Open Geospatial Consortium Inc.

Date: 2007-05-02

Reference number of this OGC® project document: **OGC 07-039r1**

Version: 0.0.9

Category: OGC® Best Practices

Editor: Carl Reed on behalf of Google Earth staff

KML 2.1 Reference – An OGC Best Practice

Warning

This is an OGC Best Practices document. It is not an OGC Standard and may not be referred to as an OGC Standard. It is subject to change without notice. However, this document is an official position of the OGC membership on this particular technology topic.

Document type: OGC® Candidate specification
Document subtype: OGC Best Practices document
Document stage: Draft
Document language: English

Preamble to "KML 2.1 - An OGC Best Practice"

Google submitted KML (formerly Keyhole Markup Language) to the Open Geospatial Consortium (OGC) to be evolved within the OGC consensus process with the following goal: KML Version 3.0 will be an adopted OpenGIS implementation specification that will have been harmonized with relevant OpenGIS specifications that comprise the OGC standards baseline. There are four objectives for this standards work:

- That there be one international standard language for expressing geographic annotation and visualization on existing or future web-based online maps (2d) and earth browsers (3d).

- That KML be aligned with international best practices and standards, thereby enabling greater uptake and interoperability of earth browser implementations.

- That the OGC and Google will work collaboratively to insure that the KML implementer community is properly engaged in the process and that the KML community is kept informed of progress and issues.

- That the OGC process will be used to insure proper life-cycle management of the KML candidate specification, including such issues as backwards compatibility.

The OGC has developed a broad Standards Baseline. Google and the OGC believe that having KML fit within that family will encourage broader implementation and greater interoperability and sharing of earth browser content and context.

What information sharing space is KML targeted at? KML is an XML language focused on geographic visualization, including annotation of maps and images. Geographic visualization includes not only the presentation of graphical data on the globe, but also the control of the user's navigation in the sense of where to go and where to look.

From this perspective, KML is complementary to most of the existing OGC specifications including key standards such as GML (Geography Markup Language), WFS (Web Feature Service) and WMS (Web Map Service). Currently, KML (v2.1) utilizes certain geometry elements derived from GML (version 2.1.2). These elements include point, line-string, linear-ring, and polygon.

The OGC and Google have agreed that there can be additional harmonization of KML with GML (e.g. to use the same geometry representation) in the future. The Mass Market Geo Working Group in the OGC will define additional harmonization activities. Other OGC specifications such as Context and SLD will be considered.

Google submitted the KML Reference Manual to the OGC. OGC staff put the reference document into the OGC Document Template. During the April Technical Committee meetings, the OGC membership approved release of the document as an OGC Best

Practices Paper. This document is based entirely on the current KML 2.1 reference documentation from Google. This OGC Best Practices document is provided to the community to initiate the process of KML standardization within the OGC consensus process.

Contents

i. Preface

This document is being submitted by Google, Inc to the OGC for future consideration as an OGC® Implementation Specification. This will be a multi-phase project. Phase one is to format the KML 2.1 Reference document into the standard OGC document template. The next step is to have the OGC membership consider the document for approval as an OGC Best Practices paper. The third step will be to do some initial harmonization with existing OGC Abstract and Implementation specifications, specifically with Topic 1 (Feature) and GML Geometry and Topic 2 (Spatial Referencing) and coordinate reference systems (CRS). The fourth and final step will be to propose KML as an RFC for consideration as an OGC adopted Implementation Specification.

ii. Submitting organizations

The following organizations submitted this Implementation Specification to the Open Geospatial Consortium Inc.:

a) Google, Inc.

iii. Submission contact points

All questions regarding this submission should be directed to the editor:

CONTACT	COMPANY	EMAIL
Carl Reed	OGC	creed at opengeospatial.org

iv. Revision history

Date	Release	Author	Paragraph modified	Description
3-7-06	0.0.5	Carl Reed	New	Initial Version
4/17/07	0.0.9	Carl Reed	Various	Updates based on comments received during Mass Market GEO WG meeting as well as a request from Google to remove 3 elements.
5/2/07	0.0.9	Carl Reed	Various	Add preamble and edit document for posting as a BP.

v. Changes to the OGC® Abstract Specification

The OGC® Abstract Specification does not require changes to accommodate this OGC® standard.

Note: This document is being initially submitted for consideration for approval as an OGC Best Practices document. The plan is to then, working jointly with the KML community, to begin phased and non-disruptive harmonization activities with various adopted OGC Abstract and Implementation Specifications.

Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights. However, to date, no such rights have been claimed or identified.

Introduction

KML is a file format used to display geographic data in an Earth browser, such as Google Earth, Google Maps, and Google Maps for Mobile. KML uses a tag-based structure with nested elements and attributes and is based on the XML standard.

You can create KML files with the Google Earth user interface, or you can use an XML or simple text editor to enter "raw" KML from scratch. KML files and their related images (if any) can be compressed using the ZIP format into KMZ archives. To share your KML and KMZ files, you can e-mail them, host them locally for sharing within a private internet, or host them publicly on a web server. Just as web browsers display HTML files, Earth browsers such as Google Earth display KML files. Once you've properly configured your server and shared the URL (address) of your KML files, anyone who's installed Google Earth can view the KML files hosted on your public web server

Introductory element — Main element — Part 1:

1 Scope

KML is a file format used to display geographic data in an earth browser, such as Google Earth, Google Maps, and Google Maps for mobile. A KML file is processed in much the same way that HTML (and XML) files are processed by web browsers. Like HTML, KML has a tag-based structure with names and attributes used for specific display purposes. Thus, Google Earth and Maps act as browsers for KML files.

You can use KML to:

- Specify icons and labels to identify locations on the planet surface
- Create different camera positions to define unique views for each of your features
- Use image overlays attached to the ground or screen
- Define styles to specify feature appearance
- Write HTML descriptions of features, including hyperlinks and embedded images
- Use folders for hierarchical grouping of features
- Dynamically fetch and update KML files from remote or local network locations
- Fetch KML data based on changes in the 3D viewer
- Display COLLADA textured 3D objects
- Annotate the Earth

2 Conformance

There are currently no conformance clauses for this specification.

3 Normative references

The following normative documents contain provisions that, through reference in this text, constitute provisions of this part of OGC 06-015. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of OGC 06-015 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies.

KML 2.1 Reference Document, 2007, Google Inc.
http://earth.google.com/kml/kml_tags_21.html

Abstract Specification Topic 0: Overview, OGC document 99-100r1

Guidelines for Successful OGC Interface Specifications, OGC document 00-014r1

4 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

4.1
Feature
Feature refers to any KML element that is derived from Feature: Document, Folder, GroundOverlay, NetworkLink, Placemark, and ScreenOverlay.

4.2
Bounding Box
A *bounding box* is a volume that encloses a set of objects or data points.

4.3
Geometry
Geometry refers to any geometric element in KML: Point, Polygon, LinearRing, LineString, Model, MultiGeometry.

4.4
Overlay
Overlay refers to the elements derived from Overlay: GroundOverlay and Screen Overlay.

4.5
Placemark
A Placemark is a Feature with associated Geometry.

5 Conventions

5.1 Symbols (and abbreviated terms)

Some frequently used abbreviated terms:

COTS	Commercial Off The Shelf
KML	Keyhole Markup Language
ISO	International Organization for Standardization
OGC	Open Geospatial Consortium
UML	Unified Modeling Language
XML	eXtended Markup Language
1D	One Dimensional
2D	Two Dimensional
3D	Three Dimensional

5.2 UML Notation

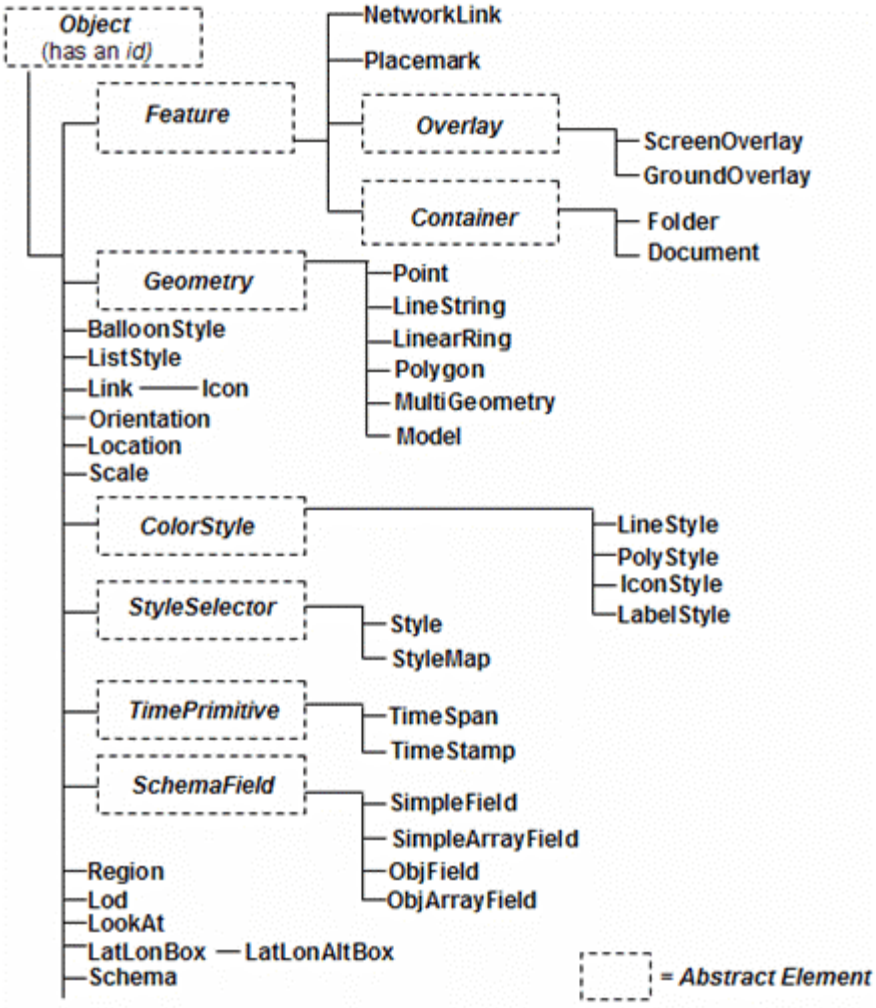
There is no UML associated with this candidate specification.

6 Overview of the KML Reference Model

Section 7 contains an alphabetical reference for all KML elements.

This section provides a class diagram of the KML element types. The class tree for KML elements is shown below. In this diagram, elements to the right on a particular branch in the tree are extensions of the elements to their left. For example, Placemark is a special kind of Feature. It contains all of the elements that belong to Feature, and it adds some elements that are specific to the Placemark element.

6.1 KML Class Tree



Note that abstract elements (shown in italics and in dotted boxes in the diagram) are not actually used in KML files. They are a useful way for a single element to serve as the programmatic foundation for multiple similar (but different) derived elements. Understanding this object-oriented hierarchy is also a good way for you to learn KML, since you can easily see groupings of related elements.

All elements derived from Object can have an id assigned to them. This id is used by the KML update mechanism (see <Update>) for files loaded with a NetworkLink. It is also used by shared styles (see <Style>). The id is a standard XML ID.

Because KML is an XML grammar and file format, tag names are case-sensitive and must appear exactly as shown here. If you're familiar with XML, you will also be interested in the KML 2.1 Schema. When you are editing KML text files, you can load this Schema into any XML editor and validate your KML code with it.

6.2 About this Reference

Each reference entry includes a Syntax section that lists the elements contained in the main element. This Syntax section is an informal listing and uses simple shorthand to summarize the elements, their default values (if any), and the type of values they contain. This section can be copied and used as a template for any non-abstract element in a KML file.

6.3 KML Field

KML uses common XML types such as *boolean, string, double, float,* and *int*. In addition, it defines a number of field element types. The following table lists some of the most commonly used types defined in KML and links to sample elements that use them:

Field Type	*Value*	*Example Use*
anglepos90	a value ≥0 and ≤90	See <tilt> in <LookAt>
angle90	a value ≥−90 and ≤90	See <latitude> in <Model>
angle180	a value ≥−180 and ≤180	See <longitude> in <Model>
angle360	a value ≥−360 and ≤360	See <heading>, <tilt>, and <roll> in <Orientation>
vec2	x=*double* xunits=*kml:unitsEnum* y=*double* yunits=*kml:unitsEnum*	See <hotSpot> in <IconStyle>, <ScreenOverlay>
color	hexBinary value: *aabbggrr*	See any element that extends *<ColorStyle>*
dateTime	*dateTime, date, gYearMonth, gYear*	See <TimeSpan> and <TimeStamp>
altitudeModeEnum	clampToGround, relativeToGround, absolute	See <LookAt> and <Region>
colorModeEnum	normal, random	See any element that extends *<ColorStyle>*

refreshModeEnum	onChange, onInterval, onExpire	See <Link>
styleStateEnum	normal, highlight	See <StyleMap>
unitsEnum	fraction, pixels, insetPixels	See <hotSpot> in <IconStyle>, <ScreenOverlay>
viewRefreshEnum	never, onRequest, onStop, onRegion	See <Link>

7 KML Reference Elements

The following section defines each of the elements available in KML.

7.1 *<BalloonStyle>*

7.1.1 Syntax

```
<BalloonStyle id="ID">
  <!-- specific to BalloonStyle -->
  <bgColor>ffffffff</bgColor>            <!-- kml:color -->
  <textColor>ff000000</textColor>        <!-- kml:color -->
  <text>...</text>                       <!-- string -->
</BalloonStyle>
```

7.1.2 Description

Specifies how the description balloon for placemarks is drawn. The <bgColor>, if specified, is used as the background color of the balloon. See <Feature> for a diagram illustrating how the default description balloon appears in Google Earth.

7.1.3 Elements specific to BalloonStyle

<bgColor> (default=ffffffff)

Background color of the balloon (optional). Color and opacity (alpha) values are expressed in hexadecimal notation. The range of values for any one color is 0 to 255 (00 to ff). The order of expression is *aabbggrr,* where *aa=alpha* (00 to ff); *bb=blue* (00 to ff); *gg=green* (00 to ff); *rr=red* (00 to ff). For alpha, 00 is fully transparent and ff is fully opaque. For example, if you want to apply a blue color with 50 percent opacity to an overlay, you would specify the following: <bgColor>7fff0000</bgColor>, where *alpha*=0x7f, *blue*=0xff, *green*=0x00, and *red*=0x00. **Note:** The use of the <color> element within <BalloonStyle> has been deprecated. Use <bgColor> instead.

<textColor>

Foreground color for text. The default is black (00000000).

<text>

Text displayed in the balloon. If no text is specified, Google Earth supplies one. You can add entities to the <text> tag using the following format to refer to a child element of Feature: **$[name]**, **$[description]**, **$[address]**, **$[id]**, **$[Snippet]**. Google Earth looks in the current Feature for the corresponding string entity and substitutes that information in the balloon. (Any entity that can logically be converted to a string can be referenced in this way.) To include *To here - From here* driving directions in the balloon, use the **$[geDirections]** tag.

For example, in the following KML excerpt, **$[name]** and **$[description]** fields will be replaced by the <name> and <description> fields found in the Feature elements that use this BalloonStyle:

<text>This is $[name], whose description is:
$[description]</text>

7.1.4 Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>BalloonStyle.kml</name>
  <open>1</open>
  <Style id="exampleBalloonStyle">
    <BalloonStyle>
      <!-- a background color for the balloon -->
      <bgColor>ffffffbb</bgColor>
      <!-- styling of the balloon text -->
      <text><![CDATA[
      <b><font color="#CC0000" size="+3">$[name]</font></b>
      <br/><br/>
      <font face="Courier">$[description]</font>
      <br/><br/>
      Extra text that will appear in the description balloon
      <br/><br/>
      <!-- insert the to/from hyperlinks -->
      $[geDirections]
      ]]></text>
    </BalloonStyle>
  </Style>
  <Placemark>
    <name>BalloonStyle</name>
    <description>An example of BalloonStyle</description>
    <styleUrl>#exampleBalloonStyle</styleUrl>
    <Point>
      <coordinates>-122.370533,37.823842,0</coordinates>
    </Point>
  </Placemark>
</Document>
</kml>
```

7.1.5 Extends

- *<ColorStyle>*

7.1.6 Contained by

- <Style>

7.2 <*ColorStyle*>

7.2.1 Syntax

```
<!-- abstract element; do not create -->
<!-- ColorStyle id="ID" -->          <!--
IconStyle,LabelStyle,LineStyle,PolyStyle -->
  <color>ffffffff</color>            <!-- kml:color -->
  <colorMode>normal</colorMode>      <!-- kml:colorModeEnum: normal or
random -->
<!-- /ColorStyle -->
```

7.2.2 Description

This is an abstract element and cannot be used directly in a KML file. It provides elements for specifying the color and color mode of extended style types.

7.2.3 Elements specific to ColorStyle

<color> (default=ffffffff)

Color and opacity (alpha) values are expressed in hexadecimal notation. The range of values for any one color is 0 to 255 (00 to ff). For alpha, 00 is fully transparent and ff is fully opaque. The order of expression is *aabbggrr,* where *aa=alpha* (00 to ff); *bb=blue* (00 to ff); *gg=green* (00 to ff); *rr=red* (00 to ff). For example, if you want to apply a blue color with 50 percent opacity to an overlay, you would specify the following: <color>7fff0000</color>, where *alpha*=0x7f, *blue*=0xff, *green*=0x00, and *red*=0x00.

<colorMode> (default=normal)

Values for <colorMode> are **normal** (no effect) and **random**. A value of **random** applies a random linear scale to the base <color> as follows.

- To achieve a truly random selection of colors, specify a base <color> of white (ffffffff).
- If you specify a single color component (for example, a value of ff0000ff for *red*), random color values for that one component (red) will be selected. In this case, the values would range from 00 (*black*) to ff (*full red*).
- If you specify values for two or for all three color components, a random linear scale is applied to each color component, with results ranging from black to the maximum values specified for each component.

The opacity of a color comes from the alpha component of <color> and is never randomized.

7.2.4 Examples

None.

7.2.5 Extends

- *<Object>*

7.2.6 Extended by

- <IconStyle>
- <LabelStyle>
- <LineStyle>
- <PolyStyle>

7.3 <Container>

7.3.1 Syntax

```
<!-- abstract element; do not create -->
<!-- Container id="ID" -->                <!-- Document,Folder -->
  <!-- inherited from Feature element -->
  <name>...</name>                        <!-- string -->
  <visibility>1</visibility>              <!-- boolean -->
  <open>1</open>                          <!-- boolean -->
  <address>...</address>                  <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                   <!-- string -->
  <phoneNumber>...</phoneNumber>          <!-- string -->
  <Snippet maxLines="2">...</Snippet>     <!-- string -->
  <description>...</description>          <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>                <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- specific to Container -->
  <!-- 0 or more Features -->
<!-- /Container -->
```

7.3.2 Description

This is an abstract element and cannot be used directly in a KML file. A Container element holds one or more Features and allows the creation of nested hierarchies.

7.3.3 Elements specific to Container

None.

7.3.4 Examples

None.

7.3.5 Extends

- *<Feature>*

7.3.6 Extended by

- <Document>
- <Folder>

7.4 <Document>

7.4.1 Syntax

```
<Document id="ID">
<!-- inherited from Feature element -->
  <name>...</name>                       <!-- string -->
  <visibility>1</visibility>             <!-- boolean -->
  <open>1</open>                         <!-- boolean -->
  <address>...</address>                 <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                  <!-- string -->
  <phoneNumber>...</phoneNumber>         <!-- string -->
  <Snippet maxLines="2">...</Snippet>    <!-- string -->
  <description>...</description>         <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>               <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- specific to Document -->
  <!-- 0 or more Schema elements -->
  <!-- 0 or more Feature elements -->
</Document>
```

7.4.2 Description

A Document is a container for features, styles, and schemas. This element is required if your KML file uses schemas or shared styles. It is recommended that all Styles be defined in a Document, each with an **id**, and then later referenced by a <styleUrl> for a given Feature or StyleMap.

7.4.3 Elements specific to Document

None.

7.4.4 Examples

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>Document.kml</name>
  <open>1</open>
  <Style id="exampleStyleDocument">
    <LabelStyle>
      <color>#ff0000cc</color>
    </LabelStyle>
  </Style>
  <Placemark>
    <name>Document Feature 1</name>
    <styleUrl>#exampleStyleDocument</styleUrl>
```

```
    <Point>
      <coordinates>-122.371,37.816,0</coordinates>
    </Point>
  </Placemark>
  <Placemark>
    <name>Document Feature 2</name>
    <styleUrl>#exampleStyleDocument</styleUrl>
    <Point>
      <coordinates>-122.370,37.817,0</coordinates>
    </Point>
  </Placemark>
</Document>
</kml>
```

7.4.5 Extends

- <*Container*>

7.4.6 Contains

- 0 or more elements derived from <*Feature*>
- 0 or more elements derived from <*StyleSelector*>
- 0 or more elements derived from <Schema>

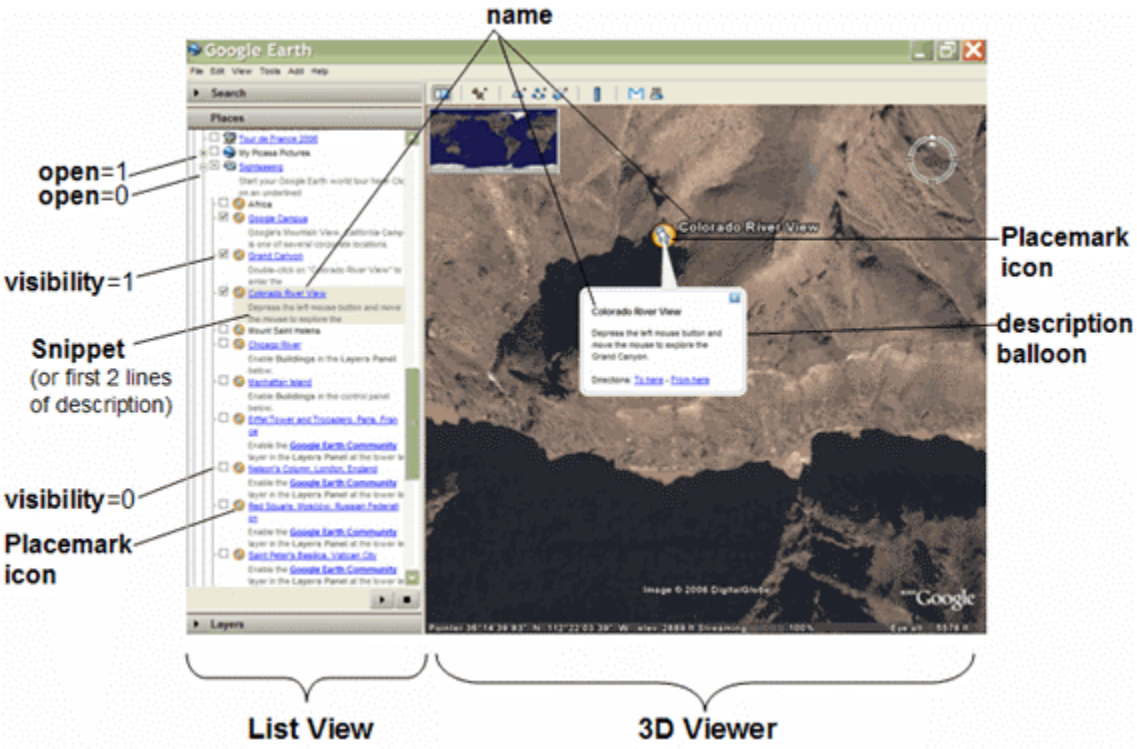
7.5 *<Feature >*

7.5.1 Syntax

```
<!-- abstract element; do not create -->
<!-- Feature id="ID" -->                <!-- Document,Folder,
                                             NetworkLink,Placemark,
                                             GroundOverlay,ScreenOverlay -
->
  <name>...</name>                      <!-- string -->
  <visibility>1</visibility>            <!-- boolean -->
  <open>1</open>                        <!-- boolean -->
  <address>...</address>                <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                 <!-- string -->
  <phoneNumber>...</phoneNumber>        <!-- string -->
  <Snippet maxLines="2">...</Snippet>   <!-- string -->
  <description>...</description>        <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>              <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>              <!-- user-defined data -->
<-- /Feature -->
```

7.5.2 Description

This is an abstract element and cannot be used directly in a KML file. The following diagram shows how some of a Feature's elements appear in Google Earth.



7.5.3 Elements specific to Feature

<name>

User-defined text displayed in the 3D viewer as the label for the object (for example, for a Placemark, Folder, or NetworkLink).

<visibility>

Boolean value (default=1). Specifies whether the feature is drawn in the 3D viewer when it is initially loaded. In order for a feature to be visible, the <visibility> tag of all its ancestors must also be set to 1. In the Google Earth List View, each Feature has a checkbox that allows the user to control visibility of the Feature.

<open>

Boolean value (default=1). Specifies whether a Folder appears closed or open when first loaded into the Places panel. 0=collapsed, 1=expanded. See also <ListStyle>.

<address>

A string value representing an unstructured address written as a standard street, city, state address, and/or as a postal code. You can use the <address> tag to specify the location of a point instead of using latitude and longitude coordinates. (However, if a <Point> is provided, it takes precedence over the <address>.) To find out which locales are supported for this tag in Google Earth, go to the Google Maps Help.

<AddressDetails>

A structured address, formatted as xAL, or eXtensible Address Language, an international standard for address formatting. <AddressDetails> is used by KML for geocoding in Google Maps only. For details, see the Google Maps API documentation. Currently, Google Earth does not use this element; use <address> instead.

<phoneNumber>

A string value representing a telephone number. This element is used by Google Maps Mobile only. The industry standard for Java-enabled cellular phones is RFC2806. For more information, see http://www.google.com/url?sa=D&q=http%3A%2F%2Fwww.ietf.org%2Frfc%2Frfc2806.txt.

<Snippet maxLines="2" **>**

A short description of the feature. In Google Earth, this description is displayed in the Places panel under the name of the feature. If a Snippet is not supplied, the first two lines of the <description> are used. In Google Earth, if a Placemark contains both a description and a Snippet, the <Snippet> appears beneath the Placemark in the Places panel, and the <description> appears in the Placemark's description balloon. This tag does not support HTML markup. <Snippet> has a **maxLines** attribute, an integer that specifies the maximum number of lines to display. Default for **maxLines** is 2.

<description>

User-supplied text that appears in the description balloon when the user clicks on either the feature name in the Places panel or the Placemark icon in the 3D viewer. This text also appears beneath the feature name in the Places panel if no <Snippet> tag is specified for the feature.

The <description> element supports plain text as well as a subset of HTML formatting elements, including tables (see KML example below). It does not support other web-based technology, such as dynamic page markup (PHP, JSP, ASP), scripting languages (VBScript, Javascript), nor application languages (Java, Python).

If your description contains no HTML markup, Google Earth attempts to format it, replacing newlines with
 and wrapping URLs with anchor tags. A valid URL string for the World Wide Web is automatically converted to a hyperlink to that URL (e.g., *http://www.google.com*). Consequently, you do not need to surround a URL with the tags in order to achieve a simple link.

When using HTML to create a hyperlink around a specific word, or when including images in the HTML, you must use HTML entity references or the CDATA element to escape angle brackets, apostrophes, and other special characters. The CDATA element tells the XML parser to ignore special characters used within the brackets. This element takes the form of:

```
<![CDATA[ special characters here ]]>
```

If you prefer not to use the CDATA element, you can use entity references to replace all the special characters.

```
<description><![CDATA[This is an image
<img src="C:/Documents and Settings/user/My Documents/icon.jpg">
and we have a link http://www.google.com.]]></description>
```

<LookAt>

Defines a camera viewpoint associated with any element derived from Feature. See <LookAt>.

<styleUrl>

URI (a URI equals [URL]#ID) of a <Style> or <StyleMap> defined in a Document. If the style is in the same file, use a # reference. If the style is defined in an external file, use a full URL along with # referencing. Examples are

```
<styleUrl>#myIconStyleID</styleUrl>
<styleUrl>http://someserver.com/somestylefile.xml#restaurant</styleUrl>
```

<StyleSelector>

One or more Styles and StyleMaps can be defined to customize the appearance of any element derived from Feature or of the Geometry in a Placemark. (See <BalloonStyle>, <ListStyle>, <StyleSelector>, and the styles derived from <ColorStyle>.) A style defined within a Feature is called an "inline style" and applies only to the Feature that contains it. A style defined as the child of a <Document> is called a "shared style." A shared style must have an *id* defined for it. This id is referenced by one or more Features within the <Document>. In cases where a style element is defined both in a shared style and in an inline style for a Feature—that is, a Folder, GroundOverlay, NetworkLink, Placemark, or ScreenOverlay—the value for the Feature's inline style takes precedence over the value for the shared style.

<Region>

Features and geometry associated with a Region are drawn only when the Region is active. See <Region>.

<Metadata>

User-defined data. This element is used to annotate elements that extend *Feature*. Google Earth does not process anything inside the Metadata element, but it does preserve the element as part of the file. If the file is saved out of Google Earth, it will preserve the Metadata element.

7.5.4 Example of a Feature: Placemark with Lengthy Sample Description

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Placemark>
  <name>Feature.kml</name>
  <Snippet maxLines="4">
```

```
    The snippet is a way of
    providing an alternative
    description that will be
    shown in the List view.
  </Snippet>
  <description>
    <![CDATA[
      Styles: <i>Italics</i>, <b>Bold</b>, <u>Underlined</u>,
      <s>Strike Out</s>, subscript<sub>subscript</sub>,
      superscript<sup>superscript</sup>,
      <big>Big</big>, <small>Small</small>, <tt>Typewriter</tt>,
      <em>Emphasized</em>, <strong>Strong</strong>, <code>Code</code>
      <hr />
      Fonts:
      <font color="red">red by name</font>,
      <font color="#408010">leaf green by hexadecimal RGB</font>,
      <font size=1>size 1</font>, <font size=2>size 2</font>,
      <font size=3>size 3</font>, <font size=4>size 4</font>,
      <font size=5>size 5</font>, <font size=6>size 6</font>,
      <font size=7>size 7</font>,
      <font face=times>Times</font>,
      <font face=verdana>Verdana</font>,
      <font face=arial>Arial</font>
      <br/>
      <hr />
      Links:
      <a href="http://doc.trolltech.com/3.3/qstylesheet.html">
      QT Rich Text Rendering
      </a>
      <br />
      <hr />
      Alignment:
      <br />
      <p align=left>left</p><p align=center>center</p>
      <p align=right>right</p>
      <hr />
      Ordered Lists:
      <br />
      <ol><li>First</li><li>Second</li><li>Third</li></ol>
      <ol type="a"><li>First</li><li>Second</li><li>Third</li></ol>
      <ol type="A"><li>First</li><li>Second</li><li>Third</li></ol>
      <hr />
      Unordered Lists:
      <br />
      <ul><li>A</li><li>B</li><li>C</li></ul>
      <ul type="circle"><li>A</li><li>B</li><li>C</li></ul>
      <ul type="square"><li>A</li><li>B</li><li>C</li></ul>
      <hr />
      Definitions:
      <br />
      <dl>
      <dt>Scrumpy</dt>
      <dd>Hard English cider from the west country</dd>
      <dt>Pentanque</dt>
      <dd>A form of boules where the goal is to throw metal ball as
      close as possible to a jack</dd>
```

```
      </dl>
      <hr />
      Block Quote:
      <br />
      <blockquote>
      We shall not cease from exploration<br />
      And the end of all our exploring<br />
      Will be to arrive where we started<br />
      And know the place for the first time
      </blockquote>
      <br />
      <hr />
      Centered:
      <br />
      <center>See, I have a Rhyme assisting<br />
      my feeble brain,<br />
      its tasks oft-times resisting!</center>
      <hr />
      Headings:
      <br />
      <h1>Header 1</h1>
      <h2>Header 2</h2>
      <h3>Header 3</h3>
      <h3>Header 4</h4>
      <h3>Header 5</h5>
      <hr />
      Images:
      <br />
      <img src="http://earth.google.com/images/googleearth.gif" />
      <br />
      <i>Scaled image</i>
      <br />
      <img src="http://earth.google.com/images/googleearth.gif"
      width="100" />
      <br />
      <hr />
      Tables:
      <table border="1" padding="3" width="300">
      <tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
      <tr><td>a</td><td>b</td><td>c</td><td>d</td><td>e</td></tr>
      </table>
    ]]>
  </description>
  <Point>
    <coordinates>-122.378927,37.826793,0</coordinates>
  </Point>
</Placemark>
</kml>
```

7.5.5 Extends

- *<Object>*

7.5.6 Extended by

- *<Container>*
- <Overlay>
- <Placemark>

7.6 <Folder >

7.6.1 Syntax

```
<Folder id="ID">
  <!-- inherited from Feature element -->
  <name>...</name>                        <!-- string -->
  <visibility>1</visibility>              <!-- boolean -->
  <open>1</open>                          <!-- boolean -->
  <address>...</address>                  <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                   <!-- string -->
  <phoneNumber>...</phoneNumber>          <!-- string -->
  <Snippet maxLines="2">...</Snippet>     <!-- string -->
  <description>...</description>          <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>                <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- specific to Folder -->
  <!-- 0 or more Feature elements -->
</Folder>
```

7.6.2 Description

A Folder is used to arrange other Features hierarchically (Folders, Placemarks, NetworkLinks, or Overlays). A Feature is visible only if it and all its ancestors are visible.

7.6.3 Elements specific to Folder

None.

7.6.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Folder>
  <name>Folder.kml</name>
  <open>1</open>
  <description>
    A folder is a container that can hold multiple other objects
  </description>
  <Placemark>
    <name>Folder object 1 (Placemark)</name>
    <Point>
      <coordinates>-122.377588,37.830266,0</coordinates>
    </Point>
```

```
  </Placemark>
  <Placemark>
    <name>Folder object 2 (Polygon)</name>
    <Polygon>
      <outerBoundaryIs>
        <LinearRing>
          <coordinates>
            -122.377830,37.830445,0
            -122.377576,37.830631,0
            -122.377840,37.830642,0
            -122.377830,37.830445,0
          </coordinates>
        </LinearRing>
      </outerBoundaryIs>
    </Polygon>
  </Placemark>
  <Placemark>
    <name>Folder object 3 (Path)</name>
    <LineString>
      <tessellate>1</tessellate>
      <coordinates>
        -122.378009,37.830128,0 -122.377885,37.830379,0
      </coordinates>
    </LineString>
  </Placemark>
</Folder>
</kml>
```

7.6.5 Extends

- *<Container>*

7.6.6 Contains

- Any element derived from *<Feature>*

7.7 *<Geometry >*

7.7.1 Syntax

Syntax

```
<!-- abstract element; do not create -->
<!-- Geometry id="ID" -->                    <!--
Point,LineString,LinearRing,
                                                Polygon,MultiGeometry,Model
-->
<!-- /Geometry -->
```

7.7.2 Description

This is an abstract element and cannot be used directly in a KML file. It provides a placeholder object for all derived Geometry objects.

<extrude>

Boolean value (default=0). Specifies whether to connect the geometric primitive (icon, line, polygon) to the ground. Extrusion requires that the geometry's <altitudeMode> be either **relativeToGround** or **absolute** and that within the <coordinates> element, the *altitude* component be greater than 0 (that is, in the air).

<tessellate>

Boolean value (default=0). Specifies whether to allow lines and paths to follow the terrain. This specification applies only to LineStrings (paths) and LinearRings (polygons) that have an <altitudeMode> of **clampToGround**. Very long lines should enable tessellation so that they follow the curvature of the earth (otherwise, they may go underground and be hidden).

<altitudeMode>

Specifies how *altitude* components in the <coordinates> element are interpreted. Possible values are

- **clampToGround** - (default) Indicates to ignore an altitude specification (for example, in the <coordinates> tag).
- **relativeToGround** - Sets the altitude of the element relative to the actual ground elevation of a particular location. For example, if the ground elevation of a location is exactly at sea level and the altitude for a point is set to 9 meters, then the elevation for the icon of a point placemark elevation is 9 meters with this mode. However, if the same coordinate is set over a location where the ground

elevation is 10 meters above sea level, then the elevation of the coordinate is 19 meters. A typical use of this mode is for placing telephone poles or a ski lift.

- **absolute** - Sets the altitude of the coordinate relative to sea level, regardless of the actual elevation of the terrain beneath the element. For example, if you set the altitude of a coordinate to 10 meters with an *absolute* altitude mode, the icon of a point placemark will appear to be at ground level if the terrain beneath is also 10 meters above sea level. If the terrain is 3 meters above sea level, the placemark will appear elevated above the terrain by 7 meters. A typical use of this mode is for aircraft placement.

7.7.3 Elements specific to Geometry

None.

7.7.4 Examples

None

7.7.5 Extends

- *<Object>*

7.7.6 Extended by

- <Point>
- <LineString>
- <LinearRing>
- <Polygon>
- <MultiGeometry>

7.8 <GroundOverlay >

7.8.1 Syntax

```
<GroundOverlay id="ID">
  <!-- inherited from Feature element -->
  <name>...</name>                        <!-- string -->
  <visibility>1</visibility>              <!-- boolean -->
  <open>1</open>                          <!-- boolean -->
  <address>...</address>                  <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                   <!-- string -->
  <phoneNumber>...</phoneNumber>          <!-- string -->
  <Snippet maxLines="2">...</Snippet>     <!-- string -->
  <description>...</description>          <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>                <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- inherited from Overlay element -->
  <color>ffffffff</color>                    <!-- kml:color -->
  <drawOrder>0</drawOrder>                   <!-- int -->
  <Icon>...</Icon>

  <!-- specific to GroundOverlay -->
  <altitude>0</altitude>                     <!-- double -->
  <altitudeMode>clampToGround</altitudeMode>
     <!-- kml:altitudeModeEnum: clampToGround or absolute -->
  <LatLonBox>
    <north>...</north>                       <! kml:angle90 -->
    <south>...</south>                       <! kml:angle90 -->
    <east>...</east>                         <! kml:angle180 -->
    <west>...</west>                         <! kml:angle180 -->
    <rotation>0</rotation>                   <! kml:angle180 -->
  </LatLonBox>
</GroundOverlay>
```

7.8.2 Description

This element draws an image overlay draped onto the terrain.

7.8.3 Elements specific to GroundOverlay

<altitude>

Specifies the distance above the earth's surface, in meters, and is interpreted according to <altitudeMode>.

<altitudeMode>

Specifies how the <altitude>is interpreted. Possible values are

- **clampToGround** - (default) Indicates to ignore the altitude specification and drape the overlay over the terrain.
- **absolute** - Sets the altitude of the overlay relative to sea level, regardless of the actual elevation of the terrain beneath the element. For example, if you set the altitude of an overlay to 10 meters with an absolute altitude mode, the overlay will appear to be at ground level if the terrain beneath is also 10 meters above sea level. If the terrain is 3 meters above sea level, the overlay will appear elevated above the terrain by 7 meters.

<LatLonBox> *(required)*

Specifies where the top, bottom, right, and left sides of a bounding box for the ground overlay are aligned. The <north>, <south>, <east>, and <west> elements are required.

- **<north>** *(required)* Specifies the latitude of the north edge of the bounding box, in decimal degrees from 0 to ±90.
- **<south>** *(required)* Specifies the latitude of the south edge of the bounding box, in decimal degrees from 0 to ±90.
- **<east>** *(required)* Specifies the longitude of the east edge of the bounding box, in decimal degrees from 0 to ±180. (For overlays that overlap the meridian of 180° longitude, values can extend beyond that range.)
- **<west>** *(required)* Specifies the longitude of the west edge of the bounding box, in decimal degrees from 0 to ±180. (For overlays that overlap the meridian of 180° longitude, values can extend beyond that range.)
- **<rotation>** *(optional)* specifies a rotation of the overlay about its center, in degrees. Values can be ±180. The default is 0 (north). Rotations are specified in a clockwise direction.

```
<LatLonBox>
   <north>48.25475939255556</north>
   <south>48.25207367852141</south>
   <east>-90.86591508839973</east>
   <west>-90.8714285289695</west>
   <rotation>39.37878630116985</rotation>
</LatLonBox>
```

7.8.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<GroundOverlay>
```

```
   <name>GroundOverlay.kml</name>
   <color>7fffffff</color>
   <drawOrder>1</drawOrder>
   <Icon>
      <href>http://www.google.com/intl/en/images/logo.gif</href>
      <refreshMode>onInterval</refreshMode>
      <refreshInterval>86400</refreshInterval>
      <viewBoundScale>0.75</viewBoundScale>
   </Icon>
   <LatLonBox>
      <north>37.83234</north>
      <south>37.832122</south>
      <east>-122.373033</east>
      <west>-122.373724</west>
      <rotation>45</rotation>
   </LatLonBox>
</GroundOverlay>
</kml>
```

7.8.5 Extends

- *<Feature>*
- *<Overlay>*

7.8.6 Contained by

- <Document>
- <Folder>

7.9 <Icon>

7.9.1 Syntax

```
<Icon id="ID">
  <!-- specific to Icon -->
  <href>...</href>                      <!-- anyURI -->
  <refreshMode>onChange</refreshMode>
    <!-- kml:refreshModeEnum: onChange, OnInterval, or OnExpire -->
  <refreshInterval>4</refreshInterval>  <!-- float -->
  <viewRefreshMode>never</viewRefreshMode>
    <!-- kml:viewRefreshModeEnum: never, onStop, onRequest, onRegion -->
  <viewRefreshTime>4</viewRefreshTime>  <!-- float -->
  <viewBoundScale>1</viewBoundScale>    <!-- float -->
  <viewFormat>...</viewFormat>          <!-- string -->
  <httpQuery>...</httpQuery>            <!-- string -->
  </Icon>
```

7.9.2 Description

Defines an image associated with an Icon style or overlay. <Icon> has the same child elements as <Link>. The required <href> child element defines the location of the image to be used as the overlay or as the icon for the placemark. This location can either be on a local file system or a remote web server.

```
<Icon>
  <href>C:/Documents and Settings/All Users/Documents/My Pictures/Sample
Pictures/Sunset.jpg</href>
</Icon>
```

7.9.3 Elements specific to Icon

<href>

An HTTP address or a local file specification used to load an icon.

<x>, <y>, <w>, <h>

Use of these elements within <Icon> has been deprecated.

<refreshMode>

For a description of <refreshMode> and the other elements listed below, see <Link>.

<refreshInterval>

<viewRefreshMode>

http://earth.google.com/kml/kml_tags_21.html - refreshmode**<viewRefreshTime>**

<viewBoundScale>

<viewFormat>

<httpQuery>

http://earth.google.com/kml/kml_tags_21.html - viewrefreshmode

http://earth.google.com/kml/kml_tags_21.html - viewrefreshtime

7.9.4 Examples

None.

7.9.5 Extends

None.

7.9.6 Contained by

- <GroundOverlay>
- <ScreenOverlay>
- <IconStyle>

7.10 <IconStyle>

7.10.1 Syntax

```
<IconStyle id="ID">
  <!-- inherited from ColorStyle -->
  <color>ffffffff</color>            <!-- kml:color -->
  <colorMode>normal</colorMode>      <!-- kml:colorModeEnum:normal or
random -->

  <!-- specific to IconStyle -->
  <scale>1</scale>                   <!-- float -->
  <heading>0</heading>               <!-- float -->
  <Icon>
    <href>...</href>
  </Icon>
  <hotSpot x="0.5"  y="0.5"
    xunits="fraction" yunits="fraction"/>    <!-- kml:vec2Type -->
</IconStyle>
```

7.10.2 Description

Specifies how icons for point Placemarks are drawn, both in the Places panel and in the 3D viewer of Google Earth. The <Icon> element specifies the icon image. The <scale> element specifies the *x, y* scaling of the icon.

7.10.3 Elements specific to IconStyle

<heading>

Compass direction, in degrees. Default=0 (North). (See diagram.) Values range from 0 to ±180 degrees.

<scale>

Resizes the icon (default=1).

Note: The <geomScale> tag has been deprecated. Use <scale> instead.

<Icon>

A custom Icon. In <IconStyle>, the only child element of <Icon> is <href>:

- **<href>**: An HTTP address or a local file specification used to load an icon.

<hotSpot x="0.5" **y="0.**5" **xunits="**fraction" **yunits="**fraction"**>**

Specifies the position within the Icon that is "anchored" to the <Point> specified in the Placemark. The *x* and *y* values can be specified in three different ways: as *pixels*

(**"pixels"**), as *fractions* of the icon (**"fraction"***)*, or as *inset pixels* (**"insetPixels"**), which is an offset in pixels from the upper right corner of the icon. The *x* and *y* positions can be specified in different ways—for example, *x* can be in pixels and *y* can be a fraction. The origin of the coordinate system is in the lower left corner of the icon.

- **x** - Either the number of pixels, a fractional component of the icon, or a pixel inset indicating the *x* component of a point on the icon.
- **y** - Either the number of pixels, a fractional component of the icon, or a pixel inset indicating the *y* component of a point on the icon.
- **xunits** - Units in which the *x* value is specified. Default=**fraction**. A value of **fraction** indicates the *x* value is a fraction of the icon. A value of **pixels** indicates the *x* value in pixels. A value of **insetPixels** indicates the indent from the right edge of the icon.
- **yunits** - Units in which the *y* value is specified. Default=**fraction**. A value of **fraction** indicates the *y* value is a fraction of the icon. A value of **pixels** indicates the *y* value in pixels. A value of **insetPixels** indicates the indent from the top edge of the icon.

7.10.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
   <Style id="randomColorIcon">
      <IconStyle>
         <color>ff00ff00</color>
         <colorMode>random</colorMode>
         <scale>1.1</scale>
         <Icon>

<href>http://maps.google.com/mapfiles/kml/pal3/icon21.png</href>
         </Icon>
      </IconStyle>
   </Style>
   <Placemark>
      <name>IconStyle.kml</name>
      <styleUrl>#randomColorIcon</styleUrl>
      <Point>
         <coordinates>-122.36868,37.831145,0</coordinates>
      </Point>
   </Placemark>
</Document>
</kml>
```

7.10.5 Extends

- *<ColorStyle>*

7.10.6 Contained by

- <Style>

7.10.7 Contains

- <Icon>

7.11 <kml >

7.11.1 Syntax

<kml xmlns="http://earth.google.com/kml/2.1"> ... </kml>

7.11.2 Description

The root element of a KML document. This element is required. It follows the xml declaration at the beginning of the file.

7.11.3 Elements specific to kml

N.A.

7.11.4 Examples

N.A.

7.11.5 Extends

N.A.

7.11.6 Contained by

N.A.

7.12 <LabelStyle>

7.12.1 Syntax

```
<LabelStyle id="ID">
  <!-- inherited from ColorStyle -->
  <color>ffffffff</color>            <!-- kml:color -->
  <colorMode>normal</colorMode>      <!-- kml:colorModeEnum: normal or
random -->

  <!-- specific to LabelStyle -->
  <scale>1</scale>                   <!-- float -->
</LabelStyle>
```

7.12.2 Description

Specifies how the <name> of a Feature is drawn in the 3D viewer. A custom color, color mode, and scale for the label (name) can be specified.

Note: The <labelColor> tag is deprecated. Use <LabelStyle> instead.

7.12.3 Elements specific to LableStyle

<scale>

Resizes the label (default=1).

7.12.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
   <Style id="randomLabelColor">
      <LabelStyle>
         <color>ff0000cc</color>
         <colorMode>random</colorMode>
         <scale>1.5</scale>
      </LabelStyle>
   </Style>
   <Placemark>
      <name>LabelStyle.kml</name>
      <styleUrl>#randomLabelColor</styleUrl>
      <Point>
         <coordinates>-122.367375,37.829192,0</coordinates>
      </Point>
   </Placemark>
</Document>
</kml>
```

7.12.5 Extends

- *<ColorStyle>*

7.12.6 Contained by

- <Style>

7.13 <LinearRing>

7.13.1 Syntax

```
<LinearRing id="ID">
  <!-- specific to LinearRing -->
  <extrude>0</extrude>                        <!-- boolean -->
  <tessellate>0</tessellate>                  <!-- boolean -->
  <altitudeMode>clampToGround</altitudeMode>
    <!-- kml:altitudeModeEnum: clampToGround, relativeToGround, or
absolute -->
  <coordinates>...</coordinates>              <!-- lon,lat[,alt] tuples
-->
</LinearRing>
```

7.13.2 Description

Defines a closed line string, typically the outer boundary of a Polygon. Optionally, a LinearRing can also be used as the inner boundary of a Polygon to create holes in the Polygon. A Polygon can contain multiple <LinearRing> elements used as inner boundaries.

7.13.3 Elements specific to LinearRing

<extrude>

Boolean value (default=0). Specifies whether to connect the geometric primitive (icon, line, polygon) to the ground. Extrusion requires that the geometry's <altitudeMode> be either **relativeToGround** or **absolute** and that within the <coordinates> element, the *altitude* component be greater than 0 (that is, in the air).

<tessellate>

Boolean value (default=0). Specifies whether to allow lines and paths to follow the terrain. This specification applies only to LineStrings (paths) and LinearRings (polygons) that have an <altitudeMode> of **clampToGround**. Very long lines should enable tessellation so that they follow the curvature of the earth (otherwise, they may go underground and be hidden).

<altitudeMode>

Specifies how *altitude* components in the <coordinates> element are interpreted. Possible values are

- **clampToGround -** (default) Indicates to ignore an altitude specification (for

example, in the <coordinates> tag).

- **relativeToGround** - Sets the altitude of the element relative to the actual ground elevation of a particular location. For example, if the ground elevation of a location is exactly at sea level and the altitude for a point is set to 9 meters, then the elevation for the icon of a point placemark elevation is 9 meters with this mode. However, if the same coordinate is set over a location where the ground elevation is 10 meters above sea level, then the elevation of the coordinate is 19 meters. A typical use of this mode is for placing telephone poles or a ski lift.

- **absolute** - Sets the altitude of the coordinate relative to sea level, regardless of the actual elevation of the terrain beneath the element. For example, if you set the altitude of a coordinate to 10 meters with an *absolute* altitude mode, the icon of a point placemark will appear to be at ground level if the terrain beneath is also 10 meters above sea level. If the terrain is 3 meters above sea level, the placemark will appear elevated above the terrain by 7 meters. A typical use of this mode is for aircraft placement.

<coordinates> *(required)*

Four or more tuples, each consisting of floating point values for *longitude*, *latitude*, and *altitude*. The *altitude* component is optional. Do not include spaces within a tuple. The last coordinate must be the same as the first coordinate. Coordinates are expressed in decimal degrees only.

7.13.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Placemark>
  <name>LinearRing.kml</name>
  <Polygon>
    <outerBoundaryIs>
      <LinearRing>
        <coordinates>
          -122.365662,37.826988,0
          -122.365202,37.826302,0
          -122.364581,37.82655,0
          -122.365038,37.827237,0
          -122.365662,37.826988,0
        </coordinates>
      </LinearRing>
    </outerBoundaryIs>
  </Polygon>
</Placemark>
</kml>
```

7.13.5 Extends

- <*Geometry*>

7.13.6 Contained by

- <MultiGeometry>
- <Placemark>
- <innerBoundaryIs>
- <outerBoundaryIs>

7.14 <LineString>

7.14.1 Syntax

```
<LineString id="ID">
  <!-- specific to LineString -->
  <extrude>0</extrude>                    <!-- boolean -->
  <tessellate>0</tessellate>              <!-- boolean -->
  <altitudeMode>clampToGround</altitudeMode>
    <!-- kml:altitudeModeEnum: clampToGround, relativeToGround, or
absolute -->
  <coordinates>...</coordinates>          <!-- lon,lat[,alt] -->
</LineString>
```

7.14.2 Description

Defines a connected set of line segments. Use <LineStyle> to specify the color, color mode, and width of the line. When a LineString is extruded, the line is extended to the ground, forming a polygon that looks somewhat like a wall. For extruded LineStrings, the line itself uses the current LineStyle, and the extrusion uses the current PolyStyle. See the *KML Tutorial* for examples of LineStrings (or *paths*).

7.14.3 Elements specific to LineString

<extrude>

Boolean value (default=0). Specifies whether to connect the geometric primitive (icon, line, polygon) to the ground. Extrusion requires that the geometry's <altitudeMode> be either **relativeToGround** or **absolute** and that within the <coordinates> element, the *altitude* component be greater than 0 (that is, in the air).

<tessellate>

Boolean value (default=0). Specifies whether to allow lines and paths to follow the terrain. This specification applies only to LineStrings (paths) and LinearRings (polygons) that have an <altitudeMode> of **clampToGround**. Very long lines should enable tessellation so that they follow the curvature of the earth (otherwise, they may go underground and be hidden).

<altitudeMode>

Specifies how *altitude* components in the <coordinates> element are interpreted. Possible values are

- **clampToGround** - (default) Indicates to ignore an altitude specification (for example, in the <coordinates> tag).
- **relativeToGround** - Sets the altitude of the element relative to the actual ground

elevation of a particular location. For example, if the ground elevation of a location is exactly at sea level and the altitude for a point is set to 9 meters, then the elevation for the icon of a point placemark elevation is 9 meters with this mode. However, if the same coordinate is set over a location where the ground elevation is 10 meters above sea level, then the elevation of the coordinate is 19 meters. A typical use of this mode is for placing telephone poles or a ski lift.

- **absolute** - Sets the altitude of the coordinate relative to sea level, regardless of the actual elevation of the terrain beneath the element. For example, if you set the altitude of a coordinate to 10 meters with an *absolute* altitude mode, the icon of a point placemark will appear to be at ground level if the terrain beneath is also 10 meters above sea level. If the terrain is 3 meters above sea level, the placemark will appear elevated above the terrain by 7 meters. A typical use of this mode is for aircraft placement.

<coordinates> *(required)*

Two or more coordinate tuples, each consisting of floating point values for *longitude*, *latitude*, and *altitude*. The *altitude* component is optional. Insert a space between tuples. Do not include spaces within a tuple.

7.14.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>LineString.kml</name>
  <open>1</open>
  <LookAt>
    <longitude>-122.36415</longitude>
    <latitude>37.824553</latitude>
    <altitude>0</altitude>
    <range>150</range>
    <tilt>50</tilt>
    <heading>0</heading>
  </LookAt>
  <Placemark>
    <name>unextruded</name>
    <LineString>
      <extrude>1</extrude>
      <tessellate>1</tessellate>
      <coordinates>
        -122.364383,37.824664,0 -122.364152,37.824322,0
      </coordinates>
    </LineString>
  </Placemark>
  <Placemark>
    <name>extruded</name>
    <LineString>
      <extrude>1</extrude>
      <tessellate>1</tessellate>
```

```
      <altitudeMode>relativeToGround</altitudeMode>
      <coordinates>
        -122.364167,37.824787,50 -122.363917,37.824423,50
      </coordinates>
    </LineString>
  </Placemark>
</Document>
</kml>
```

7.14.5 Extends

- *<Geometry>*

7.14.6 Contained by

- <MultiGeometry>
- <Placemark>

7.15 <LineStyle>

7.15.1 Syntax

```
<LineStyle id="ID">
  <!-- inherited from ColorStyle -->
  <color>ffffffff</color>           <!-- kml:color -->
  <colorMode>normal</colorMode>     <!-- colorModeEnum: normal or random -->

  <!-- specific to LineStyle -->
  <width>1</width>                  <!-- float -->
 </LineStyle>
```

7.15.2 Description

Specifies the drawing style (color, color mode, and line width) for all line geometry. Line geometry includes the outlines of outlined polygons and the extruded "tether" of Placemark icons (if extrusion is enabled).

7.15.3 Elements specific to LineStyle

<width> (default=1)

Width of the line, in pixels.

7.15.4 Example

The following example shows a 50 percent opaque red line with a width of 4 pixels.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>LineStyle.kml</name>
  <open>1</open>
  <Style id="linestyleExample">
    <LineStyle>
      <color>7f0000ff</color>
      <width>4</width>
    </LineStyle>
  </Style>
  <Placemark>
    <name>LineStyle Example</name>
    <styleUrl>#linestyleExample</styleUrl>
    <LineString>
      <extrude>1</extrude>
      <tessellate>1</tessellate>
      <coordinates>
        -122.364383,37.824664,0 -122.364152,37.824322,0
      </coordinates>
```

```
    </LineString>
  </Placemark>
</Document>
</kml>
```

7.15.5 Extends

- *<ColorStyle>*

7.15.6 Contained by

- <Style>

7.16 <Link>

7.16.1 Syntax

```
<Link id="ID">
  <!-- specific to Link -->
  <href>...</href>                      <!-- anyURI -->
  <refreshMode>onChange</refreshMode>
    <!-- refreshModeEnum: onChange, OnInterval, or OnExpire -->
  <refreshInterval>4</refreshInterval>  <!-- float -->
  <viewRefreshMode>never</viewRefreshMode>
    <!-- viewRefreshModeEnum: never, onStop, onRequest, onRegion -->
  <viewRefreshTime>4</viewRefreshTime>  <!-- float -->
  <viewBoundScale>1</viewBoundScale>    <!-- float -->
  <viewFormat>BBOX=[bboxWest],[bboxSouth],[bboxEast],[bboxNorth]</viewFormat>
                                        <!-- string -->
  <httpQuery>...</httpQuery>            <!-- string -->
</Link>
```

7.16.2 Description

<Link> specifies the location of any of the following:

- KML files fetched by network links
- Image files used by icons in icon styles, ground overlays, and screen overlays
- Model files used in the <Model> element

The file is conditionally loaded and refreshed, depending on the refresh parameters supplied here. Two different sets of refresh parameters can be specified: one set is based on *time* (<refreshMode> and <refreshInterval>) and one is based on the current "camera" *view* (<viewRefreshMode> and <viewRefreshTime>). In addition, Link specifies whether to scale the bounding box parameters that are sent to the server (<viewBoundScale> and provides a set of optional viewing parameters that can be sent to the server (<viewFormat>) as well as a set of optional parameters containing version and language information.

When a file is fetched, the URL that is sent to the server is composed of three pieces of information:

- the *href* (Hypertext Reference) that specifies the file to load.
- an *arbitrary format string* that is created from (a) parameters that you specify in the <viewFormat> element or (b) bounding box parameters (this is the default and is used if no <viewFormat> element is included in the file).
- a second *format string that is specified in the <httpQuery>* element.

If the file specified in <href> is a local file, the <viewFormat> and <httpQuery> elements

are not used.

The <Link> element is new in KML 2.1. It replaces the <Url> element of <NetworkLink> contained in earlier KML releases and adds functionality for the <Region> element (also introduced in KML 2.1). In Google Earth releases 3.0 and earlier, the <Link> element is ignored.

7.16.3 Elements specific to Link

<href> *(required)*

A URL (either an HTTP address or a local file specification). When the parent of <Link> is a NetworkLink, <href> is a KML file. When the parent of <Link> is a Model, <href> is a COLLADA file. When the parent of <Link> is an Overlay, <href> is an image. Relative URLs can be used in this tag and are evaluated relative to the enclosing KML file.

<refreshMode>

Specifies a time-based refresh mode, which can be one of the following:

- **onChange** - refresh when the file is loaded and whenever the Link parameters change (the default).
- **onInterval** - refresh every *n* seconds (specified in <*refreshInterval*>).
- **onExpire** - refresh the file when the expiration time is reached. For files fetched via NetworkLinkControls, the <expires> time takes precedence over expiration times specified in HTTP headers. If no <*expires*> time is specified, the HTTP *max-age* header is used (if present). If *max-age* is not present, the *Expires* HTTP header is used (if present). (See Section RFC261b of the *Hypertext Transfer Protocol - HTTP 1.1* for details on HTTP header fields.)

<refreshInterval>

Indicates to refresh the file every *n* seconds. Default = 4 seconds.

<viewRefreshMode>

Specifies how the link is refreshed when the "camera" changes.

Can be one of the following:

- **never** (default) - Ignore changes in the view. Also ignore <viewFormat> parameters, if any.
- **onStop** - Refresh the file *n* seconds after movement stops, where *n* is specified in <viewRefreshTime>.
- **onRequest** - Refresh the file only when the user explicitly requests it. (For

example, in Google Earth, the user right-clicks and selects Refresh in the Context menu.)

- **onRegion** - Refresh the file when the Region becomes active. See <Region>.

<viewRefreshTime>

After camera movement stops, specifies the number of seconds to wait before refreshing the view. Default = 4 seconds. (See <viewRefreshMode> and **onStop** above.)

<viewBoundScale>

Scales the BBOX parameters before sending them to the server. Default = 1. A value <1 specifies to use less than the full view (screen). A value >1 specifies to fetch an area that extends beyond the edges of the current view.

<viewFormat>

Specifies the format of the query string that is appended to the Link's <href> before the file is fetched.

(If the <href> specifies a local file, this element is ignored.)

If you specify a <ViewRefreshMode> of **onStop** and do not include the <viewFormat> tag in the file, the following information is automatically appended to the query string:

BBOX=[bboxWest],[bboxSouth],[bboxEast],[bboxNorth]

This information matches the Web Map Service (WMS) bounding box specification.

If you specify an empty <viewFormat> tag, no information is appended to the query string.

You can also specify a custom set of viewing parameters to add to the query string. If you supply a format string, it is used instead of the BBOX information. If you also want the BBOX information, you need to add those parameters along with the custom parameters.

You can use any of the following parameters in your format string (and Google Earth will substitute the appropriate current value at the time it creates the query string):

- **[lookatLon]**, **[lookatLat]** - longitude and latitude of the point the camera is looking at
- **[lookatRange]**, **[lookatTilt]**, **[lookatHeading]** - camera parameters (see descriptions of <range>, <tilt>, and <heading> in <LookAt>)
- **[horizFov]**, **[vertFov]** - horizontal, vertical field of view for the camera
- **[horizPixels]**, **[vertPixels]** - size in pixels of the 3D viewer
- **[terrainEnabled]** - indicates whether the 3D viewer is showing terrain

<httpQuery>

Appends information to the query string, based on the parameters specified. (Google Earth substitutes the appropriate current value at the time it creates the query string.) The following parameters are supported:

- **[clientVersion]**
- **[kmlVersion]**
- **[clientName]**
- **[language]**

7.16.4 Example

```
<NetworkLink>
  <name>NE US Radar</name>
  <flyToView>1</flyToView>
  <Link>
<href>http://www.example.com/geotiff/NE/MergedReflectivityQComposite.kml</
href>
    <refreshMode>onInterval</refreshMode>
    <refreshInterval>30</refreshInterval>
    <viewRefreshMode>onStop</viewRefreshMode>
    <viewRefreshTime>7</viewRefreshTime>
<viewFormat>BBOX=[bboxWest],[bboxSouth],[bboxEast],[bboxNorth];CAMERA=\
 [lookatLon],[lookatLat],[lookatRange],[lookatTilt],[lookatHeading];VIEW=\

[horizFov],[vertFov],[horizPixels],[vertPixels],[terrainEnabled]</viewForm
at>
  </Link>
</NetworkLink>
```

7.16.5 Extends

- *<Object>*

7.16.6 Contained by

- <Model>
- <NetworkLink>

7.16.7 See Also

- <NetworkLinkControl>
- <Region>

7.17 <ListStyle>

7.17.1 Syntax

```
<ListStyle id="ID">
  <!-- specific to ListStyle -->
  <bgColor>ffffffff</bgColor>        <!-- kml:color -->
  <listItemType>check</listItemType> <!-- kml:listItemTypeEnum:check,
                                          checkOffOnly,checkHideChildren,
                                      radioFolder -->
  <ItemIcon>                         <!-- 0 or more ItemIcon elements -->
    <state>open</state>
      <!-- kml:itemIconModeEnum:open, closed, error, fetching0, fetching1, or
fetching2 -->
    <href>...</href>                 <!-- anyURI -->
  </ItemIcon>
</ListStyle>
```

7.17.2 Description

Specifies how a Feature is displayed in the list view. The *list view* is a hierarchy of containers and children; in Google Earth, this is the Places panel.

7.17.3 Elements specific to ListStyle

<bgColor> (default=ffffffff)

Background color for the Snippet. Color and opacity values are expressed in hexadecimal notation. The range of values for any one color is 0 to 255 (00 to ff). For alpha, 00 is fully transparent and ff is fully opaque. The order of expression is *aabbggrr,* where *aa=alpha* (00 to ff); *bb=blue* (00 to ff); *gg=green* (00 to ff); *rr=red* (00 to ff). For example, if you want to apply a blue color with 50 percent opacity to an overlay, you would specify the following: <color>7fff0000</color>, where *alpha*=0x7f, *blue*=0xff, *green*=0x00, and *red*=0x00.

<listItemType> (default=check)

Specifies how a Folder is displayed in the list view. Possible values are:

- **check** (default) - The Feature's visibility is tied to its item's checkbox.
- **radioFolder** - When specified for a Folder, only one of the Folder's items is visible at a time
- **checkOffOnly** - When specified for a Folder, prevents all items from being made visible at once—that is, the user can turn everything in the Folder off but cannot turn everything on at the same time. This setting is useful for Folders containing

large amounts of data.

- **checkHideChildren** - Use a normal checkbox for visibility but do not display the Folder's children in the list view. A checkbox allows the user to toggle visibility of the child objects in the viewer.

<ItemIcon>

Icon used in the List view that reflects the state of a Folder or Link fetch. Icons associated with the **open** and **closed** modes are used for Folders. Icons associated with the **error** and **fetching0**, **fetching1**, and **fetching2** modes are used for Network Links.

- **<state>** - specifies the current state of the NetworkLink or Folder. Possible values are **open**, **closed**, **error**, **fetching0**, **fetching1**, and **fetching2**. These values can be combined by inserting a space between two values (no comma).

<href> - specifies the URI of the image used in the List View for the Feature.

7.17.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>ListStyle.kml</name>
  <open>1</open>
  <Style id="bgColorExample">
    <ListStyle>
      <bgColor>ff336699</bgColor>
    </ListStyle>
  </Style>
  <Style id="checkHideChildrenExample">
    <ListStyle>
      <listItemType>checkHideChildren</listItemType>
    </ListStyle>
  </Style>
  <Style id="radioFolderExample">
    <ListStyle>
      <listItemType>radioFolder</listItemType>
    </ListStyle>
  </Style>
  <Folder>
    <name>ListStyle Examples</name>
    <open>1</open>
    <Folder>
      <name>bgColor example</name>
      <open>1</open>
      <Placemark>
        <name>pl1</name>
        <Point>
          <coordinates>-122.362815,37.822931,0</coordinates>
        </Point>
      </Placemark>
      <Placemark>
        <name>pl2</name>
```

```
      <Point>
        <coordinates>-122.362825,37.822931,0</coordinates>
      </Point>
    </Placemark>
    <Placemark>
      <name>pl3</name>
      <Point>
        <coordinates>-122.362835,37.822931,0</coordinates>
      </Point>
    </Placemark>
    <styleUrl>#bgColorExample</styleUrl>
  </Folder>
  <Folder>
    <name>checkHideChildren example</name>
    <open>1</open>
    <Placemark>
      <name>pl4</name>
      <Point>
        <coordinates>-122.362845,37.822941,0</coordinates>
      </Point>
    </Placemark>
    <Placemark>
      <name>pl5</name>
      <Point>
        <coordinates>-122.362855,37.822941,0</coordinates>
      </Point>
    </Placemark>
    <Placemark>
      <name>pl6</name>
      <Point>
        <coordinates>-122.362865,37.822941,0</coordinates>
      </Point>
    </Placemark>
    <styleUrl>#checkHideChildrenExample</styleUrl>
  </Folder>
  <Folder>
    <name>radioFolder example</name>
    <open>1</open>
    <Placemark>
      <name>pl7</name>
      <Point>
        <coordinates>-122.362875,37.822951,0</coordinates>
      </Point>
    </Placemark>
    <Placemark>
      <name>pl8</name>
      <Point>
        <coordinates>-122.362885,37.822951,0</coordinates>
      </Point>
    </Placemark>
    <Placemark>
      <name>pl9</name>
      <Point>
        <coordinates>-122.362895,37.822951,0</coordinates>
      </Point>
    </Placemark>
```

```
      <styleUrl>#radioFolderExample</styleUrl>
    </Folder>
  </Folder>
</Document>
</kml>
```

7.17.5 Extends

- *<Object>*

7.17.6 Contained By

- <Style>

7.18 <LookAt>

7.18.1 Syntax

```
<LookAt id="ID">
  <longitude></longitude>      <!-- kml:angle180 -->
  <latitude></latitude>        <!-- kml:angle90 -->
  <altitude>0</altitude>       <!-- double -->
  <range></range>              <!-- double -->
  <tilt>0</tilt>               <!-- float -->
  <heading>0</heading>         <!-- float -->
  <altitudeMode>clampToGround</altitudeMode>
    <!--kml:altitudeModeEnum:clampToGround, relativeToGround, absolute -->
</LookAt>
```

7.18.2 Description

Defines a camera that is associated with any element derived from Feature. In Google Earth, double-clicking an item in the Places panel (or double-clicking an icon in the 3D viewer) causes the view to "fly to" the LookAt viewpoint.

7.18.3 Elements specific to LookAt

<longitude> *(required)*

Longitude of the point the camera is looking at. Angular distance in degrees, relative to the Prime Meridian. Values west of the Meridian range from −180 to 0 degrees. Values east of the Meridian range from 0 to 180 degrees.

<latitude> *(required)*

Latitude of the point the camera is looking at. Degrees north or south of the Equator (0 degrees). Values range from −90 degrees to 90 degrees.

<altitude>

Distance from the earth's surface, in meters. Default=0. (See <altitudeMode> for how this value is interpreted.)

<range> *(required)*

Distance in meters from the point specified by <longitude>, <latitude>, and <altitude> to the LookAt position. (See diagram below.)

<tilt>

Angle between the direction of the LookAt position and the normal to the surface of the earth. (See diagram below.) Values range from 0 to 90 degrees. Values for <tilt> cannot

be negative. A <tilt> value of 0 degrees indicates viewing from directly above. A <tilt> value of 90 degrees indicates viewing along the horizon.

<heading>

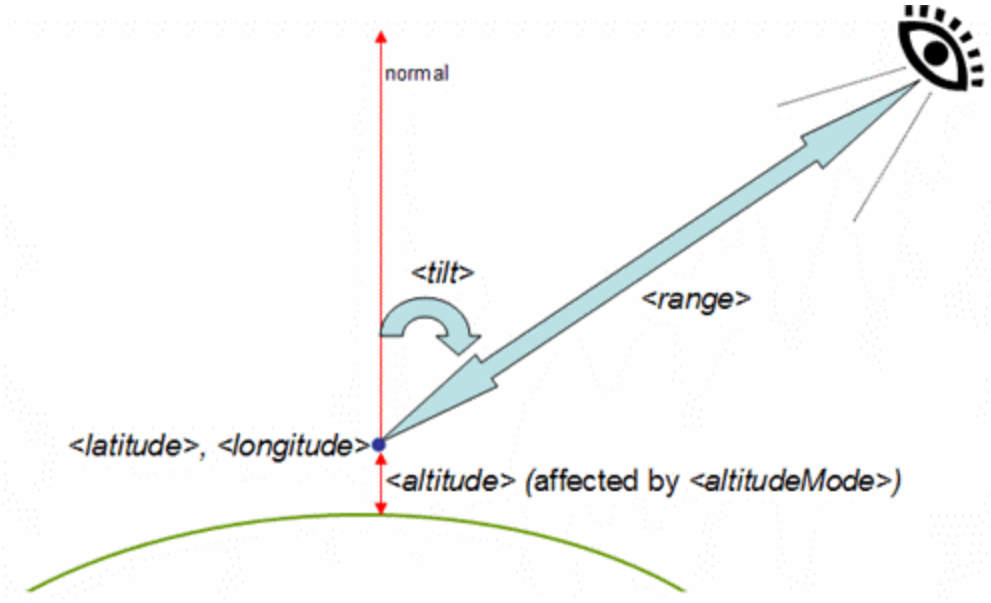
Compass direction, in degrees. Default=0 (North). (See diagram below.) Values range from 0 to ±180 degrees.

<altitudeMode>

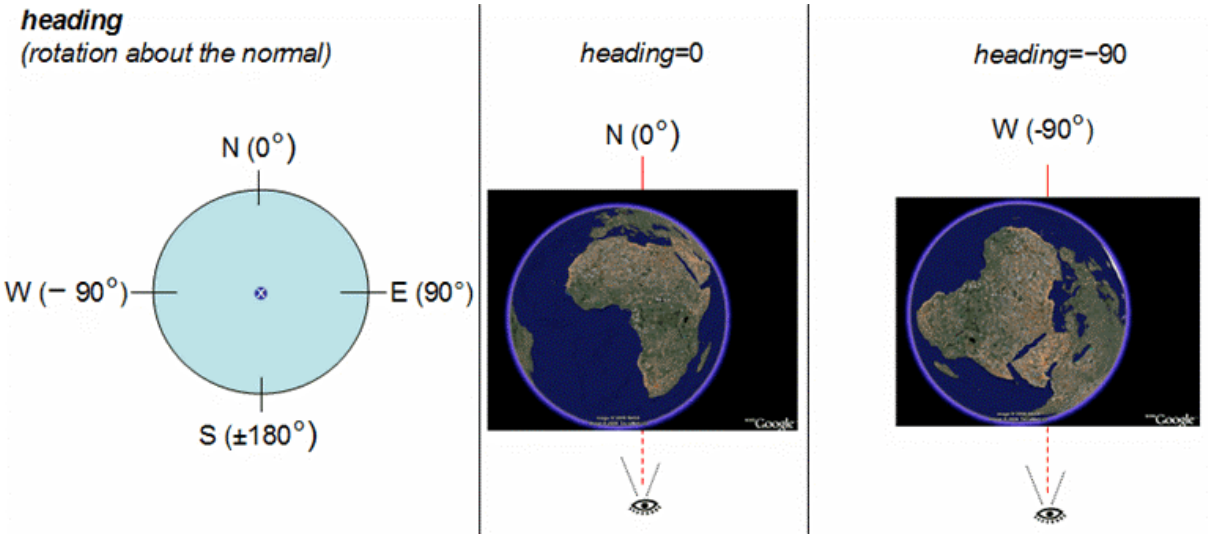
Specifies how the <altitude> specified for the LookAt point is interpreted. Possible values are as follows:

- **clampToGround** - (default) Indicates to ignore the <altitude> specification and place the LookAt position on the ground.
- **relativeToGround** - Interprets the <altitude> as a value in meters above the ground.
- **absolute** - Interprets the <altitude> as a value in meters above sea level.

This diagram illustrates the <range>, <tilt>, and <altitude> elements:



This diagram illustrates the <heading> element:



7.18.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Placemark>
  <name>LookAt.kml</name>
  <LookAt>
    <longitude>-122.363</longitude>
    <latitude>37.81</latitude>
    <altitude>2000</altitude>
    <range>500</range>
    <tilt>45</tilt>
    <heading>0</heading>
    <altitudeMode>relativeToGround</altitudeMode>
  </LookAt>
  <Point>
    <coordinates>-122.363,37.82,0</coordinates>
  </Point>
</Placemark>
</kml>
```

7.18.5 Extends

- *<Object>*

7.18.6 Contained by

- Any element derived from *<Feature>*

7.19 <Model>

7.19.1 Syntax

```
<Model id="ID">
  <!-- specific to Model -->
  <altitudeMode>clampToGround</altitudeMode>
    <!--kml:altitudeModeEnum: clampToGround,relativeToGround,or absolute -
->
  <Location>
    <longitude></longitude> <!-- kml:angle180 -->
    <latitude></latitude>   <!-- kml:angle90 -->
    <altitude>0</altitude>  <!-- double -->
  </Location>
  <Orientation>
    <heading>0</heading>    <!-- kml:angle360 -->
    <tilt>0</tilt>          <!-- kml:angle360 -->
    <roll>0</roll>          <!-- kml:angle360 -->
  </Orientation>
  <Scale>
    <x>1</x>                <!-- double -->
    <y>1</y>                <!-- double -->
    <z>1</z>                <!-- double -->
  </Scale>
  <Link>...</Link>
</Model>
```

7.19.2 Description

A 3D object described in a COLLADA file (referenced in the <Link> tag). COLLADA files have a *.dae* file extension. Models are created in their own coordinate space and then located, positioned, and scaled in Google Earth. See the KML 2.1 Tutorial for more detail.

Google Earth supports the COLLADA common profile, with the following exceptions:

- KML supports only triangles and lines as primitive types. The maximum number of triangles allowed is 21845.
- KML does not support animation or skinning.

7.19.3 Elements specific to Model

<altitudeMode>

Specifies how the <altitude> specified in <Location> is interpreted. Possible values are as follows:

- **clampToGround** - (default) Indicates to ignore the <altitude> specification and place the Model on the ground.
- **relativeToGround** - Interprets the <altitude> as a value in meters above the ground.
- **absolute** - Interprets the <altitude> as a value in meters above sea level.

<Location>

Specifies the exact coordinates of the Model's origin in latitude, longitude, and altitude. **Latitude** and **longitude** measurements are standard lat-lon projection with WGS84 datum. Google Earth uses simple cylindrical projection (or Plate Carrée), which is a simple map projection where the meridians and parallels are equidistant parallel lines, with the two sets crossing at right angles. **Altitude** is distance above the earth's surface, in meters, and is interpreted according to <altitudeMode>.

```
<Location>
  <longitude>39.55375305703105</longitude>
  <latitude>-118.9813220168456</latitude>
  <altitude>1223</altitude>
</Location>
```

<Orientation>

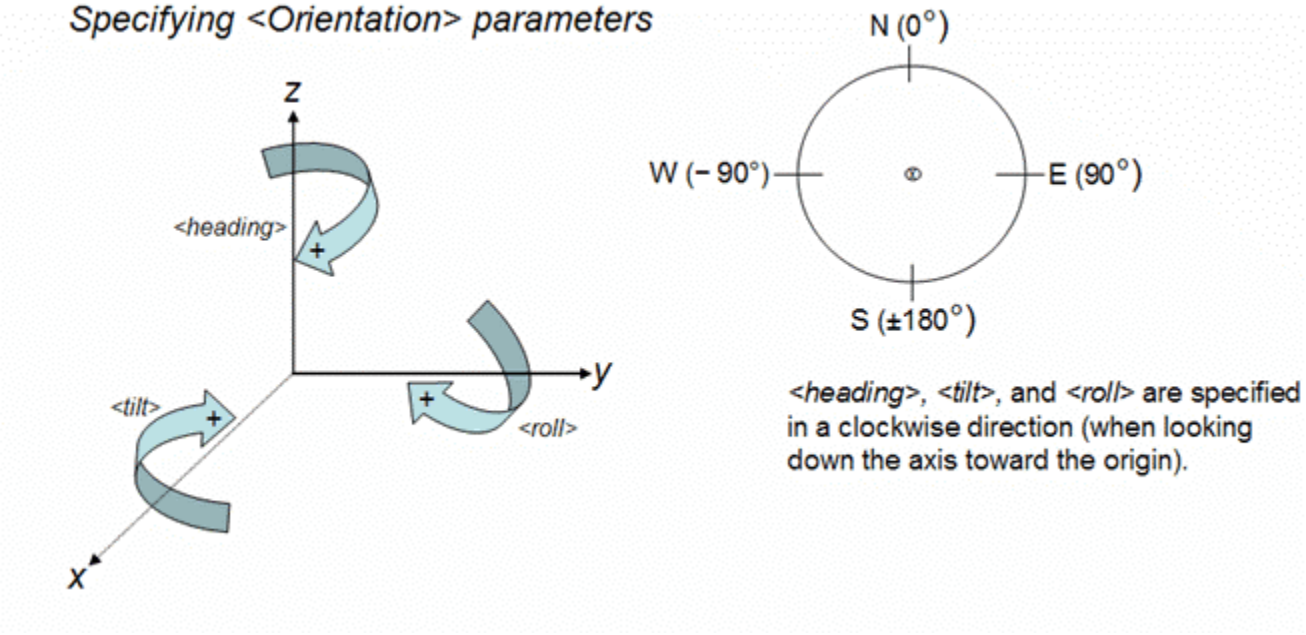
Describes rotation of a 3D model's coordinate system to position the object in Google Earth. See diagram below.

```
<Orientation>
  <heading>45.0</heading>
  <tilt>10.0</tilt>
  <roll>0.0</roll>
</Orientation>
```

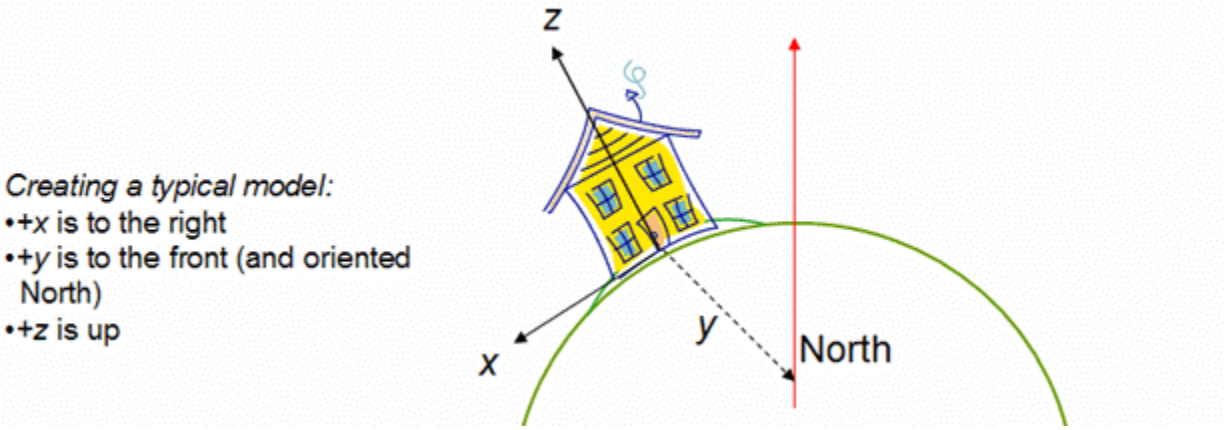
<heading> - rotation about the *z* axis. A value of 0 (the default) equals North. A positive rotation is clockwise around the *z* axis and specified in degrees from 0 to ±180.

<tilt> - rotation about the *x* axis. Default equals 0. A positive rotation is clockwise around the *x* axis and specified in degrees from 0 to ±180.

<roll> - rotation about the *y* axis. Default equals 0. A positive rotation is clockwise around the y axis and specified in degrees from 0 to ±180.



This diagram illustrates the typical orientation of a model's axes:



<Scale>

Scales a model along the *x*, *y*, and *z* axes in the model's coordinate space.

```
<Scale>
  <x>2.5</x>
  <y>2.5</y>
  <z>3.5</z>
</Scale>
```

<Link>

7.19.4 Example

```
<Model id="khModel543">
  <altitudeMode>relativeToGround</altitudeMode>
  <Location>
    <longitude>39.55375305703105</longitude>
    <latitude>-118.9813220168456</latitude>
    <altitude>1223</altitude>
  </Location>
  <Orientation>
    <heading>45.0</heading>
    <tilt>10.0</tilt>
    <roll>0.0</roll>
  </Orientation>
  <Scale>
    <x>1.0</x>
    <y>1.0</y>
    <z>1.0</z>
  </Scale>
  <Link>
    <href>house.dae</href>
    <refreshMode>once</refreshMode>
  </Link>
</Model>
```

7.19.5 Extends

- *<Geometry>*

7.19.6 Contained by

- <MultiGeometry>
- <Placemark>

7.20 <MultiGeometry>

7.20.1 Syntax

```
<MultiGeometry id="ID">
  <!-- specific to MultiGeometry -->
  <!-- 0 or more Geometry elements -->
</MultiGeometry>
```

7.20.2 Description

A container for one or more geometry primitives associated with the same feature.

Note: The <GeometryCollection> tag has been deprecated. Use <MultiGeometry> instead.

7.20.3 Elements specific to MultiGeometry

0 or more <Geometry> elements

7.20.4 Example

```
<Placemark>
  <name>SF Marina Harbor Master</name>
  <visibility>0</visibility>
  <MultiGeometry>
    <LineString>
      <!-- north wall -->
      <coordinates>
        -122.4425587930444,37.80666418607323,0
        -122.4428379594768,37.80663578323093,0
      </coordinates>
    </LineString>
    <LineString>
      <!-- south wall -->
      <coordinates>
        -122.4425509770566,37.80662588061205,0
        -122.4428340530617,37.8065999493009,0
      </coordinates>
    </LineString>
  </MultiGeometry>
</Placemark>
```

7.20.5 Extends

- *<Geometry>*

7.20.6 Contained by

- <MultiGeometry>
- <Placemark>

7.21 <NetworkLink>

7.21.1 Syntax

```
<NetworkLink id="ID">
  <!-- inherited from Feature element -->
  <name>...</name>                        <!-- string -->
  <visibility>1</visibility>              <!-- boolean -->
  <open>1</open>                          <!-- boolean -->
  <address>...</address>                  <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                   <!-- string -->
  <phoneNumber>...</phoneNumber>          <!-- string -->
  <Snippet maxLines="2">...</Snippet>     <!-- string -->
  <description>...</description>          <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>                <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- specific to NetworkLink -->
  <Link>...</Link>
  <refreshVisibility>0</refreshVisibility> <!-- boolean -->
  <flyToView>0</flyToView>                 <!-- boolean -->
</NetworkLink>
```

7.21.2 Description

References a KML file or KMZ archive on a local or remote network. Use the <Link> element to specify the location of the KML file. Within that element, you can define the refresh options for updating the file, based on time and camera change. NetworkLinks can be used in combination with Regions to handle very large datasets efficiently.

7.21.3 Elements specific to NetworkLink

<Link> *(required).* See <Link>.

<refreshVisibility>

Boolean value (default=0). A value of 0 leaves the visibility of features within the control of the Google Earth user. Set the value to 1 to reset the visibility of features each time the NetworkLink is refreshed. For example, suppose a Placemark within the linked KML file has <visibility> set to 1 and <refreshVisibility> set to 1. When the file is first loaded into Google Earth, the user can clear the check box next to the item to turn off display in the 3D viewer. However, when the network link is refreshed, the item will be made visible

again, since its original visibility state was TRUE.

Note: This tag has been superseded by the <Update> tag, introduced in KML 2.1.

<flyToView>

Boolean value (default=0). A value of 1 causes Google Earth to fly to the view of the root element in the refreshed file. In this case, Google Earth flies to the <LookAt> view of the parent Document, not the <LookAt> of the Placemarks contained within the Document.

7.21.4 Example

```
<Document>
  <visibility>1</visibility>
  <NetworkLink>
    <name>NE US Radar</name>
    <flyToView>1</flyToView>
    <Link>...</Link>
    <refreshVisibility>1</refreshVisibility>
  </NetworkLink>
</Document>
```

7.21.5 Extends

- *<Feature>*

7.21.6 Contained by

- any element derived from *<Container>*

7.22 <NetworkLinkControl>

7.22.1 Syntax

```
<NetworkLinkControl>
  <minRefreshPeriod>0</minRefreshPeriod>          <!-- float -->
  <cookie>...</cookie>                            <!-- string -->
  <message>...</message>                          <!-- string -->
  <linkName>...</linkName>                        <!-- string -->
  <linkDescription>...</linkDescription>          <!-- string -->
  <linkSnippet maxLines="2">...</linkSnippet>     <!-- string -->
  <expires>...</expires>                          <!-- kml:dateTime --
>
  <Update>...</Update>                            <!--
Change,Create,Delete -->
  <LookAt>...</LookAt>
</NetworkLinkControl>
```

7.22.2 Description

Controls the behavior of files fetched by a <NetworkLink>.

7.22.3 Elements specific to NetworkLinkControl

<minRefreshPeriod> (default=0)

Specified in seconds, <minRefreshPeriod> is the minimum allowed time between fetches of the file. This parameter allows servers to throttle fetches of a particular file and to tailor refresh rates to the expected rate of change to the data. For example, a user might set a link refresh to 5 seconds, but you could set your minimum refresh period to 3600 to limit refresh updates to once every hour.

<cookie>

Use this http://earth.google.com/kml/kml_tags_21.html - cookie element to append the text to the URL query on the next refresh of the network link. You can use this data in your script to provide more intelligent handling on the server side, including version querying and conditional file delivery.

<message>

You can deliver a pop-up message, such as usage guidelines for your network link. The message appears when the network link is first loaded into Google Earth, or when it is changed in the network link control.

<linkName>

You can control the name of the network link from the server, so that changes made to

the name on the client side are overridden by the server.

<linkDescription>

You can control the description of the network link from the server, so that changes made to the description on the client side are overridden by the server.

<linkSnippet maxLines="2" **>**

You can control the snippet for the network link from the server, so that changes made to the snippet on the client side are overridden by the server. <linkSnippet> has a **maxLines** attribute, an integer that specifies the maximum number of lines to display. Default for **maxLines** is 2.

<expires>

You can specify a date/time at which the link should be refreshed. This specification takes effect only if the <refreshMode> in <Link> has a value of **onExpire**. See <refreshMode>.

<Update>

With <Update>, you can specify any number of Change, Create, and Delete tags for a *.kml* file or *.kmz* archive that has previously been loaded with a network link. See <Update>.

7.22.4 Example

```
<NetworkLinkControl>
   <message>This is a pop-up message. You will only see this
once</message>
   <cookie>cookie=sometext</cookie>
   <linkName>New KML features</linkName>
   <linkDescription><![CDATA[KML now has new features
available!]]></linkDescription>
</NetworkLinkControl>
```

7.22.5 Extends

- This is a root element

7.22.6 Contained by

- <kml>

7.22.7 See Also

- <Update>

7.23 *<Object>*

7.23.1 Syntax

```
<!-- abstract element; do not create -->
<Object id="ID" targetId="NCName">
</Object>
```

7.23.2 Description

This is an abstract base class and cannot be used directly in a KML file. It provides the **id** attribute, which allows unique identification of a KML element, and the **targetId** attribute, which is used to reference objects that have already been loaded into Google Earth. The **id** attribute must be assigned if the <Update> mechanism is to be used.

7.23.3 Elements specific to Overlay

N.A.

7.23.4 Examples

N.A.

7.23.5 Extends

N.A.

7.23.6 Contained by

N.A.

7.24 *<Overlay>*

7.24.1 Syntax

```
<!-- abstract element; do not create -->
<!-- Overlay id="ID" -->                    <!--
GroundOverlay,ScreenOverlay -->
  <!-- inherited from Feature element -->
  <name>...</name>                        <!-- string -->
  <visibility>1</visibility>              <!-- boolean -->
  <open>1</open>                          <!-- boolean -->
  <address>...</address>                  <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                   <!-- string -->
  <phoneNumber>...</phoneNumber>          <!-- string -->
  <Snippet maxLines="2">...</Snippet>     <!-- string -->
  <description>...</description>          <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>                <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- specific to Overlay -->
  <color>ffffffff</color>                    <!-- kml:color -->
  <drawOrder>0</drawOrder>                   <!-- int -->
  <Icon>
    <href>...</href>
  </Icon>
<!-- /Overlay -->
```

7.24.2 Description

This is an abstract element and cannot be used directly in a KML file. *<Overlay>* is the base type for image overlays drawn on the planet surface or on the screen. <Icon> specifies the image to use and can be configured to reload images based on a timer or by camera changes. This element also includes specifications for stacking order of multiple overlays and for adding color and transparency values to the base image.

7.24.3 Elements specific to Overlay

<color>

Color values are expressed in hexadecimal notation, including opacity (alpha) values. The order of expression is alpha, blue, green, red (*aabbggrr*). The range of values for any one color is 0 to 255 (00 to ff). For opacity, 00 is fully transparent and ff is fully opaque.

For example, if you want to apply a blue color with 50 percent opacity to an overlay, you would specify the following: <color>7fff0000</color>

Note: The <geomColor> element has been deprecated. Use <color> instead.

<drawOrder> (default=0)

This element defines the stacking order for the images in overlapping overlays. Overlays with higher <drawOrder> values are drawn on top of overlays with lower <drawOrder> values.

<Icon> See also <Icon>.

Defines the image associated with the Overlay. The <href> element defines the location of the image to be used as the Overlay. This location can be either on a local file system or on a web server. If this element is omitted or contains no <href>, a rectangle is drawn using the color and size defined by the ground or screen overlay.

```
<Icon>
   <href>icon.jpg</href>
</Icon>
```

7.24.4 Example

None.

7.24.5 Extends

- *<Feature>*

7.24.6 Extended by

- <GroundOverlay>
- <ScreenOverlay>

7.25 <PlaceMark>

7.25.1 Syntax

```
<Placemark id="ID">
  <!-- inherited from Feature element -->
  <name>...</name>                        <!-- string -->
  <visibility>1</visibility>              <!-- boolean -->
  <open>1</open>                          <!-- boolean -->
  <address>...</address>                  <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                   <!-- string -->
  <phoneNumber>...</phoneNumber>          <!-- string -->
  <Snippet maxLines="2">...</Snippet>     <!-- string -->
  <description>...</description>          <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>                <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>
  <!-- specific to Placemark element -->
  <Geometry>...</Geometry>
</Placemark>
```

7.25.2 Description

A Placemark is a Feature with associated Geometry. In Google Earth, a Placemark appears as a list item in the Places panel. A Placemark with a Point has an icon associated with it that marks a point on the earth in the 3D viewer. (In the Google Earth 3D viewer, a Point Placemark is the only object you can click or roll over. Other Geometry objects do not have an icon in the 3D viewer. To give the user something to click, you would need to add a Point to the Polygon.)

7.25.3 Elements specific to Placemark

- 0 or one *<Geometry>* elements

7.25.4 Example

```
<Placemark>
  <name>Google Earth - New Placemark</name>
  <description>Some Descriptive text.</description>
  <LookAt>
    <longitude>-90.86879847669974</longitude>
    <latitude>48.25330383601299</latitude>
    <range>440.8</range>
```

```
    <tilt>8.3</tilt>
    <heading>2.7</heading>
  </LookAt>
  <Point>
    <coordinates>-90.86948943473118,48.25450093195546,0</coordinates>
  </Point>
</Placemark>
```

7.25.5 Extends

- *<Feature>*

7.25.6 Contained by

- <Document>
- <Folder>

7.25.7 See Also

- <Icon>

7.26 <Point>

7.26.1 Syntax

```
<Point id="ID">
  <!-- specific to Point -->
  <extrude>0</extrude>                    <!-- boolean -->
  <tessellate>0</tessellate>              <!-- boolean -->
  <altitudeMode>clampToGround</altitudeMode>
    <!-- kml:altitudeModeEnum: clampToGround, relativeToGround, or
absolute -->
  <coordinates>...</coordinates>          <!-- lon,lat[,alt] -->
</Point>
```

7.26.2 Description

A geographic location defined by longitude, latitude, and (optional) altitude. When a Point is contained by a Placemark, the point itself determines the position of the Placemark's name and icon. When a Point is extruded, it is connected to the ground with a line. This "tether" uses the current LineStyle.

7.26.3 Elements specific to Point

<extrude>

Boolean value (default=0). Specifies whether to connect the geometric primitive (icon, line, polygon) to the ground. Extrusion requires that the geometry's <altitudeMode> be either **relativeToGround** or **absolute** and that within the <coordinates> element, the *altitude* component be greater than 0 (that is, in the air).

<tessellate>

Boolean value (default=0). Specifies whether to allow lines and paths to follow the terrain. This specification applies only to LineStrings (paths) and LinearRings (polygons) that have an <altitudeMode> of **clampToGround**. Very long lines should enable tessellation so that they follow the curvature of the earth (otherwise, they may go underground and be hidden).

<altitudeMode>

Specifies how *altitude* components in the <coordinates> element are interpreted. Possible values are

- **clampToGround** - (default) Indicates to ignore an altitude specification (for example, in the <coordinates> tag).
- **relativeToGround** - Sets the altitude of the element relative to the actual ground

elevation of a particular location. For example, if the ground elevation of a location is exactly at sea level and the altitude for a point is set to 9 meters, then the elevation for the icon of a point placemark elevation is 9 meters with this mode. However, if the same coordinate is set over a location where the ground elevation is 10 meters above sea level, then the elevation of the coordinate is 19 meters. A typical use of this mode is for placing telephone poles or a ski lift.

- **absolute** - Sets the altitude of the coordinate relative to sea level, regardless of the actual elevation of the terrain beneath the element. For example, if you set the altitude of a coordinate to 10 meters with an **absolute** altitude mode, the icon of a point placemark will appear to be at ground level if the terrain beneath is also 10 meters above sea level. If the terrain is 3 meters above sea level, the placemark will appear elevated above the terrain by 7 meters. A typical use of this mode is for aircraft placement.

<coordinates> *(required)*

A single tuple consisting of floating point values for longitude, latitude, and altitude (in that order). Longitude and latitude values are in degrees, where

- *longitude* ≥ -180 and <= 180
- *latitude* ≥ -90 and ≤ 90
- *altitude* values (optional) are in meters above sea level

Do not include spaces between the three values that describe a coordinate.

7.26.4 Example

```
<Point>
  <coordinates>-90.86948943473118,48.25450093195546</coordinates>
</Point>
```

7.26.5 Extends

- *<Geometry>*

7.26.6 Contained by

- <MultiGeometry>
- <Placemark>

7.27 <Polygon>

7.27.1 Syntax

```
<Polygon id="ID">
  <!-- specific to Polygon -->
  <extrude>0</extrude>                          <!-- boolean -->
  <tessellate>0</tessellate>                    <!-- boolean -->
  <altitudeMode>clampToGround</altitudeMode>
    <!-- kml:altitudeModeEnum: clampToGround, relativeToGround, or
absolute -->
  <outerBoundaryIs>
    <LinearRing>
      <coordinates>...</coordinates>            <!-- lon,lat[,alt] -->
    </LinearRing>
  </outerBoundaryIs>
  <innerBoundaryIs>
    <LinearRing>
      <coordinates>...</coordinates>            <!-- lon,lat[,alt] -->
    </LinearRing>
  </innerBoundaryIs>
</Polygon>
```

7.27.2 Description

A Polygon is defined by an outer boundary and 0 or more inner boundaries. The boundaries, in turn, are defined by LinearRings. When a Polygon is extruded, its boundaries are connected to the ground to form additional polygons, which gives the appearance of a building. When a Polygon is extruded, each point is extruded individually. Extruded Polygons use <Polystyle> for their color, color mode, and fill.

7.27.3 Elements specific to Polygon

<extrude>

Boolean value (default=0). Specifies whether to connect the geometric primitive (icon, line, polygon) to the ground. Extrusion requires that the geometry's <altitudeMode> be either **relativeToGround** or **absolute** and that within the <coordinates> element, the *altitude* component be greater than 0 (that is, in the air).

<tessellate>

Boolean value (default=0). Specifies whether to allow lines and paths to follow the terrain. This specification applies only to LineStrings (paths) and LinearRings (polygons) that have an <altitudeMode> of **clampToGround**. Very long lines should enable tessellation so that they follow the curvature of the earth (otherwise, they may go

underground and be hidden).

<altitudeMode>

Specifies how *altitude* components in the <coordinates> element are interpreted. Possible values are

- **clampToGround** - (default) Indicates to ignore an altitude specification (for example, in the <coordinates> tag).
- **relativeToGround** - Sets the altitude of the element relative to the actual ground elevation of a particular location. For example, if the ground elevation of a location is exactly at sea level and the altitude for a point is set to 9 meters, then the elevation for the icon of a point placemark elevation is 9 meters with this mode. However, if the same coordinate is set over a location where the ground elevation is 10 meters above sea level, then the elevation of the coordinate is 19 meters. A typical use of this mode is for placing telephone poles or a ski lift.
- **absolute** - Sets the altitude of the coordinate relative to sea level, regardless of the actual elevation of the terrain beneath the element. For example, if you set the altitude of a coordinate to 10 meters with an **absolute** altitude mode, the icon of a point placemark will appear to be at ground level if the terrain beneath is also 10 meters above sea level. If the terrain is 3 meters above sea level, the placemark will appear elevated above the terrain by 7 meters. A typical use of this mode is for aircraft placement.

<outerBoundaryIs> *(required)*

Contains a <LinearRing> element.

<innerBoundaryIs>

Contains a <LinearRing> element. A Polygon can contain multiple <innerBoundaryIs> elements, which create multiple cut-outs inside the Polygon.

7.27.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>Polygon.kml</name>
  <open>1</open>
  <Placemark>
    <name>hollow box</name>
    <Polygon>
      <extrude>1</extrude>
      <altitudeMode>relativeToGround</altitudeMode>
      <outerBoundaryIs>
        <LinearRing>
          <coordinates>
```

```
            -122.366278,37.818844,30
            -122.365248,37.819267,30
            -122.365640,37.819861,30
            -122.366669,37.819429,30
            -122.366278,37.818844,30
          </coordinates>
        </LinearRing>
      </outerBoundaryIs>
      <innerBoundaryIs>
        <LinearRing>
          <coordinates>
            -122.366212,37.818977,30
            -122.365424,37.819294,30
            -122.365704,37.819731,30
            -122.366488,37.819402,30
            -122.366212,37.818977,30
          </coordinates>
        </LinearRing>
      </innerBoundaryIs>
    </Polygon>
  </Placemark>
</Document>
</kml>
```

7.27.5 Extends

- *<Geometry>*

7.27.6 Contained by

- <MultiGeometry>
- <Placemark>

7.28 <PolyStyle>

7.28.1 Syntax

```
<PolyStyle id="ID">
  <!-- inherited from ColorStyle -->
  <color>ffffffff</color>            <!-- kml:color -->
  <colorMode>normal</colorMode>      <!-- kml:colorModeEnum: normal or
random -->

  <!-- specific to PolyStyle -->
  <fill>1</fill>                     <!-- boolean -->
  <outline>1</outline>               <!-- boolean -->
</PolyStyle>
```

7.28.2 Description

Specifies the drawing style for all polygons, including polygon extrusions (which look like the walls of buildings) and line extrusions (which look like solid fences).

7.28.3 Elements specific to PolyStyle

<fill>

Boolean value (default=1). Specifies whether to fill the polygon.

<outline>

Boolean value (default=1). Specifies whether to outline the polygon. Polygon outlines use the current LineStyle.

7.28.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>PolygonStyle.kml</name>
  <open>1</open>
  <Style id="examplePolyStyle">
    <PolyStyle>
      <color>ff0000cc</color>
      <colorMode>random</colorMode>
    </PolyStyle>
  </Style>
  <Placemark>
```

```
    <name>hollow box</name>
    <styleUrl>#examplePolyStyle</styleUrl>
    <Polygon>
      <extrude>1</extrude>
      <altitudeMode>relativeToGround</altitudeMode>
      <outerBoundaryIs>
        <LinearRing>
          <coordinates>
            -122.3662784465226,37.81884427772081,30
            -122.3652480684771,37.81926777010555,30
            -122.365640222455,37.81986126286519,30
            -122.36666937925,37.81942987753481,30
            -122.3662784465226,37.81884427772081,30
          </coordinates>
        </LinearRing>
      </outerBoundaryIs>
      <innerBoundaryIs>
        <LinearRing>
          <coordinates>
            -122.366212593918,37.81897719083808,30
            -122.3654241733188,37.81929450992014,30
            -122.3657048517827,37.81973175302663,30
            -122.3664882465854,37.81940249291773,30
            -122.366212593918,37.81897719083808,30
          </coordinates>
        </LinearRing>
      </innerBoundaryIs>
    </Polygon>
  </Placemark>
</Document>
</kml>
```

7.28.5 Extends

- *<ColorStyle>*

7.28.6 Contained by

- <Style>

7.29 <Region>

7.29.1 Syntax

```
<Region id="ID">
  <LatLonAltBox>
    <north></north>                          <!-- required; kml:angle90 -->
    <south></south>                          <!-- required; kml:angle90 -->
    <east></east>                            <!-- required; kml:angle180 -->
    <west></west>                            <!-- required; kml:angle180 -->
    <minAltitude>0</minAltitude>             <!-- float -->
    <maxAltitude>0</maxAltitude>             <!-- float -->
    <altitudeMode>clampToGround</altitudeMode>
      <!-- kml:altitudeModeEnum: clampToGround, relativeToGround, or absolute -->
  </LatLonAltBox>
  <Lod>
    <minLodPixels>0</minLodPixels>           <!-- float -->
    <maxLodPixels>-1</maxLodPixels>          <!-- float -->
    <minFadeExtent>0</minFadeExtent>         <!-- float -->
    <maxFadeExtent>0</maxFadeExtent>         <!-- float -->
  </Lod>
</Region>
```

7.29.2 Description

A region contains a *bounding box* (<LatLonAltBox>) that describes an area of interest defined by geographic coordinates and altitudes. In addition, a Region contains an *LOD (level of detail) extent* (<Lod>) that defines a validity range of the associated Region in terms of projected screen size. A Region is said to be "active" when the bounding box is within the user's view and the LOD requirements are met. Objects associated with a Region are drawn only when the Region is active. When the <viewRefreshMode> is **onRegion**, the Link or Icon is loaded only when the Region is active. See the KML 2.1 Tutorial for more details. In a Container or NetworkLink hierarchy, this calculation uses the Region that is the closest ancestor in the hierarchy.

7.29.3 Elements specific to Region

<LatLonAltBox>*(required)*

A bounding box that describes an area of interest defined by geographic coordinates and altitudes. Default values and required fields are as follows:

- <altitudeMode> (defaults to *clampToGround*). Possible values are **clampToGround**, **relativeToGround**, and **absolute**. Also see <LatLonBox>.
- **<minAltitude>** Defaults to 0; specified in meters above sea level (and is affected by the <altitudeMode> specification).

- **<maxAltitude>** Defaults to 0; specified in meters above sea levell (and is affected by the <altitudeMode> specification).

- **<north>** *(required)* Specifies the latitude of the north edge of the bounding box, in decimal degrees from 0 to ±90.

- **<south>** *(required)* Specifies the latitude of the south edge of the bounding box, in decimal degrees from 0 to ±90.

- **<east>** *(required)* Specifies the longitude of the east edge of the bounding box, in decimal degrees from 0 to ±180.

- **<west>** *(required)* Specifies the longitude of the west edge of the bounding box, in decimal degrees from 0 to ±180.

```
<LatLonAltBox>
  <north>43.374</north>
  <south>42.983</south>
  <east>-0.335</east>
  <west>-1.423</west>
  <minAltitude>0</minAltitude>
  <maxAltitude>0</maxAltitude>
</LatLonAltBox>
```

<Lod>

Lod is an abbreviation for *Level of Detail*. <Lod> describes the size of the projected region on the screen that is required in order for the region to be considered "active." Also specifies the size of the pixel ramp used for fading in (from transparent to opaque) and fading out (from opaque to transparent). See diagram below for a visual representation of these parameters.

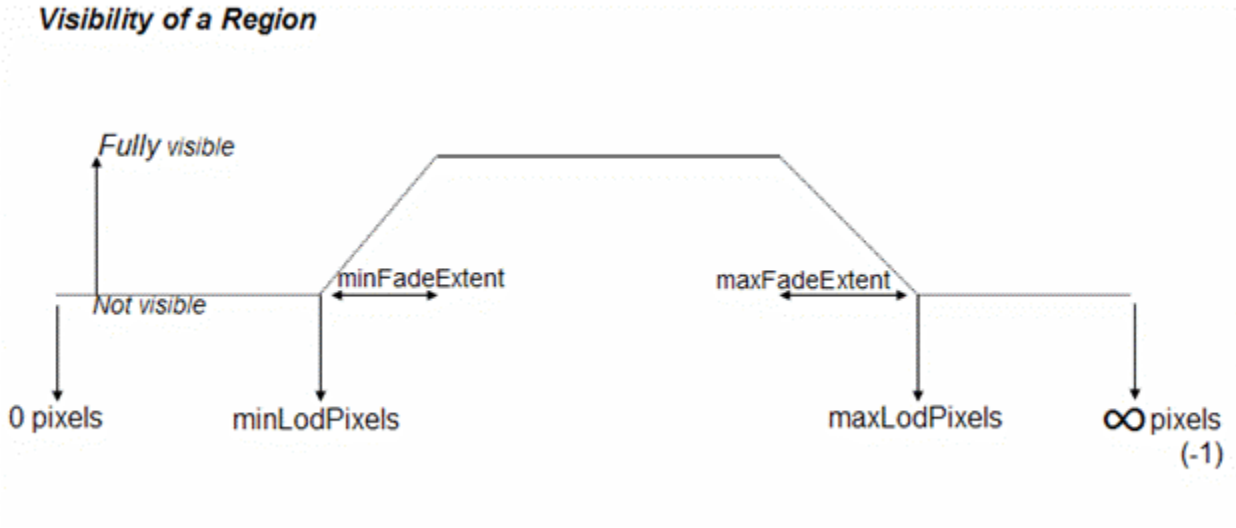
```
<Lod>
  <minLodPixels>256</minLodPixels>
  <maxLodPixels>-1</maxLodPixels>
  <minFadeExtent>0</minFadeExtent>
  <maxFadeExtent>0</maxFadeExtent>
</Lod>
```

- **<minLodPixels>** (default = 0) Measurement in screen pixels that represents the minimum limit of the visibility range for a given Region. Google Earth calculates the size of the Region when projected onto screen space. Then it computes the square root of the Region's area (if, for example, the Region is square and the viewpoint is directly above the Region, and the Region is not tilted, this measurement is equal to the width of the projected Region). If this measurement falls within the limits defined by <minLodPixels> and <maxLodPixels> (and if the <LatLonAltBox> is in view), the Region is active. If this limit is not reached, the associated geometry is considered to be too far from the user's viewpoint to be drawn.

- **<maxLodPixels>** (default = −1) Measurement in screen pixels that represents the maximum limit of the visibility range for a given Region. A value of −1, the default, indicates "active to infinite size."

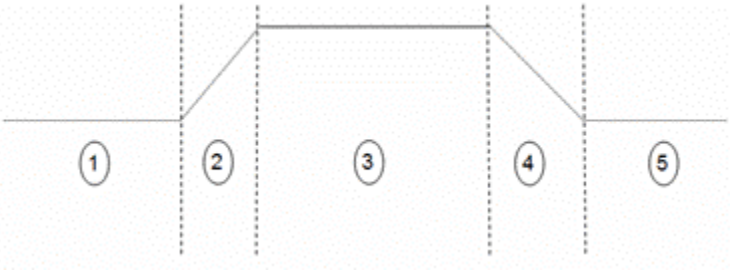
- **<minFadeExtent>** (default = 0) Distance over which the geometry fades, from fully opaque to fully transparent. This ramp value, expressed in screen pixels, is applied at the minimum end of the LOD (visibility) limits.

- **<maxFadeExtent>** (default = 0) Distance over which the geometry fades, from fully transparent to fully opaque. This ramp value, expressed in screen pixels, is applied at the maximum end of the LOD (visibility) limits.



In the following diagram, if *P*=the calculated projected pixel size, the circled numbers indicate the following:

```
if (P < minLodPixels)
  opacity=0                                  //#1 in diagram
else if(P < minLodPixels + minFadeExtent)
  opacity=(P - minLodPixels)/minFadeExtent   //#2 in diagram
else if (P < maxLodPixels - maxFadeExtent)
  opacity=1                                  //#3 in diagram
else if (P < maxLodPixels)
  opacity=(maxLodPixels-P)/maxFadeExtent     //#4 in diagram
else
  opacity=0                                  //#5 in diagram
```



7.29.4 Example

```
<Region>
  <LatLonAltBox>
    <north>50.625</north>
    <south>45</south>
    <east>28.125</east>
    <west>22.5</west>
    <minAltitude>10</minAltitude>
    <maxAltitude>50</maxAltitude>
  </LatLonAltBox>
  <Lod>
    <minLodPixels>128</minLodPixels>
    <maxLodPixels>1024</maxLodPixels>
    <minFadeExtent>128</minFadeExtent>
    <maxFadeExtent>128</maxFadeExtent>
  </Lod>
</Region>
```

7.29.5 Extends

- *<Object>*

7.29.6 Contained by

- any element derived from *<Feature>*

7.30 *<SchemaField>*

7.30.1 Syntax

```
<!-- abstract element; do not create -->
<!-- SchemaField name="string" type="enum" -->      <!--
SimpleField,SimpleArrayField,

ObjField,ObjArrayField -->
<!-- /SchemaField -->
```

7.30.2 Description

<SchemaField> is an abstract element and cannot be used directly in a KML file. The derived classes, which can be added to a <Schema> are as follows:

<SimpleField>

A schema field for simple types such as ints, floats, and strings.

<SimpleArrayField>

A schema field for arrays of simple types (ints, floats, and string).

<ObjField>

A schema field for a pointer to a schema object.

<ObjArrayField>

A schema field for an array of pointers to schema objects.

The **type** attribute for any of these schema fields can be one of the following:

- **string**
- **int**
- **uint**
- **short**
- **ushort**
- **float**
- **double**

- **bool**

7.30.3 Elements specific to SchemaField

None.

7.30.4 Example

```
<Schema name="S_countyp020_DDISSSSD" parent="Placemark">
  <SimpleField name="AREA" type="double"></SimpleField>
  <SimpleField name="PERIMETER" type="double"></SimpleField>
  <SimpleField name="COUNTYP020" type="int"></SimpleField>
  <SimpleField name="STATE" type="string"></SimpleField>
  <SimpleField name="COUNTY" type="string"></SimpleField>
  <SimpleField name="FIPS" type="string"></SimpleField>
  <SimpleField name="STATE_FIPS" type="string"></SimpleField>
  <SimpleField name="SQUARE_MIL" type="double"></SimpleField>
</Schema>
```

7.30.5 Extended by the elements

For simplicity, these lightweight elements are described above.

- <SimpleField>
- <SimpleArrayField>
- <ObjField>
- <ObjArrayField>

7.30.6 Contained by

- <Schema>

7.31 <ScreenOverlay>

7.31.1 Syntax

```
<ScreenOverlay id="ID">
  <!-- inherited from Feature element -->
  <name>...</name>                      <!-- string -->
  <visibility>1</visibility>            <!-- boolean -->
  <open>1</open>                        <!-- boolean -->
  <address>...</address>                <!-- string -->
  <AddressDetails xmlns="urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">...
      </AddressDetails>                 <!-- string -->
  <phoneNumber>...</phoneNumber>        <!-- string -->
  <Snippet maxLines="2">...</Snippet>   <!-- string -->
  <description>...</description>        <!-- string -->
  <LookAt>...</LookAt>
  <TimePrimitive>...</TimePrimitive>
  <styleUrl>...</styleUrl>              <!-- anyURI -->
  <StyleSelector>...</StyleSelector>
  <Region>...</Region>
  <Metadata>...</Metadata>

  <!-- inherited from Overlay element -->
  <color>ffffffff</color>                  <!-- kml:color -->
  <drawOrder>0</drawOrder>                 <!-- int -->
  <Icon>...</Icon>

  <!-- specific to ScreenOverlay -->
  <overlayXY x="double" y="double" xunits="fraction" yunits="fraction"/>
    <!-- vec2Type -->
    <!-- xunits and yunits can be one of: fraction, pixels, or insetPixels
-->
  <screenXY x="double" y="double" xunits="fraction" yunits="fraction"/>
    <!-- vec2Type -->
  <rotationXY x="double" y="double" xunits="fraction" yunits"fraction"/>
    <!-- vec2Type -->
  <size x="double" y="double" xunits="fraction" yunits="fraction"/>
    <!-- vec2Type -->
  <rotation>0</rotation>                   <!-- float -->
 </ScreenOverlay>
```

7.31.2 Description

This element draws an image overlay fixed to the screen. Sample uses for ScreenOverlays are compasses, logos, and heads-up displays. ScreenOverlay sizing is determined by the <size> element. Positioning of the overlay is handled by mapping a point in the image specified by <overlayXY> to a point on the screen specified by <screenXY>. Then the image is rotated by <rotation> degrees about a point relative to the screen specified by <rotationXY>.

7.31.3 Elements specific to ScreenOverlay

<overlayXY>

Specifies a point on (or outside of) the overlay image that is mapped to the screen coordinate (<*screenXY*>). It requires *x* and *y* values, and the units for those values.

The *x* and *y* values can be specified in three different ways: as *pixels* (**"pixels"**), as *fractions* of the image (**"fraction"**), or as *inset pixels* (**"insetPixels"**), which is an offset in pixels from the upper right corner of the image. The *x* and *y* positions can be specified in different ways—for example, *x* can be in pixels and *y* can be a fraction. The origin of the coordinate system is in the lower left corner of the image.

- **x** - Either the number of pixels, a fractional component of the image, or a pixel inset indicating the *x* component of a point on the overlay image.
- **y** - Either the number of pixels, a fractional component of the image, or a pixel inset indicating the *y* component of a point on the overlay image.
- **xunits** - Units in which the *x* value is specified. Default=**"fraction"**. A value of **"fraction"** indicates the *x* value is a fraction of the image. A value of **"pixels"** indicates the *x* value in pixels. A value of **"insetPixels"** indicates the indent from the right edge of the image.
- **yunits** - Units in which the *y* value is specified. Default=**"fraction"**. A value of **"fraction"** indicates the *y* value is a fraction of the image. A value of **"pixels"** indicates the *y* value in pixels. A value of **"insetPixels"** indicates the indent from the top edge of the image.

<screenXY>

Specifies a point relative to the screen origin that the overlay image is mapped to. The *x* and *y* values can be specified in three different ways: as *pixels* (**"pixels"**), as *fractions* of the screen (**"fraction"**), or as *inset pixels* (**"insetPixels"**), which is an offset in pixels from the upper right corner of the screen. The *x* and *y* positions can be specified in different ways—for example, *x* can be in pixels and *y* can be a fraction. The origin of the coordinate system is in the lower left corner of the screen.

- **x** - Either the number of pixels, a fractional component of the screen, or a pixel inset indicating the *x* component of a point on the screen.
- **y** - Either the number of pixels, a fractional component of the screen, or a pixel inset indicating the *y* component of a point on the screen.
- **xunits** - Units in which the *x* value is specified. Default=**"fraction"**. A value of **"fraction"** indicates the *x* value is a fraction of the screen. A value of **"pixels"** indicates the *x* value in pixels. A value of **"insetPixels"** indicates the indent from the right edge of the screen.

- **yunits** - Units in which the *y* value is specified. Default=*fraction*. A value of *fraction* indicates the *y* value is a fraction of the screen. A value of **"pixels"** indicates the *y* value in pixels. A value of **"insetPixels"** indicates the indent from the top edge of the screen.

For example, <*screenXY* x=".5" y=".5" xunits="fraction" yunits="fraction"/> indicates a point in the middle of the screen.

Here are some examples:

Center the image:

```
<ScreenOverlay>
  <overlayXY x="0.5" y="0.5" xunits="fraction" yunits="fraction"/>
  <screenXY x="0.5" y="0.5" xunits="fraction" yunits="fraction"/>
</ScreenOverlay>
```

Place the image on the top left:

```
<ScreenOverlay>
  <overlayXY x="0" y="1" xunits="fraction" yunits="fraction"/>
  <screenXY x="0" y="1" xunits="fraction" yunits="fraction"/>
</ScreenOverlay>
```

Place the image at the right of the screen:

```
<ScreenOverlay>
  <overlayXY x="1" y="1" xunits="fraction" yunits="fraction"/>
  <screenXY x="1" y="1" xunits="fraction" yunits="fraction"/>
</ScreenOverlay>
```

<rotationXY>

Point relative to the screen about which the screen overlay is rotated.

<size>

Specifies the size of the image for the screen overlay, as follows:

- A value of −1 indicates to use the native dimension
- A value of 0 indicates to maintain the aspect ratio
- A value of *n* sets the value of the dimension

For example:

To force the image to retain its original *x* and *y* dimensions, set the values to −1:

```
<size x="-1" y="-1" xunits="fraction" yunits="fraction"/>
```

To force the image to retain its horizontal dimension, but to take up 20 percent of the vertical screen space:

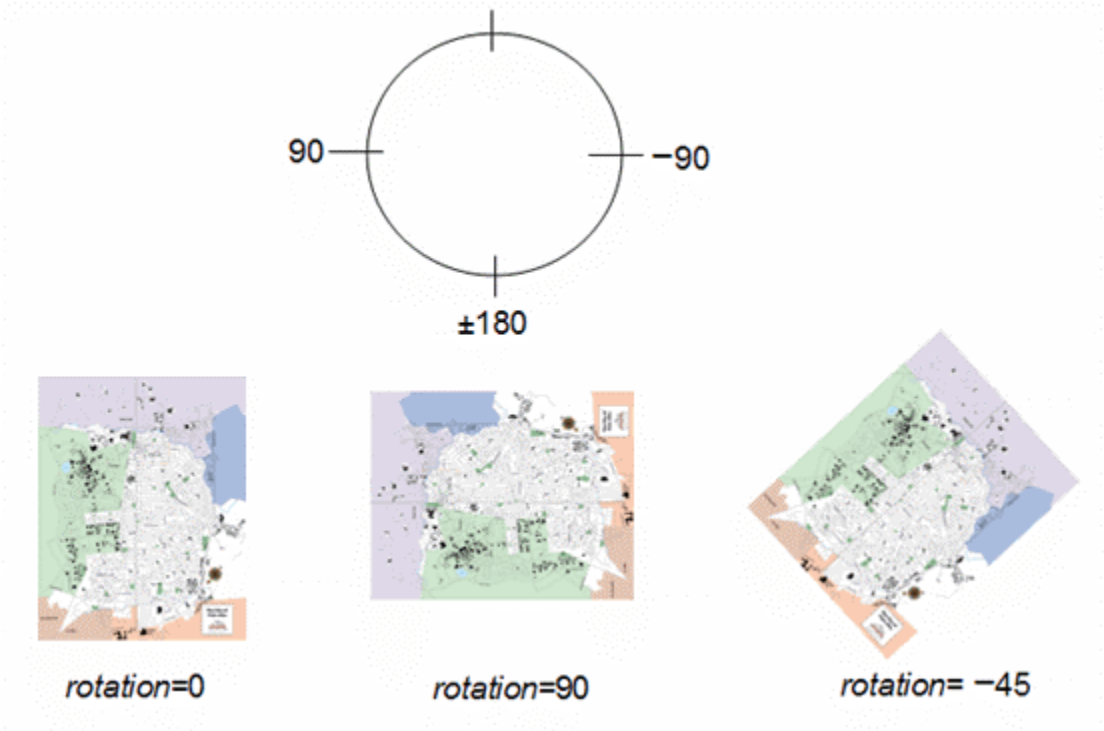
```
<size x="-1" y="0.2" xunits="fraction" yunits="fraction"/>
```

To force the image to resize to 100px by 500px:

```
<size x="100" y="500" xunits="pixels" yunits="pixels"/>
```

<rotation>

Indicates the angle of rotation of the parent object. A value of 0 means no rotation. The value is an angle in degrees counterclockwise starting from north. Use ±180 to indicate the rotation of the parent object from 0. The center of the <rotation>, if not (.5,.5), is specified in <rotationXY>.



7.31.4 Example

The following example places an image at the exact center of the screen, using the original width, height, and aspect ratio of the image.

```
<ScreenOverlay id="khScreenOverlay756">
  <name>Simple crosshairs</name>
  <description>This screen overlay uses fractional positioning
   to put the image in the exact center of the screen</description>
  <Icon>
    <href>http://myserver/myimage.jpg</href>
  </Icon>
  <overlayXY x="0.5" y="0.5" xunits="fraction" yunits="fraction"/>
  <screenXY x="0.5" y="0.5" xunits="fraction" yunits="fraction"/>
  <rotation>39.37878630116985</rotation>
  <size x="0" y="0" xunits="pixels" yunits="pixels"/>
</ScreenOverlay>
```

7.31.5 Extends

- *<Feature>*
- *<Overlay>*

7.31.6 Contained by

- <Document>
- <Folder>

7.32 <Style>

7.32.1 Syntax

```
<Style id="ID">
<!-- extends StyleSelector -->

<!-- specific to Style -->
  <IconStyle>...</IconStyle>
  <LabelStyle>...</LabelStyle>
  <LineStyle>...</LineStyle>
  <PolyStyle>...</PolyStyle>
  <BalloonStyle>...</BalloonStyle>
  <ListStyle>...</ListStyle>
</Style>
```

7.32.2 Description

A Style defines an addressable style group that can be referenced by StyleMaps and Features. Styles affect how Geometry is presented in the 3D viewer and how Features appear in the Places panel of the List view. Shared styles are collected in a <Document> and must have an **id** defined for them so that they can be referenced by the individual Features that use them.

Use an **id** to refer to the style from a <styleUrl>.

7.32.3 Elements specific to Style

- <BalloonStyle>
- <IconStyle>
- <LabelStyle>
- <LineStyle>
- <ListStyle>
- <PolyStyle>

7.32.4 Example

```
<Document>
  <!-- Begin Style Definitions -->
  <Style id="myDefaultStyles">
    <IconStyle>
      <color>a1ff00ff</color>
      <scale>1.399999976158142</scale>
      <Icon>
```

```
        <href>http://myserver.com/icon.jpg</href>
      </Icon>
    </IconStyle>
    <LabelStyle>
      <color>7fffaaff</color>
      <scale>1.5</scale>
    </LabelStyle>
    <LineStyle>
      <color>ff0000ff</color>
      <width>15</width>
    </LineStyle>
    <PolyStyle>
      <color>7f7faaaa</color>
      <colorMode>random</colorMode>
    </PolyStyle>
  </Style>
  <!-- End Style Definitions -->
  <!-- Placemark #1 -->
  <Placemark>
    <name>Google Earth - New Polygon</name>
    <description>Here is some descriptive text</description>
    <styleUrl>#myDefaultStyles</styleUrl>
    . . .
  </Placemark>
  <!-- Placemark #2 -->
  <Placemark>
    <name>Google Earth - New Path</name>
    <styleUrl>#myDefaultStyles</styleUrl>
      . . . .
  </Placemark>
</Document>
</kml>
```

7.32.5 Extends

- *<StyleSelector>*

7.32.6 Contained by

- any *<Feature>*

7.33 <StyleMap>

7.33.1 Syntax

```
<StyleMap id="ID">
  <!-- extends StyleSelector -->
  <!-- elements specific to StyleMap -->
  <Pair id="ID">
    <key>normal</key>              <!-- kml:styleStateEnum:  normal or
highlight
    <styleUrl>...</styleUrl>       <!-- anyURI -->
  </Pair>
</StyleMap>
```

7.33.2 Description

A <StyleMap> maps between two different icon styles. Typically a <StyleMap> element is used to provide separate normal and highlighted styles for a placemark, so that the highlighted version appears when the user mouses over the icon in Google Earth.

7.33.3 Elements specific to StyleMap

<Pair> *(required)*

Defines a key/value pair that maps a mode (*normal* or *highlight*) to the predefined <styleUrl>. <Pair> contains two elements (both are required):

- **<key>**, which identifies the key
- **<styleUrl>**, which references the style. For referenced style elements that are local to the KML document, a simple # referencing is used. For styles that are contained in external files, use a full URL along with # referencing.

  For example:

```
<Pair>
  <key>normal</key>

<styleUrl>http://myserver.com/populationProject.xml#example_style_off</
styleUrl>
</Pair>
```

7.33.4 Example

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <name>StyleMap.kml</name>
```

```
  <open>1</open>
  <Style id="normalState">
    <IconStyle>
      <scale>1.0</scale>
      <Icon>

<href>http://maps.google.com/mapfiles/kml/pal3/icon55.png</href>
      </Icon>
    </IconStyle>
    <LabelStyle>
      <scale>1.0</scale>
    </LabelStyle>
  </Style>
  <Style id="highlightState">
    <IconStyle>
      <Icon>

<href>http://maps.google.com/mapfiles/kml/pal3/icon60.png</href>
      </Icon>
      <scale>1.1</scale>
    </IconStyle>
    <LabelStyle>
      <scale>1.1</scale>
      <color>ff0000c0</color>
    </LabelStyle>
  </Style>
  <StyleMap id="styleMapExample">
    <Pair>
      <key>normal</key>
      <styleUrl>#normalState</styleUrl>
    </Pair>
    <Pair>
      <key>highlight</key>
      <styleUrl>#highlightState</styleUrl>
    </Pair>
  </StyleMap>
  <Placemark>
    <name>StyleMap example</name>
    <styleUrl>#styleMapExample</styleUrl>
    <Point>
      <coordinates>-122.368987,37.817634,0</coordinates>
    </Point>
  </Placemark>
</Document>
</kml>
```

7.33.5 Extends

- *<StyleSelector>*

7.33.6 Contained By

- any *<StyleSelector>*

7.34 *<StyleSelector>*

7.34.1 Syntax

Syntax

```
<!-- abstract element; do not create -->
<!-- StyleSelector id="ID" -->                    <!-- Style,StyleMap -->
<!-- /StyleSelector -->
```

7.34.2 Description

This is an abstract element and cannot be used directly in a KML file. It is the base type for the <Style> and <StyleMap> elements. The StyleMap element selects a style based on the current mode of the Placemark. An element derived from StyleSelector is uniquely identified by its **id** and its *url*.

7.34.3 Elements specific to StyleSelector

This abstract element does not contain any child elements.

7.34.4 Example

None.

7.34.5 Extends

- *<Object>*

7.34.6 Extended By

- <Style>
- <StyleMap>

7.35 *<TimePrimitive>*

7.35.1 Syntax

```
<!-- abstract element; do not create -->
<!-- TimePrimitive id="ID" -->            <!-- TimeSpan,TimeStamp -->
  <!-- extends Object -->
<!-- /TimePrimitive -->
```

7.35.2 Description

This is an abstract element and cannot be used directly in a KML file. This element is extended by the <TimeSpan> and <TimeStamp> elements.

7.35.3 Elements specific to TimePrimitive

None.

7.35.4 Example

None.

7.35.5 Extends

- <Object>

7.35.6 Extended By

- <TimeSpan>
- <TimeStamp>

7.36 <TimeSpan>

7.36.1 Syntax

```
<TimeSpan id="ID">
  <begin>...</begin>     <!-- kml:dateTime -->
  <end>...</end>         <!-- kml:dateTime -->
</TimeSpan>
```

7.36.2 Description

Represents an extent in time bounded by begin and end *dateTimes*.

If <begin> or <end> is missing, then that end of the period is unbounded (see Example below).

The *dateTime* is defined according to XML Schema time (see XML Schema Part 2: Datatypes Second Edition). The value can be expressed as *yyyy-mm-dd*T*hh:mm:ss*zzzzzz, where T is the separator between the date and the time, and the time zone is either Z (for UTC) or *zzzzzz*, which represents ±*hh:mm* in relation to UTC. Additionally, the value can be expressed as a date only. See <TimeStamp> for examples.

7.36.3 Elements specific to TimeSpan

<begin>

Describes the beginning instant of a time period. If absent, the beginning of the period is unbounded.

<end>

Describes the ending instant of a time period. If absent, the end of the period is unbounded.

7.36.4 Example

The following example shows the time period representing California's statehood. It contains only a <begin> tag because California became a state on August 1, 1876, and continues to be a state:

```
<Placemark>
  <name>California</name>
  .
```

```
  .
  .
  <TimeSpan>
    <begin>1876-08-01</begin>
  </TimeSpan>
</Placemark>
```

7.36.5 Extends

- *<TimePrimitive>*

7.36.6 Contained by

- any element derived from *<Feature>*

7.37 <TimeStamp>

7.37.1 Syntax

```
<TimeStamp id=ID>
  <when>...</when>      <!-- kml:dateTime -->
</TimeStamp>
```

7.37.2 Description

Represents a single moment in time. This is a simple element and contains no children. Its value is a *dateTime*, specified in XML time (see XML Schema Part 2: Datatypes Second Edition). The precision of the TimeStamp is dictated by the *dateTime* value in the <when> element.

7.37.3 Elements specific to TimeStamp

<when>

Specifies a single moment in time. The value is a *dateTime*, which can be one of the following:

- *dateTime* gives second resolution
- *date* gives day resolution
- *gYearMonth* gives month resolution
- *gYear* gives year resolution

The following examples show different resolutions for the <when> value:

- *gYear* (*YYYY*)

```
<TimeStamp>
  <when>1997</when>
</TimeStamp>
```

- *gYearMonth* (*YYYY-MM*)

```
<TimeStamp>
  <when>1997-07</when>
</TimeStamp>
```

- *date* (*YYYY-MM-DD*)

```
<TimeStamp>
  <when>1997-07-16</when>
</TimeStamp>
```

- *dateTime* (*YYYY-MM-DD***T***hh:mm:ss***Z**)

Here, T is the separator between the calendar and the hourly notation of time, and Z indicates UTC. (Seconds are required.)

```
<TimeStamp>
  <when>1997-07-16T07:30:15Z</when>
</TimeStamp>
```

- *dateTime* (*YYYY-MM-DD***T***hh:mm:sszzzzzz)*
  This example gives the local time and then the ± conversion to UTC.

```
<TimeStamp>
  <when>1997-07-16T10:30:15+03:00</when>
</TimeStamp>
```

7.37.4 Examples

See above.

7.37.5 Extends

- *<TimePrimitive>*

7.37.6 Contained by

- any element that extends *<Feature>*

7.38 <Update>

7.38.1 Syntax

```
<Update>
  <targetHref>...<targetHref>    <!-- URL -->
  <Change>...</Change>
  <Create>...</Create>
  <Delete>...</Delete>
</Update>
```

7.38.2 Description

Specifies an addition, change, or deletion to KML data that has already been loaded using the specified URL. The <targetHref> specifies the *.kml* or *.kmz* file whose data (within Google Earth) is to be modified. <Update> is always contained in a NetworkLinkControl. Furthermore, the file containing the NetworkLinkControl must have been loaded by a NetworkLink. See the KML 2.1 Tutorial for a detailed example of how Update works.

7.38.3 Elements specific to Update

Can contain any number of <Change>, <Create>, and <Delete> elements, which will be processed in order.

<targetHref> *(required)*

A URL that specifies the *.kml* or *.kmz* file whose data (within Google Earth) is to be modified by an <Update> element. This KML file must already have been loaded via a <NetworkLink>. In that file, the element to be modified must already have an explicit **id** attribute defined for it.

<Change>

Modifies the values in an element that has already been loaded with a <NetworkLink>. Within the Change element, the child to be modified must include a **targetId** attribute that references the original element's **id**.

This update can be considered a "sparse update": in the modified element, only the values listed in <Change> are replaced; all other values remained untouched. When <Change> is applied to a set of coordinates, the new coordinates replace the current coordinates.

Children of this element are the element(s) to be modified, which are identified by the **targetId** attribute.

<Create>

Adds new elements to a Folder or Document that has already been loaded via a

<NetworkLink>. The <targetHref> element in <Update> specifies the URL of the *.kml* or *.kmz* file that contained the original Folder or Document. Within that file, the Folder or Document that is to contain the new data must already have an explicit **id** defined for it. This **id** is referenced as the **targetId** attribute of the Folder or Document within <Create> that contains the element to be added.

Once an object has been created and loaded into Google Earth, it takes on the URL of the original parent Document of Folder. To perform subsequent updates to objects added with this Update/Create mechanism, set <targetHref> to the URL of the original Document or Folder (not the URL of the file that loaded the intervening updates).

<Delete>

Deletes features from a complex element that has already been loaded via a <NetworkLink>. The <targetHref> element in <Update> specifies the *.kml* or *.kmz* file containing the data to be deleted. Within that file, the element to be deleted must already have an explicit **id** defined for it. The <Delete> element references this **id** in the **targetId** attribute.

Child elements for <Delete>, which are the only elements that can be deleted, are Document, Folder, GroundOverlay, Placemark, and ScreenOverlay.

7.38.4 Examples

Example of <Change>

```
<NetworkLinkControl>
  <Update>
    <targetHref>http://www/~sam/January14Data/Point.kml</targetHref>
    <Change>
      <Point targetId="point123">
        <coordinates>-95.48,40.43,0</coordinates>
      </Point>
    </Change>
  </Update>
</NetworkLinkControl>
```

Example of <Create>

This example creates a new Placemark in a previously created Document that has an **id** of "region24." Note that if you want to make subsequent updates to "placemark891," you will still use *http://myserver.com/Point.kml* as the <targetHref>.

```
<Update>
  <targetHref>http://myserver.com/Point.kml</targetHref>
  <Create>
    <Document targetId="region24">
      <Placemark id="placemark891">
        <Point>
          <coordinates>-95.48,40.43,0</coordinates>
        </Point>
      </Placemark>
    </Document>
```

```
  </Create>
</Update>
```

Example of <Delete>

This example deletes a Placemark previously loaded into Google Earth. (This Placemark may have been loaded directly by a NetworkLink with the specified URL, or it may have been loaded by a subsequent Update to the original Document.)

```
<Update>
  <targetHref>http://www.foo.com/Point.kml</targetHref>
  <Delete>
    <Placemark targetId="pa3556"></>
  </Delete>
</Update>
```

7.38.5 Extends

N.A.

7.38.6 Contained by

- <NetworkLinkControl>

7.39 <Url>

7.39.1 Syntax

N.A.

7.39.2 Description

NOTE: This element is deprecated in KML Release 2.1 and has been replaced by <Link>, which provides the additional functionality of Regions. The <Url> tag will still work in Google Earth, but use of the newer <Link> tag is encouraged.

Use this element to set the location of the link to the KML file, to define the refresh options for the server and viewer changes, and to populate a variable to return useful client information to the server.

7.39.3 Elements specific to

7.39.4 Examples

7.39.5 Extends

7.39.6 Contained by

Annex A
(normative)

Annex title

Annex B
(informative)