

Globe Component Description



Welcome to TGlobe

- **TGlobe** is a 3D Interactive Mapping/GIS component for Borland Delphi* written in 100% native Delphi code.
- **TGlobe** displays a three-dimensional Spherical, Mercator or Cartesian projection of the world and allows you to interact in realtime with the countries and objects displayed.
- **TGlobe** has a built in, configurable zoom and rotation features, can manage multiple layers of data and supports Thematic mapping.
- The globe data is stored to an accuracy of one **1500th of a second of a degree**, that is **TGlobe** is able to display data points that are approximately 1 inch apart in the physical world.

[Installation](#)

[Component Reference](#)

[New Features](#)

[Version Control](#)

[Screen Shots](#)

[Order Source Code](#)

[Copyright and Disclaimer](#)

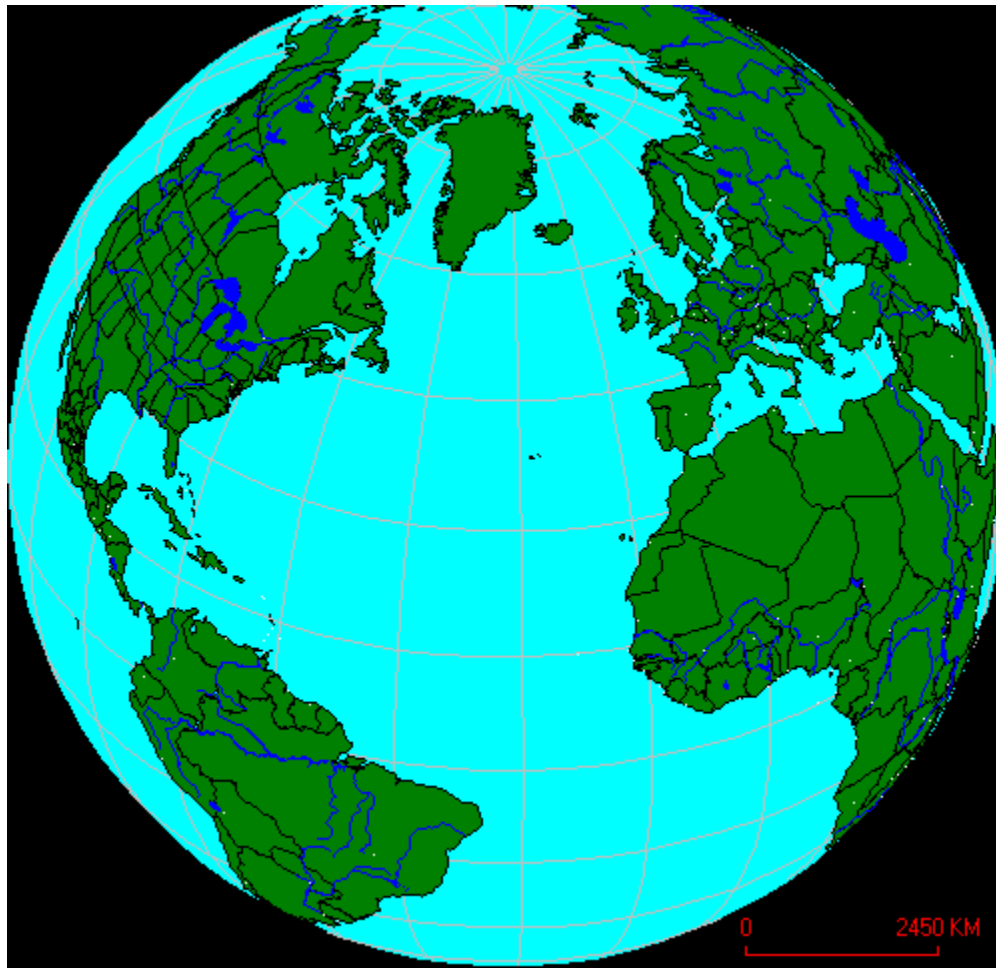
For technical support, comments and suggestions Please contact **Graham Knight** at

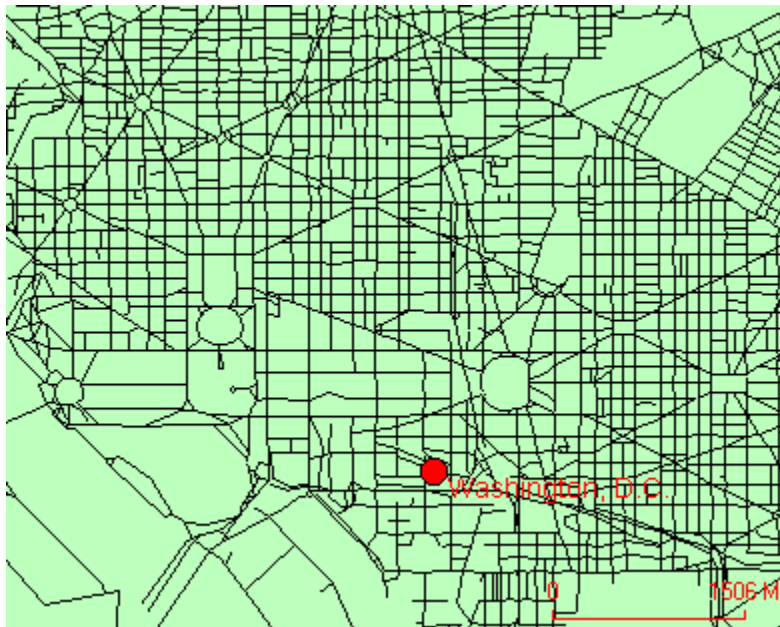
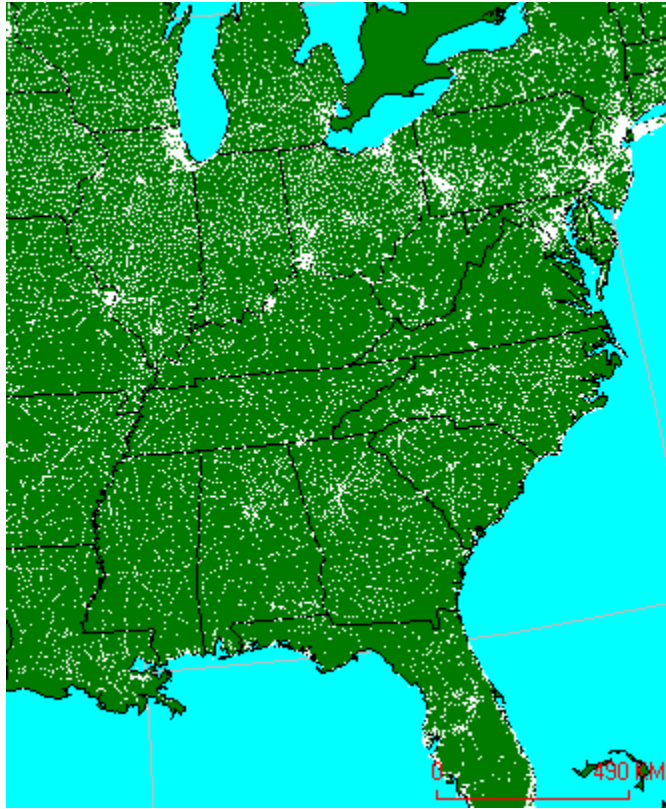
Email: gknight@helmstone.co.uk

Please specify the version of TGlobe that you are using and the version of Delphi you are developing under with any technical queries.

* Borland Delphi copyright of Borland International, Inc.

Screen Shots







Ordering TGlobe Source

If you would like to register or order the TGlobe source code then fill in the form below. Please contact me if you have any questions.

Email: gknight@helmstone.co.uk

TGlobe costs just **£ 25 GBP** to register and a further **£ 25 GBP** to receive the Source code to the TGlobe component.

The registered version has the copyright message removed, and includes the source for the Map Editor.

If you cannot send a cheque in Sterling then add **£ 5 GBP** to the amount and send a cheque converted into your local currency, e.g. the equivalent of **£ 30 GBP** or **£ 55 GBP**. The extra **£ 5 GBP** is a flat fee charged by my Bank to convert a foreign cheque to Sterling.

Please supply the following information with your registration. Your email address is very important so that I can send notifications of changes and updates to you.

Customer Information:

Name..... :
Company..... :
Address..... :
City..... :
State..... :
ZIP.(Postal Code). :
Country..... :

EMail..... :

Shipping & Handling:

Normally I will send the registered and source versions by email once the registration has been received. If you require the software to be sent to you by floppy disk and snail mail then please add a further **£ 5 GBP** to cover the cost of shipping and handling.

<input type="checkbox"/> TGlobe component registration only :	£ 25 GBP	\$ 49 USD
<input type="checkbox"/> TGlobe component registration + source :	£ 50 GBP	\$ 90 USD
<input type="checkbox"/> If floppy disk shipping is required please add	£ 5 GBP	\$ 8 USD

Please make cheques payable to **Graham Knight** and send the remittance to :

**Graham Knight
54 Matlock Road
Brighton
Sussex BN1 5BF
England**

For purchases by bank transfer please contact me for information.

Email: gknight@helmstone.co.uk

New Features

Version 2.20 has the following changes and features

- Integrated Help file with context sensitive footnotes.
- Support for Thematic mapping (see [TGlobe.OnRenderAttributes](#) event)
- [TGlobePen.PenWidth](#) property to control the width of lines drawn with the pen.
- [TGlobeObject.UserObject](#) property so the object can have an external object associated with it.
- The **ZoomMin** and **ZoomMax** profile properties now work by measuring the visible distance in the Globe window.
- Improved rotation of the sphere code to avoid Gimbal lock.
- [ScaleUnitsToDevice\(\)](#) now replaces all other **ScaleToDevice()** functions.
- [GlobeUnitsToDecimal\(\)](#) function to mirror [DecimalToGlobeUnits\(\)](#) function.
- Added [EllipsoidDistanceLLToLL\(\)](#) Calculates the distance between two points using an ellipsoid model of the earth (WGS84) for more accurate results.
- All defined constants that represent values in GlobeUnits now start with GU_.
- [TGlobe.Print](#) sends the globe image to the default printer.
- [TGlobe.OnObjectCreate](#) and [TGlobe.OnObjectFree](#) events
- [TGlobe.ProgressMessage\(\)](#) Method to enable progress messages to be passed to the application during long processing sessions.
- [TGlobe.RenderGreatCircleLine](#) renders a great circle line from one coordinate to another.

Version 2.10 has the following features

- This version has had lots of changes even since the last beta so I would recommend that this document is examined carefully so that you are fully aware of the changes and new features and concepts.
- Overlays have been removed as a concept and has been replaced by Layers. Now the globe data is loaded directly into layers. Each Layer is held in a separate globe data file. This makes managing the layer data much easier, You can control which layers are loaded and even adjust the order of loaded layers.
- The Profile file controls the loading of Layer files and options associated with each Layer e.g. Visible or Hidden, Maximum and Minimum zoom visibility, maximum text height and font scaling.
- Added Animation layers to Profile options. Animation layers are rendered after the non animation layers have been displayed on the screen, this means that moving objects on the animation layers display very quickly as the underlying globe data does not need to be re-generated for each movement of the the animated objects.
- Two new object types
- [TGlobeSymbol](#) displays a single character symbol from a True Type font.
- [TGlobeGrid](#), displays a grid at the specified location with definable major and minor grid steps.
- The **TGlobe.GlobeFile** property has been replaced by [TGlobe.ProfileName](#).
- Added [TGlobe.GlobeUnitsPerInch](#) property to set and read the scale of the globe in GlobeUnits.
- Added [TGlobeObject.Layer](#) : TGlobeLayer returns the Layer structure associated with the object.
- Added [TLLPointList.AddLL\(\)](#) to simplify the creation of PointLists for polyline and polygon objects.
- [DecimalToGlobeUnits\(\)](#) converts a Latitude or Longitude coordinate in floating point or decimal representation to TGlobe GlobeUnits.
- [GlobeUnitsToDecimal\(\)](#) does the opposite of the above.
- [DistanceLLToLL\(\)](#) Calculates the distance between two points.

Version Control

- 2.20
 - Include Context sensitive help file, Print facility, Thematic mapping support
- 2.11
 - TGlobePen now has a Width attribute.
- 2.10
 - Production Version - many changes and fixes.
- 2.07beta
 - Fixed code for 16bit version, Added Profile code. Added TGlobeLayer Object.
- 2.06beta
 - Added TLLPointList Object to make defining and manipulating polyline/polygon objects easier.
- 2.05beta
 - Bugfixes to Mercator projection.
- 2.04beta
 - Fixes to SaveToFile method.
- 2.03beta
 - Added ProjectionModel to allow the easier addition of new projections.
- 2.02beta
 - Performace enhancements and much improved resolution to 1000th of a second of arc.
 - Internal projection model to allow easy development of new types of data projections.
- 2.00beta
 - Complete re-build to make the internals of TGlobe properly object orientated.
 - Now much easier to add new types of display objects to the globe.
 - Added many new functions to facilitate interaction with the globe.

Component Installation

If you already have a copy of TGlobe installed then you should remove the component from the library before installing the updated component and compiling any applications.

The TGlobe component must be installed before compiling and running the Demo application. If you do not install the component you will receive errors when you try to compile and run the Demo application.

The components for the Delphi 1 IDE comprise of the following files:

- Globe1.dcu

- Globe1.dcr

The components for the Delphi 2 IDE comprise of the following files:

- Globe2.dcu

- Globe2.dcr

and the Delphi 3 IDE :

- Globe3.dcu

- Globe3.dcr

Copy these files to the installation to directory of your choice then install the components into your IDE before attempting to compile and run the Demo program.

The Globe component installs itself onto the Samples Palette.

The rest of the files are required to build and run the Demo program. The TGLOBE.PRF file and all of the *.DAT files should be placed in the same directory as the Demo program so that it can be loaded on startup.

Installing the TGLOBE component in the Delphi 1 IDE:

- Select Options | Install components from Delphi's menu.

- Click the Add button to open the Add Module dialog box.

- Click the Browse button to open the Add Module file selection dialog box.

- Select the full drive and directory name path to the globe1.dcu for the Delphi 1 IDE and click the OK button.

Installing the TGLOBE component in the Delphi 2 IDE:

- Select Options | Install components from Delphi's menu.

- Click the Add button to open the Add Module dialog box.

- Click the Browse button to open the Add Module file selection dialog box.

- Select the full drive and directory name path to the globe2.dcu for the Delphi 2 IDE and click the OK button.

To install under the Delphi 3 IDE:

- Select component | Install component from Delphi's menu.

- Click the Browse button and locate the globe3.dcu file.

- Select the full drive and directory name path to the globe.dcu and click the open button.

Copyright and Disclaimer

Copyright © 1997,1998 Graham Knight. All rights reserved.

Disclaimer

The Author cannot and does not guarantee that any functions contained in the Software will meet your requirements, or that its operations will be error free. The entire risk as to the Software performance or quality, or both, is solely with the user and not the Author. You assume responsibility for the selection of the component to achieve your intended results, and for the installation, use, and results obtained from the Software.

The Author makes no warranty, either implied or expressed, including without limitation any warranty with respect to this Software documented here, its quality, performance, or fitness for a particular purpose. In no event shall the Author be liable to you for damages, whether direct or indirect, incidental, special, or consequential arising out the use of or any defect in the Software, even if the Author has been advised of the possibility of such damages, or for any claim by any other party.

All other warranties of any kind, either express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose, are expressly excluded.

Globe Object Hierachy

The TGlobe component is actually a manager component for a collection of container objects. The globe component holds a number of Layers each layer contains a collection of globe objects and their associated attributes. Each layer is independant of any other layer, but the TGlobe component can bring all of the layers together. For example the method `Globe.ObjectAtXY(X, Y)`; will search each layer in turn looking for an object at the specified location.

```
TGlobe = class( TCustomControl )
TGlobeLayer = class( TObject )
TLLPointList = class( TObject )

* TGlobePersistant = class( TObject )
    * TGlobeAttribute = class( TGlobePersistant )
        * TGlobePen = class( TGlobeAttribute )
        * TGlobeBrush = class( TGlobeAttribute )
        * TGlobeFont = class( TGlobeAttribute )
    * TGlobeObject = class( TGlobePersistant )
        TGlobeText = class( TGlobeObject )
        TGlobePointText = class( TGlobeText )
        TGlobeSymbol = class( TGlobeText )
        TGlobePolyLine = class( TGlobeObject )
        TGlobePolyGon = class( TGlobePolyLine )
        TGlobeGrid = class( TGlobeObject )
```

Objects marked with an * are internal to the operation of TGlobe and should never be instantiated by the user.

The attribute objects, TGlobePen, TGlobeBrush and TGlobeFont should not be instantiated by the user, these should always be created using the `Class.Define()` method. This method will create a new instance of the object if necessary otherwise it will return an existing object if it exactly matches the supplied parameters.

Types and Exceptions

Types

[TDrawGrid](#)

[TGlobeMouseButton](#)

[TGlobeObjectState](#)

[TGlobeProgressEvent](#)

[TGlobeProjection](#)

[TGlobeRenderAttributesEvent](#)

[TGlobeObjectNotifyEvent](#)

[TGlobeTitle](#)

[TGlobeUnits](#)

[TGreatCircle](#)

[TPointLL](#)

Exceptions

EGlobeException = class(Exception);

Globe Constants

The following constants are defined in the TGlobe module.

Constant	Value	Meaning	16 bit
GLOBE_RESOLUTION	= (360*60*60*1500) or (360 * 60) for 16bit	Maximum resolution of the TGlobe object.	Yes
GU_DEGREE	= GLOBE_RESOLUTION div 360		One degree in G
GU_MINUTE	= GU_DEGREE div 60	One Minute of Arc in GlobeUnits.	Yes
GU_SECOND	= GU_DEGREE div 3600	One Second of Arc in GlobeUnits.	No
GU_THIRD	= GU_DEGREE div 21600	One Third of a second of Arc in GlobeUnits.	No
GU_KILOMETER	= GU_MINUTE / 1.852	One Kilometer in GlobeUnits.	No
GU_METER	= GU_KILOMETER / 1000.0	One Meter in GlobeUnits.	No
GU_CENTIMETER	= GU_METER / 100.0	One Centimeter in GlobeUnits.	No
GU_NAUTICALMILE	= GU_MINUTE	One Nautical Mile in GLOBEUnits	Yes
GU_MILE	= GU_KILOMETER * 1.609344	One Mile in GlobeUnits.	No
GU_YARD	= GU_MILE / 1760	One Yard in GlobeUnits	No
GU_FOOT	= GU_YARD / 3	One Foot in GlobeUnits.	No
GU_INCH	= GU_FOOT / 12	One Inch in GlobeUnits.	No
GU_360_DEGREE	= (360 * GU_DEGREE)	360 degrees in GlobeUnits.	Yes
GU_180_DEGREE	= (180 * GU_DEGREE)	180 degrees in GlobeUnits.	Yes
GU_90_DEGREE	= (90 * GU_DEGREE)	90 degrees in GlobeUnits.	Yes
PolarRadius	= 6356752.3142451793 * GU_METER		
EquatorialRadius	= 6378137.0 * GU_METER		
Flattening	= abs((EquatorialRadius - PolarRadius) / EquatorialRadius)		
EccentricitySquared	= (2.0 * Flattening) - (Flattening * Flattening)		
EarthRadius	= 3437.7 * GU_MINUTE Yes	In minutes of a degree or nautical miles	
MAXPOINTS	= 10000 Yes	Maximum number of points in an object	

Utility Routines

The following utility routines are defined in the TGlobe module.

[AngleToRadians](#)
[DecimalToGlobeUnits](#)
[DistanceLLToLL](#)
[EllipsoidDistanceLLToLL](#)
[GlobeUnitsFrom](#)
[GlobeUnitsTo](#)
[GlobeUnitsToDecimal](#)
[LLAsString](#)
[LLToStr](#)
[MaxVal](#)
[MinVal](#)
[PointLL](#)
[RadiansToAngle](#)
[sIFE](#)
[StrToLL](#)
[StrToUnits](#)
[UnitsToStr](#)

Creating New Globe Object Classes

Here is an example of creating a simple globe object. This object displays a scaled triangle at the origin location for the object, this could be used to plot an object moving about the globe e.g. a ship or plane. The displayed triangle is not projected onto the surface of the globe. To plot the object directly onto the surface of the globe the `Globe.RenderPolygon()` method should be used.

This object can be saved with the data file. The object needs to be registered with the Globe object using the `Globe.RegisterGlobeClass()` method, this should be done in the `FormCreate` method before any loading or saving of globe data files has occurred.

If the object does not need to be saved to a data file then it is not necessary to implement the `.SaveToStream` and `.LoadFromStream` methods. It is also important that the `Globe.RegisterGlobeClass()` method is not called so that the Globe object does not try to save or load this class.

The `CalcProjectionData` method is implemented so that the object can be selected using the mouse. If the object is for display purposes only then this method is not required.

```
TGlobeTriangle = class(TGlobeObject)
public
    FPen : TGlobePen;
    FBrush : TGlobeBrush;
    FFont : TGlobeFont;

    constructor Create( ParentLayer : TGlobeLayer; const sTitle : string;
        iLayer : integer; iLong, iLat : integer );

    procedure Render; override;
    procedure CalcProjectionData; override;
    function AttributeList( var aAttrs : TGlobeAttributeArray ) : integer;
override;
    procedure LoadFromStream( AStream : TStream ); override;
    procedure SaveToStream( AStream : TStream ); override;
end;

constructor TGlobeTriangle.Create( ParentLayer : TGlobeLayer;
    const sTitle : string; iLong, iLat : integer );
begin
    inherited Create( ParentLayer, sTitle, iLong, iLat );

    with ParentLayer.Globe do
    begin
        FPen := TGlobePen.Define( ParentLayer, Pen.Color, Pen.Style );
        FBrush := TGlobeBrush.Define( ParentLayer, Brush.Color, Brush.Style );
        FFont := TGlobeFont.Define( ParentLayer, 'Arial', Font.Color, FontUnit,
            Font.Size, Font.Angle, Font.Style );
    end;
end;

procedure TGlobeTriangle.Calcprojectiondata;
begin
    with Origin do
        BoundsRect := Rect( X - 30, Y - 30, X + 30, Y + 30 );
```

```
end;
```

```
function TGlobeTriangle.AttributeList( var aAttrs : TGlobeAttributeArray ) :  
integer;  
begin  
    aAttrs[0] := FPen;  
    aAttrs[1] := FBrush;  
    aAttrs[2] := FFont;  
    Result := 3  
end;
```

```
procedure TGlobeTriangle.Render;  
var  
    iTmp : integer;  
    pt : TPoint;  
begin  
    with Overlay.Globe do  
        begin  
            RenderAttributes( State, [ FPen, FBrush, FFont ] );  
  
            LLToDeviceXY( Origin.X, Origin.Y, pt );  
            with pt do  
                begin  
                    { Scale the size to the globe }  
                    iTmp := ScaleMinutesToDevice( 30 ) + 1;  
  
                    GlobeCanvas.Polygon( [  
                        Point( X , Y + iTmp ), Point( X - iTmp, Y - iTmp ),  
                        Point( X + iTmp, Y - iTmp ) ] );  
  
                    SetBkMode( GlobeCanvas.Handle, TRANSPARENT );  
                    GlobeCanvas.TextOut( X - iTmp, Y + iTmp, Title );  
                    SetBkMode( GlobeCanvas.Handle, OPAQUE );  
                end;  
            end;  
        end;  
    end;
```

```
procedure TGlobeTriangle.LoadFromStream( AStream : TStream );  
{-----}  
function ReadSmallInt : SmallInt;  
begin  
    AStream.Read( Result, SizeOf( SmallInt ) );  
end;  
begin  
    inherited LoadFromStream( AStream );  
  
    FPen := TGlobePen( Layer.Attributes[ReadSmallInt] );  
    FBrush := TGlobeBrush( Layer.Attributes[ReadSmallInt] );  
    FFont := TGlobeFont( Layer.Attributes[ReadSmallInt] );  
end;
```

```
procedure TGlobeTriangle.SaveToStream( AStream : TStream );  
{-----}
```

```
    procedure WriteSmallInt( iVal : SmallInt );
    begin
        AStream.Write( iVal, SizeOf( SmallInt ));
    end;
begin
    inherited SaveToStream( AStream );

    { only save the references to the attributes }
    WriteSmallInt( FPen.Index );
    WriteSmallInt( FBrush.Index );
    WriteSmallInt( FFont.Index );
end;
```


Globe Unit

[Object hierarchy](#)

[Types and exceptions](#)

[Constants](#)

The TGlobe component is actually a manager component for a collection of container objects. The globe component holds a number of Layers each layer contains a collection of globe objects and their associated attributes. Each layer is independant of any other layer, but the TGlobe component can bring all of the layers together. For example the method `Globe.ObjectAtXY(X, Y);` will search each layer in turn looking for an object at the specified location.

For the 32 bit versions of the component (Delphi 2 and 3), Longitude and Latitude are defined in 1500th of a second of arc. This gives a resolution for plotting points down to less than 1 inch apart. All points held internally by the component are stored at this resolution. Therefore internal points are store in GlobeUnits (i.e. 1500th of a Second of arc).

For the 16 bit version (Delphi 1) of the component Longitude and Latitude are define in minutes of a degree which gives an accuracy of one Nautical mile. GlobeUnits are therefore defined as one minute of arc.

The supplied data files are defined in minutes of a degree, this allows reasonable accuracy i.e. a point on the globe can be specified to an accuracy of one nautical mile and all points on the globe can fit within a 16 bit integer value. For the 32bit version these data points are converted to GlobeUnits when they are loaded..

Longitude can range between -180 degrees to +180 degrees about the Greenwich Meridian and Latitude can range between -90 degrees to +90 degrees about the equator.

TLLPointList object

[See also](#)

[Properties](#)

[Methods](#)

Unit

[Globe](#)

Description

This object is used to create and maintain a list of points that are associated with the TGlobePolyline and TGlobePolygon objects. If the list of points associated with an object is modified then the object.RedrawObject method should be called to ensure that the object is correctly updated on the screen.

See also

[TGlobePolyline](#)
[TGlobePolygon](#)

Properties

▶ Run-time only

🔑 Key properties



Capacity



Count











LLArray



Points

Methods

 Key methods

	<u>Destroy</u>
	<u>Add</u>
	<u>AddLL</u>
	<u>Assign</u>
	<u>Clear</u>
	<u>Delete</u>
	<u>Insert</u>
	<u>LoadFromStream</u>
	<u>SaveToStream</u>

Capacity property

Applies to

TLLPointList object

Declaration

property Capacity: integer;

Description

This is the number of points that can be stored in the object without allocating more memory. This can be set to allocate space for points before they are written to the list for performance reasons. If this is set the new capacity must be greater than the count.

Run-time only

Count property

Applies to

TLLPointList object

Declaration

property Count: integer;

Description

The number of points currently held in the list. This can be set to any value, if set to a value greater than the current capacity of the object then the capacity is automatically increased.

Run-time only

LLArray property

Applies to

[TLLPointList](#) object

Declaration

property LLArray: PTPointLLArray;

Description

A pointer to the allocated array of points. This gives direct access to the array of points, the [Count](#) property should be used to avoid overrunning the end of the array.

Run-time only

Read-only

Points property

Applies to

TLLPointList object

Declaration

property Points: TPointLL;

Description

Index access to each point in the list. This is the default property for the object therefore you can access all of the point in the object as if they are in an array e.g.

PointLL := PointList.Points[10];

Run-time only

Add method

Applies to

TLLPointList object

Declaration

```
function Add(ptLL: TPointLL): integer;
```

Description

Appends a point to the end of the List. The capacity of the list is increased as necessary to accomodate the new point.

AddLL method

Applies to

[TLLPointList](#) object

Declaration

```
function AddLL(iLong, iLat: integer): integer;
```

Description

Appends a point to the end of the List, the point is specified by passing its longitude and latitude. The capacity of the list is increased as necessary to accomodate the new point.

Assign method

Applies to

TLLPointList object

Declaration

procedure Assign(aNewList: TLLPointList);

Description

Copies the contents of the supplied PointList to the current list. Any points in the current list will be overwritten.

Clear method

Applies to

TLLPointList object

Declaration

```
procedure Clear;
```

Description

Clears the object and deletes all points and allocated memory.

Delete method

Applies to

TLLPointList object

Declaration

procedure Delete(iIndex: integer);

Description

Deletes the indexed point from the list.

Insert method

Applies to

TLLPointList object

Declaration

procedure Insert(iIndex: integer; ptLL: TPointLL);

Description

Inserts the point into the position specified by the Index.

LoadFromStream method

Applies to

TLLPointList object

Declaration

procedure LoadFromStream(AStream: TStream; Origin: TPointLL);

Description

Method to load a list of points from the data file into an array.

SaveToStream method

Applies to

TLLPointList object

Declaration

```
procedure SaveToStream(AStream: TStream);
```

Description

Method to save the array of points to the data file.

TPrjModel object

[Properties](#)

[Methods](#)

Unit

[Globe](#)

Description

This abstract object is the base class for the objects that control the interpretation of the globe data and transforms it from logical coordinates to physical coordinates.

Properties

▶ Run-time only

▶ ExtentsLL

▶ GlobeUnitsPerInch

 PointsPerPixel

 ScaleFactor

 XOrigin

 XRotation

 YOrigin

 YRotation

 ZRotation

Methods

[ArrayLLToXY](#)

[LLToXY](#)

[PaintBackground](#)

[RectLLVisible](#)

[XYToLL](#)

[UpdateModel](#)

ExtentsLL property

Applies to

TPriModel object

Declaration

property ExtentsLL: TRect;

Description

The logical extents of the image to display. This is used to control the zooming of the projection data when zooming to see the maximum possible.

For a sphere this is :-

-EARTH_RADIUS, -EARTH_RADIUS, EARTH_RADIUS, EARTH_RADIUS;

And for a Mercator or Cartesian projection this is :-

-DEGREES_180, -DEGREES_90, DEGREES_180, DEGREES_90

Run-time only

Read-only

GlobeUnitsPerInch property

Applies to

TPriModel object

Declaration

property GlobeUnitsPerInch: integer;

Description

Calculates the number of GlobeUnits that are displayed per screen inch. This will vary based on the scalefactor.

Run-time only

Read-only

PointsPerPixel property

Applies to

[TPriModel](#) object

Declaration

property PointsPerPixel: Integer;

Description

The number of logical points that are displayed per screen pixel in [GlobeUnits](#).

Run-time only

Read-only

ScaleFactor property

Applies to

TPriModel object

Declaration

property ScaleFactor: Extended;

Description

The current scalefactor that is being used to display the globe.

Run-time only

XOrigin property

Applies to

[TPriModel](#) object

Declaration

property XOrigin: Integer;

Description

The current X origin in screen coordinates for the positioning of the globe image.

Run-time only

XRotation property

Applies to

[TPrjModel](#) object

Declaration

property XRotation: integer;

Description

The current value of rotation about the X axis.

Run-time only

Read-only

YOrigin property

Applies to

[TPrjModel](#) object

Declaration

property YOrigin: Integer;

Description

The current Y origin in screen coordinates for the positioning of the globe image.

Run-time only

YRotation property

Applies to

[TPrjModel](#) object

Declaration

property YRotation: integer;

Description

The current value of rotation about the Y axis.

Run-time only

Read-only

ZRotation property

Applies to

[TPrjModel](#) object

Declaration

property ZRotation: integer;

Description

The current value of rotation about the Z axis.

Run-time only

Read-only

ArrayLLToXY method

Applies to

[TPrjModel](#) object

Declaration

```
function ArrayLLToXY(apLL: TLLPointList; Obj: TGlobeObject): integer; virtual;
```

Description

Converts an array of logical coordinates into an array of screen coordinates.

LLToXY method

Applies to

[TPrjModel](#) object

Declaration

```
function LLToXY(iLong, iLat: integer; iIndex: integer): Boolean; virtual;  
abstract;
```

Description

Converts logical coordinates into screen coordinates.

PaintBackground method

Applies to

[TPrjModel](#) object

Declaration

```
function PaintBackground: integer; virtual;
```

Description

Paints the background image for the projection for the Spherical projection this is a circle the the Mercator and Cartesian projections this is a rectangle.

RectLLVisible method

Applies to

[TPrjModel](#) object

Declaration

```
function RectLLVisible(RectLL: TRect; var OutCodeTLBR: Byte): Boolean;  
virtual;
```

Description

Checks the supplied logical rectangle to see if it is visible on the screen. The OutCodeTLBR is set to control the clipping of the object before display.

XYToLL method

Applies to

[TPRJModel](#) object

Declaration

```
function XYToLL(iX, iY, iIndex: integer): Boolean; virtual; abstract;
```

Description

Converts screen coordinates into logical coordinates.

UpdateModel method

Applies to

[TPrjModel](#) object

Declaration

```
procedure UpdateModel(iRotX, iRotY, iRotZ: integer; eSF: Extended); virtual;
```

Description

When the rotation or scaling of the data is modified this is called to update the projection model before the globe objects are re-displayed.

About the TSphericalPrj object

Purpose

This projection is the default projection for the Globe Component. It displays the globe as a three dimensional sphere.

Tasks

The Spherical projection is selected by setting the [Globe.Projection](#) property to **gpSpherical**.

About the TCartesianPrj object

Purpose

This projection displays a flat rectangular projection with all the latitude and longitude points are equally spaced.

Tasks

The Cartesian projection is selected by setting the [Globe.Projection](#) property to **gpCartesian**.

TGlobePersistant object

Unit

Globe

Description

This is the root class for all objects and their associated attributes that can be saved to or loaded from a globe data file.

This is an abstract class and should never be instantiated. All classes derived from TGlobePersistant must implement the CreateObject, SaveToStream and LoadFromStream methods.

Actually there should be very little reason to directly derive any object from TGlobePersistant, all objects should be derived from TGlobeAttribute or TGlobeObject. Objects derived from TGlobeAttribute or TGlobeObject should implement the SaveToStream and LoadFromStream methods if they need to be stored into the data file.

TGlobeAttribute object

[Properties](#)

[Tasks](#)

Unit


[Globe](#)


Description

This is the root class for all attributes associated with each globe object. All attributes can be stored in the globe data file.

This is an abstract class and should never be instantiated. All attributes used to draw objects are derived from this class.

Properties

 Run-time only

 Key properties

 [Index](#)



 [Layer](#)



 [Referenced](#)

About the TGlobeAttribute Class

[TGlobeAttribute reference](#)

Purpose

This is an abstract class and is only used to derive other attribute classes. All attributes that are to be managed by TGlobe must be derived from this class.

Index property

Applies to

TGlobeAttribute object

Declaration

property Index: integer;

Description

This returns the attributes position in the layers attribute list. This is used when saving an object to a stream, instead of saving the attribute data to the stream with each object, only the index of the attribute is saved, and the attribute list is saved separately to the objects.

Run-time only

Read-only

Layer property

Applies to

TGlobeAttribute object

Declaration

property Layer: TGlobeLayer;

Description

Returns the Layer object associated with this attribute.

Run-time only

Referenced property

Applies to

TGlobeAttribute object

Declaration

property Referenced: Boolean;

Description

Used by the TGlobe.SaveToFile method. This is used to ensure that any unused attributes are not saved to the file.

Run-time only

TGlobePen object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

The pen attribute defines the color and style of lines used to draw the globe objects.

Properties

 Run-time only

 Key properties



 PenColor



 PenStyle



 PenWidth

Methods

🔑 Key methods



Define

LoadFromStream

RenderAttribute

SaveToStream

About the TGlobePen object

[TGlobePen reference](#)

Purpose

The TGlobePen object is derived from the [TGlobeAttribute](#) object. It is used by the Globe objects to draw their lines.

Tasks

To create a new pen for an object the [TGlobePen.Define\(\)](#) class method must be used. This method will check to see if an existing attribute has all of the same properties as the pen that is begin defined. If a matching pen is found then that is returned otherwise a new pen object is created and returned.

Define method

Applies to

TGlobePen object

Declaration

```
Class function TGlobePen.Define( ParentLayer : TGlobeLayer; Color : TColor;  
Style : TPenStyle; Width : Byte ) : TGlobePen;
```

Description

Defines a new Globe Pen attribute for the supplied Layer. If a Pen attribute already exists that matches the supplied parameters then it will be returned otherwise a new Pen is created and returned.

Example:

```
ObjectPen := TGlobePen.Define( Layer, clBlue, psDot, 1 );
```

PenColor property

Applies to

TGlobePen object

Declaration

property PenColor: TColor;

Description

Color used to draw the pen with.

Run-time only

PenStyle property

Applies to

TGlobePen object

Declaration

property PenStyle: TPenStyle;

Description

Styles used to draw the Pen.

Run-time only

PenWidth property

Applies to

TGlobePen object

Declaration

property PenWidth: integer;

Description

Width of line to draw with the Pen.

Run-time only

TGlobeBrush object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

The brush attribute defines the color and style used to fill the globe objects when drawn.

Properties

 Run-time only

 Key properties



 BrushColor



 BrushStyle

Methods

🔑 Key methods



Define

LoadFromStream

RenderAttribute

SaveToStream

About the TGlobeBrush object

[TGlobeBrush reference](#)

Purpose

The TGlobeBrush object is derived from the TGlobeAttribute object. It is used by the polygon object to fill the area bounded by the polygon lines.

Tasks

To create a new brush for an object the TGlobeBrush.Define() class method must be used. This method will check to see if an existing attribute has all of the same properties as the brush that is being defined. If a matching brush is found then that is returned otherwise a new brush object is created and returned.

Define Method

Applies to

TGlobeBrush object

Declaration

```
Class function TGlobeBrush.Define( ParentLayer : TGlobeLayer; Color : TColor;  
Style : TBrushStyle ) : TGlobeBrush;
```

Description

Defines a new Globe Brush attribute for the supplied Layer. If a Brush attribute already exists that matches the supplied parameters then it will be returned otherwise a new Brush is created and returned.

Example:

```
ObjectBrush := TGlobeBrush.Define( Layer, clRed, bsSolid );
```

BrushColor property

Applies to

TGlobeBrush object

Declaration

property BrushColor: TColor;

Description

Color used to draw the Brush with.

Run-time only

BrushStyle property

Applies to

[TGlobeBrush](#) object

Declaration

property BrushStyle: [TBrushStyle](#);

Description

Styles used to draw the Brush.

Run-time only

TGlobeFont object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

The font attribute defines the font, face, size, style and angle used to display the globe object text.

Properties

 Run-time only

 Key properties



 FontColor



 FontSize



 FontAngle



 FontFace




 FontStyles



 FontUnit

Methods

 Key methods



[Define](#)

[GetTextHeight](#)

[LoadFromStream](#)

[RenderAttribute](#)

[SaveToStream](#)

About the TGlobeFont object

[TGlobeFont reference](#)

Purpose

The TGlobeFont object is derived from the TGlobeAttribute object. It is used by the Globe objects render their text. The font object can control the angle of the text rendered as well as all of the standard font properties.

Tasks

To create a new font for an object the TGlobeFont.Define() class method must be used. This method will check to see if an existing attribute has all of the same properties as the font that is begin defined. If a matching font is found then that is returned otherwise a new font object is created and returned.

Define method

Applies to

TGlobeFont object

Declaration

```
Class function TGlobeFont.Define( ParentLayer : TGlobeLayer; const Face :  
String;  
    Color : TColor; TextUnit : TGlobeUnits;  
    Size, Angle : integer; Styles : TFontStyles ) : TGlobeFont;
```

Description

Defines a new Globe Font attribute for the supplied Layer. If a Font attribute already exists that matches the supplied parameters then it will be returned otherwise a new Font is created and returned.

Example:

```
ObjectFont := TGlobeFont.Define( Layer, 'Arial', clBlack,  
    NauticalMile, 20, 0, [fsBold] );
```


FontColor property

Applies to

TGlobeFont object

Declaration

property FontColor: TColor;

Description

Color used to draw the Text with.

Run-time only

FontStyles property

Applies to

TGlobeFont object

Declaration

property FontStyles: TFontStyles;

Description

Styles used to render the font.

Run-time only

FontFace property

Applies to

TGlobeFont object

Declaration

property FontFace: string;

Description

The name of the font to render the text with. If the text is to be scaled then this should be a TrueType font.

Run-time only

FontSize property

Applies to

[TGlobeFont](#) object

Declaration

property FontSize : integer;

Description

The size of the font to display. If the font is to be scaled then this is used in conjunction with the [FontUnit](#) property to calculate the final pixel size that is displayed on the globe.

Run-time only

FontUnit property

Applies to

TGlobeFont object

Declaration

property FontUnit: TGlobeUnits;

Description

This property and the FontSize property are used to control the size of the rendered text. If set to NauticalMile then the text will be rendered at a size of FontSize * NauticalMile.

Run-time only

GetTextHeight method

Applies to

[TGlobeFont](#) object

Declaration

```
function GetTextHeight: integer;
```

Description

This calculates and returns the text height in display pixels.

TGlobeObject object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)


Description

This is the root object for all globe objects that can be drawn on the globe. This is an abstract class, all objects that are to be displayed on the globe must be derived from this class.

Properties

 Run-time only

 Key properties

 Attributes

 BoundsRectLL

 Index



 Layer




 Origin




 Selected

 State

 Title

Methods

 Key methods

[Create](#)

[CreateObject](#)

[IsVisible](#)



[LLInObject](#)

[CalcBoundsRect](#)



[Delete](#)

[LoadFromStream](#)



[RedrawObject](#)

[Render](#)

[SaveToStream](#)

About the TGlobeObject class

[TGlobeObject reference](#)

Purpose

The TGlobeObject Class is an abstract class that all objects have to be derived from if they are to be managed by the [TGlobe](#) component.

Attributes property

Applies to

TGlobeObject object

Declaration

property Attributes: TGlobeAttribute;

Description

Accesses all of the attributes held in the object. If the Index goes out of range then nil is returned.

Run-time only

BoundsRectLL property

Applies to

[TGlobeObject](#) object

Declaration

property BoundsRectLL: TRect;

Description

This is the bounding rectangle surrounding the object, this is specified in latitude and longitude coordinates the coordinates are in [GlobeUnits](#).

Run-time only

Index property

Applies to

[TGlobeObject](#) object

Declaration

property Index: integer;

Description

Readonly property returns the index of the object in the layers object list.

Run-time only

Read-only

Layer property

Applies to

TGlobeObject object

Declaration

property Layer: TGlobeLayer;

Description

Returns the Layer structure associated with the object. This can be used to move an object between layers by assigning the destination layer to this property.

Run-time only

Origin property

Applies to

TGlobeObject object

Declaration

property Origin: TPointLL;

Description

This is the initial location in Longitude and Latitude for the object, by modifying this value objects can be moved across the globe.

Run-time only

Selected property

Applies to

TGlobeObject object

Declaration

property Selected: Boolean;

Description

Sets or clears the osSelected Flag in the object State. If the osSelected flag is set then the object will appear using the SelectedPen and SelectedColor properties instead of their associated attributes.

Run-time only

State property

Applies to

TGlobeObject object

Declaration

property State: TGlobeObjectStateSet;

Description

This is a set of flags that hold the current state of the object TGlobeObjectStateSet is defined as one or more of the following values :-

osReset	When set to this it causes the CalcProjectionData method to be called so that any optimisations of the objects associated data can be made.
osRedraw	When the object is modified in some way e.g. its origin is moved then this causes the object to be re-drawn.
osVisible	The object can be seen and therefore needs to be drawn.
osSelected	The object is the selected object and should be drawn with different attributes.
osTiny	This object is too small at this scale to be displayed.

Run-time only

Title property

Applies to

TGlobeObject object

Declaration

property Title: TGlobeTitle;

Description

The title associated with the object. TGlobeTitle is defined as string[32].

Run-time only

UserObject

Applies to

TGlobeObject object

Declaration

property UserObject: TObject;

Description

General purpose property for the user to associate some data with the TGlobeObject.

Create method

Applies to

TGlobeObject object

Declaration

```
constructor Create(ParentLayer: TGlobeLayer; const sTitle: string; iLong,  
iLat: integer);
```

Description

Must be called by all derived classes. This initialises the object and set the default data required by all TGlobeObjects.

IsVisible method

Applies to

[TGlobeObject](#) object

Declaration

```
function IsVisible: Boolean; virtual;
```

Description

This function decides whether the object is on the visible side of the globe, is in the displayed viewing rectangle and whether it should be painted on the screen.

LLInObject method

Applies to

[TGlobeObject](#) object

Declaration

```
function LLIInObject(iLong, iLat: integer): Boolean; virtual;
```

Description

Returns True if the specified Longitude and Latitude lie within the BoundsRectLL of the object. Descendant classes should override this method to accurately test whether or not the Longitude and Latitude lie within the objects bounds.

CalcBoundsRect method

Applies to

[TGlobeObject](#) object

Declaration

```
procedure CalcBoundsRect; virtual; abstract;
```

Description

Called when the object is created and whenever the globe is zoomed, this is used to calculate a bounding rectangle for the object. This is used to control the display and interaction of the object. All descendant class must override this method.

Delete method

Applies to

TGlobeObject object

Declaration

procedure Delete;

Description

Method used to delete an object from a layer, the object is destroyed.

RedrawObject method

Applies to

TGlobeObject object

Declaration

procedure RedrawObject;

Description

Cause the osReset Flag to be set in the object's State and then re-drawn onto the globe. This should be called when the attributes or points associated with an object have been altered.

Render method

Applies to

TGlobeObject object

Declaration

procedure Render; **virtual; abstract;**

Description

Called when the object needs to be painted onto the globe. All descendant class must override this method.

TGlobeText object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

The text object defines the location and text to display on the globe. The minimum size of the text is controlled by the TGlobe.MinTextSize property. If the scaled size of the text falls below the minimum size the text will not be displayed.

Properties


 Run-time only

 Key properties



 ObjectFont

Methods

 Key methods



[Create](#)

[CalcBoundsRect](#)

[LoadFromStream](#)

[Render](#)

[SaveToStream](#)

About the TGlobeText object

[TGlobeText reference](#)

Purpose

The text object defines the location and text to display on the globe. The minimum size of the text is controlled by the [TGlobe.MinTextHeight](#) property. If the scaled size of the text falls below the minimum size the text will not be displayed.

ObjectFont property

Applies to

TGlobeText object

Declaration

property ObjectFont: TGlobeFont;

Description

The Font used to draw the text with.

Run-time only

Create method

Applies to

TGlobeText object

Declaration

constructor Create(ParentLayer: TGlobeLayer; **const** sTitle: **string**; iLong, iLat: integer);

Description

Creates a new text object associated with a layer. The new text object is returned so that its other properties may be modified.

On creation the object will be assigned the default properties assigned to the Globe object itself.

Example:

```
Globe.LayerNew( 'Test' );

with TGlobeText.Create( Globe.LayerByName( 'Test' ), 'Hello World',
    DecimalToGlobeUnits( 10.0 ), DecimalToGlobeUnits( 51.3 )) do
begin
    ObjectFont := TGlobeFont.Define( Layer,
        'Arial', clBlack, NauticalMile, 20, 0, [fsBold] );
end;
```


TGlobePointText object

[Properties](#)

[Methods](#)

[Tasks](#)

Unit

[Globe](#)

Description

Has all of the same methods and properties as the [TGlobeText](#) object except it displays a circle at the origin location for the object. As for the TGlobeText Object the text will not be displayed if the scaled size of the text is less than the [TGlobe.MinTextHeight](#), but a single pixel will always be displayed to locate the position of the text. The diameter of the circle that is displayed will be the same as the Font.Size property associated with the object.

Methods

🔑 Key methods



Create

CalcBoundsRect

Render

About the TGlobePointText object

[TGlobePointText reference](#)

Purpose

This object displays a point or circle located at its [Origin](#), followed by its [Title](#).

TGlobeSymbol object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

Has all of the same methods and properties as the TGlobeText object except it displays a a single character or symbol from the ObjectFont at the origin location for the object.

Properties


 Run-time only

 Key properties



 ObjectSymbol

Methods

 Key methods



[Create](#)

[CalcBoundsRect](#)

[LoadFromStream](#)

[Render](#)

[SaveToStream](#)

About the TGlobeSymbol object

[TGlobeSymbol reference](#)

Purpose

Has all of the same methods and properties as the TGlobeText object except it displays a a single character or symbol from the ObjectFont at the origin location for the object.

ObjectSymbol property

Applies to

[TGlobeSymbol](#) object

Declaration

property ObjectSymbol: Smallint;

Description

The index number of the symbol that is displayed by the object.

Run-time only

Create method

Applies to

[TGlobeSymbol](#) object

Declaration

constructor Create(ParentLayer: [TGlobeLayer](#); const sTitle: string; iLong, iLat: integer; iSymbol: integer);

Description

Creates a new symbol object associated with a layer. The new symbol object is returned so that its other properties may be modified.

On creation the object will be assigned the default properties assigned to the Globe object itself.

Example:

```
Globe.LayerNew( 'Test' );

with TGlobeSymbol.Create( Globe.LayerByName( 'Test' ), 'AirCraft', 0, 0, 81
) do
  ObjectFont := TGlobeFont.Define( Globe.LayerByName( 'Test' ),
    'WingDings', clRed, GlobeUnit, 20, 900, [fsBold] );
```

TGlobePolyLine object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

Draws a multipoint line e.g. a river or a road.

Properties

 Run-time only

 Key properties




 LLPointList



 ObjectPen

Methods

 Key methods



[Create](#)

[CreateObject](#)

[Destroy](#)

[IsVisible](#)

[LLInObject](#)

[CalcBoundsRect](#)

[LoadFromStream](#)

[Render](#)

[SaveToStream](#)

[SetOrigin](#)

About the TGlobePolyLine object

[TGlobePolyLine reference](#)

Purpose

Draws a multipoint line e.g. a river.

LLPointList property

Applies to

TGlobePolyLine object

Declaration

property LLPointList: TLLPointList;

Description

The PointList object, used to manipulate the points in the object.

Run-time only

ObjectPen property

Applies to

TGlobePolyLine object

Declaration

property ObjectPen: TGlobePen;

Description

The pen property used to draw the object.

Run-time only

Create method

Applies to

TGlobePolyLine object

Declaration

constructor Create(ParentLayer: TGlobeLayer; const sTitle: string; const LLPointList: TLLPointList);

Description

Creates a new polyline object associated with a layer. The new polyline object is returned so that its other properties may be modified.

On creation the object will be assigned the default properties assigned to the Globe object itself.

Example:

```
LLPointList := TLLPointList.Create;
LLPointList.Add( PointLL( -GU_DEGREE, -GU_DEGREE ));
LLPointList.Add( PointLL( GU_DEGREE, -GU_DEGREE ));
LLPointList.Add( PointLL( GU_DEGREE, GU_DEGREE ));
LLPointList.Add( PointLL( -GU_DEGREE, GU_DEGREE ));

Globe.Pen.Color := clGreen; { set the default pen property }
Globe.Pen.Style := psDot;
Globe.Pen.Width := 2;

TGlobePolyline.Create( Globe.LayerNew( 'Test' ), 'Polyline', LLPointList );

LLPointList.Free;
```


TGlobePolygon object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

Draws a filled area e.g. a country or state. This object is derived from the [TGlobePolyline](#) object and inherits all of its properties and methods.

Properties


 Run-time only

 Key properties



 ObjectBrush

Methods

 Key methods



Create

LLInObject

About the TGlobePolygon object

[TGlobePolygon reference](#)

Purpose

Draws a filled area e.g. a country or state. This object is derived from the [TGlobePolyline](#) object and inherits all of its properties and methods.

ObjectBrush property

Applies to

TGlobePolygon object

Declaration

property ObjectBrush: TGlobeBrush;

Description

The Brush property used to fill the object.

Run-time only

Create method

Applies to

TGlobePolygon object

Declaration

constructor Create(ParentLayer: TGlobeLayer; **const** sTitle: **string**; **const** LLPointList: TLLPointList);

Description

Creates a new polygon object associated with a layer. The new polygon object is returned so that its other properties may be modified.

On creation the object will be assigned the default properties assigned to the Globe object itself.

Example:

```
LLPointList := TLLPointList.Create;
LLPointList.Add( PointLL( -GU_DEGREE, -GU_DEGREE ));
LLPointList.Add( PointLL( GU_DEGREE, -GU_DEGREE ));
LLPointList.Add( PointLL( GU_DEGREE, GU_DEGREE ));
LLPointList.Add( PointLL( -GU_DEGREE, GU_DEGREE ));

with TGlobePolygon.Create( Globe.LayerNew( 'Test' ), 'Polygon', LLPointList
) do
begin
    ObjectPen := TGlobePen.Define( Layer, clBlue, psDot, 1 );
    ObjectBrush := TGlobeBrush.Define( Layer, clRed, bsSolid );
end;

LLPointList.Free;
```

TGlobeGrid object

[Properties](#)

[Methods](#)

[Tasks](#)


Unit


[Globe](#)

Description

Displays a localised grid at the specified location. The grids major and minor grid spacing and pens can be defined.

Properties

 Run-time only

 Key properties



 MajorPen



 MajorX



 MajorY



 MinorPen




 MinorX



 MinorY

Methods

 Key methods



[Create](#)

[LoadFromStream](#)

[Render](#)

[SaveToStream](#)

About the TGlobeGrid object

[TGlobeGrid reference](#)

Purpose

This object is used to display a localised grid on the Globe.

MajorPen property

Applies to

TGlobeGrid object

Declaration

property MajorPen: TGlobePen;

Description

This pen is used to draw the Major grid lines.

Run-time only

MajorX property

Applies to

TGlobeGrid object

Declaration

property MajorX: integer;

Description

This property defines the spacing of the Major longitude grid lines.

Run-time only

MajorY property

Applies to

TGlobeGrid object

Declaration

property MajorY: integer;

Description

This property defines the spacing of the Major latitude grid lines.

Run-time only

MinorPen property

Applies to

TGlobeGrid object

Declaration

property MinorPen: TGlobePen;

Description

This pen is used to draw the Minor grid lines.

Run-time only

MinorX property

Applies to

TGlobeGrid object

Declaration

property MinorX: integer;

Description

This property defines the spacing of the Minor longitude grid lines.

Run-time only

MinorY property

Applies to

TGlobeGrid object

Declaration

property MinorY: integer;

Description

This property defines the spacing of the Minor latitude grid lines.

Run-time only

Create method

Applies to

TGlobeGrid object

Declaration

```
constructor Create(ParentLayer: TGlobeLayer; const sTitle: string;  
GridBoundsLL: TRect; iMajorX, iMajorY, iMinorX, iMinorY: integer);
```

Description

The GridBoundsLL is a rectangle that defines the bounding area that the grid will be displayed over. The iMajorX and iMajorY values define the spacing of the Major grid lines, these will be drawn using the MajorPen attribute. The iMinorX and iMinorY values define the spacing of the Minor grid lines, these will be drawn using the MinorPen attribute.

The Major and Minor grid values are defined in units of angle e.g. GU_MINUTE or GU_SECOND

On creation the object will be assigned the default properties assigned to the Globe object itself.

Example:

```
if Globe.LayerCount = 0 then Globe.LayerNew( 'Test' );  
  
with TGlobeGrid.Create( Globe[0], 'TestGrid',  
    Rect( -ONE_DEGREE, -ONE_DEGREE, ONE_DEGREE, ONE_DEGREE ),  
    ONE_MINUTE * 10, ONE_MINUTE * 10, ONE_MINUTE, ONE_MINUTE ) do  
begin  
    MinorPen := TGlobePen.Define( Globe[0], clSilver, psDot );  
    MajorPen := TGlobePen.Define( Globe[0], clBlack, psSolid );  
end;
```

TGlobeLayer object

[Properties](#)

[Methods](#)


Unit


[Globe](#)

Description

Each Layer manages the objects and their associated attributes within it. Each layer has a number of properties that control how that layer is displayed. The properties for each layer can be set automatically by defining a Profile .PRF file that has the same name as the data file.

Properties

 Run-time only

 Key properties



 Animated

 Attributes



 Enabled



 Filename

 Globe

 Index



 MaxFontHeight



 Name

 ObjectCount

 Objects



 ScaleFont



 Visible



 ZoomMax




 ZoomMin




 ZoomUnits



Methods

 Key methods

 [Create](#)
[Destroy](#)
[IsLayerVisible](#)
 [ObjectAtLL](#)
 [ObjectByTitle](#)
 [Clear](#)
[CopyAttributes](#)
 [LoadFromFile](#)
[Paint](#)
 [SaveToFile](#)

Animated property

Applies to

TGlobeLayer object

Declaration

property Animated: Boolean;

Description

If set to True then this layer is rendered after the non animation layers have been displayed on the screen, this means that objects on this layer are display very quickly as the underlying globe data does not need to be re-generated for each movement of the the animated objects.

Run-time only

Attributes property

Applies to

TGlobeLayer object

Declaration

property Attributes: TGlobeAttribute;

Description

Array of attributes associated with the objects in this layer. All the objects with the layer share a common set of attributes. These attributes are stored here. If lots of objects have the same definition of a Brush i.e. the same brush color and brush style then each of the objects will refer to the same attribute stored in this list. If an attribute on this list is modified it will change the appearance of all the objects that share that attribute.

Run-time only

Read-only

Enabled property

Applies to

TGlobeLayer object

Declaration

property Enabled: Boolean;

Description

If set to false then the objects that are on this layer will be displayed but will not be searched by the Globe object when the ObjectAtLL() and ObjectAtXY() methods are called.

Run-time only

Filename property

Applies to

[TGlobeLayer](#) object

Declaration

```
property Filename: string;
```

Description

The name of the file associated with the Layer.

Run-time only

Globe property

Applies to

TGlobeLayer object

Declaration

property Globe: TGlobe;

Description

The parent TGlobe object that owns this Layer.

Run-time only

Index property

Applies to

TGlobeLayer object

Declaration

property Index: integer;

Description

Index of the layer in Layer list of the Globe object. If this is written to then the layer will be moved to that position in the Globe Layer list.

Run-time only

MaxFontHeight property

Applies to

TGlobeLayer object

Declaration

property MaxFontHeight: integer;

Description

If non zero this controls the maximum screen height in pixels of any text that is displayed on this layer.

Run-time only

Name property

Applies to

TGlobeLayer object

Declaration

property Name: **string**;

Description

A name that is associated with the layer.

Run-time only

ObjectCount property

Applies to

[TGlobeLayer](#) object

Declaration

property ObjectCount: integer;

Description

Count of the number of objects that are members of this layer.

Run-time only

Read-only

Objects property

Applies to

TGlobeLayer object

Declaration

property Objects: TGlobeObject;

Description

The array of objects held by the layer. This is the default property for the layer so the objects can be referenced using the following methods:-

```
Globe.Layer[iLayer].Objects[iObject];  
Globe.Layer[iLayer][iObject];  
Globe[iLayer][iObject];
```

Run-time only

Read-only

ScaleFont property

Applies to

[TGlobeLayer](#) object

Declaration

property ScaleFont: Boolean;

Description

If True then the text size adjusted to match the scale of the displayed data on the globe. Otherwise the text is display at a fixed size.

Run-time only

Visible property

Applies to

TGlobeLayer object

Declaration

property Visible: Boolean;

Description

If set to false then the objects that are on this layer will not be displayed.

Run-time only

ZoomMax property

Applies to

[TGlobeLayer](#) object

Declaration

property ZoomMax: integer;

Description

If non zero this property controls the maximum point at which data on this layer is displayed.

Run-time only

ZoomMin property

Applies to

[TGlobeLayer](#) object

Declaration

property ZoomMin: integer;

Description

If non zero this property controls the minimum point at which data on this layer is displayed.

Run-time only

ZoomUnits property

Applies to

[TGlobeLayer](#) object

Declaration

property ZoomUnits: [TGlobeUnits](#);

Description

Used to define the units that the ZoomMax and ZoomMin values are defined in e.g. KM for kilometers or NM for nautical miles. See [GlobeUnitsTo\(\)](#) for a list of valid units.

Run-time only

Create method

Applies to

[TGlobeLayer](#) object

Declaration

constructor `Create(ParentGlobe: TGlobe);`

Description

Creates a new empty layer associated with a TGlobe object. It is easier and better to create new layers using the [Globe.LayerNew](#) method.

IsLayerVisible method

Applies to

[TGlobeLayer](#) object

Declaration

```
function IsLayerVisible: Boolean;
```

Description

Checks all associated attributes of the Layer and determines whether or not the layer is visible and needs to be displayed.

ObjectAtLL method

Applies to

TGlobeLayer object

Declaration

```
function ObjectAtLL(iLong, iLat: integer): TGlobeObject;
```

Description

Returns the object found at the supplied Longitude and Latitude. If no object in the layer exists at this location then **nil** is returned.

ObjectByTitle method

Applies to

TGlobeLayer object

Declaration

```
function ObjectByTitle(const sTitle: string; ObjStart: TGlobeObject):  
TGlobeObject;
```

Description

Finds an object in the object list which has a matching Title. the ObjStart parameter specifies where in the list the search will start from. If the ObjStart parameter is nil then the search will start from the beginning of the list, if an object is supplied then the search will start from the next object in the list. If no object is found then nil is returned. The returned object can be passed to LocateObject to rotate the globe.

Example:

```
obj := Globe.ObjectByTitle( 'FRANCE', nil );  
  
if obj = nil then  
    ShowMessage( 'FRANCE not found in this Layer' );
```

Clear method

Applies to

[TGlobeLayer](#) object

Declaration

```
procedure Clear;
```

Description

Deletes all data associated with the Layer.

CopyAttributes method

Applies to

TGlobeLayer object

Declaration

procedure CopyAttributes (Obj: TGlobeObject);

Description

Copys the attributes associated with an object into the current layer. This is necessary when moving objects between layers. This routine is called internally by the TGlobeObject.Layer assign code.

LoadFromFile method

Applies to

[TGlobeLayer](#) object

Declaration

```
procedure LoadFromFile(const sFileName: string);
```

Description

Loads a Globe data file into the Layer.

Paint method

Applies to

[TGlobeLayer](#) object

Declaration

```
procedure Paint;
```

Description

Paints all the objects associated with this layer. This is called internally by the TGlobe.Paint method

SaveToFile method

Applies to

TGlobeLayer object

Declaration

```
procedure SaveToFile(const sFileName: string; iFormat: integer);
```

Description

Saves the data associated with the Layer to a file. If the file already exists then it will be overwritten. The iFormat parameter specifies whether the data should be saved to 16 bit accuracy or 32 bit accuracy.

If the accuracy of the data is not greater than one Nautical Mile then the data can be stored to 16 bit accuracy. This will also make the data file smaller and possible to be loaded using the 16 bit Delphi 1 version of the component.

If the data is of greater accuracy than one Nautical Mile then it must be stored using 32 bit accuracy.

iFormat can be one of the following values:

```
TG_DATAFILE_16BIT 0  
TG_DATAFILE_32BIT 1
```

TGlobe component

[Properties](#)

[Methods](#)

[Events](#)

[Tasks](#)


Unit


[Globe](#)

Description

This is the main component class, it combines all of the overlays together and provides the main user interface routines.

Properties

 Run-time only

 Key properties

Align

Background

 Bitmap

 Brush

Color

 DataDirectory

DragCursor

DragMode

 DrawGrid

Enabled

Font

 FontAngle

 FontUnit

 GlobeCanvas

 GlobeCentreXY

 GlobeColor

 GlobeUnitsPerInch

 LatitudeStep

 Layers

 LongitudeStep

 MinTextHeight

 MouseCapture

 MouseRotateButton

 MouseSelectButton

 MouseZoomButton

ParentShowHint

 Pen

PopupMenu

 Profile

 ProfileChanged

 ProfileName

 Projection

 ProjectionModel

 ScaleFactor

 ScaleFactor

 ScrollBars

 SelectedBrush

 SelectedObject

 SelectedObject

 SelectedPen

ShowHint

 ViewRect

Visible

 XRotation

 XRotationInDegs

 XRotationInMins

 XRotationInSecs

 YRotation

 YRotationInDegs


 YRotationInMins

 YRotationInSecs



ZRotation
ZRotationInDegs
ZRotationInMins
ZRotationInSecs

Methods

 Key methods

[DeviceXYToLL](#)

[LayerByName](#)

[LayerCount](#)



[LayerNew](#)

[LLToDeviceXY](#)

[ObjectAtLL](#)



[ObjectAtXY](#)

[ObjectByTitle](#)

[ProgressMessage](#)

[ScaleUnitsToDevice](#)



[Clear](#)



[LoadProfile](#)

[ObjectLocate](#)

[PanViewRect](#)



[Print](#)

 [Redraw](#)

[RegisterGlobeClass](#)

[RenderAttributes](#)

[RenderLine](#)


[RenderPolygon](#)

[RenderPolyLine](#)

[RenderTextOut](#)

[SaveProfile](#)

Events

 Key events

[OnClick](#)
[OnDbClick](#)
[OnDragDrop](#)
[OnDragOver](#)
[OnEndDrag](#)
[OnMouseDown](#)
[OnMouseMove](#)
[OnMouseUp](#)
[OnObjectCreate](#)

[OnObjectFree](#)

 [OnPaint](#)

[OnPrint](#)

[OnProgress](#)



[OnRender](#)

[OnRenderAttributes](#)



[OnSelected](#)

[OnZoomed](#)

About the TGlobe object

[TGlobe reference](#)

Purpose

This is the controlling component. The component encapsulates the layers and their attributes and objects. The Globe component provides the interface to allow the user to interact with all of the layers as a whole.

Background property

Applies to

TGlobe object

Declaration

property Background: TBitmap;

Description

This can be set to a bitmap that will be tiled and displayed behind the globe, e.g. the bitmap could show a starfield behind the globe.

Bitmap property

Applies to

TGlobe object

Declaration

property Bitmap: TBitmap;

Description

This is the bitmap used by the object to draw all of the globe objects onto before blitting it to the controls window.

This can be used to capture an image of the object for use in an external application.

Run-time only

Read-only

Brush property

Applies to

TGlobe object

Declaration

property Brush: TBrush;

Description

This is the brush that will by default be associated with any objects when they are created.

DataDirectory property

Applies to

TGlobe object

Declaration

```
property DataDirectory: string;
```

Description

This is the default location for the data files. This will be overridden by the **[Global Settings]** section, **DataDirectory** tag in the Profile file if present.

DrawGrid property

Applies to

TGlobe object

Declaration

property DrawGrid: TDrawGrid;

Description

This can be set to either **dgNone**, **dgBehind** or **dgInFront** this will allow the grid to be switched off, or displayed behind or in front of the country objects.

FontAngle property

Applies to

TGlobe object

Declaration

property FontAngle: integer;

Description

This is the display angle in tenths of a degree that will by default be associated with any objects when they are created.

FontUnit property

Applies to

TGlobe object

Declaration

property FontUnit: TGlobeUnits;

Description

This property and the FontSize property are used to control the size of the rendered text. If set to NauticalMile then the text will be rendered at a size of FontSize * NauticalMile.

GlobeCanvas property

Applies to

TGlobe object

Declaration

property GlobeCanvas: TCanvas;

Description

This canvas encapsulates the drawing surface used by the object. Drawing operations onto this canvas actually go to the internally held bitmap which is then blitted onto the window.

Run-time only

Read-only

GlobeCentreXY property

Applies to

TGlobe object

Declaration

property GlobeCentreXY: TPoint;

Description

Sets or returns the X,Y coordinates of the centre of the Globe.

Run-time only

GlobeColor property

Applies to

TGlobe object

Declaration

property GlobeColor: TColor;

Description

This controls the surface color of the globe active area. This is the area that all valid globe points are plotted onto, i.e. This will be the color of the sea.

GlobeUnitsPerInch property

Applies to

TGlobe object

Declaration

property GlobeUnitsPerInch: integer;

Description

Sets or returns the number of Globe Units per inch, this can be used to monitor the displayed scale or set the display to a specific scale. e.g.

```
Globe.GlobeUnitsPerInch := GlobeUnitsFrom( 100, 'KM' );
```

This will set the scalefactor for the globe so that one inch on the display represents 100 kilometers.

Run-time only

LatitudeStep property

Applies to

TGlobe object

Declaration

property LatitudeStep: integer;

Description

This property controls the spacing of the globe's Latitude grid lines, the value is in degrees and set relative to the equator. Setting the latitude step to 15 will display a grid line for every hour difference from GMT. If this property is set to zero then the Latitude lines will not be displayed.

Layers property

Applies to

TGlobe object

Declaration

property Layers: TGlobeLayer;

Description

This is the list of layers associated with the Globe object.

This is also the default object so layers can be referenced directly by using Globe[iLayer].

Run-time only

Read-only

LongitudeStep property

Applies to

TGlobe object

Declaration

property LongitudeStep: integer;

Description

This property controls the spacing of the globe's Longitude grid lines, the value is in degrees and set relative to the Greenwich meridian. Setting the longitude step to 15 will display a grid line for every hour difference from GMT. If this property is set to zero then the Longitude lines will not be displayed.

MinTextHeight property

Applies to

TGlobe object

Declaration

property MinTextHeight: integer;

Description

This controls the minimum size of text. If the scaled text is small than this value then it is not displayed. If the value is set to 0 then the text will always be displayed.

MouseRotateButton property

Applies to

TGlobe object

Declaration

property MouseRotateButton: TGlobeMouseButton;

Description

This can be set to **mzbLeft**, **mzbRight**, **mzbMiddle** or **mzbNone**. By setting this to one of the mouse buttons an automatic rotate facility is enabled. If it is set to mzbLeft then by pressing and holding the left mouse button the globe can be rotated by dragging the mouse.

MouseSelectButton property

Applies to

TGlobe object

Declaration

property MouseSelectButton: TGlobeMouseButton;

Description

This can be set to **mzbLeft**, **mzbRight**, **mzbMiddle** or **mzbNone**. By setting this to one of the mouse buttons an automatic select facility is enabled. If it is set to mzbLeft then by pressing the Left mouse button over an object that object will be set as the globe selected object.

MouseZoomButton property

Applies to

TGlobe object

Declaration

property MouseZoomButton: TGlobeMouseButton;

Description

This can be set to **mzbLeft**, **mzbRight**, **mzbMiddle** or **mzbNone**. By setting this to one of the mouse buttons an automatic zoom facility is enabled. If it is set to mzbRight then by pressing and holding the right mouse button a window can be drawn on the globe, when the mouse button is released the globe will expand to display the selected area of the globe.

Pen property

Applies to
TGlobe object

Declaration
property Pen: TPen;

Description
This is the pen that will by default be associated with any objects when they are created.

Profile property

Applies to

TGlobe object

Declaration

property Profile: TInifile;

Description

This is the TInifile object used to access a loaded profile. The .ini file should contain a section called LayerList, this section lists all of the globe data files that should be loaded into the globe layers. Each entry in the LayerList section can have a corresponding section that gives information specific to the referenced data file.

[Global Settings]

DataDirectory

Tags in this section affect all layers loaded in this profile.

This specifies the location for the data files and overrides the DataDirectory property associated with the Globe object.

[LayerList]

This section contains a list of all the data files that make up the layers of the profile. Each entry is numbered sequentially starting from 0.

[Layerfilename.ext]

Tags in a section with the same name as the data file and are therefore specific to that data file.

Animated

If set to 1 then the Layer is optimised for animated objects. The objects are painted onto the displayed canvas after the normal layers have been displayed.

Name

The name associated with the Layer

Enabled

Whether or not the Layer is enabled for object selection.

Visible

Whether or not the Layer is initially visible.

MaxFontHeight

If non zero this specifies the maximum pixel size that the displayed text will be allowed to grow to.

ScaleFont

If set to 0 then the text height is not scaled as the globe is zoomed.

ZoomMax

If non zero this property controls the maximum point at which data on this layer is displayed.

ZoomMin

If non zero this property controls the minimum point at which data on this layer is displayed.

ZoomUnits

Used to define the units that the ZoomMax and ZoomMin values are defined in e.g. KM for kilometers or NM for nautical miles. See GlobeUnitsTo() for a list of valid units

Run-time only

ProfileChanged property

Applies to

TGlobe object

Declaration

property ProfileChanged: Boolean;

Description

Flag to indicate if any of the profile properties associated with each layer have been altered since they were loaded from the profile file.

Run-time only

ProfileName property

Applies to

TGlobe object

Declaration

```
property ProfileName: string;
```

Description

This is the name of the current Profile .PRF file. You can assign a new name, this will cause the current profile to be closed and the new profile .PRF file opened and loaded. If you assign a blank string to this property then the current profile is closed.

Projection property

Applies to

TGlobe object

Declaration

property Projection: TGlobeProjection;

Description

This can be set to **gpSpherical**, **gpMercator** or **gpCartesian**. If set to **gpSpherical** then a 3D Globe is displayed that can be zoomed or rotated. If **gpMercator** is set then the World is displayed as a 2 dimensional map, the X and Y Rotation properties have no effect when **gpMercator** is set.

ProjectionModel property

Applies to

TGlobe object

Declaration

property ProjectionModel: TPrjModel;

Description

This is the current projection object being used to display the globe. The Projection model controls how the globe data is interpreted and then displayed on the screen.

Run-time only

ScaleFactor property

Applies to

[TGlobe](#) object

Declaration

property ScaleFactor: Extended;

Description

This property is the value used to scale all objects on the globe. $1/\text{ScaleFactor}$ is the number of logical lat/long points that fit inside a single device pixel. An object that is say 100 logical points long can be scaled by multiplying by the ScaleFactor, the result gives the proportional length of the object relative to the globe.

Run-time only

ScrollBars property

Applies to
TGlobe object

Declaration
property ScrollBars: TScrollStyle;

Description
This property controls the display of the horizontal or vertical scroll bars for the TGlobe window.

SelectedBrush property

Applies to

TGlobe object

Declaration

property SelectedBrush: TBrush;

Description

This is the Brush used to fill objects when they have the **osSelected** flag set in their State property.

SelectedObject property

Applies to

TGlobe object

Declaration

property SelectedObject: TGlobeObject;

Description

Points to the currently selected object or nil if no object is selected. The Selected object is set by the User to any object on the Globe. When the Globe is subsequently painted the selected object is highlighted using the SelectedBrush and SelectedPen properties.

Run-time only

SelectedPen property

Applies to

[TGlobe](#) object

Declaration

property SelectedPen: [TPen](#);

Description

This is the Pen used to Draw objects when they have the **osSelected** flag set in their [State](#) property.

ViewRect property

Applies to

[TGlobe](#) object

Declaration

property ViewRect: TRect;

Description

This defines the size of the view of the globe that is displayed. This rectangle is specified in [GlobeUnits](#). The rectangle is used to zoom into or out of the Globe.

To display the whole of the Globe set this rectangle to :

Globe.ViewRect := Rect(0,0,0,0);

Run-time only

XRotation property

Applies to
[TGlobe](#) object

Declaration
`property XRotation: integer;`

Description

These set the amount that the globe is rotated about the X axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen and the Z axis coming out of the screen. The value of rotation is set in [GlobeUnits](#).

Run-time only

XRotationInDeps property

Applies to

TGlobe object

Declaration

property XRotationInDeps: integer;

Description

These set the amount that the globe is rotated about the X axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in degrees.

XRotationInMins property

Applies to

TGlobe object

Declaration

property XRotationInMins: integer;

Description

These set the amount that the globe is rotated about the X axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in minutes of a degree.

Run-time only

XRotationInSecs property

Applies to

TGlobe object

Declaration

property XRotationInSecs: integer;

Description

These set the amount that the globe is rotated about the X axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in Seconds of a degree

Run-time only

YRotation property

Applies to

[TGlobe](#) object

Declaration

property YRotation: integer;

Description

These set the amount that the globe is rotated about the Y axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen and the Z axis coming out of the screen. The value of rotation is set in [GlobeUnits](#).

Run-time only

YRotationInDeps property

Applies to

TGlobe object

Declaration

property YRotationInDeps: integer;

Description

These set the amount that the globe is rotated about the Y axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in degrees.

YRotationInMins property

Applies to

TGlobe object

Declaration

property YRotationInMins: integer;

Description

These set the amount that the globe is rotated about the Y axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in minutes of a degree.

Run-time only

YRotationInSecs property

Applies to

TGlobe object

Declaration

property YRotationInSecs: integer;

Description

These set the amount that the globe is rotated about the Y axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in Seconds of a degree

Run-time only

ZRotation property

Applies to

[TGlobe](#) object

Declaration

property ZRotation: integer;

Description

These set the amount that the globe is rotated about the Z axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen and the Z axis coming out of the screen. The value of rotation is set in [GlobeUnits](#).

Run-time only

ZRotationInDeps property

Applies to

TGlobe object

Declaration

property ZRotationInDeps: integer;

Description

These set the amount that the globe is rotated about the Z axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in degrees.

ZRotationInMins property

Applies to

TGlobe object

Declaration

property ZRotationInMins: integer;

Description

These set the amount that the globe is rotated about the Z axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in minutes of a degree.

Run-time only

ZRotationInSecs property

Applies to

TGlobe object

Declaration

property ZRotationInSecs: integer;

Description

These set the amount that the globe is rotated about the Z axes. The axes are orientated in the standard way with the Y axes going from the bottom to the top and the X-axes going from then left to right of the screen. The value of rotation is set in Seconds of a degree

Run-time only

DeviceXYToLL method

Applies to

TGlobe object

Declaration

```
function DeviceXYToLL(iX, iY: integer; var ptLL: TPointLL): Boolean;
```

Description

Converts screen X, Y coordinates to a Longitude, Latitude coordinate pair. Returns true if the X,Y coordinates lie over the globe and therefore the returned point contains valid data.

LayerByName method

Applies to

TGlobe object

Declaration

```
function LayerByName(const sName: string): TGlobeLayer;
```

Description

Searches the Layer List for a layer with the supplied name. If the layer is not found then nil is returned.

LayerCount method

Applies to

TGlobe object

Declaration

```
function LayerCount: integer;
```

Description

Returns a count of the number of layers loaded by the Globe object.

LayerNew method

Applies to

TGlobe object

Declaration

```
function LayerNew(const sName: string): TGlobeLayer;
```

Description

Creates a new layer and appends it to the end of the Layer List. The layer is given the supplied name. The layer's FileName property is set to the supplied name but with the extension .DAT appended.

LLToDeviceXY method

Applies to

TGlobe object

Declaration

```
function LLToDeviceXY(iLong, iLat: integer; var ptXY: TPoint): Boolean;
```

Description

Converts the iLong / iLat coords to a device XY coordinate. The boolean return if True indicates that the point is visible, if False then the point lies on the far side of the globe.

ObjectAtLL method

Applies to

TGlobe object

Declaration

```
function ObjectAtLL(iLong, iLat: integer): TGlobeObject;
```

Description

Returns the object located at the coordinates supplied, if no object found then nil is returned. This will search all enabled Overlays for an object at the location.

ObjectAtXY method

Applies to

TGlobe object

Declaration

```
function ObjectAtXY(iX, iY: integer): TGlobeObject;
```

Description

Converts the screen X, Y coordinates to Globe coordinates and returns the object found at that location, or nil if none found. The X, Y coordinates of the Mouse events can be passed directly into this method to retrieve the object located under the mouse cursor.

ObjectByTitle method

Applies to

TGlobe object

Declaration

```
function ObjectByTitle(const sTitle: string; ObjStart: TGlobeObject):  
TGlobeObject;
```

Description

Searches all layers in the Globe and finds an object in the object list which has a matching Title. the ObjStart parameter specifies where in the list the search will start from. If the ObjStart parameter is nil then the search will start from the beginning of the list, if an object is supplied then the search will start from the next object in the list. If no object is found then nil is returned.

Example:

```
if InputQuery( 'Title', 'Enter title to find', sTitle ) then  
begin  
    iCount := 0;  
    obj := nil;  
    while true do  
    begin  
        obj := Globe.ObjectByTitle( sTitle, obj );  
  
        if obj = nil then  
            break;  
  
        Inc( icount );  
    end;  
    ShowMessage( format( 'Found %d objects', [icount] ) );  
end;
```

ProgressMessage method

Applies to

TGlobe object

Declaration

```
function ProgressMessage(iMsgType: integer; const sMsg: string): Boolean;
```

Description

Sends a progress message to the globe object. This will be passed onto the application if the application has defined an OnProgressMessage event handler.

ScaleUnitsToDevice method

Applies to
TGlobe object

Declaration

```
function ScaleUnitsToDevice(Value: integer; Units : TGlobeUnits): integer;
```

Description

Scales a value in the specified units to the number of pixels it would represent on the screen.

Clear method

Applies to

TGlobe object

Declaration

```
procedure Clear;
```

Description

Clears the Globe database of all data and objects. All layers are cleared.

LoadProfile method

Applies to

[TGlobe](#) object

Declaration

```
procedure LoadProfile(const sProfileName: string);
```

Description

Loads up a [profile](#) file and processes the sections, all layers are loaded and their associated attributes are set.

ObjectLocate method

Applies to

TGlobe object

Declaration

procedure ObjectLocate (Obj: TGlobeObject);

Description

Adjusts the XRotation and YRotation of the globe to bring the supplied object into the viewing area.

PanViewRect method

Applies to

[TGlobe](#) object

Declaration

```
procedure PanViewRect(iX, iY: integer);
```

Description

Scrolls the current view of the globe to the left, right, up or down by iX and iY [GlobeUnits](#).

Print method

Applies to

TGlobe object

Declaration

```
procedure Print;
```

Description

Prints the rendered Globe image to the current default printer.

Redraw method

Applies to
TGlobe object

Declaration
`procedure Redraw;`

Description
Forces the Globe data to be painted to the screen.

RegisterGlobeClass method

Applies to

[TGlobe](#) object

Declaration

procedure RegisterGlobeClass (AClass: TGlobePersistantClass);

Description

If you derive a new class from [TGlobeObject](#) or [TGlobeAttribute](#) and you want the object to be stored in the data file then it needs to be registered with the TGlobe object. Only registered objects can be loaded from or saved to the data file.

RenderAttributes method

Applies to

TGlobe object

Declaration

```
procedure RenderAttributes(State: TGlobeObjectStateSet; const Attrs: array of  
TGlobeAttribute);
```

Description

This will set the GlobeCanvas attributes to the array of passed TGlobeAttribute objects.

RenderGreatCircleLine method

Applies to

TGlobe object

Declaration

```
procedure RenderGreatCircleLine (ptFromLL, ptToLL: TPointLL; CircleType :  
TGreatCircleType );
```

Description

Draws a projected great circle line between the supplied points using the circle type supplied. The line will follow the shortest path over the curvature of the earth.

RenderLine method

Applies to

TGlobe object

Declaration

procedure RenderLine(ptFromLL, ptToLL: TPointLL; iSteps: integer);

Description

Draws a projected line between the supplied points using the number of steps supplied. The Line will follow the curvature of the earth. A large number of steps will display a smooth line but will take more time to draw.

RenderPolygon method

Applies to

TGlobe object

Declaration

```
procedure RenderPolygon(const Points: array of TPointLL);
```

Description

Draws a filled polygon using the passed array of points to define the Longitude and Latitude vertices of the polygon. The polygon will be drawn using the current pen and brush associated with the GlobeCanvas. The polygon is projected onto the surface of the globe.

RenderPolyLine method

Applies to

TGlobe object

Declaration

```
procedure RenderPolyLine(const Points: array of TPointLL);
```

Description

Draws a polyline using the passed array of points to define the Longitude and Latitude vertices of the polyLine. The polyline will be drawn using the current pen associated with the GlobeCanvas. The polyline is projected onto the surface of the globe.

RenderTextOut method

Applies to

[TGlobe](#) object

Declaration

```
procedure RenderTextOut(iLong, iLat: integer; const sText: ShortString);
```

Description

Draws the text at the Longitude and Latitude supplied. The text will be drawn using the current font attributes associated with the Globe object. The text is projected onto the surface of the globe. The Font will be scaled using the [TGlobe.FontUnits](#) property.

SaveProfile method

Applies to

TGlobe object

Declaration

```
procedure SaveProfile;
```

Description

Writes the profile data held in the Layers to the current profile file.

OnPaint event

Applies to

TGlobe object

Declaration

property OnPaint: TNotifyEvent;

Description

This event occurs after the globe objects have been painted to the GlobeCanvas and its image has been blitted to the Window. This is a good time to draw to the window canvas directly.

OnPrint event

Applies to

TGlobe object

Declaration

property OnPrint: TNotifyEvent;

Description

This event occurs after the globe data has been sent to the printer.

OnProgress event

Applies to

[TGlobe](#) object

Declaration

property OnProgress: [TGlobeProgressEvent](#);

Description

This event is called when the TGlobe.ProgressMessage is called. See [TGlobe.ProgressMessage](#) for the MsgType vales. If the Abort flag is set by the called method then this is passed back to the ProgressMessage calling routine for action if applicable.

OnRender event

Applies to

TGlobe object

Declaration

property OnRender: TNotifyEvent;

Description

This event occurs after the globe objects have been painted to the GlobeCanvas but before the GlobeCanvas has been blitted to the window. This is a good time to write any extra information to the canvas e.g. Titles etc.

OnRenderAttributes event

Applies to

TGlobe object

Declaration

property OnRenderAttributes: TGlobeRenderAttributesEvent;

Description

This is called just before an object is rendered. GlobeObject is the object that is about to be rendered. The attributes used to render the object can be overridden here so that the object is displayed differently dependant on some external condition. This is useful for Thematic mapping purposes. The Done parameter should be set to True so that the object does not use its default attributes.

Example:

```
procedure TForm1.GlobeRenderAttributes( Sender: TObject;  
    GlobeObject: TGlobeObject; var Done: Boolean );  
const  
    aColors: array[0..13] of TColor =  
(clBlack, clMaroon, clGreen, clOlive, clPurple,  
    clTeal, clGray, clSilver, clRed, clLime, clYellow, clBlue, clFuchsia, clWhite);  
begin  
    if GlobeObject is TGlobePolygon then  
    begin  
        Globe.GlobeCanvas.Brush.Color := aColors[( GlobeObject.Index mod 14)];  
        Globe.GlobeCanvas.Pen.Color := aColors[( GlobeObject.Index mod 14)];  
        Done := True;  
    end;  
end;
```

OnObjectFree event

Applies to

TGlobe object

Declaration

property OnObjectFree: TGlobeObjectNotifyEvent;

Description

This is called when a globe object is just about to be destroyed.

Example:

```
procedure TForm1.GlobeObjectCreate( Sender: TObject;  
    GlobeObject: TGlobeObject );  
begin  
    if GlobeObject is TGlobePolygon then { track the number of polygons  
freed }  
        Dec( giPolygonCount );  
end;
```

OnObjectCreate event

Applies to

TGlobe object

Declaration

property OnObjectCreate: TGlobeObjectNotifyEvent;

Description

This is called when a globe object is created or loaded from the stream. The object is only guaranteed to have its Title and Origin initialised at this point.

Example:

```
procedure TForm1.GlobeObjectCreate( Sender: TObject;  
  GlobeObject: TGlobeObject );  
begin  
  if GlobeObject is TGlobePolygon then { count the number of polygons  
created }  
    Inc( giPolygonCount );  
end;
```

OnSelected event

[Example](#)

Applies to

[TGlobe](#) object

Declaration

property OnSelected: TNotifyEvent;

Description

This event occurs whenever an object is selected by setting the [TGlobe.SelectedObject](#) property.

OnSelected event example

```
{ The form holds a listbox with a list of all the objects in the layer. This
will select the line in the listbox that corresponds to the selected object }
```

```

procedure TForm.GlobeSelected(Sender: TObject);
begin
    if Globe.SelectedObject = nil then
        lbxLocations.ItemIndex := -1
    else
        with lbxLocations do
            for idx := 0 to Items.Count -1 do
                if Items.objects[idx] = Globe.SelectedObject then
                    begin
                        ItemIndex := idx;
                        Break;
                    end;
                end;
            end;
        end;
    end;
end;

```

OnZoomed event

Applies to

TGlobe object

Declaration

property OnZoomed: TNotifyEvent;

Description

This event occurs whenever the Globe is zoomed in or out.

TDrawGrid type

Unit

[Globe](#)

Declaration

```
type TDrawGrid = (dgNone, dgBehind, dgInFront);
```

Description

Controls the display of the Latitude and Longitude grid. If set to dgBehind, the grid is drawn before any other objects are rendered. If set to dgInFront then the grid is drawn after all other objects have been rendered.

TGlobeMouseButton type

Unit

[Globe](#)

Declaration

```
type TGlobeMouseButton = (mzbLeft, mzbRight, mzbMiddle, mzbNone);
```

Description

Used to specify the mouse button for the [Globe.MouseSelectButton](#), [Globe.MouseZoomButton](#) and [Globe.MouseRotateButton](#) properties

TGlobeObjectState type

Unit

[Globe](#)

Declaration

```
type TGlobeObjectState = (osReset, osRedraw, osVisible, osSelected, osTiny,  
osRenderAttribute);
```

Description

Used by the render routines to decided wether or not to display the object and which attributes to use to display the object.

TGlobeProgressEvent type

Unit

Globe

Declaration

```
type TGlobeProgressEvent = procedure (Sender: TObject; MsgType: integer; const
MsgText: string; var Abort: Boolean) of object;
```

Description

This event is called when the TGlobe.ProgressMessage is called. See TGlobe.ProgressMessage above for the MsgType vales. If the Abort flag is set by the called method then this is passed back to the ProgressMessage calling routine for action if applicable.

TGlobeProjection type

Unit

[Globe](#)

Declaration

```
type TGlobeProjection = (gpSpherical, gpCartesian, gpMercator);
```

Description

Specifies the type of projection to use to display the globe information. See [Globe.Projection](#) property.

TGreatCircle type

Unit

[Globe](#)

Declaration

```
type TGreatCircleType = ( gcShortest, gcLongest, gcCircle );
```

Description

Specifies the type of circle to draw when rendering a great circle line.

gcShortest	Connects the points with the shortest great circle line.
gcLongest	Connects the points with the longest great circle line.
gcCircle	Connects the points with a complete great circle line.

TGlobeRenderAttributesEvent type

Unit

Globe

Declaration

```
type TGlobeRenderAttributesEvent = procedure(Sender: TObject; GlobeObject: TGlobeObject; var Done: Boolean) of object;
```

Description

This is the type of the event handler that is called when the OnRenderAttributes event occurs. This event occurs before each object is rendered. At this time the called routine can setup the GlobeCanvas attributes and set the **Done** parameter to True. The object will then be rendered using the attributes set in the GlobeCanvas and not the attributes associated with the object.

TGlobeObjectNotifyEvent type

Unit

[Globe](#)

Declaration

```
type TGlobeObjectNotifyEvent = procedure(Sender: TObject; GlobeObject:  
TGlobeObject) of object;
```

Description

This is the type of the event handler that is called when the [OnObjectCreate](#) or [OnObjectFree](#) event occurs.

TGlobeTitle type

Unit

Globe

Declaration

```
type TGlobeTitle = string[64];
```

Description

This is a Title associated with an object.

TGlobeUnits type

Unit

Globe

Declaration

```
type TGlobeUnits = (Degree, NauticalMile, Mile, Yard, Foot, Inch, Kilometer,  
Meter, CentiMeter, GlobeUnit);
```

Description

These values are used specify the type of units used in an object.

TPointLL type

Unit

[Globe](#)

Declaration

```
type TPointLL = TPoint;
```

Description

This is a TPoint Object that holds a Longitude and Latitude coordinate pair specified in [GlobeUnits](#).

AngleToRadians function

Unit

Globe

Declaration

```
function AngleToRadians(iAngle: integer): Extended;
```

Description

Converts an angle in GlobeUnits to radians.

GlobeUnits

All latitude and longitude values are stored internally as GlobeUnits.

GlobeUnits are in the range $-(360 * 60 * 60 * 1500)$ to $+(360 * 60 * 60 * 1500)$.

DecimalToGlobeUnits function

Unit

Globe

Declaration

```
function DecimalToGlobeUnits(Value: Extended): integer;
```

Description

Converts a Longitude or Latitude value in decimal to GlobeUnits.

DistanceLLToLL function

Unit

[Globe](#)

Declaration

```
function DistanceLLToLL(FromLL, ToLL: TPointLL): integer;
```

Description

Calculates the great circle distance from one Longitude, Latitude point to another. The result is in [GlobeUnits](#).

EllipsoidDistanceLLToLL function

Unit

Globe

Declaration

```
function EllipsoidDistanceLLToLL(FromLL, ToLL: TPointLL): integer;
```

Description

Calculates the great circle distance from one Longitude, Latitude point to another. The result is in GlobeUnits. This function uses an ellipsoid model of the earth (WGS84) to calculate the distance. This function is slower than DistanceLLToLL but is more accurate over longer distances.

GlobeUnitsFrom function

Unit

[Globe](#)

Declaration

```
function GlobeUnitsFrom(iValue: integer; Units: TGlobeUnits): integer;
```

Description

Converts the iValue from the units specified by Units into [GlobeUnits](#). See [GlobeUnitsTo](#) for valid values for sUnits.

GlobeUnitsTo function

Unit

Globe

Declaration

```
function GlobeUnitsTo(iValue: integer; Units: TGlobeUnits): integer;
```

Description

Converts the iValue from GlobeUnits into the units specified by Units.

Units	TGloveUnit	Values	Valid for Delphi 1
D	Degree	Yes	
NM	NauticalMile		Yes
MI	Mile	No	
YD	Yard	No	
FT	Foot	No	
IN	Inch	No	
KM	Kilometer		No
M	Meter	No	
CM	Centimeter		No
GU	GlobeUnit		Yes

GlobeUnitsToDecimal function

Unit

Globe

Declaration

```
function GlobeUnitsToDecimal(iValue: integer): Extended;
```

Description

Converts a value in GlobeUnits into its Longitude, Latitude Decimal representation. The returned value is a floating point value in degrees and a fraction of a degree.

LLAsString function

Unit

[Globe](#)

Declaration

```
function LLAsString(iLong, iLat: integer): string;
```

Description

Converts the Longitude, Latitude coordinates in [GlobeUnits](#) to a string representation.

LLToStr function

Unit

[Globe](#)

Declaration

```
function LLToStr(iValue: integer; const sFmt: string): string;
```

Description

Converts a Longitude or Latitude value to a string dependant on the formatting specified in sFmt. The format parameter can contain the following specifiers:-

'D'	The number of degrees are returned.
'M'	The number of minutes are returned.
'S'	The number of seconds are returned.
'T'	The number of thirds are returned.
'U'	The number of remain GlobeUnits are returned.
'E'	The letter 'E' or 'W' is returned dependant on the value.
'N'	The letter 'N' or 'S' is returned dependant on the value.

e.g.

```
Result := LLToStr( iLong, '%d.%m.%s%E ' ) + LLToStr( iLat, '%d.%m.%s%N' );
```

This is used by the [LLAsString\(\)](#) method below to produce its result.

MaxVal function

Unit

Globe

Declaration

```
function MaxVal(lLeft, lRight: integer): integer;
```

Description

Returns the larger of the two paramters.

MinVal function

Unit

Globe

Declaration

```
function MinVal(lLeft, lRight: integer): integer;
```

Description

Returns the smaller of the two parameters.

PointLL function

Unit

[Globe](#)

Declaration

```
function PointLL(iLong, iLat: integer): TPointLL;
```

Description

Creates a [TPointLL](#) object from a pair of Longitude and Latitude values. Longitude and Latitude should be defined in [GlobeUnits](#).

RadiansToAngle function

Unit

[Globe](#)

Declaration

```
function RadiansToAngle(rad: Extended): integer;
```

Description

Converts an angle in radians to [GlobeUnits](#).

sIFE function

Unit

[Globe](#)

Declaration

```
function sIFE(bCond: Boolean; const sLeft, sRight: string): string;
```

Description

Returns a string based on the bCond parameter. If bCond is True then the sLeft string is returned otherwise the sRight string is returned.

StrToLL function

Unit

[Globe](#)

Declaration

```
function StrToLL(sValue: String; const sFmt: string): integer;
```

Description

Performs the opposite function to LLToStr. Converts a string representing a Latitude or Longitude to [GlobeUnits](#). The sFmt parameter is used to control the interpretation of the sValue string (see LLToStr for format specifiers).

StrToUnits function

Unit

[Globe](#)

Declaration

```
function StrToUnits(const sUnits: String): TGlobeUnits;
```

Description

Converts a Units string into a TGlobeUnits type value. See [GlobeUnitsTo\(\)](#) for valid Units strings.

UnitsToStr function

Unit

[Globe](#)

Declaration

```
function UnitsToStr(Units: TGlobeUnits): String;
```

Description

Converts a TGlobeUnits type value to its string representation. See [GlobeUnitsTo\(\)](#) for valid TGlobeUnit type values.

