Change Log for GPS.NET Version 1.4.6

This release contains two major feature enhancements: Automatic Device Discovery and Automatic Connection Recovery. These features will virtually eliminate the need to specify serial port settings and (even better) minimize your own technical support issues from customers who can't connect to their devices. Connection Recovery means that GPS.NET will automatically detect if a sudden disconnect occurs (like when a wireless devices goes out of range) and will attempt to reconnect every few seconds.

Changes which could break compatibility are colored Red.

- The **Receiver** class is now configured by default to automatically discover GPS devices.
- The **Receiver** class will automatically detect devices that have gone out of range and re-connect to them automatically.
- The **Receiver** class now implements **IComponent** and can be dropped onto a form (but is still multithreaded).
- New **DeviceSettingsDetector** class allows developers the ability to extend device detection easily, and create their own device detection user interfaces.
- New **DeviceSettings** class encapsulates COM port, baud rate, data bits, stop bits, and parity.
- New **BaudRateComboBox** automatically populates a ComboBox with valid baud rates found during device discovery.
- New **ComPortComboBox** automatically populates a ComboBox with valid baud rates found during device discovery.
- The **ComPort** enumeration now contains a value named **Automatic**.
- The **BaudRate** enumeration now contains a value names **Automatic**.
- The **SatelliteViewer** control now supports transparent backgrounds – partial support for VS2002/2003 (as best as possible; VS2002/2003 have a different definition of "transparent"), and full support for VS2005.
- The **SatelliteViewer** now supports circular regions. This improves support for transparency.
- The **SatelliteViewer** now scales satellite icons. This improves the look of the control at smaller sizes.
- All multithreading environments have been profiled and tuned for the best balance between CPU usage and responsiveness.
- The assembly has been reanalyzed with FxCop.
- The **Connecting** event of the **Receiver** class has changed from type **EventHandler** to **DeviceSettingsEventHandler**.
- The **Disconnecting** event of the **Receiver** class has changed from type **EventHandler** to **DeviceSettingsEventHandler**.
- Q10268 – FIXED: Icons for user controls are broken
- Q10232 – FIXED (by InstallShield): Cannot change installation folder during setup.
- Q10254 – FIXED: Incorrect Hemisphere for Latitudes under 1° west of Prime Meridian and Latitudes under 1° south of equator.

Late-Breaking Changes for Version 1.4.5

- Q10277 – FIXED: System.FormatException via ParsingErrorOccurred in Angle.Parse method, caused by lack of international support in a small section of code.
- Q10223 – FIXED: File not found "SatelliteViewerBitmap.bmp" when loading a form containing a SatelliteViewer or attempting to add a SatelliteViewer.

Late-Breaking Changes for Version 1.4.4

- The trial screen was working incorrectly when building WindowsCE applications. A special edition of GPS.NET for WindowsCE is now available during setup.
- The **Equals** method of **UtmPosition** class was not implemented. This caused any calls to **.Equals()** to return an incorrect value, especially when trying to update the current position.
- Q10189 - A "Cannot call Invoke or InvokeAsync on a control until a window handle has been created." exception was thrown in the example application. This was caused by an invalid call to **DoEvents**. The **DoEvents** call has been removed.
- Q10190 - The **Receiver.UtmPosition** property no longer automatically converts the latitude/longitude position because the conversion results in a loss of accuracy. Now, the **UtmPosition** property stores the value which is reported directly from the GPS device. This allows millimeter-accuracy GPS applications to function properly with no loss in accuracy.
- The **FixLikelihood** property had no associated event to notify that the likelihood has changed. A **FixLikelihoodChanged** event has been added to the Receiver class.
- The **ProtocolEventHandler** and **ManufacturerEventHandler** delegates were changed to **DeviceEventHandler** because they have the same signature.
- Changing the **RotationMode** property of a **SatelliteViewer** caused the compass to redraw correctly, but satellites were drawn incorrectly.
- In some cases, a **NullReferenceException** error would occur when creating a new instance of the **Receiver** class. This was caused by an uninitialized ellipsoid. This is now fixed.

Change Log for GPS.NET Version 1.4.0

This file describes each improvement which has been made to GPS.NET since the previous version was released in April 2004. Existing users of GPS.NET should read this document carefully to be aware of any minor code changes required after upgrading.

Major Improvements

- The "Visualization Tools" product has been merged with the SDK. The namespace, however, is still "StormSource.Gps.Controls". You may remove any reference to **StormSource.Gps.Controls.dll** after upgrading.
- Support for the Garmin® binary protocol has been added! You may now work with waypoints, routes, and tracks to and from any Garmin device.
- Support for the Garmin simple text protocol has been added.
- Support for UTM positions and conversions to and from lat/long.
- New trigonometric functions for calculating minimum speed, travel time, instersection of points for one position or a collection of positions.
- Brand new data parsing engine intelligently determines the protocol of data being transmitted from the GPS device, even if you change the protocol on-the-fly!
- Much improved international support for local cultures and varying numeric formats.
- GPS.NET is now rigorously tested using automated tests made using NUnit software.
- GPS.NET complies with .NET Design Guidelines enforced via FxCop software.
- GPS.NET is now available for .NET Framework version 1.1 and the installation will not break on machines using only Visual Studio® 2003. (GPS.NET version 1.3 required both 1.0 and 1.1 frameworks to be installed)
- Classes now implement the **ISerializable** interface to facilitate persistance. (Excludes .NET Compact Framework)
- Classes now implement the **IComparable** interface wherever possible to facilitate sorting.
- Added **IndexOf**, **Contains**, and **CopyTo** methods to all collections in accordance with .NET Design Guidelines.
- Documentation on the web now uses an MSDN-like tree-view. This will make it much easier to look up documentation online based on a member's name.

Documentation and Examples

A subsequent release of version 1.4 will be made which will include completed documentation and more examples, including examples written in C#.

Licensing Changes for Developers Using Previous Versions

Previous versions required developers to jump through a few hoops to successfully install GPS.NET on client machines. This included the addition of a "**licenses.lic**" embedded resource, creation of a run-time "**licensing.licx**" resource, compilation of licenses using the .NET License Compiler, and a second DLL (Xheo.Licensing.dll) before it would work on deployed machines. In version 1.4, this complexity has all been discarded -- you may now remove all licensing files as well as the Xheo DLL from your projects and use the new **SetLicenseKey** method of the **Receiver** class.

Remove from your Visual Studio development or setup project any references to the following files, as they are no longer needed:

- Runtime.lic
- Xheo.Licensing.DLL

Compatibility and Changes to the GPS.NET Framework

In order to achieve the highest level of quality and ease of use, some refactoring has taken place. Developers already using version 1.3 should examine items listed below in Red to determine how much effort is required to upgrade. It is estimated that about 5-30 minutes would be required to upgrade for most applications.

- Support for COM Interop has been discontinued due to the rapid growth of the GPS.NET framework, and the fact that the .NET framework must be redistributed with any pre-.NET application. The "pure COM" version of GPS.NET is now recommended for developers using pre-.NET platforms such as Visual Basic 6 and unmanaged C++.
- The **Degree** class has been replaced with the **Latitude** and **Longitude** classes. These classes have the same members as the **Degree** class but have been separated to allow for better enforcement of hemispheres and automatic adjustment of hemispheres.
- The **PositionDilutionOfPrecision** property of the **Receiver** class has changed to **MeanDilutionOfPrecision**.
- The enumerative value **DilutionOfPrecisionType.Position** has changed to **DilutionOfPrecisionType.Mean**.
- The **PositionDilutionOfPrecisionChanged** event of the **Receiver** class has changed to **MeanDilutionOfPrecisionChanged**.
- The **MagneticVariation** property now returns a **Longitude** object.
- The **ToLocalTime** method of the **Receiver** class was removed since it was provided only for COM Interop users, and COM Interop support has been discontinued.
- The **ToCardinalDirection** method of the **Azimuth** class has changed to the **CardinalDirection** property.
- The **Indicator** property of the former **Degree** class has been changed to **LatitudeHemisphere** and **LongitudeHemisphere**.
- **DilutionOfPrecisionChanged** event of the **DilutionOfPrecision** class has changed to **ValueChanged**
- The **Value** property of the **DilutionOfPrecision** class has changed from an **Integer** to a **Double** type.
- The **SentenceType** property of the **SentenceEventArgs** class has changed to **Type**.
- The **Prc** property of the **Satellite** class has changed to **PseudoRandomCode**.

Receiver Class Improvements

- New **MagneticBearing** property (Azimuth type)
- New $PGRMM sentence (current Datum)
- New $HCHDG sentence (Heading change/eTrex built-in compass)
- New **Timeout** event when sending commands to a GPS device.
- New **Device** property exposes hardware-specific information (see Device class).
- **Altitude**, **Bearing** and **Speed** are now shared.
- New shared **UtmPosition** property.
- $GPVTG sentence added
- $GPRMB sentence (recommended minimum navigation)
- $PGRMZ sentence added (Garmin® devices only)
- $PGRMM sentence added (Garmin® devices only)
- $GPRTE sentence added (Garmin® devices only)
- New **AltitudeAboveWgs1984** property.
- New **DifferentialGps** property and class which reports differential GPS information such as station ID, time since last DGPS update, and date/time of last fix is now available.
- Fixed error in $GPGSA sentence where satellite fix information was assumed to be present when it was not actually specified.
- New **Send** method to send arbitrary commands
- *FIX:* In some cases the **BaudRate** and **ComPort** properties were overridden when calling the **Start** method.
- If information is not understood after a size limit is reached, the Receiver will attempt to auto-detect the baud rate and protocol.
- *FIX:* Fixed error where GPS device would apparently stop reporting data when a fix was received. The bug was during the processing of GPGSA messages.
- New **PositionReceived** event reports the latest position from the GPS device, *even if the position has not actually changed*. Useful for position-averaging techniques.

New **Angle** class:

- This class serves as the base class for the **Azimuth**, **Latitude** and **Longitude** classes, since they are so similar.
- This class can be inherited in your own projects.
- Includes shared members for maximum and minimum values (0-359.9)
- *FIX:* Fixed rounding error caused when calculating hours, minutes, and seconds after the DecimalDegrees had changed. This caused some degree measurements to report a **Seconds** of 60 instead of zero.

- Angular measurements are now automatically normalized if **Seconds** or **Minutes** are set to sixty or above. (i.e. 1°60'120" becomes 2°1'1")
- Much improved **Parse** method supports more flexible H/M/S measurements. The degree symbol and other symbols are no longer required for string-based parsing. Values like "105 12 45" are now acceptable.

New **Radian** class:

- Facilitates easy conversion between angles and radians for custom trigonometric functions.

Azimuth class:

- Now derives from the **Angle** class.
- Shared fields are now available for any of sixteen cardinal directions, such as **North**, **NorthWest** and **NorthNorthWest**.
- New shared **CurrentBearing** property returns the current direction of travel.
- New shared **CurrentMagneticBearing** returns the current direction of travel adjusted by magnetic variation.
- Values set in this class are automatically normalized to fit between 0° and 359.9999°.

Distance class:

- Added support for nautical mile distance types.
- New **ToNauticalMiles** method.
- New shared **CurrentAltitude** property.
- New shared **CurrentAltitudeAboveWgs1984** property for height above the WGS84 ellipsoid (an imaginary oval shape intended to fit as much of the Earth inside it as possible).

Latitude and **Longitude** classes (formerly the **Degree** class):

- Fixed error where **ToString** did not produce the same results as the string passed to **Parse**.
- These class are derived from the new **Angle** base class.
- New shared members for the equator, both poles, the Prime Meridian, Tropic of Capricorn and Tropic of Cancer.
- New read/write **DecimalMinutes** property.
- Includes shared members for common latitudelongitude values, such as **Equator**, **PrimeMeridian**, **TropicOfCancer**, etc.
- New shared **CurrentLatitude** and **CurrentLongitude** properties.
- The values of the **DecimalDegrees** and **Hours** properties of the **Latitude** class are automatically normalized to be between -90° and 90°.
- The values of the **DecimalDegrees** and **Hours** properties of the **Longitude** class are automatically normalized to be between -180° and 180°.

Position Class

- New shared **Parse** method for converting string-based positional measurements into a position object.
- New **TimeTo** method calculates the travel time to a destination hen given a speed. Also provided as a shared method.
- New **SpeedTo** method calculates the minimum speed required to reach a destination in a given period of time. Also provided as a shared method.
- New **Translate** method adjusts a position by the given bearing and distance.
- New **TranslateTo** method returns a new position object relative shifted by the given bearing and distance. Also provided as a shared method.
- New **IntersectionOf** method returns the point at which two positions intersect, when given two positions and two bearings. Also provided as a shared method.
- New **ToUtmPosition** method converts the current position (ragardless of its coordinate system!) into a UTM northing/easting/zone coordinate.
- New **ToEllipsoid** method converts a position to any of several ellipsoids using Molodensky transforms.
- New shared **CurrentPosition** property returns the current location.

- The **DistanceTo** method has been improved to use a more accurate version of the Haversine formula. Error has been reduced from 0.5% to 0.02% even over large distances.
- This class is now inheritable!

New **UtmPosition** class:

- Provides the ability to easily work with Universal Transverse Mercator (UTM) coordinates, and convert to and from Position objects.
- Provides the same methods as the Position class for distance, bearing, time, speed, translation, conversion, and each methods work across UTM zone boundaries!
- Includes a **ToPosition** class to easily convert from UTM to lat/long coordinates (and back).
- This class is now inheritable!

New **PositionCollection** class:

- This overridable class provides ad-hoc position-related features such as total distance, time to travel through points, speed required to travel through all points within a certain time, and translation of a group of points.
- This class is now inheritable!

New **Waypoint** class:

- This powerful class allows for the management of GPS device waypoints while requiring no knowledge of waypoint communications protocols. Waypoints can be easily transferred between GPS devices by simply calling the **Save** method. This class is derived from the **Position** class and can therefore be used with any Position-based method. This class supports all Garmin waypoint types, from D100 to D155!
- Derives from the **Position** class, making it easy to use distance, bearing, translation, and conversion to **UtmPosition** objects.

New **Device** class:

- This class stores hardware-specific information about the device, such as the manufacturer, model, current data protocol (i.e. NMEA, Garmin, etc.), almanac information and, when possible, remaining battery life.
- **Protocol** property: Indicates the data protocol currently being received from the GPS device. When the protocol is changed, the new ProtocolChanged event is raised.
- **Manufacturer** property: Indicates the manufacturer of the GPS device. This property is typically Unknown unless information is received which uniquely identifies the device.
- **BatteryLife** property (used by Magellan devices)

DilutionOfPrecision class:

- New **RatingChanged** event for notification of important changes in positional confidence level.
- New **CurrentMean** shared property returns overall positional accuracy.
- New **CurrentHorizontal** shared property returns current horizontal DOP.
- New **CurrentVertical** shared property returns current altitude accuracy.
- New **ToString** method returns DOP reads as "Value (Rating)".
- The **Value** property has changed from an **Integer** to a **Double** type for greater accuracy.

Satellite class:

- Now derives from the **CelestialBody** class. This allows for more functionality and custom drawing of satellite information.

New **PositionError** class (for Garmin devices)

- Reports horizontal, vertical, mean error in measurable distances.

New **Geodesy** class:

This class provides access to shared collections of international ellipsoids, datums, and local datums, and functionality to convert between over 200 coordinate systems. Information is organized in the following classes:

- Ellipsoid
- Datum
- MolodenskyTransform

NOTE: Grid conversions could not be completed in time to make the 1.4 release date. They will be included in version 1.5.

Exception Management

- New **ParsingErrorOccurred** event allows separation between recoverable warnings (parsing errors) and non-recoverable errors (via **ErrorOccurred** event)
- *FIX:* In some cases, errors which occured when handling GPS.NET events caused an exception to be thrown back inside GPS.NET code. This would result in a false error report. Raised events are now within **Try..Catch** blocks to prevent confusing errors raised within a developer's own event code.

SatelliteCollection Class

- **Item** method of **SatelliteCollection** now throws an **IndexOutOfRangeException** instead of a **GpsException**.