CLLocation Class Reference

Data Management: Device Information

2010-05-11

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

CLLocation Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CoreLocation.framework
Availability	Available in iOS 2.0 and later.
Declared in	CLLocation.h

Overview

A `CLLocation` object represents the location data generated by a `CLLocationManager` object. This object incorporates the geographical coordinates and altitude of the device's location along with values indicating the accuracy of the measurements and when those measurements were made. In iOS, this class also reports information about the speed and heading in which the device is moving.

Typically, you use a `CLLocationManager` object to create instances of this class based on the last known location of the user's device. You can create instances yourself, however, if you want to cache custom location data or get the distance between two points.

This class is designed to be used as is and should not be subclassed.

Tasks

Initializing a Location Object

`– initWithLatitude:longitude:` (page 11)
Initializes and returns a location object with the specified latitude and longitude.

`– initWithCoordinate:altitude:horizontalAccuracy:verticalAccuracy:timestamp:` (page 10)
Initializes and returns a location object with the specified coordinate information.

Location Attributes

`coordinate` (page 7) *property*
The geographical coordinate information. (read-only)

`altitude` (page 6) *property*
The altitude measured in meters. (read-only)

`horizontalAccuracy` (page 8) *property*
The radius of uncertainty for the location, measured in meters. (read-only)

`verticalAccuracy` (page 9) *property*
The accuracy of the altitude value in meters. (read-only)

`timestamp` (page 8) *property*
The time at which this location was determined. (read-only)

`– description` (page 9)
Returns the location data in a formatted text string.

Measuring the Distance Between Coordinates

`– distanceFromLocation:` (page 9)
Returns the distance (in meters) from the receiver's location to the specified location.

`– getDistanceFrom:` (page 15) Deprecated in iOS 3.2
Returns the distance (in meters) from the receiver's location to the specified location. (Deprecated. Use the `distanceFromLocation:` (page 9) method instead.)

Getting Speed and Course Information

`speed` (page 8) *property*
The instantaneous speed of the device in meters per second.

`course` (page 7) *property*
The direction in which the device is traveling.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

altitude

The altitude measured in meters. (read-only)

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) CLLocationDistance altitude
```

Discussion
Positive values indicate altitudes above sea level. Negative values indicate altitudes below sea level.

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability

Available in iOS 2.0 and later.

See Also

`@property verticalAccuracy` (page 9)

Declared In

`CLLocation.h`

coordinate

The geographical coordinate information. (read-only)

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) CLLocationCoordinate2D coordinate
```

Discussion

When running in the simulator, Core Location assigns a fixed set of coordinate values to this property. You must run your application on an iOS-based device to get real location values.

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

course

The direction in which the device is traveling.

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) CLLocationDirection course
```

Discussion

Course values are measured in degrees starting at due north and continuing clockwise around the compass. Thus, north is 0 degrees, east is 90 degrees, south is 180 degrees, and so on. Course values may not be available on all devices. A negative value indicates that the direction is invalid.

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability

Available in iOS 2.2 and later.

Declared In

`CLLocation.h`

horizontalAccuracy

The radius of uncertainty for the location, measured in meters. (read-only)

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) CLLocationAccuracy horizontalAccuracy
```

Discussion

The location's latitude and longitude identify the center of the circle, and this value indicates the radius of that circle. A negative value indicates that the location's latitude and longitude are invalid.

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

speed

The instantaneous speed of the device in meters per second.

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) CLLocationSpeed speed
```

Discussion

This value reflects the instantaneous speed of the device in the direction of its current heading. A negative value indicates an invalid speed. Because the actual speed can change many times between the delivery of subsequent location events, you should use this property for informational purposes only.

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability

Available in iOS 2.2 and later.

Declared In

`CLLocation.h`

timestamp

The time at which this location was determined. (read-only)

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) NSDate *timestamp
```

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

verticalAccuracy

The accuracy of the altitude value in meters. (read-only)

```
@property(readonly, NS_NONATOMIC_IPHONEONLY) CLLocationAccuracy verticalAccuracy
```

Discussion
The value in the `altitude` property could be plus or minus the value indicated by this property. A negative value indicates that the altitude value is invalid.

Determining the vertical accuracy requires a device with GPS capabilities. Thus, on some earlier iOS-based devices, this property always contains a negative value.

Special Considerations

In iOS, this property is declared as `nonatomic`. In Mac OS X, it is declared as `atomic`.

Availability
Available in iOS 2.0 and later.

See Also
`@property altitude` (page 6)

Declared In
`CLLocation.h`

Instance Methods

description

Returns the location data in a formatted text string.

```
- (NSString *)description
```

Return Value
A string of the form "<<*latitude*>, <*longitude*>> +/- <*accuracy*>m (speed <*speed*> kph / heading <*heading*>) @ <*date-time*>", where <*latitude*>, <*longitude*>, <*accuracy*>, <*speed*>, and <*heading*> are formatted floating point numbers and <*date-time*> is a formatted date string that includes date, time, and time zone information.

Discussion
The returned string is intended for display purposes only.

Availability
Available in iOS 2.0 and later.

Declared In
`CLLocation.h`

distanceFromLocation:

Returns the distance (in meters) from the receiver's location to the specified location.

```
- (CLLocationDistance)distanceFromLocation:(const CLLocation *)location
```

Parameters

location

The other location.

Return Value

The distance (in meters) between the two locations.

Discussion

This method measures the distance between the two locations by tracing a line between them that follows the curvature of the Earth. The resulting arc is a smooth curve and does not take into account specific altitude changes between the two locations.

Availability

Available in iOS 3.2 and later.

Declared In

`CLLocation.h`

initWithCoordinate:altitude:horizontalAccuracy:verticalAccuracy:timestamp:

Initializes and returns a location object with the specified coordinate information.

```
- (id)initWithCoordinate:(CLLocationCoordinate2D)coordinate
    altitude:(CLLocationDistance)altitude
    horizontalAccuracy:(CLLocationAccuracy)hAccuracy
    verticalAccuracy:(CLLocationAccuracy)vAccuracy timestamp:(NSDate *)timestamp
```

Parameters

coordinate

A coordinate structure containing the latitude and longitude values.

altitude

The altitude value for the location.

hAccuracy

The accuracy of the coordinate value. Specifying a negative number indicates that the coordinate value is invalid.

vAccuracy

The accuracy of the altitude value. Specifying a negative number indicates that the altitude value is invalid.

timestamp

The time to associate with the location object. Typically, you would set this to the current time.

Return Value

A location object initialized with the specified information.

Discussion

Typically, you acquire location objects from the location service, but you can use this method to create new location objects for other uses in your application.

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

initWithLatitude:longitude:

Initializes and returns a location object with the specified latitude and longitude.

```
- (id)initWithLatitude:(CLLocationDegrees)latitude
    longitude:(CLLocationDegrees)longitude
```

Parameters

latitude

The latitude of the coordinate point.

longitude

The longitude of the coordinate point.

Return Value

A location object initialized with the specified coordinate point.

Discussion

Typically, you acquire location objects from the location service, but you can use this method to create new location objects for other uses in your application. When using this method, the other properties of the object are initialized to appropriate values. In particular, the `altitude` and `horizontalAccuracy` properties are set to 0, the `verticalAccuracy` property is set to -1 to indicate that the altitude value is invalid, and the `timestamp` property is set to the time at which the instance was initialized.

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

Constants

CLLocationDegrees

Represents a latitude or longitude value specified in degrees.

```
typedef double CLLocationDegrees;
```

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

CLLocationCoordinate2D

A structure that contains a geographical coordinate using the WGS 84 reference frame.

```
typedef struct {
    CLLocationDegrees latitude;
    CLLocationDegrees longitude;
} CLLocationCoordinate2D;
```

Fields

`latitude`

The latitude in degrees. Positive values indicate latitudes north of the equator. Negative values indicate latitudes south of the equator.

`longitude`

The longitude in degrees. Measurements are relative to the zero meridian, with positive values extending east of the meridian and negative values extending west of the meridian.

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

CLLocationAccuracy

Represents the accuracy of a coordinate value in meters.

```
typedef double CLLocationAccuracy;
```

Availability

Available in iOS 2.0 and later.

Declared In

`CLLocation.h`

Accuracy Constants

Constant values you can use to specify the accuracy of a location.

```
extern const CLLocationAccuracy kCLLocationAccuracyBestForNavigation;
extern const CLLocationAccuracy kCLLocationAccuracyBest;
extern const CLLocationAccuracy kCLLocationAccuracyNearestTenMeters;
extern const CLLocationAccuracy kCLLocationAccuracyHundredMeters;
extern const CLLocationAccuracy kCLLocationAccuracyKilometer;
extern const CLLocationAccuracy kCLLocationAccuracyThreeKilometers;
```

Constants

`kCLLocationAccuracyBestForNavigation`

Use the highest possible accuracy and combine it with additional sensor data. This level of accuracy is intended for use in navigation applications that require precise position information at all times and are intended to be used only while the device is plugged in.

Available in iOS 4.0 and later.

Declared in `CLLocation.h`.

`kCLLocationAccuracyBest`

Use the highest-level of accuracy.

Available in iOS 2.0 and later.

Declared in `CLLocation.h`.

`kCLLocationAccuracyNearestTenMeters`

Accurate to within ten meters of the desired target.

Available in iOS 2.0 and later.

Declared in `CLLocation.h`.

`kCLLocationAccuracyHundredMeters`

Accurate to within one hundred meters.

Available in iOS 2.0 and later.

Declared in `CLLocation.h`.

`kCLLocationAccuracyKilometer`

Accurate to the nearest kilometer.

Available in iOS 2.0 and later.

Declared in `CLLocation.h`.

`kCLLocationAccuracyThreeKilometers`

Accurate to the nearest three kilometers.

Available in iOS 2.0 and later.

Declared in `CLLocation.h`.

CLLocationSpeed

Represents the speed at which the device is moving in meters per second.

```
typedef double CLLocationSpeed;
```

Availability
Available in iOS 2.2 and later.

Declared In
`CLLocation.h`

CLLocationDirection

Represents a direction that is measured in degrees relative to true north.

```
typedef double CLLocationDirection;
```

Discussion
Direction values are measured in degrees starting at due north and continue clockwise around the compass. Thus, north is 0 degrees, east is 90 degrees, south is 180 degrees, and so on. A negative value indicates an invalid direction.

Availability
Available in iOS 2.2 and later.

Declared In
`CLLocation.h`

Specifying an Invalid Coordinate

Use this constant whenever you want to indicate that a coordinate is invalid.

```
const CLLocationCoordinate2D kCLLocationCoordinate2DInvalid
```

Constants

`kCLLocationCoordinate2DInvalid`

An invalid coordinate value.

Available in iOS 4.0 and later.

Declared in `CLLocation.h`.

Deprecated CLLocation Methods

A method identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in iOS 3.2

getDistanceFrom:

Returns the distance (in meters) from the receiver's location to the specified location. (Deprecated in iOS 3.2. Use the `distanceFromLocation:` (page 9) method instead.)

```
- (CLLocationDistance)getDistanceFrom:(const CLLocation *)location
```

Parameters

location

The other location.

Return Value

The distance (in meters) between the two locations.

Discussion

This method measures the distance between the two locations by tracing a line between them that follows the curvature of the Earth. The resulting arc is a smooth curve and does not take into account specific altitude changes between the two locations.

Availability

Available in iOS 2.0 and later.

Deprecated in iOS 3.2.

Declared In

`CLLocation.h`

Document Revision History

This table describes the changes to *CLLocation Class Reference*.

Date	Notes
2010-05-11	Updated to include symbols introduced in iOS 4.0.
2010-02-25	Updated for iOS 3.2.
2009-08-04	Updated the document to reflect the availability of the interfaces in Mac OS X v10.6.
2009-02-06	Documented units for the speed property.
2008-10-23	Documented new properties in iOS v2.2.
2008-05-27	New document that describes the class for representing the geographical coordinates and altitude of a device's location, and related values.