
CLLocationManagerDelegate Protocol Reference

Data Management: Device Information



2010-05-11



Apple Inc.
© 2010 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

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

Apple, the Apple logo, iPhone, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

iOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

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

CLLocationManagerDelegate Protocol Reference 5

Overview 5

Tasks 5

 Responding to Location Events 5

 Responding to Heading Events 5

 Responding to Region Events 6

Instance Methods 6

 locationManager:didEnterRegion: 6

 locationManager:didExitRegion: 6

 locationManager:didFailWithError: 7

 locationManager:didUpdateHeading: 8

 locationManager:didUpdateToLocation:fromLocation: 8

 locationManager:monitoringDidFailForRegion:withError: 9

 locationManagerShouldDisplayHeadingCalibration: 9

Document Revision History 11

CLLocationManagerDelegate Protocol Reference

Conforms to	NSObject
Framework	/System/Library/Frameworks/CoreLocation.framework
Availability	Available in iOS 2.0 and later.
Declared in	CLLocationManagerDelegate.h

Overview

The `CLLocationManagerDelegate` protocol defines the methods used to receive location and heading updates from a `CLLocationManager` object. The methods of this protocol are optional.

Upon receiving a successful location or heading update, you can use the result to update your user interface or perform other actions. Similarly, if the location or heading could not be determined, you might want to stop updates for a short period of time and try again later. You can use the `stopUpdatingLocation`, `stopMonitoringSignificantLocationChanges`, `stopUpdatingHeading`, or `stopMonitoringForRegion:` methods of `CLLocationManager` to stop location, heading, and region updates.

The methods of your delegate object are called from the thread in which you started the corresponding location services. That thread must itself have an active run loop, like the one found in your application's main thread.

Tasks

Responding to Location Events

- [locationManager:didUpdateToLocation:fromLocation:](#) (page 8)
Tells the delegate that a new location value is available.
- [locationManager:didFailWithError:](#) (page 7)
Tells the delegate that the location manager was unable to retrieve a location value.

Responding to Heading Events

- [locationManager:didUpdateHeading:](#) (page 8)
Tells the delegate that the location manager received updated heading information.

- [locationManagerShouldDisplayHeadingCalibration:](#) (page 9)
Asks the delegate whether the heading calibration alert should be displayed.

Responding to Region Events

- [locationManager:didEnterRegion:](#) (page 6)
Tells the delegate that the user entered the specified region.
- [locationManager:didExitRegion:](#) (page 6)
Tells the delegate that the user left the specified region.
- [locationManager:monitoringDidFailForRegion:withError:](#) (page 9)
Tells the delegate that a region monitoring error occurred.

Instance Methods

locationManager:didEnterRegion:

Tells the delegate that the user entered the specified region.

```
(void)locationManager:(CLLocationManager *)managerdidEnterRegion:(CLRegion *)region
```

Parameters

manager

The location manager object reporting the event.

region

An object containing information about the region that was entered.

Discussion

Because regions are a shared application resource, every active location manager object delivers this message to its associated delegate. It does not matter which location manager actually registered the specified region. And if multiple location managers share a delegate object, that delegate receives the message multiple times.

The region object provided may not be the same one that was registered. As a result, you should never perform pointer-level comparisons to determine equality. Instead, use the region's identifier string to determine if your delegate should respond.

Availability

Available in iOS 4.0 and later.

Declared In

CLLocationManagerDelegate.h

locationManager:didExitRegion:

Tells the delegate that the user left the specified region.

```
(void)locationManager:(CLLocationManager *)managerdidExitRegion:(CLRegion *)region
```

Parameters*manager*

The location manager object reporting the event.

region

An object containing information about the region that was exited.

Discussion

Because regions are a shared application resource, every active location manager object delivers this message to its associated delegate. It does not matter which location manager actually registered the specified region. And if multiple location managers share a delegate object, that delegate receives the message multiple times.

The region object provided may not be the same one that was registered. As a result, you should never perform pointer-level comparisons to determine equality. Instead, use the region's identifier string to determine if your delegate should respond.

Availability

Available in iOS 4.0 and later.

Declared In

`CLLocationManagerDelegate.h`

locationManager:didFailWithError:

Tells the delegate that the location manager was unable to retrieve a location value.

```
- (void)locationManager:(CLLocationManager *)manager didFailWithError:(NSError *)error
```

Parameters*manager*

The location manager object that was unable to retrieve the location.

error

The error object containing the reason why the location or heading could not be retrieved.

Discussion

Implementation of this method is optional. You should implement this method, however.

If the location service is unable to retrieve a location fix right away, it reports a `kCLLocationErrorLocationUnknown` error and keeps trying. In such a situation, you can simply ignore the error and wait for a new event.

If the user denies your application's use of the location service, this method reports a `kCLLocationErrorDenied` error. Upon receiving such an error, you should stop the location service.

If a heading could not be determined because of strong interference from nearby magnetic fields, this method returns `kCLLocationErrorHeadingFailure`.

Availability

Available in iOS 2.0 and later.

See Also

`CLLocationError`

Declared In

`CLLocationManagerDelegate.h`

locationManager:didUpdateHeading:

Tells the delegate that the location manager received updated heading information.

```
- (void)locationManager:(CLLocationManager *)manager didUpdateHeading:(CLHeading *)newHeading
```

Parameters

manager

The location manager object that generated the update event.

newHeading

The new heading data.

Discussion

Implementation of this method is optional but expected if you start heading updates using the `startUpdatingHeading` method.

The location manager object calls this method after you initially start the heading service. Subsequent events are delivered when the previously reported value changes by more than the value specified in the `headingFilter` property of the location manager object.

Availability

Available in iOS 3.0 and later.

Declared In

`CLLocationManagerDelegate.h`

locationManager:didUpdateToLocation:fromLocation:

Tells the delegate that a new location value is available.

```
- (void)locationManager:(CLLocationManager *)manager didUpdateToLocation:(CLLocation *)newLocation fromLocation:(CLLocation *)oldLocation
```

Parameters

manager

The location manager object that generated the update event.

newLocation

The new location data.

oldLocation

The location data from the previous update. If this is the first update event delivered by this location manager, this parameter is `nil`.

Discussion

Implementation of this method is optional. You should implement this method, however.

By the time this message is delivered to your delegate, the new location data is also available directly from the `CLLocationManager` object. The *newLocation* parameter may contain the data that was cached from a previous usage of the location service. You can use the `timestamp` property of the location object to determine how recent the location data is.

Availability

Available in iOS 2.0 and later.

Declared In

CLLocationManagerDelegate.h

locationManager:monitoringDidFailForRegion:withError:

Tells the delegate that a region monitoring error occurred.

```
- (void)locationManager:(CLLocationManager *)manager monitoringDidFailForRegion:(CLRegion *)region withError:(NSError *)error
```

Parameters*manager*

The location manager object reporting the event.

region

The region for which the error occurred.

error

An error object containing the error code that indicates why region monitoring failed.

Discussion

If an error occurs while trying to monitor a given region, the location manager sends this message to its delegate. Region monitoring might fail because the region itself cannot be monitored or because there was a more general failure in configuring the region monitoring service.

Although implementation of this method is optional, it is recommended that you implement it if you use region monitoring in your application.

Availability

Available in iOS 4.0 and later.

Declared In

CLLocationManagerDelegate.h

locationManagerShouldDisplayHeadingCalibration:

Asks the delegate whether the heading calibration alert should be displayed.

```
- (BOOL)locationManagerShouldDisplayHeadingCalibration:(CLLocationManager *)manager
```

Parameters*manager*

The location manager object coordinating the display of the heading calibration alert.

Return Value

YES if you want to allow the heading calibration alert to be displayed or NO if you do not.

Discussion

Core Location may call this method in an effort to calibrate the onboard hardware used to determine heading values. Typically, Core Location calls this method at the following times:

- The first time heading updates are ever requested
- When Core Location observes a significant change in magnitude or inclination of the observed magnetic field

If you return `YES` from this method, Core Location displays the heading calibration alert on top of the current window immediately. The calibration alert prompts the user to move the device in a particular pattern so that Core Location can distinguish between the Earth's magnetic field and any local magnetic fields. The alert remains visible until calibration is complete or until you explicitly dismiss it by calling the `dismissHeadingCalibrationDisplay` method. In this latter case, you can use this method to set up a timer and dismiss the interface after a specified amount of time has elapsed.

Note: The calibration process is able to filter out only those magnetic fields that move with the device. To calibrate a device that is near other sources of magnetic interference, the user must either move the device away from the source or move the source in conjunction with the device during the calibration process.

If you return `NO` from this method or do not provide an implementation for it in your delegate, Core Location does not display the heading calibration alert. Even if the alert is not displayed, calibration can still occur naturally when any interfering magnetic fields move away from the device. However, if the device is unable to calibrate itself for any reason, the value in the `headingAccuracy` property of any subsequent events will reflect the uncalibrated readings.

Availability

Available in iOS 3.0 and later.

Declared In

`CLLocationManagerDelegate.h`

Document Revision History

This table describes the changes to *CLLocationManagerDelegate Protocol Reference*.

Date	Notes
2010-05-11	Updated to include symbols introduced in iOS 4.0.
2009-07-28	Updated the document to reflect the availability of the interfaces in Mac OS X v10.6.
2009-05-12	Updated for iOS 3.0.
2008-05-27	New document that describes the protocol for receiving location updates from a CLLocationManager object.

REVISION HISTORY

Document Revision History