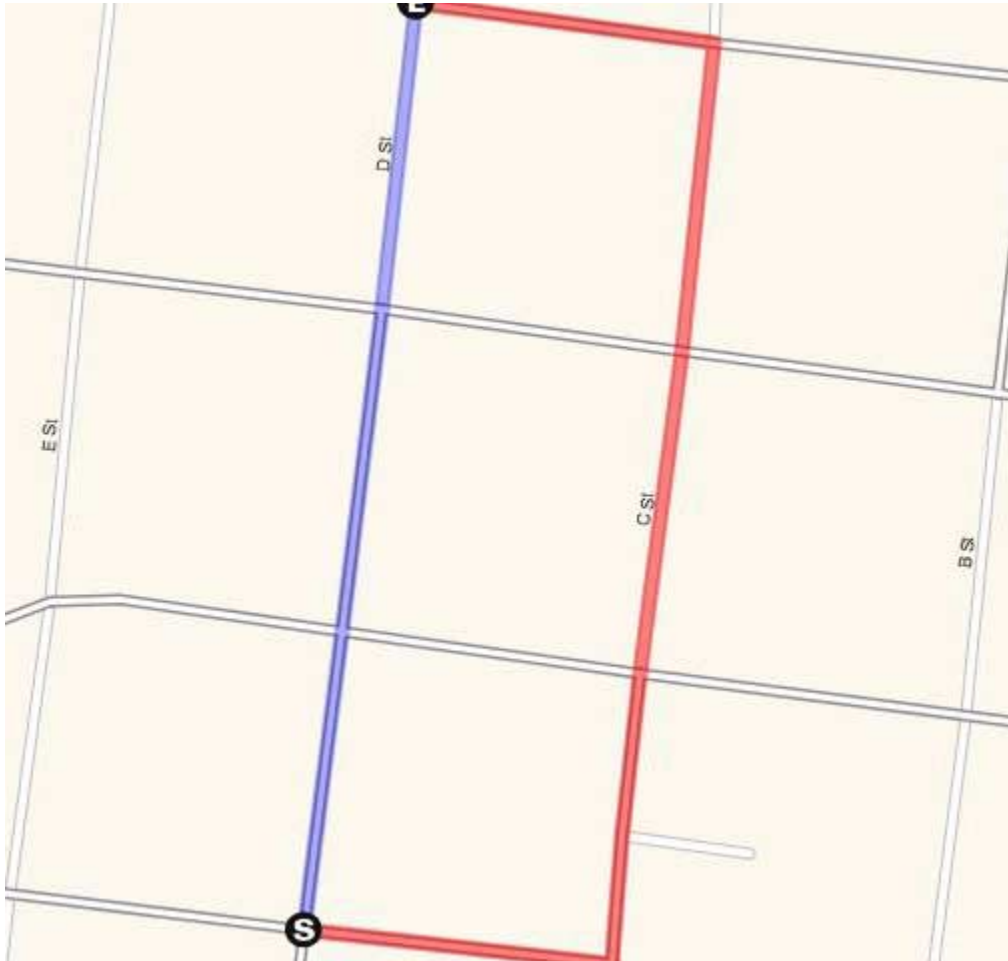
```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  version="1.0.1" lang="en-US">
```

## Map Request Using Different Line Styles (2 of 2)

---

```
<RequestHeader clientName="clientname" clientPassword="password" />
<Request methodName="PortrayMapRequest" version="1.0.1" requestID="1">
  <PortrayMapRequest>
    <Output width="640" height="480" format="image/png"
      bitsPerPixel="24" content="URL">
      <BoundingBox>
        <gml:pos>37.974552 -122.530731</gml:pos>
        <gml:pos>37.971451 -122.532402</gml:pos>
      </BoundingBox>
    </Output>
    <Overlay>
      <RouteHandle routeID="6fe6d4d8-906f-08be-00f4-d39b00003fa4"
        serviceID="ADSKWSG" />
      <Style>
        <Name>1</Name>
      </Style>
    </Overlay>
    <Overlay>
      <RouteHandle routeID="6fe6da18-906f-08be-01f2-16cb3fc14239"
        serviceID="ADSKWSG" />
      <Style>
        <Name>2</Name>
      </Style>
    </Overlay>
  </PortrayMapRequest>
</Request>
</XLS>
```

The map appears as follows.



Displaying routes with different line styles

## Displaying Positions with Precision

To display a position, along with a circle representing the quality of that position, use a `Position` element within the `Overlay` element of your `PortrayMapRequest`.

The following XML fragment contains the position and precision information returned from a location request. (For details about making location requests, see “Location Request with Quality of Position,” on page 56.)

### Location with Quality of Position

---

```
<Position>
  <gml:Point>
    <gml:pos>37.78467 -122.40719</gml:pos>
  </gml:Point>
  <gml:CircleByCenterPoint srsName="ignored" numArc="1">
    <gml:pos>37.78467 -122.40719</gml:pos>
    <gml:radius uom="M">300</gml:radius>
  </gml:CircleByCenterPoint>
</Position>
```

The following request illustrates the use of position and precision information in a PortrayMapRequest.

### Map Request with Position and Precision (1 of 2)

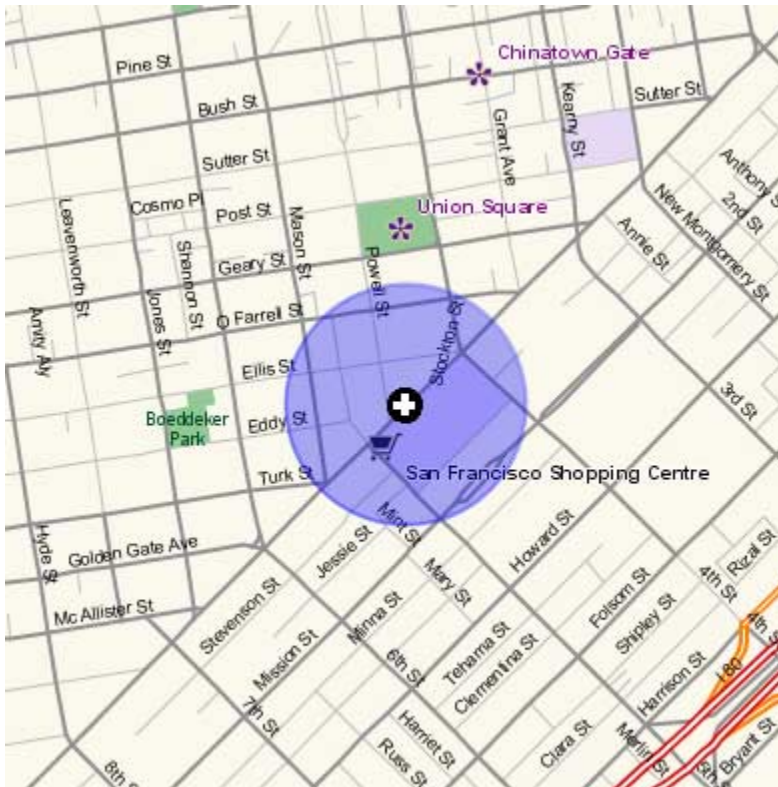
---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
lang="en" version="1.0.0">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest" version="1.0.0" requestID="1">
    <PortrayMapRequest>
      <Output width="400" height="400" format="image/png" bitsPerPixel="24"
content="URL">
        <CenterContext SRS="RequiredButIgnored">
          <CenterPoint>
            <gml:pos>37.78467 -122.40719</gml:pos>
          </CenterPoint>
          <Radius unit="M">
            1000.0
          </Radius>
        </CenterContext>
      </Output>
    <Overlay>
      <Position>
        <gml:Point>
          <gml:pos>37.78467 -122.40719</gml:pos>
        </gml:Point>
        <gml:CircleByCenterPoint srsName="ignored" numArc="1">
          <gml:pos>37.78467 -122.40719</gml:pos>
```

## Map Request with Position and Precision (2 of 2)

```
<gml:radius uom="M">300</gml:radius>
</gml:CircleByCenterPoint>
</Position>
<Style>
<StyleContent>LL - Buddy,*9*SOLID*2*OPAQUE*0.33*9*2*SOLID*0.33</StyleContent>
</Style>
</Overlay>
</PortrayMapRequest>
</Request>
</XLS>
```

The map appears as follows.



Map with position and precision

## Drawing with Position Overlays

The `Position` element of an `Overlay` in a `PortrayMapRequest` can be used to draw a variety of objects on top of your maps. More complicated drawings can be created by using multiple `Overlay` elements.

**Note** The `Position` element has been extended significantly from its original purpose in order to support advanced drawing. Developers should watch for more changes and improvements in this functionality in future versions of `LocationLogic`.

The following `PortrayMapRequest` draws a circle and several ellipses on a map.

### Drawing Request with Multiple Position Overlays (1 of 4)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" lang="en"
version="1.0.0">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest" version="1.0.0" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480" format="image/png" content="URL"
bitsPerPixel="24">
        <CenterContext SRS="RequiredButIgnored">
          <CenterPoint>
            <gml:pos>37.55 -121.6</gml:pos>
          </CenterPoint>
          <Radius unit="M">
            100000.0
          </Radius>
        </CenterContext>
      </Output>
      <Overlay>
        <Position>
          <gml:Point>
            <gml:pos>37.55 -121.6</gml:pos>
          </gml:Point>
          <gml:CircleByCenterPoint gid="54321"
srsName="http://srsURL.com/srsindex.htm#wgs80" numArc="1">
            <gml:pos>37.55 -121.6</gml:pos>
            <gml:radius uom="m">100000</gml:radius>
          </gml:CircleByCenterPoint>
        </Position>
        <Style>
          <StyleContent>LL - End, *9*SOLID*2*OPAQUE*0.33*5*2*SOLID*1.0</StyleContent>
        </Style>
      </Overlay>
    </PortrayMapRequest>
  </Request>
</XLS>
```

## Drawing Request with Multiple Position Overlays (2 of 4)

---

```
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">0</rotation>
    </Ellipse>
  </Position>
  <Style>
    <StyleContent>LL - End,WALDO</StyleContent>
  </Style>
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">15</rotation>
    </Ellipse>
  </Position>
  <Style>
    <StyleContent>LL - End</StyleContent>
  </Style>
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">30</rotation>
    </Ellipse>
  </Position>
  <Style>
    <StyleContent>LL - End,*9*SOLID*2*OPAQUE*0.33*5*2*SOLID*1.0</StyleContent>
  </Style>
</Overlay>
```



## Drawing Request with Multiple Position Overlays (3 of 4)

---

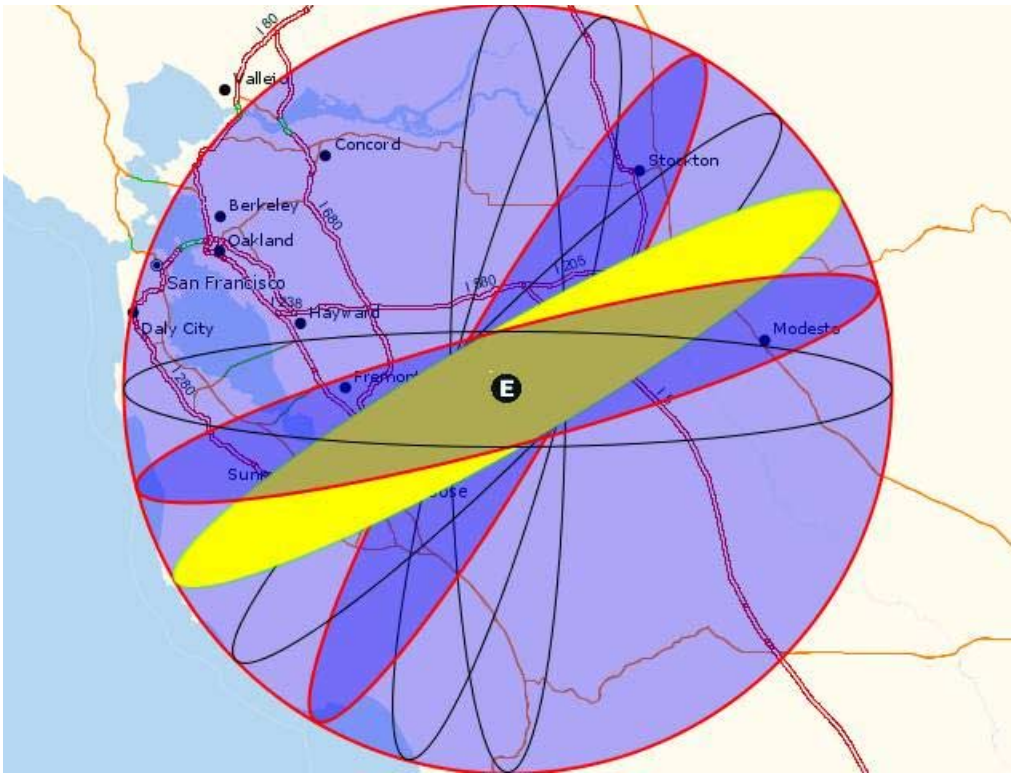
```
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">45</rotation>
    </Ellipse>
  </Position>
  <Style>
    <StyleContent>LL - End,WALDO</StyleContent>
  </Style>
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">60</rotation>
    </Ellipse>
  </Position>
  <Style>
<StyleContent>LL - End,*6*SOLID*2*OPAQUE*1.0*7*2*SOLID*0.3</StyleContent>
  </Style>
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">75</rotation>
    </Ellipse>
  </Position>
  <Style>
<StyleContent>LL - End,*9*SOLID*2*OPAQUE*0.33*5*2*SOLID*1.0</StyleContent>
  </Style>
```

## Drawing Request with Multiple Position Overlays (4 of 4)

---

```
</Overlay>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <Ellipse gid="54321" srsName="http://srsURL.com/srsindex.htm#wgs80">
      <gml:pos>37.55 -121.6</gml:pos>
      <majorAxis uom="m">100000</majorAxis>
      <minorAxis uom="m">15000</minorAxis>
      <rotation uom="m">90</rotation>
    </Ellipse>
  </Position>
  <Style>
    <StyleContent>LL - End</StyleContent>
  </Style>
</Overlay>
</PortrayMapRequest>
</Request>
</XLS>
```

The map appears as follows.



Map with multiple position overlays

## Hiding the Position Overlay Symbol

When drawing Overlays with Position elements in a `PortrayMapRequest`, you may want to hide the symbol associated with the Position element. To hide the symbol, set the symbol name in the `Style` element to `LL - INVISIBLE`. See “Map Symbols,” on page 97, for more information about symbols.

# Drawing a CircularArc

The following request demonstrates drawing with the CircularArc element.

## Position Overlay Request with a CircularArc

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" lang="en"
version="1.0.0">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="PortrayMapRequest" version="1.0.0" requestID="1">
    <PortrayMapRequest>
      <Output width="400" height="400" format="image/png" bitsPerPixel="24"
content="URL">
        <CenterContext SRS="RequiredButIgnored">
          <CenterPoint>
            <gml:pos>37.55 -121.6</gml:pos>
          </CenterPoint>
          <Radius unit="M">
            100000.0
          </Radius>
        </CenterContext>
      </Output>
      <Overlay>
        <Position>
          <gml:Point>
            <gml:pos>37.55 -121.6</gml:pos>
          </gml:Point>
          <CircularArc gid="54321"
srsName="http://srsURL.com/srsindex.htm#wgs80" numArc="1">
            <gml:pos>37.55 -121.6</gml:pos>
            <innerRadius uom="some_distance_uri">20000</innerRadius>
            <outerRadius uom="some_distance_uri">50000</outerRadius>
            <startAngle uom="some_radius_uri">-15</startAngle>
            <endAngle uom="some_radius_uri">130</endAngle>
          </CircularArc>
        </Position>
        <Style>
          <StyleContent>
            LL - INVISIBLE,*2*SOLID*2*OPAQUE*0.33*9*2*SOLID*1.0
          </StyleContent>
        </Style>
      </Overlay>
    </PortrayMapRequest>
  </Request>
</XLS>
```

The map appears as follows.



Map with a CircularArc position overlay

## Drawing Nested Polygons

The following request demonstrates drawing nested polygons using the Polygon element in a Position Overlay.

### Position Overlay Request with Nested Polygons (1 of 2)

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" lang="en"
version="1.0.0">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="PortrayMapRequest" version="1.0.0" requestID="1">
    <PortrayMapRequest>
      <Output width="400" height="400" format="image/png" bitsPerPixel="24"
content="URL">
```

## Position Overlay Request with Nested Polygons (2 of 2)

---

```
<CenterContext SRS="RequiredButIgnored">
  <CenterPoint>
    <gml:pos>37.55 -121.6</gml:pos>
  </CenterPoint>
  <Radius unit="M">
    100000.0
  </Radius>
</CenterContext>
</Output>
<Overlay>
  <Position>
    <!-- The position -->
    <gml:Point>
      <gml:pos>37.55 -121.6</gml:pos>
    </gml:Point>
    <!-- Precision polygon for the position -->
    <gml:Polygon srsName="TestCRS">
      <gml:exterior srsName="WGS84">
        <gml:pos>37.9 -121.3</gml:pos>
        <gml:pos>37.9 -121.9</gml:pos>
        <gml:pos>37.2 -121.9</gml:pos>
        <gml:pos>37.2 -121.3</gml:pos>
      </gml:exterior>
      <gml:interior srsName="WGS84">
        <gml:pos>37.8 -121.4</gml:pos>
        <gml:pos>37.8 -121.8</gml:pos>
        <gml:pos>37.3 -121.8</gml:pos>
        <gml:pos>37.3 -121.4</gml:pos>
      </gml:interior>
    </gml:Polygon>
  </Position>
  <Style>
    <StyleContent>
      LL - INVISIBLE,*9*SOLID*2*OPAQUE*0.33*5*2*SOLID*1.0
    </StyleContent>
  </Style>
</Overlay>
</PortrayMapRequest>
</Request>
</XLS>
```

The map appears as follows.



Map with a position overlay of nested polygons

## Adding Labels to Map Overlays

To add a label to a map, create an `Overlay` containing a `POI` element and set the `POIName` property to your desired label text. To construct a `PortrayMapRequest` containing both drawing elements and labels, create multiple `Overlay` elements, using `POI` elements for the labels and `Position` elements for the drawings.

## Creating a Map with a City Boundary

To create a map displaying a city boundary, first use a `DirectoryRequest` to obtain a polygon representing the boundary, and then pass the polygon as a `Position Overlay` to a `PortrayMapRequest` to render the map.

The following map shows a city boundary. A `Position Overlay` was used to draw the boundary polygon and a `POI Overlay` was used to create the label.



Map showing San Rafael city limits



The following DirectoryRequest obtains a polygon representing the city boundary.

### Request for City Boundary Polygon

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Directory request with WithinBoundary Box. Should return POIs. -->
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DirectoryRequest" version="1.0.1" requestID="1"
maximumResponses="1">
    <DirectoryRequest>
      <POILocation>
        <Nearest>
          <Address countryCode="US">
            <StreetAddress>
              <Street>111 McInnis Parkway</Street>
            </StreetAddress>
            <Place type="Municipality">San Rafael</Place>
            <Place type="CountrySubdivision">CA</Place>
          </Address>
        </Nearest>
      </POILocation>
      <POIProperties directoryType="LL.TEST.AREAS.NT_NA_POLY.CITY" />
      <ReturnProperties>
        <PropertyName>NAME</PropertyName>
        <PropertyName>FID</PropertyName>
        <PropertyName>GEOM</PropertyName>
      </ReturnProperties>
    </DirectoryRequest>
  </Request>
</XLS>
```

The following PortrayMapRequest creates the city boundary map.

### Request for City Boundary Map (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" lang="en"
version="1.0.0">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="PortrayMapRequest" version="1.0.0" requestID="1">
    <PortrayMapRequest>
      <Output width="700" height="700" format="image/png" bitsPerPixel="24"
content="URL">
        <CenterContext SRS="RequiredButIgnored">
          <CenterPoint>
```

## Request for City Boundary Map (2 of 2)

---

```
<gml:pos>37.97461 -122.53247</gml:pos>
</CenterPoint>
<Radius unit="M">
  6000.0
</Radius>
</CenterContext>
</Output>
<Overlay>
  <Position>
    <gml:Point>
      <gml:pos>37.97461 -122.53247</gml:pos>
    </gml:Point>
    <gml:Polygon>
      <gml:exterior>
        <gml:pos>37.97538 -122.48143</gml:pos>
        <gml:pos>37.9756 -122.48147</gml:pos>
        ...many linestring points omitted for brevity...
        <gml:pos>37.97511 -122.48141</gml:pos>
        <gml:pos>37.97538 -122.48143</gml:pos>
      </gml:exterior>
      <gml:interior>
        <gml:pos>37.97599 -122.50826</gml:pos>
        <gml:pos>37.9762 -122.50807</gml:pos>
        ...many linestring points omitted for brevity...
        <gml:pos>37.98973 -122.53008</gml:pos>
        <gml:pos>37.98991 -122.53017</gml:pos>
      </gml:interior>
    </gml:Polygon>
  </Position>
  <Style>
    <StyleContent>
      LL - INVISIBLE,*9*SOLID*2*OPAQUE*0.25*9*2*SOLID*0.45
    </StyleContent>
  </Style>
</Overlay>
<Overlay>
  <POI POIName="San Rafael City Limits" ID="">
    <gml:Point>
      <gml:pos>37.97461 -122.53247</gml:pos>
    </gml:Point>
  </POI>
  <Style>
    <StyleContent>LL - INVISIBLE</StyleContent>
  </Style>
</Overlay>
</PortrayMapRequest>
</Request>
</XLS>
```

## Adding an Image Map Hotspot to a Route

The Presentation service provides the ability to return clickable *hotspots* on maps. A map request can return the bounding boxes of overlaid POIs, Positions, and labels.

To include image map hotspots in a map, add the optional element, `HotSpotRequest`, to the `PortrayMapRequest` element. To receive hotspot information in the response, you must also specify an `id` attribute for the `Overlay` element in the request that is used to identify the item (POI or Position) being overlaid on the map. IDs for each `Overlay` item must be unique.

The following Portray Map request includes a `pos` `Overlay` element that indicates the location of a friend. Note that the `HotSpotRequest` element is also included.

### Image Map Hotspot Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
version="1.0.1" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="PortrayMapRequest" version="1.0.1" requestID="1">
    <PortrayMapRequest>
      <Output width="640" height="480" format="image/png" content="URL">
        <BoundingBox>
          <gml:pos>37.819 -122.460</gml:pos>
          <gml:pos>37.742 -122.364</gml:pos>
        </BoundingBox>
      </Output>
      <Overlay id="ID1">
        <Position>
          <gml:Point>
            <gml:pos>37.7805000 -122.4120000</gml:pos>
          </gml:Point>
        </Position>
        <Style>
          <Name>LL - Red Friend</Name>
        </Style>
      </Overlay>
      <HotSpotRequest />
    </PortrayMapRequest>
  </Request>
</XLS>
```

If hotspots have been requested, the Map response includes a `HotSpotList` element that contains one or more `HotSpot` elements. Each `HotSpot` element has an `id` attribute and will contain `BoundingBox` and `LabelBoundingBox` child elements. The `LabelBoundingBox` element appears in the response only if a `POI` element is overlaid and its `POIName` attribute is specified.

Hotspot coordinates are returned in *x, y* order, with the origin (0.0) at the top left corner of the image. The *x*-axis increases in value to the right, and the *y*-axis increases in value downward.

In the following response, the `PortrayMapResponse` element describes the returned map, which can be found at the indicated URL.

### Image Map Hotspot Response

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <PortrayMapResponse>
      <Map>
        <Content format="image/png" width="640" height="480">
          <URL>http://144.111.170.62:4001/geomap/
getMap?LAYERS=sfcolor.mwf&BBOX=-122.46,37.742,-
122.364,37.819&FORMAT=PNG&WIDTH=640&HEIGHT=480&HLS=1&SRS=ADSK;LL84&REQUEST=map
&UNITS=M&USER_ID=clientName&APP_SESSION_ID=null&LOGIN_SESSION_ID=5c4f68ba906fa
a3e011fe161607e4469&APP_NAME=TEST&SYMBOLS2=ID1,LL+-+Red+Friend,-
122.412,37.7805,0.007071,0.007071,,</URL>
        </Content>
        <BoundingBox>
          <gml:pos>37.819 -122.36067</gml:pos>
          <gml:pos>37.742 -122.46333</gml:pos>
        </BoundingBox>
        <HotSpotList>
          <HotSpot id="ID1">
            <BoundingBox>
              <gml:pos>325 249</gml:pos>
              <gml:pos>314 230</gml:pos>
            </BoundingBox>
          </HotSpot>
        </HotSpotList>
      </Map>
    </PortrayMapResponse>
  </Response>
</XLS>
```

## Converting from Points to Pixels

Coordinate conversion services translate between real-world points and pixels within a map. You must provide the LocationLogic server with enough context information to accurately make the conversion. The `Output` element in a `CoordinateConversionRequest` provides the context information and should match the `Output` element from the corresponding map request. The following request translates real-world points to pixels within a map.

### Point-to-pixel Coordinate Conversion Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml"
version="1.0.0" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="CoordinateConversionRequest"
version="1.0.0" requestID="1">
    <CoordinateConversionRequest>
      <PointToPixel>
        <Output width="640" height="480" format="image/gif" content="URL">
          <CenterContext SRS="RequiredButIgnored">
            <CenterPoint>
              <gml:pos>37.767921 -122.41238</gml:pos>
            </CenterPoint>
            <Radius unit="M">
              1000.0
            </Radius>
          </CenterContext>
        </Output>
        <gml:pos>37.775052 -122.424358</gml:pos>
        <gml:pos>37.76079 -122.424358</gml:pos>
        <gml:pos>37.775052 -122.400402</gml:pos>
        <gml:pos>37.76079 -122.400402</gml:pos>
      </PointToPixel>
    </CoordinateConversionRequest>
  </Request>
</XLS>
```

### Point-to-pixel Coordinate Conversion Response (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.0" xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml">
```

## Point-to-pixel Coordinate Conversion Response (2 of 2)

---

```
<ResponseHeader/>
<Response version="1.0.0" requestID="1">
  <CoordinateConversionResponse>
    <PixelToPointResult>
      <gml:pos>37.775052 -122.424358</gml:pos>
      <gml:pos>37.76079 -122.424358</gml:pos>
      <gml:pos>37.775052 -122.400402</gml:pos>
      <gml:pos>37.76079 -122.400402</gml:pos>
    </PixelToPointResult>
  </CoordinateConversionResponse>
</Response>
</XLS>
```

## Converting from Pixels to Points

The following request translates pixels within a map to real-world points.

### Pixel-to-point Coordinate Conversion Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml"
version="1.0.0" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="CoordinateConversionRequest"
version="1.0.0" requestID="1">
    <CoordinateConversionRequest>
      <PixelToPoint>
        <Output width="640" height="480" format="image/gif" content="URL">
          <CenterContext SRS="RequiredButIgnored">
            <CenterPoint>
              <gml:pos>37.767921 -122.41238</gml:pos>
            </CenterPoint>
            <Radius unit="M">
              1000.0
            </Radius>
          </CenterContext>
        </Output>
        <Pixel x="0" y="0" />
        <Pixel x="0" y="480" />
        <Pixel x="640" y="0" />
        <Pixel x="640" y="480" />
      </PixelToPoint>
    </CoordinateConversionRequest>
  </Request>
</XLS>
```

## Pixel-to-point Coordinate Conversion Response

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.0" xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml">
  <ResponseHeader/>
  <Response version="1.0.0" requestID="1">
    <CoordinateConversionResponse>
      <PixelToPointResult>
        <gml:pos>37.775052 -122.424358</gml:pos>
        <gml:pos>37.76079 -122.424358</gml:pos>
        <gml:pos>37.775052 -122.400402</gml:pos>
        <gml:pos>37.76079 -122.400402</gml:pos>
      </PixelToPointResult>
    </CoordinateConversionResponse>
  </Response>
</XLS>
```

## Using a Map URL for Coordinate Conversions

The `Output` element in a `CoordinateConversionRequest` provides context information explicitly. If you have access to the URL of a previously created map, you may implicitly specify the context information by providing the map URL instead of the `Output` element. The following request illustrates the use of a map URL for a pixel-to-point conversion. The response will be the same as if an `Output` element is used.

### Pixel-to-point Coordinate Conversion Using a Map URL

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
xmlns:gml="http://www.opengis.net/gml"
version="1.0.0" lang="en-US">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="CoordinateConversionRequest"
version="1.0.0" requestID="1">
    <CoordinateConversionRequest>
      <PixelToPoint>
        <URL><![CDATA[http://127.0.0.1:8001/geomap/
getMap?LAYERS=sfcolor.mwf&REQUEST=map&BBOX=-122.6607422,37.37542533,-
122.07949219999999,38.30191617&SRS=ADSK:LL84&WIDTH=400&HEIGHT=400&FORMAT=PNG&P
PIY=72]]></URL>
        <Pixel x="0" y="0" />
        <Pixel x="0" y="100" />
        <Pixel x="100" y="0" />
        <Pixel x="100" y="100" />
      </PixelToPoint>
    </CoordinateConversionRequest>
  </Request>
</XLS>
```



# Route Requests

The Route service enables LocationLogic client applications to request and receive routes for start and end points, subject to various travel preferences and constraints.

By default, the WSG returns travel directions in English. LocationLogic includes driving direction resource bundles that support the following languages: English, French, Italian, German, Spanish, Dutch, and Portuguese.

You can add support for additional languages by adding resource property files to the WSG. To add support for additional languages, contact your LocationLogic administrator.

## Generating Maps

You can generate maps using a route request, as well as by using a presentation request. To return a map from a route request, you must embed a `RouteMapRequest` element in the `DetermineRouteRequest`; see “Requesting Maps with Route Requests,” on page 132. For more information about creating maps with the Presentation service, see “Presentation Requests” on page 89.

## Calculating a Route Between Two Addresses

The following request retrieves a route with two waypoints—start and end points—specified by `Address` elements.

### Route Request Between Two Addresses (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  lang="en" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DetermineRouteRequest"
    version="1.0.1" requestID="1">
    <DetermineRouteRequest>
      <RoutePlan>
        <RoutePreference>Fastest</RoutePreference>
        <WayPointList>
          <StartPoint>
            <Address countryCode="us">
```

## Route Request Between Two Addresses (2 of 2)

---

```

        <StreetAddress>
            <Street>2230 Francisco St</Street>
        </StreetAddress>
        <Place type="Municipality">San Francisco</Place>
        <Place type="CountrySubdivision">CA</Place>
        <PostalCode>94123</PostalCode>
    </Address>
</StartPoint>
<EndPoint>
    <Address countryCode="us">
        <StreetAddress>
            <Street>1 Market St</Street>
        </StreetAddress>
        <Place type="Municipality">San Francisco</Place>
        <Place type="CountrySubdivision">CA</Place>
        <PostalCode>94105</PostalCode>
    </Address>
</EndPoint>
</WayPointList>
</RoutePlan>
<RouteInstructionsRequest/>
<RouteGeometryRequest/>
</DetermineRouteRequest>
</Request>
</XLS>
```

In the following response, the `RouteSummary` element specifies the route's overall characteristics, including the estimated time to travel the complete route (`TotalTime` element), the total distance covered by the route (`TotalDistance` element), and the rectangular area bounding the complete route (`BoundingBox` element).

Because the request contains a `RouteGeometryRequest` element, the response has a `RouteGeometry` element, which holds a linestring that defines the route's geometry.

The response's `RouteInstructionList` element contains turn-by-turn route directions and advisories formatted for presentation.

## Route Response Containing Instruction List (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
```

## Route Response Containing Instruction List (2 of 2)

---

```
<DetermineRouteResponse>
  <RouteSummary>
    <TotalTime>PT8M34S</TotalTime>
    <TotalDistance value="5.911" uom="KM" />
    <BoundingBox>
      <gml:pos>37.80661 -122.39258</gml:pos>
      <gml:pos>37.79318 -122.44185</gml:pos>
    </BoundingBox>
  </RouteSummary>
  <RouteGeometry>
    <gml:LineString>
      <gml:pos>37.801083 -122.44185</gml:pos>
      <gml:pos>37.801151 -122.44133</gml:pos>
      ...49 linestring points omitted for brevity...
      <gml:pos>37.79361 -122.3936</gml:pos>
      <gml:pos>37.79449 -122.39473</gml:pos>
    </gml:LineString>
  </RouteGeometry>
  <RouteInstructionsList format="text/plain" lang="en">
    <RouteInstruction>2230 Francisco St,
      San Francisco</RouteInstruction>
    <RouteInstruction>Depart (E) on FRANCISCO ST.
      (0.05 km)</RouteInstruction>
    <RouteInstruction>Turn RIGHT onto SCOTT ST.
      (0.22 km)</RouteInstruction>
    <RouteInstruction>Turn LEFT onto LOMBARD ST (US-101).
      (1.47 km)</RouteInstruction>
    <RouteInstruction>Turn LEFT onto VAN NESS AVE.
      (0.32 km)</RouteInstruction>
    <RouteInstruction>Turn RIGHT onto BAY ST.
      (1.71 km)</RouteInstruction>
    <RouteInstruction>Turn Slight RIGHT onto THE EMBARCADERO
      (EMBARCADERO PLAZA/JUSTIN HERMAN PLAZA).
      (1.87 km)</RouteInstruction>
    <RouteInstruction>Turn RIGHT onto MISSION ST.
      (0.07 km)</RouteInstruction>
    <RouteInstruction>Turn RIGHT onto STEUART ST
      (JUSTIN HERMAN PLAZA). (0.2 km)</RouteInstruction>
    <RouteInstruction>Arrive 1 Market St,
      San Francisco</RouteInstruction>
  </RouteInstructionsList>
</DetermineRouteResponse>
</Response>
</XLS>
```

## Calculating a Route Between Two Points

You also can request a route by specifying coordinates instead of street addresses. The following request substitutes the street addresses in the preceding example for their geocoded locations, and also includes these other changes:

- The XLS element's `lang` attribute specifies English ("en") as the language in which instructions are given. (ISO 639 codes are used.)
- The `RouteGeometryRequest` element is omitted, suppressing the `RouteGeometry` `linestring` in the response.
- The `RoutePlan` element's `expectedStartTime` attribute specifies when travel is expected to begin (in XML `dateTime` format). The returned route and route summary may vary by time of day, due to traffic conditions, road restrictions, and travel preferences (given by the `RoutePreference` element). In this example, the specified start time is in the GMT-08:00 (San Francisco) time zone. If `expectedStartTime` is omitted, it defaults to the current GMT date and time (regardless of where the WSG is installed). For more information about route times, see "Time Zone Support in Route Requests" on page 142.

### Route Request Between Two Points (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  lang="en"
  version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="DetermineRouteRequest"
    version="1.0.1" requestID="1">
    <DetermineRouteRequest>
      <RoutePlan expectedStartTime="2007-01-22T08:15:00-08:00">
        <RoutePreference>Fastest</RoutePreference>
        <WayPointList>
          <StartPoint>
            <Position>
              <gml:Point>
                <gml:pos>37.801082 -122.44184</gml:pos>
              </gml:Point>
            </Position>
          </StartPoint>
          <EndPoint>
            <Position>
              <gml:Point>
                <gml:pos>37.79449 -122.39473</gml:pos>
              </gml:Point>
            </Position>
          </EndPoint>
        </WayPointList>
      </RoutePlan>
    </DetermineRouteRequest>
  </Request>
</XLS>
```

## Route Request Between Two Points (2 of 2)

---

```
        </gml:Point>
      </Position>
    </EndPoint>
  </WayPointList>
</RoutePlan>
<RouteInstructionsRequest/>
</DetermineRouteRequest>
</Request>
</XLS>
```

The response follows.

## Route Response Containing Route Between Two Points (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <DetermineRouteResponse>
      <RouteSummary>
        <TotalTime>PT8M34S</TotalTime>
        <TotalDistance value="3.67293" uom="MI" />
        <BoundingBox>
          <gml:pos>37.80661 -122.39258</gml:pos>
          <gml:pos>37.79318 -122.44184</gml:pos>
        </BoundingBox>
      </RouteSummary>
      <RouteInstructionsList format="text/plain" lang="en">
        <RouteInstruction>Non-street location</RouteInstruction>
        <RouteInstruction>Depart (E) on FRANCISCO ST.
          (0.03 mi)</RouteInstruction>
        <RouteInstruction>Turn RIGHT onto SCOTT ST.
          (0.14 mi)</RouteInstruction>
        <RouteInstruction>Turn LEFT onto LOMBARD ST (US-101).
          (0.91 mi)</RouteInstruction>
        <RouteInstruction>Turn LEFT onto VAN NESS AVE.
          (0.2 mi)</RouteInstruction>
        <RouteInstruction>Turn RIGHT onto BAY ST.
          (1.06 mi)</RouteInstruction>
        <RouteInstruction>Turn Slight RIGHT onto THE EMBARCADERO
          (EMBARCADERO PLAZA/JUSTIN HERMAN PLAZA).
          (1.16 mi)</RouteInstruction>
        <RouteInstruction>Turn RIGHT onto MISSION ST.
          (0.04 mi)</RouteInstruction>
        <RouteInstruction>Turn RIGHT onto STEUART ST
```

## Route Response Containing Route Between Two Points (2 of 2)

---

```
(JUSTIN HERMAN PLAZA). (0.13 mi)</RouteInstruction>
<RouteInstruction>Arrive Non-street location
</RouteInstruction>
</RouteInstructionsList>
</DetermineRouteResponse>
</Response>
</XLS>
```

## Requesting Maps with Route Requests

The following examples show how to issue route requests that include embedded requests for maps.

Generating a map with a Route request differs from doing so with a Presentation request (via `PortrayMapRequest`) in the following ways.

- When a `RouteMapRequest` element is embedded in a `DetermineRouteRequest`, the geometry for the route is automatically overlaid on the map, so there is no need to specify an explicit `RouteGeometry` element (within the `Overlay` element) as you do with a `PortrayMapRequest`.
- If no context (neither `BboxContext` nor `CenterContext`) is specified in a `DetermineRouteRequest`, the bounds of the generated map are derived from the route. In contrast, a `PortrayMapRequest` will fail if no context is specified.

In the first request/response example below, the generated map is based on a `CenterContext` centered on the route starting point, showing an area within a radius of one kilometer.

The second request/response example includes no geographic context, so the returned map displays the entire route. The actual maps are shown following the code listings.

## Route Request for Center-Context Map (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml" lang="en" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="DetermineRouteRequest" version="1.0.1" requestID="">
    <DetermineRouteRequest distanceUnit="KM">
      <RoutePlan>
        <RoutePreference>Fastest</RoutePreference>
```

## Route Request for Center-Context Map (2 of 2)

---

```
<WayPointList>
  <StartPoint>
    <Position>
      <gml:Point>
        <gml:pos dimension="2">37.76671 -122.43108</gml:pos>
      </gml:Point>
    </Position>
  </StartPoint>
  <EndPoint>
    <Position>
      <gml:Point>
        <gml:pos dimension="2">37.75252 -122.41234</gml:pos>
      </gml:Point>
    </Position>
  </EndPoint>
</WayPointList>
</RoutePlan>
<RouteMapRequest>
  <Output width="400" height="400" format="image/png"
    bitsPerPixel="24" content="URL">
    <CenterContext SRS="WGS84">
      <CenterPoint>
        <gml:pos>37.76671 -122.43108</gml:pos>
      </CenterPoint>
      <Radius unit="M">
        1000.0
      </Radius>
    </CenterContext>
  </Output>
</RouteMapRequest>
</DetermineRouteRequest>
</Request>
</XLS>
```

## Route Response Containing Center-Context Map (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml">
  <ResponseHeader/>
  <Response requestID="" version="1.0.1">
    <DetermineRouteResponse>
      <RouteSummary>
        <TotalTime>PT7M47S</TotalTime>
        <TotalDistance uom="KM" value="3.407"/>
        <BoundingBox>
```

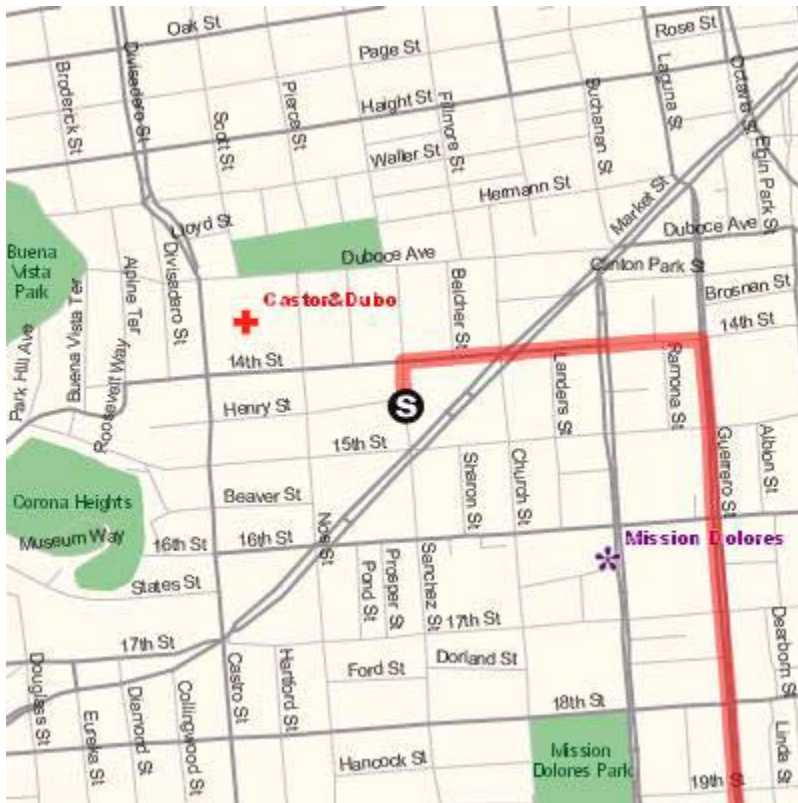
## Route Response Containing Center-Context Map (2 of 2)

---

```
<gml:pos>37.767921 -122.412376</gml:pos>
<gml:pos>37.751858 -122.43116</gml:pos>
</BoundingBox>
</RouteSummary>
<RouteMap>
  <Content height="400" width="400" format="image/png">
    <URL>http://127.0.0.1:8001/geomap/getMap?LAYERS=sfcolor.mwf&
      REQUEST=map&CENTER=-122.431080,37.766710&
      RADIUS=1000.0&AZIMUTH=0.0&SRS=ADSK:LL84&
      WIDTH=400&HEIGHT=400&FORMAT=PNG&PPIY=72&
      HLS=1&LINES=29,
      -122.431066,37.766709,-122.431061,37.766834,
      -122.431160,37.767086,-122.431130,37.767513,
      -122.430534,37.767548,-122.430038,37.767578,
      -122.428932,37.767647,-122.428604,37.767658,
      -122.428452,37.767677,-122.427841,37.767712,
      -122.426727,37.767784,-122.426559,37.767796,
      -122.425476,37.767857,-122.425293,37.767857,
      -122.424446,37.767921,-122.424294,37.766342,
      -122.424171,37.764721,-122.424133,37.764393,
      -122.424080,37.763886,-122.423988,37.763065,
      -122.423912,37.762253,-122.423843,37.761433,
      -122.423691,37.759880,-122.423614,37.759068,
      -122.423523,37.758263,-122.423462,37.757538,
      -122.423378,37.756680,-122.423302,37.755905,
      -122.423218,37.755108,1&SYMBOLS2=LL+--+Start,LL+--+Start,
      -122.43106620865505,37.76670940276802,0.007071,
      0.007071,0.0,,LL+--+End,LL+--+End,
      -122.4123764,37.7525024,0.007071,0.007071,0.0,
      &CLIENTID=clientname&
      LOGIN_SESSION_ID=1296bdc9906f09c50022a6af1373793d</URL>
    </Content>
    <CenterContext SRS="WGS84" xmlns:gml="http://www.opengis.net/gml"
      xmlns="http://www.opengis.net/xls">
      <CenterPoint>
        <gml:pos>37.76671 -122.43108</gml:pos>
      </CenterPoint>
      <Radius unit="M">
        1000.0
      </Radius>
    </CenterContext>
  </RouteMap>
</DetermineRouteResponse>
</Response>
</XLS>
```



The route and map appear as follows.



Route and center-context map

## Route Request for Map With No Geographic Context

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
      xmlns:gml="http://www.opengis.net/gml"
      lang="en" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password"/>
  <Request methodName="DetermineRouteRequest" version="1.0.1" requestID="">
    <DetermineRouteRequest distanceUnit="KM">
      <RoutePlan>
        <RoutePreference>Fastest</RoutePreference>
        <WayPointList>
          <StartPoint>
            <Position>
              <gml:Point>
                <gml:pos dimension="2">37.76671 -122.43108</gml:pos>
              </gml:Point>
            </Position>
          </StartPoint>
          <EndPoint>
            <Position>
              <gml:Point>
                <gml:pos dimension="2">37.75252 -122.41234</gml:pos>
              </gml:Point>
            </Position>
          </EndPoint>
        </WayPointList>
      </RoutePlan>
      <RouteMapRequest>
        <Output width="400" height="400" format="image/png"
              bitsPerPixel="24" content="URL"/>
      </RouteMapRequest>
    </DetermineRouteRequest>
  </Request>
</XLS>
```

## Route Response Containing Map With No Geographic Context (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml">
  <ResponseHeader/>
  <DetermineRouteResponse>
    <RouteSummary>
      <TotalTime>PT7M47S</TotalTime>
      <TotalDistance uom="KM" value="3.407"/>
      <BoundingBox>
        <gml:pos>37.767921 -122.412376</gml:pos>
        <gml:pos>37.751858 -122.43116</gml:pos>
      </BoundingBox>
    </RouteSummary>
    <RouteMap>
      <Content height="400" width="400" format="image/png">
        <URL>http://127.0.0.1:8001/geomap/getMap?LAYERS=sfcolor.mwf&
          REQUEST=map&BBOX=-122.43303835999998,37.75025144,
          -122.41049804,37.76952776&SRS=ADSK:LL84&
          WIDTH=400&HEIGHT=400&FORMAT=PNG&PPIY=72&
          HLS=1&LINES=51,
          -122.431066,37.766709,-122.431061,37.766834,
          -122.431160,37.767086,-122.431130,37.767513,
          -122.430534,37.767548,-122.430038,37.767578,
          -122.428932,37.767647,-122.428604,37.767658,
          -122.428452,37.767677,-122.427841,37.767712,
          -122.426727,37.767784,-122.426559,37.767796,
          -122.425476,37.767857,-122.425293,37.767857,
          -122.424446,37.767921,-122.424294,37.766342,
          -122.424171,37.764721,-122.424133,37.764393,
          -122.424080,37.763886,-122.423988,37.763065,
          -122.423912,37.762253,-122.423843,37.761433,
          -122.423691,37.759880,-122.423614,37.759068,
          -122.423523,37.758263,-122.423462,37.757538,
          -122.423378,37.756680,-122.423302,37.755905,
          -122.423218,37.755108,-122.423142,37.754265,
          -122.423073,37.753468,-122.423004,37.752663,
          -122.422943,37.751858,-122.422829,37.751869,
          -122.421616,37.751946,-122.421158,37.751961,
          -122.420685,37.751991,-122.420616,37.752003,
          -122.420082,37.752052,-122.419556,37.752083,
          -122.419014,37.752113,-122.418465,37.752144,
          -122.417870,37.752171,-122.417336,37.752216,
          -122.416756,37.752232,-122.416206,37.752277,
          -122.415123,37.752350,-122.414047,37.752399,
          -122.413498,37.752441,-122.412933,37.752487,
          -122.412376,37.752502,1&
          SYMBOLS2=LL+-+Start,LL+-+Start,
          -122.43106620865505,37.76670940276802,0.007071,
          0.007071,0.0,,LL+-+End,LL+-+End,
```

## Route Response Containing Map With No Geographic Context (2 of 2)

```
-122.4123764,37.7525024,0.007071,0.007071,0.0,
&amp;CLIENTID=clientname&amp;
LOGIN_SESSION_ID=1296bdc9906f09c50022a6af1373793d</URL>
</Content>
<BoundingBox>
  <gml:pos>37.77116 -122.410498</gml:pos>
  <gml:pos>37.748619 -122.433038</gml:pos>
</BoundingBox>
</RouteMap>
</DetermineRouteResponse>
</Response>
</XLS>
```

The map appears as follows.



Route and map with no geographic context

## Retrieving Previously Determined Routes with Route Handles

A route handle is an optional feature that allows a route to be stored temporarily on the WSG server. Route handles can be used to request additional information about a route, or as input to other services, such as the Alert service used to obtain traffic along a route. To obtain a route handle, request a route in the normal way, but add the attribute `provideRouteHandle="true"` to the `DetermineRouteRequest` element, as shown in the following request.

### Route Handle Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
      xmlns:gml="http://www.opengis.net/gml" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DetermineRouteRequest"
    version="1.0.1" requestID="1">
    <DetermineRouteRequest
      provideRouteHandle="true">
      <RoutePlan>
        <RoutePreference>Fastest</RoutePreference>
        <WayPointList>
          <StartPoint>
            <Position>
              <gml:Point>
                <gml:pos>37.801082 -122.44184</gml:pos>
              </gml:Point>
            </Position>
          </StartPoint>
          <EndPoint>
            <Position>
              <gml:Point>
                <gml:pos>37.79449 -122.39473</gml:pos>
              </gml:Point>
            </Position>
          </EndPoint>
        </WayPointList>
      </RoutePlan>
      <RouteInstructionsRequest/>
    </DetermineRouteRequest>
  </Request>
</XLS>
```

The following response includes the route summary and instructions. The `RouteHandle` element's `routeID` attribute contains the unique identifier for the stored route.

### Route Handle Response with Route Summary and Instructions

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <DetermineRouteResponse>
      <RouteHandle serviceID="ADSKWSG"
        routeID="ca7ed4c5-906f-aa44-013c-bbbfa8b57ec0" />
      <RouteSummary>
        <TotalTime>PT8M34S</TotalTime>
        <TotalDistance value="5.911" uom="KM" />
        <BoundingBox>
          <gml:pos>37.80661 -122.39258</gml:pos>
          <gml:pos>37.79318 -122.44184</gml:pos>
        </BoundingBox>
      </RouteSummary>
      <RouteInstructionsList format="text/plain" lang="en">
        <RouteInstruction>Non-street location</RouteInstruction>
        <RouteInstruction>Depart (E) on FRANCISCO ST.
          (0.05 km)</RouteInstruction>
        ...7 route instructions omitted for brevity...
        <RouteInstruction>Arrive Non-street
          location</RouteInstruction>
      </RouteInstructionsList>
    </DetermineRouteResponse>
  </Response>
</XLS>
```

To retrieve a stored route, use a route request and set the `RouteHandle` element's `routeID` attribute to the route's unique identifier. Note that by default, a route handle expires in 15 minutes. If you do not submit route-handle requests within that time limit, you will receive an error message indicating that the requested route identifier was not found in the route cache. To modify the route handle cache time limit (expiration) value, contact your LocationLogic administrator.

The following request asks for the stored route's geometry:

### Route Handle Geometry Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls"
  xmlns:gml="http://www.opengis.net/gml" version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password" />
  <Request methodName="DetermineRouteRequest"
    version="1.0.1" requestID="1">
    <DetermineRouteRequest
      provideRouteHandle="true">
        <RouteHandle routeID="ca7ed4c5-906f-aa44-013c-bbbfa8b57ec0"
          serviceID="ADSKWMSG" />
        <RouteGeometryRequest />
      </DetermineRouteRequest>
    </Request>
  </XLS>
```

The following response contains the geometry of the previously determined route.

### Route Handle Geometry Response (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
  xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <DetermineRouteResponse>
      <RouteHandle serviceID="ADSKWMSG"
        routeID="ca7ed4c5-906f-aa44-013c-bbbfa8b57ec0" />
      <RouteSummary>
        <TotalTime>PT8M34S</TotalTime>
        <TotalDistance value="5911" uom="M" />
        <BoundingBox>
          <gml:pos>37.80661 -122.39258</gml:pos>
          <gml:pos>37.79318 -122.44184</gml:pos>
        </BoundingBox>
      </RouteSummary>
      <RouteGeometry>
        <gml:LineString>
          <gml:pos>37.801083 -122.44184</gml:pos>
          <gml:pos>37.80115 -122.44133</gml:pos>
          ...49 linestring points omitted for brevity...
          <gml:pos>37.79361 -122.3936</gml:pos>
          <gml:pos>37.79449 -122.39473</gml:pos>
        </gml:LineString>
      </RouteGeometry>
    </DetermineRouteResponse>
  </Response>
</XLS>
```

```
</DetermineRouteResponse>  
</Response>  
</XLS>
```

## Specifying Features to Avoid in Route Requests

You can use the `AvoidList` element in route requests to specify route restrictions. Following are the types of features you can avoid:

- POIs
- Positions
- Areas of Interest (AOIs)
- Tollways
- Bridges
- Tunnels
- Ferries

When specified as elements to avoid, `Address`, `POI`, and `Position` will result in the closest road network link being avoided. The maximum number of road network links to avoid when specifying travel restrictions is a configurable property, with a default of 1000. To change this property, contact your `LocationLogic` administrator.

## Time Zone Support in Route Requests

When you make a route service request, you should specify the desired time zone in the `RoutePlan` element's `expectedStartTime` attribute. If you omit the time zone, the WSG uses GMT (Greenwich Mean Time), regardless of where the WSG is physically installed. This could lead to unexpected results, particularly if your application is processing requests from multiple time zones.

## Time-Dependent and Time-Independent Route Service Responses

When your application makes a Route service request, the WSG uses the presence or absence of the `expectedStartTime` attribute to determine which type of route to calculate:

- `expectedStartTime` specified—The Route service calculates a time-dependent route.
- `expectedStartTime` omitted—The Route service calculates a time-independent route.



**Note** A time-independent route means that *all* time restrictions are applied (so as to allow for the case where a route has been requested during a time when restrictions are in effect).

When determining what time to use as a basis for calculating the Route response, the XML Web Services take into account whether the `expectedStartTime` attribute is specified, and if so, whether a time zone is specified.

The following table provides examples for each possible combination:

Specification of <code>expectedStartTime</code>	Example Route Calculation
Request specifies <code>expectedStartTime</code> ; contains a time zone.	Request specifies time of 4:30 PST (GMT - 8 hours). Server is running MST. Time is normalized to 12:30 GMT (MST is ignored). Route is calculated based on 12:30 GMT.
Request specifies <code>expectedStartTime</code> ; omits the time zone.	Request specifies time of 5:30 but does not specify a time zone. Server is running MST. Time is normalized to 12:30 GMT. Route is calculated based on 12:30 GMT.
Request does not specify <code>expectedStartTime</code> .	Request does not specify a time. A time-independent route is calculated.

# Alert Requests

The Alert service enables LocationLogic client applications to create subscriptions in order to receive alert messages about traffic events that occur along a route.

## Listing Subscriptions

The following request retrieves a list of an application's subscriptions.

### List Subscriptions Request

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientName" clientPassword="password" />
  <Request methodName="SubscriptionRequest" version="1.0.1" requestID="1">
    <SubscriptionRequest>
      <ListSubscriptions />
    </SubscriptionRequest>
  </Request>
</XLS>
```

The response contains a list of subscriptions.

### List Subscriptions Response

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <SubscriptionResponse>
      <Subscription id="143">
        <Name>Morning Commute</Name>
        <Type>TrafficAlongRoute</Type>
      </Subscription>
      ... additional subscriptions omitted for brevity ...
    </SubscriptionResponse>
  </Response>
</XLS>
```

## Creating a Subscription for Traffic Along a Route

You create a subscription for traffic along a specific, predetermined route. To link that route to your subscription, you must obtain a route handle (see “Retrieving Previously Determined Routes with Route Handles” on page 139), and use it when you create the subscription.

### Create Subscription Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
<RequestHeader clientName="clientName" clientPassword="password" />
  <Request methodName="SubscriptionRequest" version="1.0.1" requestID="1">
    <SubscriptionRequest>
      <CreateSubscription>
        <Name>Morning Commute</Name>
        <msid type="MSISDN">46708123456790</msid>
        <AlertURL>http://www.mysvlt.com/alertlistener</AlertURL>
        <ActivationPeriod>
          <Start>2003-01-31T07:30:00.000-05:00</Start>
          <Reoccurs>
            <ByDay>MO TU WE TH FR</ByDay>
            <ByHour>7</ByHour>
            <ByMinute>30</ByMinute>
            <Duration>PT30M</Duration>
          </Reoccurs>
        </ActivationPeriod>
        <TrafficAlongRoute>
          <Source>TrafficFeedFeatureCategory</Source>
          <RouteHandle routeID="123ABC" />
          <AlertUpdateMode>Periodic</AlertUpdateMode>
        </TrafficAlongRoute>
      </CreateSubscription>
    </SubscriptionRequest>
  </Request>
</XLS>
```

You will receive the following response to a successful subscription request. The `id` value can be used in subsequent requests to activate, deactivate, or delete the subscription.

### Create Subscription Response (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.0">
  <ResponseHeader/>
  <Response version="1.0.0" requestID="1">
    <SubscriptionResponse>
```

## Create Subscription Response (2 of 2)

---

```
<Subscription id="741"/>
</SubscriptionResponse>
</Response>
</XLS>
```

## Editing a Subscription

You can edit any existing subscription that was created using LocationLogic XML Web Services, whether or not the subscription is currently active. For optimum performance, you should include only those fields in the `SubscriptionDetail` element for which information is changing. The following request changes a subscription's activation period.

### Edit Subscription Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" xmlns:gml="http://www.opengis.net/gml"
version="1.0.1">
  <RequestHeader clientName="clientname" clientPassword="password">
    <Request methodName="SubscriptionRequest" version="1.0.1" requestID="1">
      <SubscriptionRequest>
        <EditSubscription>
          <SubscriptionDetail id="subscriptionID
```

The response follows.

### Edit Subscription Response (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS version="1.0.1" xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
```

## Edit Subscription Response (2 of 2)

---

```
<Response requestID="1" version="1.0.1">
  <SubscriptionResponse>
    <Subscription id="741" />
  </SubscriptionResponse>
</Response>
</XLS>
```

## Activating a Subscription

You can activate a previously created and deactivated subscription any time with the following request:

### Subscription Activation Request

---

```
<?xml version="1.0" encoding="UTF-8" ?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientName" clientPassword="password" />
  <Request methodName="SubscriptionRequest" version="1.0.1" requestID="1">
    <SubscriptionRequest>
      <ActivateSubscription>
        <Subscription id="subscriptionID" />
        <msid type="MSISDN">MSID</msid>
      </SubscriptionRequest>
    </ActivateSubscription>
  </SubscriptionRequest>
</Request>
</XLS>
```

The response follows.

### Subscription Activation Response

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <SubscriptionResponse>
      <Subscription id="741" />
    </SubscriptionResponse>
  </Response>
</XLS>
```

## Deactivating a Subscription

You can deactivate a previously created, active subscription any time with the following request:

### Deactivate Subscription Request

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientName" clientPassword="password" />
  <Request methodName="SubscriptionRequest" version="1.0.1" requestID="1">
    <SubscriptionRequest>
      <DeactivateSubscription>
        <Subscription id="subscriptionID" />
        <msid type="MSISDN">MSID</msid>
      </DeactivateSubscription>
    </SubscriptionRequest>
  </Request>
</XLS>
```

The response follows:

### Deactivate Subscription Response

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS version="1.0.1" xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.opengis.net/xls">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <SubscriptionResponse>
      <Subscription id="741" />
    </SubscriptionResponse>
  </Response>
</XLS>
```

## Deleting a Subscription

You can delete an active or inactive subscription with the following request:

### Delete Subscription Request (1 of 2)

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1">
  <RequestHeader clientName="clientName" clientPassword="password" />
  <Request methodName="SubscriptionRequest" version="1.0.1" requestID="1">
    <SubscriptionRequest>
```

## Delete Subscription Request (2 of 2)

---

```
<DeleteSubscription>
  <Subscription id="subscriptionID" />
    <msid type="MSISDN">MSID</msid>
  </Subscription>
</DeleteSubscription>
</SubscriptionRequest>
</Request>
</XLS>
```

The response follows:

## Delete Subscription Response

---

```
<?xml version="1.0" encoding="UTF-8"?>
<XLS xmlns="http://www.opengis.net/xls" version="1.0.1"
xmlns:gml="http://www.opengis.net/gml">
  <ResponseHeader />
  <Response version="1.0.1" requestID="1">
    <SubscriptionResponse>
      <Subscription id="741" />
    </SubscriptionResponse>
  </Response>
</XLS>
```

## Receiving Alert Messages

When subscriptions are triggered, the WSG sends out an XML message to the Alert URL specified when the subscription was created. Alerts are sent only for incidents, not for subscription status changes (such as subscription cancellations). Alerts are sent only when there is something to report, such as a traffic incident along a route. So it is possible for an application to receive no alerts during a subscription activation period.

Applications should have a dedicated listener to receive the alert messages and quickly respond. This ensures that further WSG alert messages are not delayed due to client message processing.

**Note** The WSG ignores all data in the alert message response.

The following XML snippet illustrates an alert message for traffic along a route:

```
<XLSAlert>
  <Subscription id="subscriptionID">
    <Name>Morning Commute</Name>
    <Type>TrafficAlongRoute</Type>
```

```
</Subscription>
<TrafficAlongRouteAlert numberOfEvents="1">
  <TrafficEvent>
    <Description>10:05,A21,MONCALIERI,TROFARELLO,VERY SLOW
    </Description>
    <BoundingBox>
      <gml:pos>51.2 -109.5</gml:pos>
      <gml:pos>51.3 -109.4</gml:pos>
    </BoundingBox>
  </TrafficEvent>
</TrafficAlongRouteAlert>
</XLSAlert>
```



# Error Codes

The Web Services Gateway provides error codes for applications using LocationLogic XML Web Services. This appendix lists the available error codes related to each of the XML Web Services schemas.

# A

## In this appendix

- Geoservice error codes
- Location provisioning result codes
- Client provisioning result codes
- Auxiliary result codes

# Geoservice Error Codes

LocationLogic XML Web Services assign error codes to the following types of errors:

- **Implementation Errors**—Arise from conditions that are unexpected or unrecoverable, and cannot be corrected by client applications. Implementation errors are logged using the LocationLogic platform’s logging mechanism.
- **Validation Errors**—Occur during the validation portion of the request and during conversion of WSG types to LocationLogic Java `Lbs` types. The client application can fix these errors by modifying the request data to conform to the schema.
- **Unsupported Errors**—Produced from request documents that pass validation but access an unsupported part of the XLS schema. Unlike validation errors, unsupported errors do not signal the presence of bad data. The client application can fix these errors by modifying the request document.

## Geoservice Error Codes

The following error codes indicate the type of error generated by the XML validator or support checker:

Error Code	Description
100	Unknown
101	Request version mismatch
102	Response version mismatch
103	Value not recognized
104	Inconsistent
105	Security

## Geoservice Implementation Error Codes

The following error codes indicate implementation errors. The range for implementation error codes is 10000—10999.

### Geoservice Implementation Error Codes (1 of 3)

Error Code	Description
10000	Geocode address failed
10001	No geocode matches
10002	No error message available
10003	Unexpected content type
10004	Value not set
10005	Unsupported route preference type
10006	Unsupported request type
10007	Unsupported POI location format
10008	Unsupported feature property type
10009	Unsupported client distance units
10010	Unsupported boundary type
10011	Unexpected exception type
10012	Unexpected error type
10013	Cannot format negative time
10014	Time exceeds maximum long value
10015	Significant figures format not available
10016	Number of significant figures invalid

## Geoservice Implementation Error Codes (2 of 3)

Error Code	Description
10017	Null or empty language code
10018	Non-string property comparison
10019	Negative lat-lon accuracy
10020	Negative distance value
10021	Missing request element
10022	Missing location group element
10023	Missing comparison property
10024	Mismatched SRS values
10025	Maximum significant figures invalid
10026	Maximum seconds-to-milliseconds overflow
10027	Invalid URL
10028	Invalid property value for default maximum distance
10029	Invalid number of <code>pos</code> elements
10030	Invalid bounding box string
10031	Insufficient polygon coordinates
10032	DPI and radius both null
10033	Distance property not found
10034	Client units not set;
10035	Center or Bounding Box context require

### Geoservice Implementation Error Codes (3 of 3)

Error Code	Description
10036	Cannot format negative distance
10037	Unexpected image format
10038	Runtime error stack trace
10039	Unsupported distance unit
10040	Null or empty language
10041	Unsupported bits per pixel
10042	Unexpected number of matches
10043	Bounding box X-axis error
10044	Bounding box Y-axis error
10045	Zero maximum response
10046	Subscription—UUID general error
10047	Subscription—Non-unique type
10048	No match for subscription type string
10049	Unknown subscription abbreviation type
10050	Invalid subscription status
10051	Invalid subscription detail type
10052	Unknown character encoding

## Geoservice Validation Error Codes

The following error codes indicate validation errors. The range for validation error codes is 11000—11999:

### Geoservice Validation Error Codes (1 of 3)

Error Code	Description
11000	Invalid client name
11001	Invalid request document
11002	Invalid response document
11003	Invalid lang attribute
11004	Empty or missing client name
11005	Request element type does not match method name attribute
11006	Invalid Pos element
11007	Missing exterior element on polygon
11008	Route instructions not available for specified language
11009	Directory type not specified
11010	Invalid POI
11012	Authentication failure
11013	Password missing
11014	DPI not positive
11015	Display scale not positive
11016	Radius not positive
11017	Line style is not an integer

### Geoservice Validation Error Codes (2 of 3)

Error Code	Description
11018	Image width is missing
11019	Image height is missing
11020	BoundingBox or CenterContext is missing
11021	Maximum responses too small
11022	Maximum avoid links exceeded
11023	OpenLS schema version mismatch
11024	Route handle supplied not found in cache
11025	Route handle cache not enabled
11026	Minimum distance is greater or equal than maximum distance
11027	Position element is missing Point
11028	Invalid URL string
11029	Non-unique overlay IDs
11030	Invalid day in a Reoccurs element
11031	Invalid hour in a Reoccurs element
11032	Invalid minute in a Reoccurs element
11033	Missing map format
11034	Start and end are equal; they must be different
11035	End time is earlier than start time; end must be later than start
11036	Missing subscription ID

### Geoservice Validation Error Codes (3 of 3)

Error Code	Description
11037	Invalid number format
11038	Invalid free-form address
11039	Illegal argument
11040	Subscription not recognized as created by the WSG
11041	Subscription could not be found for specified id and user
11042	Position not available for the submitted device id(s)
11043	Invalid polygon style
11044	Invalid symbol name
11045	POI ID value required
11046	Invalid int property value
11047	Invalid long property value
11048	Invalid float property value
11049	Invalid double property value
11050	Invalid date property value
11051	At least one POI property value must be supplied
11052	Unsupported property type



## Geoservice Unsupported Feature and Format Error Codes

The following error codes indicate unsupported feature and format errors. The range for unsupported feature and format error codes is 12000—12999

### Geoservice Unsupported Feature and Format Error Codes

Error Code	Description
12000	Freeform address not supported
12001	Unsupported dimension
12002	<code>nearestCriterion of Easiest</code> not supported
12003	Unsupported radius UOM (unit of measure)
12004	Pinpoint directory requests using <code>Address</code> not supported
12005	Unsupported geometry shape
12006	Unsupported image format
12007	Unsupported <code>AvoidFeatureType</code>
12008	Unsupported <code>DistanceUnitType</code>
12009	Unsupported reverse geocode preference type

## Geoservice Generic Error Codes

The following are generic error codes that may be generated by the WSG:

Error Code	Description
0	OK
-1	Unknown error

# Location Provisioning Result Codes

These codes are returned in the `resultId` attribute of the `<result>` element, which is always included in the location provisioning responses. The result code ranges are taken from the MLP (Mobile Location Protocol) specification and reused for location provisioning for consistency.:

Location Provisioning Result Codes (1 of 2)

Location Provisioning Result Code	Description
0	OK
1	System failure
2	Unspecified error
3	Unauthorized application
4	MSID does not exist
5	MSID already exists
6	Client authorization already exists
7	Requestor authorization already exists
8	Client authorization does not exist
9	Requestor authorization does not exist
10	Cannot find privacy profile for MSID
11	Default privacy has not been supplied
12	A privacy profile for the MSID already exists
13	The client does not exist
14	Document does not contain any requests

## Location Provisioning Result Codes (2 of 2)

Location Provisioning Result Code	Description
15	The <code>qopLimit</code> is negative
16	The <code>qopLimit</code> is too large
101	Request cannot be handled due to congestion in the location server
102	Request cannot be handled due to congestion in the mobile network
103	Unsupported version
104	Format error
105	Syntax error
106	<code>Protocol</code> element not supported
107	Service not supported
108	<code>Protocol</code> element attribute not supported
109	Invalid <code>Protocol</code> element value
110	Unknown character encoding
111	Document error
112	Document validation error

# Client Provisioning Result Codes

These codes are returned in the `resultId` attribute of the `<result>` element, which is always included in the client provisioning responses. The result code ranges are taken from the MLP (Mobile Location Protocol) specification and reused for client provisioning for consistency.:

## Client Provisioning Result Codes (1 of 2)

Client Provisioning Result Code	Description
0	OK
1	System failure
2	Unspecified error
3	Unauthorized application
4	Client ID does not exist
5	The role is unknown
101	Request cannot be handled due to congestion in the location server
102	Request cannot be handled due to congestion in the mobile network
103	Unsupported version
104	Format error
105	Syntax error
106	Protocol element not supported
107	Service not supported
108	Protocol element attribute not supported

### Client Provisioning Result Codes (2 of 2)

Client Provisioning Result Code	Description
109	Invalid <code>Protocol</code> element value
110	Unknown character encoding
111	Document error
112	Document validation error
113	Illegal number format error

# Auxiliary Result Codes

These codes are returned in the `resultId` attribute of the `<result>` element, which is always included in the *llaux* responses. The result code ranges are taken from the MLP (Mobile Location Protocol) specification and reused for auxiliary services for consistency.

## Auxiliary Result Codes (1 of 2)

Auxiliary Result Code	Description
0	OK
1	System failure
2	Unspecified error
3	Unauthorized application
4	Debug logging has not been enabled within LocationLogic
101	Request cannot be handled due to congestion in the location server
102	Request cannot be handled due to congestion in the mobile network
103	Unsupported version
104	Format error
105	Syntax error
106	<code>Protocol</code> element not supported
107	Service not supported
108	<code>Protocol</code> element attribute not supported
109	Invalid <code>Protocol</code> element value

**Auxiliary Result Codes (2 of 2)**

Auxiliary Result Code	Description
110	Unknown character encoding
111	Document error
112	Document validation error
113	Empty document





# Index

## Symbols

.NET 2

## A

- adding a client 39
- Alert service 4, 22, 31
  - activating subscriptions 147
  - common requests 144
  - conditions for subscriptions 32
  - continuous subscriptions 32
  - creating a subscription for traffic along a route 145
  - deactivating subscriptions 148
  - deleting subscriptions 148
  - editing a subscription 146
  - listing subscriptions 144
  - periodic 32
  - receiving alert messages 149
  - subscription triggers 32
  - temporal filters 32
  - See also* Alert requests
- AlertUpdateMode element, and subscription notification 32
- alt\_acc 59
- AOI element 76
- application authentication 37
- applications
  - configuring SSL properties 11
  - security 9
- Area of Interest. *See* AOI element
- authentication
  - application 37
- Auxiliary result type codes 164
- auxiliary services 4, 33
- available privacy setting levels 24
- available services 22
- avoiding links 142
- AvoidList element 142
- azimuth 28

## B

- bounding box 30
- BREW 2

## C

- CenterContext output type 28
- Client Provisioning
  - service 3
- client provisioning 22
  - requests 39
  - result type codes 162
- clientName attribute 37
- clientPassword 37
- clientProv.xsd 5
- communication security 9
- content management service 22
- content, required knowledge of 6
- Coordinate Conversion 30
- coordinate conversion 30
- coordinate order
  - image map hotspots 122
- coordinate system, used by WSG 17, 60
- creating privacy profiles 48

## D

- Determination of device location 4
- DetermineRouteRequest element 139
- deviceType 37
- digital certificate, WSG 10
- Directory requests
  - directory type 71
  - finding places along a route 77
  - map URL, specific area 89
  - map URL, surrounding a point 91
  - nearest places 71
  - SQL WHERE queries 79
  - within circular boundary 76
- Directory service 4, 26
  - common requests 71
  - feature and feature properties 27
  - feature categories 27
  - See also* Directory requests, Directory responses
- DirectoryResponse element 72
- directoryType attribute 71
- DirectoryUpdateRequest 84
- documentation, text conventions in viii

## E

- enc (MLP element) 59
- encoding
  - string literals for HTTP 12, 80
  - XML documents 11
- encryption 59
- encryption attribute 59
- encryption. *See* security
- eqop (MLP element) 58
- error codes
  - client provisioning 162
  - error type 152
  - generic errors 159
  - geoservices 152
  - implementation errors 153
  - location provisioning 160
  - unsupported feature and format errors 159
  - validation errors 156
- error type codes 152
- errorCode attribute 8
- errors
  - attributes of 8
  - handling of 7
  - LocationLogic 7
  - stack trace 7
- exceptions
  - See* errors
- expectedStartTime attribute 130, 142
- Extended Attributes of a POI 75
- Extensible Markup Language. *See* XML

## F

- feature category 26
- feature category path 71
- freeFormAddress element 64
- Fuzzy POI Name Search 81

## G

- Gateway requests
  - QoP
- generic error codes 159
- Geocode requests 55
  - search modes 67
  - street address 60
  - street intersection 63, 65
- Geocode responses
  - free-form address 65
  - multiple matches 62
  - street address 62
  - street intersection 66
- GeocodedAddress element 61
- GeocodeMatchCode element 66
- geocoding 23

- Geocoding/Reverse Geocoding service. *See*
  - Geocoding requests, Geocoding responses,
  - Location Utility service
- Geographic Information Systems (GIS) x
- geographical data
  - coordinates 17
  - geometry 18
  - linestring 18
  - location 25
  - ordinate 17
  - point 18
  - polygon 18
  - rectangle 18
  - working with 17
- geoservice error codes 152
- Geoservices 26
- GIS x
- gml elements 76
- gml4xls2.xsd* 5
- GMT 32

## H

- hor\_acc 58
- HorizontalAcc 58
- HotSpot element 122
- HotSpotList element 122
- HotSpotRequest element 121
- hotspots 28, 121
- HTTPS 9
- URLConnection 9

## I

- ID attribute of overlay element 30
- image map hotspots 121
- Implementation 152
- implementation error codes 153
- implementation errors 152, 153
- InputGatewayParameters element 58
- InputMSInformation element 55, 59
- intelligent reverse geocoding 70
- interpolation mode, reverse geocoding 70
- ISO 639 two-letter language
  - code standard x
  - setting response language with 130

## J

- J2EE 2
- J2SE 2
- Java Secure Sockets Extension 10
- java.security* 11
- JavaME 2
- JSEE library 10

## L

- LabelBoundingBox 122
- lang attribute 130
- languages
  - adding support for 127
  - ISO 639 two-letter language code standard
  - website x
- latitude 17
- Lbs errors. *See* LocationLogic errors
- LBSAPI roles 22
- linestring 18
  - finding places within distance of 77
- LineString element 99
- llaux.xsd* 5
- loc\_type (MLP element) 58
- location 25
- location determination technology (LDT) 25
- location privacy management service
  - role for 23
- Location Provisioning
  - requests 48
  - result type codes 160
  - service 3
- Location service
  - role for 23
- Location Utility service 4
  - common requests 60
  - see also* Geocode requests, Geocode responses
- location-based services (LBS)
  - geocoding 26
  - reverse geocoding 26
- LocationLogic
  - error reporting 7
- LocationLogic SDK
  - about vii
- LocationLogic XML Web Services
  - Alert service 4
  - architecture 3
  - compared with XLS schema. *See* XLS
  - Directory service 4
  - Location Utility (Geocoding/Reverse Geocoding) service 4
  - Presentation service 4
  - Route service 4
  - schema 5
- locationType 58
- locationType element 58
- locProv.xsd* 5
- logging service 4, 33
- longitude 18

## M

- maintaining privacy 23
- map characteristics 28

- map context 28
- map name 28
- map overlays 28
- map rotation 28
- map symbols 97, 98
- mapping
  - roles for 23
- mapping. *See* Presentation service
- max\_loc\_age 58
- MaximumDistance element 77
- maximumResponses attribute 71
- maxLocAge 58
- methodName 38
- MinimumDistance element 77
- MLP elements, support of 58
- MMS 2
- mobile device 25
- mobile devices 25
  - locating 25
- mobile to mobile alerts 31
- mobile to stationary alerts 31
- MSID 23
- MSID\_TYPE 23
- msIDType 59
- msIDType 59
- msIDValue element 55

## N

- Nearest element 71
- nearestCriterion attribute 71
- notifications. *See* alerts
- numberOfAddresses attribute 62

## O

- OBSUBTYPE2 33
- OGC x
- Open GIS Consortium. *See* OGC
- OpenLS schema
  - LocationLogic XML Web Services extensions 5
- OpenLS Specification
  - defined 2
  - website x
- operation subtype fields 33
- OPSUBTYPE1 33
- ordinate 17
- Overlay element 96, 106, 121
- overlay element in hotspot 30

## P

- POI element 96
- POIContext element 72
- point 18

- points of interest (POIs) 26
- POIProperties element 71
- polygon 18
- PortrayMapRequest element 89
- Position element 106
- precision 25
- precision elements 19
- preferences, route 31
- Presentation requests
  - adding a point overlay 96
  - binary map 94
  - image map hotspots 121
  - route map 99
  - specifying a logical map name 95
- Presentation responses
  - image map hotspots 122
- Presentation service 4, 27
  - common requests 89
  - See also* Presentation requests, Presentation responses
- prio (MLP element) 58
- priority 58
- priority element 58
- Privacy Checking service 25
- privacy profiles 23

## Q

- QoP 56
- quality of position. *See* QoP

## R

- rectangle 18
- removing a client 47
- removing a privacy profile 53
- removing client properties or roles 46
- removing part of a privacy profile 52
- Requested QoP 58, 59
- RequestedQoP element 56, 58
- requestedsrsName attribute 17
- RequestHeader element 37
- requesting maps with a route 29
- requests
  - body element 36
  - header element 36
- retrieving a client 41
- retrieving a privacy profile 49
- Reverse Geocode requests, coordinate 68
- Reverse Geocode responses, coordinate 69
- reverse geocoding area threshold property 70
- roles 22
  - alerts 22
  - content management service 22
  - for Location service 23
  - related LBSAPI roles 22
  - services table 22

- WSG\_ALERTS 22
- WSG\_CLIENT\_PROVISIONING 22
- WSG\_DIRECTORYUPDATE 22
- WSG\_LOCATION\_PROVISIONING 23
- WSG\_LOCATION\_PROVISIONING\_MSCLIENT\_SELF 23
- WSG\_MLP\_LOCATION 23
- WSG\_OPENLS\_COORDINATECONVERSION 23
- WSG\_OPENLS\_DETERMINEROUTE 23
- WSG\_OPENLS\_DIRECTORY 22
- WSG\_OPENLS\_GATEWAY 23
- WSG\_OPENLS\_GEOCODING 23
- WSG\_OPENLS\_PORTRAYMAP 23
- WSG\_OPENLS\_REVERSEGEOCODING 23

- route handle cache, time limit 140
- route handles 139
- route preferences 31
- Route service 4, 30
  - areas to avoid 5
  - complex routes 30
  - legs 30
  - requests 71, 127
    - route between two addresses 127
    - route between two points 130
    - route handles 139
    - time zone support 142
  - routes 30
  - step modifiers 30
  - steps 30
  - street networks 31
  - time-dependent and time-independent responses 142
  - travel directions 128
  - See also* Route service requests
- RouteGeometry element 128
- RouteGeometryRequest 130
- routeID 140
- RouteInstructionList element 128
- RoutePlan element 130, 142
- RoutePreference 130
- RouteSummary element 128
- routing
  - role for 23

## S

- sample requests
  - Alert service 144
  - Directory service 71
  - Geocode 55, 63, 65, 67
  - Presentation service 89
  - Reverse Geocode 68
  - Route 127, 130
  - Route service 71
- sample responses

- Geocode 62, 65, 66
- Reverse Geocode 69
- schema 5
  - clientProv 5
  - gml4xls2 5
  - llaux 5
  - locProv 5
  - XLS2 5
- SDK, LocationLogic. *See* LocationLogic SDK
- SearchCentreDistance element 69
- secure communications 9
  - Configuring Java clients for 9
  - JDK classes and library needed 9
- Secure Sockets Layer (SSL). *See* SSL
- security 9
  - configuring SSL properties 11
  - jsse library 10
- Service and Deployment Test URLs 34
- size and format of maps 28
- SMS 2
- spatial reference system (SRS), used by WSG 17
- SQL WHERE queries 79
- SSL
  - configuring Java clients for 9
  - configuring properties for client applications 11
  - configuring properties of 11
  - defined 9
- street networks, polyline 31
- StreetAddress element 66
- StreetIntersection element 65
- Style element 96, 97, 99
- subscriber authentication 37
- subscriptions
  - alert conditions 32
  - continuous 32
  - periodic alerts 32
  - temporal filters 32
  - time-based 32
  - triggered 32
  - See also* Alert service
- symbols 30
- symbols, map 97, 98

## T

- technical knowledge needed x
- temporal filters 32
- text conventions viii
- time-based subscriptions 32
- TotalDistance element 128
- TotalTime element 128
- traffic alerts 31
- transaction logging 4, 33
- transport security 9
- travel restrictions 142
- type (MLP element) 59

## U

- unsupported elements and attributes
  - checks for 13, 14
  - NotSupported error 13
- unsupported errors 152
- unsupported feature and format error codes 159
- updating a client 45
- updating a privacy profile 51
- URLs
  - auxiliary services 34
  - client provisioning 34
  - Geoservices (OpenLS) 34
  - location privacy provisioning 34

## V

- validation
  - content 15
  - content errors 16
  - during deserialization 13
  - error codes 156
  - errors 152
  - requests 12
  - schema 13
- VerticalAcc 59

## W

- WGS84, used by WSG 17, 60
- Windows Mobile 2
- WithinBoundary element 76
- WSG, security 9
- wsg.directory.defaultmaxdistance 77
- WSG\_ALERTS role 22
- WSG\_CLIENT\_PROVISIONING role 22
- WSG\_DIRECTORYUPDATE role 22
- WSG\_LOCATION\_PROVISIONING role 23
- WSG\_LOCATION\_PROVISIONING\_MSCLIENT\_SELF role 23
- WSG\_MLP\_LOCATION role 23
- WSG\_OPENLS\_COORDINATECONVERSION role 23
- WSG\_OPENLS\_DETERMINEROUTE role 23
- WSG\_OPENLS\_DIRECTORY role 22
- WSG\_OPENLS\_GATEWAY role 23
- WSG\_OPENLS\_GEOCODING role 23
- WSG\_OPENLS\_PORTRAYMAP role 23
- WSG\_OPENLS\_REVERSEGEOCODING role 23

## X

- XLS
  - defined 2
  - schema
- XLS envelope 36
- XLS2.xsd 5

## XML

API. *See also* LocationLogic XML Web  
Services  
comments, processing of 7

## documents

encoding 11  
multiple service requests in 7