

iOS 无障碍协议

中国信息无障碍产品联盟&信息无障碍研究会 译制

20160819

翻译声明

翻译机构： [信息无障碍研究会 \(ARA\)](#) [中国信息无障碍产品联盟 \(CAPA\)](#)

译者： 刘辉

审阅： 刘彪、沈广荣

本文档翻译自苹果 iOS Developer Library 的官方文档[无障碍协议](#)，如您对翻译文档内容有异议，请将原文文档作为主要参考，原文版权由 Apple Inc 持有并保留。

本翻译文档使用请参见 [CC BY-NC-SA 3.0](#)。文档可以免费使用、分享，但请保留本链接，如您有任何内容上的修改，请发送邮件至 liuhui@siaa.org.cn，我们只是希望文档内容能够统一完整，真正帮助开发者完善产品的信息无障碍。

目录

翻译声明.....	0
1.UIAccessibility	1
1.1 任务.....	2
1.1.1 判定无障碍.....	2
1.1.1.1 isAccessibilityElement 属性.....	2
1.1.2 配置无障碍元素.....	2
1.1.2.1 accessibilityActivationPoint 属性	2
1.1.2.2 accessibilityElementsHidden 属性	3
1.1.2.3 accessibilityFrame 属性.....	4
1.1.2.4 accessibilityHint 属性.....	5
1.1.2.5 accessibilityLabel 属性.....	6
1.1.2.6 accessibilityLanguage 属性	7
1.1.2.7 accessibilityPath 属性.....	7
1.1.2.8 accessibilityTraits 属性.....	8
1.1.2.9 accessibilityValue 属性.....	9
1.1.2.10 accessibilityViewIsModal 属性	9
1.1.2.11 shouldGroupAccessibilityChildren 属性	10
1.1.2.12 accessibilityNavigationStyle 属性	11
1.2 数据类型.....	12
1.2.1 UIAccessibilityTraits	12
1.2.2 UIAccessibilityZoomType	12
1.2.3 UIAccessibilityNotifications.....	14
1.3 常量.....	15
1.3.1 无障碍特性（Accessibility Traits）	15
1.3.2 通知字典关键值（Notification Dictionary Keys）	19
1.3.3 属性字符串的语音属性（Speech Attributes for Attributed Strings）	20
1.3.4 辅助技术标识（Assistive Technology Identifiers）	22
1.3.5 UIAccessibilityNavigationStyle	22
1.4 通知.....	24
1.4.1 UIAccessibilityAnnouncementNotification.....	24
1.4.2 UIAccessibilityAnnouncementDidFinishNotification.....	25

1.4.3 UIAccessibilityBoldTextStatusDidChangeNotification	26
1.4.4 UIAccessibilityClosedCaptioningStatusDidChangeNotification	26
1.4.5	
UIAccessibilityDarkerSystemColorsStatusDidChangeNotification.....	27
1.4.6 UIAccessibilityGrayscaleStatusDidChangeNotification	28
1.4.7 UIAccessibilityGuidedAccessStatusDidChangeNotification	28
1.4.8 UIAccessibilityInvertColorsStatusDidChangeNotification	29
1.4.9 UIAccessibilityLayoutChangedNotification.....	30
1.4.10 UIAccessibilityMonoAudioStatusDidChangeNotification	31
1.4.11 UIAccessibilityPageScrolledNotification	31
1.4.12 UIAccessibilityPauseAssistiveTechnologyNotification	32
1.4.13 UIAccessibilityReduceMotionStatusDidChangeNotification ..	33
1.4.14	
UIAccessibilityReduceTransparencyStatusDidChangeNotification	34
1.4.15 UIAccessibilityResumeAssistiveTechnologyNotification.....	34
1.4.16 UIAccessibilityScreenChangedNotification	35
1.4.17 UIAccessibilitySpeakScreenStatusDidChangeNotification	36
1.4.18 UIAccessibilitySpeakSelectionStatusDidChangeNotification ..	36
1.4.19 UIAccessibilitySwitchControlStatusDidChangeNotification...	37
1.4.20 UIAccessibilityVoiceOverStatusChanged	38
2. UIAccessibilityAction.....	40
2.1 执行操作.....	41
2.1.1 – accessibilityActivate	41
2.1.2 – accessibilityIncrement.....	42
2.1.3 – accessibilityDecrement	42
2.1.4 – accessibilityScroll:	43
2.1.5 – accessibilityPerformEscape	44
2.1.6 – accessibilityPerformMagicTap	44
2.2 访问自定义操作.....	46
2.2.1 accessibilityCustomActions 属性.....	46
2.3 数据类型.....	47
2.3.1 UIAccessibilityScrollDirection.....	47
3. UIAccessibilityContainer.....	50

3.1 提供关于无障碍元素的信息.....	51
3.1.1 - accessibilityElementCount	51
3.1.2 - accessibilityElementAtIndex:	51
3.1.3 - indexOfAccessibilityElement:	52
3.1.4 accessibilityElements 属性	53
4. UIAccessibilityFocus	54
4.1 获得焦点信息.....	55
4.1.1 – accessibilityElementDidBecomeFocused	55
4.1.2 – accessibilityElementDidLoseFocus	55
4.1.3 – accessibilityElementIsFocused	56
5. UIAccessibilityIdentification	57
5.1 访问一个元素的标识符.....	58
5.1.1 accessibilityIdentifier 必填属性	58
6. UIAccessibilityReadingContent	59
6.1 访问页面上的内容.....	60
6.1.1 - accessibilityLineNumberForPoint:必填.....	60
6.1.2 - accessibilityContentForLineNumber:必填.....	61
6.1.3 - accessibilityFrameForLineNumber:必填	61
6.1.4 - accessibilityPageContent 必填	62

1. UIAccessibility

继承自: Not Applicable;

遵循: Not Applicable;

导入语句:

```
OBJECTIVE-C
```

```
@import UIKit;
```

可获得性: 在 iOS3.0 及更高版本中可获得;

UIAccessibility 非正式协议提供关于应用用户界面元素的无障碍信息。辅助应用, 例如 VoiceOver, 将此信息传递给残障用户, 帮助其使用应用。

标准 UIKit 控件和视图默认实现 UIAccessibility 方法, 因此其对辅助应用默认可访问。这意味着如果应用只使用标准控件和视图, 例如 UIButton、UISegmentedControl 和 UITableView, 开发者只需要在默认值不完整的时候, 提供应用特定的细节信息。开发者可以在 Interface Builder 中设置这些值, 或在该非正式协议中设置这些属性。

UIAccessibility 非正式协议也可由 UIAccessibilityElement 类实现, 该类呈现自定义用户界面对象。如果创建一个自定义 UIView 子类, 开发者需要创建一个 UIAccessibilityElement 实例来呈现它。在这种情景下, 开发者要保证正确设置所有 UIAccessibility 属性并返回无障碍元素的属性。

1.1 任务

1.1.1 判定无障碍

1.1.1.1 isAccessibilityElement 属性

一个布尔值，标识接收者是个辅助应用可访问的无障碍元素。

声明

```
SWIFT
```

```
var isAccessibilityElement: Bool
```

```
OBJECTIVE-C
```

```
@property(nonatomic) BOOL isAccessibilityElement
```

简介

该属性的默认值为 NO，除非接收者是个标准 UIKit 控件，该情景下，值为 YES。

辅助应用只可以获得被无障碍元素呈现的对象的信息。因此，如果实现一个应该对残障用户可访问的自定义控件或视图，将该属性设置为 YES。该方式的唯一例外是该视图只是作为其他应该被可访问项目的容器。这样的视图应该实现 UIAccessibilityContainer 协议并将该属性设置为 NO。

可获得性

在 iOS3.0 及更高的版本中可获得。

1.1.2 配置无障碍元素

1.1.2.1 accessibilityActivationPoint 属性

在屏幕坐标中，无障碍元素的激活点。

声明

SWIFT

```
var accessibilityActivationPoint: CGPoint
```

OBJECTIVE-C

```
@property(nonatomic) CGPoint accessibilityActivationPoint
```

简介

该属性的默认值是由 `accessibilityFrame` 属性给定的无障碍元素框架的中点。一个元素的激活点是当用户双击该元素时，VoiceOver 激活的特定区域。

可以指定激活点的能力允许元素在不同情景下呈现给 VoiceOver 不同的点而不会改变元素自身的样式。例如，主屏幕 APP 图标标准激活点是该图标的中点。但是当重置主屏幕上的图标时，图标的激活点变为移除控件的中点（移除控件是图标左上角的圆形 X 号）。

使用该属性保证小元素的激活点保持准确无误，尽管开发者呈现给 VoiceOver 是元素的较大版本。

可获得性

在 iOS5.0 及更高的版本中可获得。

1.1.2.2 accessibilityElementsHidden 属性

一个布尔值，标识被包含在无障碍元素中的无障碍元素是否被隐藏。

声明

SWIFT

```
var accessibilityElementsHidden: Bool
```

OBJECTIVE-C

```
@property(nonatomic) BOOL accessibilityElementsHidden
```


简介

该属性的默认值为 **NO**。可以使用该属性隐藏被新出现视图覆盖的视图。在该情景中，该隐藏视图在屏幕上仍保持可见，但是它们不能获得用户操作的焦点。

开发者可以使用该属性隐藏 VoiceOver 用户不需要注意的瞬变视图。例如，当用户调整设备音量的时候，VoiceOver 不需要去描述出现的半透明的视图，因为操作已经有足够的听觉反馈。

可获得性

在 iOS5.0 及更高的版本中可获得。

1.1.2.3 accessibilityFrame 属性

在屏幕坐标中，无障碍元素的框架。

声明

SWIFT

```
var accessibilityFrame: CGRect
```

OBJECTIVE-C

```
@property(nonatomic) CGRect accessibilityFrame
```

简介

该属性的默认值是 **CGRectZero**，除非接收者是个 **UIView** 对象或者 **UIView** 的子类，在此情景下，该属性的值为视图的框架。

无障碍元素呈现的对象不是 **UIView** 的子类，开发者必须为这些无障碍元素设置该属性，因为该对象的屏幕坐标未知。（开发者不一定要为呈现 **UIView** 子类的无障碍元素设置该属性，因为该对象的屏幕坐标已知。）

可获得性

在 iOS3.0 及更高的版本中可获得。

另请参见

- [accessibilityPath](#)
- [UIAccessibilityConvertFrameToScreenCoordinates](#)

1.1.2.4 accessibilityHint 属性

使用本地语言，在无障碍元素上执行操作的结果的简单描述。

声明

SWIFT

```
var accessibilityHint: String?
```

OBJECTIVE-C

```
@property(nonatomic, copy) NSString *accessibilityHint
```

简介

该属性的默认值为 nil，除非接收者是个 UIKit 控件，在该情景下，该属性的值是由系统根据控件类型提供的提示。

当无障碍标签不能清楚传达结果的时候，无障碍提示帮助用户理解在无障碍元素上执行操作将会发生什么。例如，如果在应用提供一个添加按钮，按钮的无障碍标签帮助用户理解点击该按钮在应用中添加值。如果，另一方面，应用允许用户在歌曲名称列表中点击标题播放歌曲，每一个列表行的无障碍标签不会告知用户该结果。为了帮助辅助应用给残障用户提供该信息，每个列表行的合适提示为“播放歌曲”。

遵循以下准则为一个无障碍元素创建提示：

- 以动词开头，使用非常简洁的短语定义操作的结果，例如“播放歌曲（Plays the song）”“购买项目（Purchases the item）”。

避免使用命令式的动词作为短语的开头，因为这会让 hint 提示听起来像个命令。例如，不要创建像“去播放歌曲（Play the song）”“去购买项目（Purchase the item）”类的提示。

- 不要在 hint 提示中重复操作方式。例如，不要创建像“点击来播放歌曲（Tap

to play the song) ”或“点击播放歌曲 (Tapping plays the song) ”类的提示。

- 不要在 hint 提示中重复控件或视图类型。例如，不要创建像“播放此行中的歌曲”“添加联系人姓名的按钮”类的提示。

可获得性

在 iOS3.0 及更高的版本中可获得。

1.1.2.5 accessibilityLabel 属性

一个简洁的标签，使用本地语言标识无障碍元素。

声明

SWIFT

```
var accessibilityLabel: String?
```

OBJECTIVE-C

```
@property(nonatomic, copy) NSString *accessibilityLabel
```

简介

该属性的默认值是 nil，除非接收者是个 UIKit 控件，在该情景下，该属性的值直接从控件标题 (title) 中获取。

注意：如果在一个 UISegmentedControl 中呈现 UIImage 对象，为每一个图像设置该属性保证该部分是正常可访问的。

如果实现一个自定义控件或视图，或在 UIKit 控件上展示一个自定义的图标，设置该属性保证无障碍元素有合适的标签。如果一个无障碍元素没有呈现一个可描述的标签，设置该属性来提供一个简短、本地化的标签，简洁标识该元素。例如，一个“播放音乐”的按钮可能会呈现一个图标告知视觉用户它的作用。为了做到可访问，但是，该按钮应该有一个无障碍 label“播放”或“播放音乐”，这样辅助应用可以将该信息提供给残障用户。注意，但是，label 应该不能包含控件类型（例如，按钮），因为该信息被包含在与无障碍元素相关联的特性 (trait) 中。

可获得性

在 iOS3.0 及更高的版本中可获得。

1.1.2.6 accessibilityLanguage 属性

用于朗读无障碍元素标签、值和提示的语言。

声明

SWIFT

```
var accessibilityLanguage: String?
```

OBJECTIVE-C

```
@property(nonatomic, strong) NSString *accessibilityLanguage
```

简介

该属性的默认值为 nil。如果未设置语言，使用用户当前的语言设置。

如果需要设置该属性，保证使用语言 ID 标签，该 ID 标签遵循 [BCP47 规范](#) 定义的格式。

可获得性

在 iOS4.0 及更高的版本中可获得。

1.1.2.7 accessibilityPath 属性

在屏幕坐标中，元素的路径。

声明

SWIFT

```
@NSCopying var accessibilityPath: UIBezierPath?
```

OBJECTIVE-C

```
@property(nonatomic, copy) UIBezierPath *accessibilityPath
```

简介

该属性的默认值是 `nil`。如果未设置路径，无障碍矩形边框被用来突显元素。

当为该属性指定一个值，辅助技术使用指定的路径对象（代替无障碍边框）突显元素。

可获得性

在 iOS7.0 及更高的版本中可获得。

1.1.2.8 accessibilityTraits 属性

无障碍特性的组合，这些特性能够最好地描述无障碍元素的特征。

声明

SWIFT

```
var accessibilityTraits: UIAccessibilityTraits
```

OBJECTIVE-C

```
@property(nonatomic) UIAccessibilityTraits accessibilityTraits
```

简介

该属性的默认值是 `UIAccessibilityTraitNone`，除非接收者是个 `UIKit` 控件，在该情景下，该属性的值是与控件相关联特性的标准设置。

如果实现一个自定义控件或视图，开发者需要去选择能够最好地描述该对象特征的所有无障碍特性，然后在它父类的特性（换句话说，使用 `super.accessibilityTraits`）中通过执行一个 `OR` 操作整合这些无障碍特性。特性的完全列表，详见 [Accessibility Traits](#)。

可获得性

在 iOS3.0 及更高的版本中可获得。

1.1.2.9 accessibilityValue 属性

无障碍元素的值，使用本地化语言。

简介

SWIFT

```
var accessibilityValue: String?
```

OBJECTIVE-C

```
@property(nonatomic, copy) NSString *accessibilityValue
```

简介

该属性的默认值为 nil，除非该接收者是个 UIKit 控件，在这种情况下，该属性呈现的是控件的值，与控件的标签不同。

当一个无障碍元素有一个静态的标签和一个动态的值，设置该属性的返回值。例如，一个呈现文本域的无障碍元素有标签“Message”，但该无障碍元素的值是当前文本域中的文本。

可获得性

在 iOS3.0 及更高的版本中可获得。

1.1.2.10 accessibilityViewIsModal 属性

一个布尔值，标识出 VoiceOver 是否应该忽略视图内的接收者的兄弟元素。

声明

SWIFT

```
var accessibilityViewIsModal: Bool
```

OBJECTIVE-C

```
@property(nonatomic) BOOL accessibilityViewIsModal
```

简介

该属性的默认值是 NO。当该属性的值为 YES 时，VoiceOver 忽略接收视图兄弟视图内的元素。

例如，在一个窗口中包含兄弟视图 A 和 B，在视图 B 上设置 accessibilityViewIsModal 为 YES，VoiceOver 会忽略视图 A 中的元素。另一方面，如果视图 B 包含一个子视图 C，且在视图 C 上设置 accessibilityViewIsModal 为 YES，VoiceOver 不会忽略视图 A 中的元素。

可获得性

在 iOS5.0 及更高的版本中可获得。

1.1.2.11 shouldGroupAccessibilityChildren 属性

一个布尔值，标识出 VoiceOver 是否应该将接收者的子元素组合在一起，不管它们在屏幕上的位置如何。

简介

SWIFT

```
var shouldGroupAccessibilityChildren: Bool
```

OBJECTIVE-C

```
@property(nonatomic) BOOL shouldGroupAccessibilityChildren
```

讨论

该属性的默认值为 NO。

例如，假设一个垂直展示项目的应用。正常情况下，VoiceOver 可以在水平行中导航这些项目。在垂直展示项目的父级视图中设置该属性的值为 YES，这会让 VoiceOver 遵守应用的分组并正确导航它们。

可获得性

在 iOS6.0 及更高的版本中可获得。

1.1.2.12 accessibilityNavigationStyle 属性

适用于对象及其元素的导航样式。

声明

SWIFT

```
var accessibilityNavigationStyle: UIAccessibilityNavigationStyle
```

OBJECTIVE-C

```
@property(nonatomic) UIAccessibilityNavigationStyle  
accessibilityNavigationStyle
```

简介

为了导航其元素，一些辅助技术让用户选择父视图或容器。该属性控制该行为是否适用于当前对象。开关控件（Switch Control）使用该技术，但是 VoiceOver 和其他辅助技术不会。

该属性的默认值为 `UIAccessibilityNavigationStyleAutomatic`。

可获得性

在 iOS8.0 及更高的版本中可获得。

1.2 数据类型

1.2.1 UIAccessibilityTraits

一个面具，包含无障碍特性的或（OR）组合，能够最好地描述一个无障碍元素的特征。

声明

SWIFT

```
typealias UIAccessibilityTraits = UInt64
```

OBJECTIVE-C

```
typedef uint64_t UIAccessibilityTraits;
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS3.0 及更高的版本中可获得。

1.2.2 UIAccessibilityZoomType

可以起作用的系统缩放类型。

声明

SWIFT

```
enum UIAccessibilityZoomType : Int {  
    case InsertionPoint  
}
```

OBJECTIVE-C

```
typedef enum {  
    UIAccessibilityZoomTypeInsertionPoint,  
} UIAccessibilityZoomType;
```

常量

- UIAccessibilityZoomTypeInsertionPoint

系统缩放类型是在文本插入点。

在 iOS5.0 及更高的版本中可获得。

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS5.0 及更高的版本中可获得。

1.2.3 UIAccessibilityNotifications

无障碍应用可以发送的通知。

声明

SWIFT

```
 typealias UIAccessibilityNotifications = UInt32
```

OBJECTIVE-C

```
 typedef uint32_t UIAccessibilityNotifications;
```

导入语句

OBJECTIVE-C

```
 @import UIKit;
```

SWIFT

```
 import UIKit
```

可获得性

在 iOS3.0 及更高的版本中可获得。

1.3 常量

1.3.1 无障碍特性（Accessibility Traits）

无障碍特性，用来告知辅助应用一个无障碍元素怎样行为或者应该怎样被对待。

声明

SWIFT

```
var UIAccessibilityTraitNone: UIAccessibilityTraits
```

```
var UIAccessibilityTraitButton: UIAccessibilityTraits
```

```
var UIAccessibilityTraitLink: UIAccessibilityTraits
```

```
var UIAccessibilityTraitSearchField: UIAccessibilityTraits
```

```
var UIAccessibilityTraitImage: UIAccessibilityTraits
```

```
var UIAccessibilityTraitSelected: UIAccessibilityTraits
```

```
var UIAccessibilityTraitPlaysSound: UIAccessibilityTraits
```

```
var UIAccessibilityTraitKeyboardKey: UIAccessibilityTraits
```

```
var UIAccessibilityTraitStaticText: UIAccessibilityTraits
```

```
var UIAccessibilityTraitSummaryElement: UIAccessibilityTraits
```

```
var UIAccessibilityTraitNotEnabled: UIAccessibilityTraits
```

```
var UIAccessibilityTraitUpdatesFrequently: UIAccessibilityTraits
```

```
var UIAccessibilityTraitStartsMediaSession: UIAccessibilityTraits
```

```
var UIAccessibilityTraitAdjustable: UIAccessibilityTraits
```

```
var UIAccessibilityTraitAllowsDirectInteraction: UIAccessibilityTraits
```

```
var UIAccessibilityTraitCausesPageTurn: UIAccessibilityTraits
```

var UIAccessibilityTraitHeader: [UIAccessibilityTraits](#)

OBJECTIVE-C

UIAccessibilityTraits UIAccessibilityTraitNone;

UIAccessibilityTraits UIAccessibilityTraitButton;

UIAccessibilityTraits UIAccessibilityTraitLink;

UIAccessibilityTraits UIAccessibilityTraitSearchField;

UIAccessibilityTraits UIAccessibilityTraitImage;

UIAccessibilityTraits UIAccessibilityTraitSelected;

UIAccessibilityTraits UIAccessibilityTraitPlaysSound;

UIAccessibilityTraits UIAccessibilityTraitKeyboardKey;

UIAccessibilityTraits UIAccessibilityTraitStaticText;

UIAccessibilityTraits UIAccessibilityTraitSummaryElement;

UIAccessibilityTraits UIAccessibilityTraitNotEnabled;

UIAccessibilityTraits UIAccessibilityTraitUpdatesFrequently;

UIAccessibilityTraits UIAccessibilityTraitStartsMediaSession;

UIAccessibilityTraits UIAccessibilityTraitAdjustable;

UIAccessibilityTraits UIAccessibilityTraitAllowsDirectInteraction;

UIAccessibilityTraits UIAccessibilityTraitCausesPageTurn;

UIAccessibilityTraits UIAccessibilityTraitHeader;

常量

- UIAccessibilityTraitNone

该无障碍元素无特性。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitButton**

该无障碍元素应该被当作一个按钮。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitLink**

该无障碍元素应该被当作一个链接。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitSearchField**

该无障碍元素应该被当作一个搜索框。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitImage**

该无障碍元素应该被当作一个图像。

该特性可以与按钮或链接特性组合使用。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitSelected**

该无障碍元素当前已被选定。

使用该特性描述一个无障碍元素的特征，例如，在一个分段控件中，呈现已选定表行或已选定段。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitPlaysSound**

该无障碍元素被激活时播放自己的声音。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitKeyboardKey**

该无障碍元素的行为像键盘键。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitStaticText**

该无障碍元素应该被当作不能改变的静态文本。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitSummaryElement**

当应用启动时，该无障碍元素提供摘要信息。

使用该属性描述一个无障碍元素的特征，该元素提供当前情况、设置或状态的摘要，例如天气应用中的当前气温。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitNotEnabled**

该无障碍元素不可用且不能响应用户交互。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitUpdatesFrequently**

该无障碍元素频繁更新其标签或值。

使用该特性描述一个无障碍元素的特征，该元素频繁更新其标签或值发送更新通知。当想要避免辅助应用处理不断的通知时，和当需要更新信息而轮询改变时，包含该特性。例如，开发者可以使用该属性描述秒表朗读的特征。

在 iOS3.0 及更高的版本中可获得。

- **UIAccessibilityTraitStartsMediaSession**

当该无障碍元素被激活时，启动一个媒体对话。

使用该特性关闭辅助应用的音频输出，例如 VoiceOver。在媒体会话期间，媒体会话不应该被打断。例如，当用户录制音频的时候，开发者可以使用该属性静音 VoiceOver。

在 iOS4.0 及更高的版本中可获得。

- **UIAccessibilityTraitAdjustable**

该无障碍元素的值在一定范围内允许持续调整。

使用该属性描述一个无障碍元素的特征，用户可以使用持续的行为调整该元素，例如滑块或选择器视图。如果为一个无障碍元素指定了该特性，开发者同时应该在 `UIAccessibilityAction` 协议中实现 `accessibilityIncrement` 和 `accessibilityDecrement` 方法。

在 iOS4.0 及更高的版本中可获得。

- **UIAccessibilityTraitAllowsDirectInteraction**

该无障碍元素允许 VoiceOver 用户直接触摸交互。

使用该特性描述一个无障碍元素的特征，该元素呈现的对象用户可以直接交互，例如呈现钢琴键盘的视图。

在 iOS5.0 及更高的版本中可获得。

- **UIAccessibilityTraitCausesPageTurn**

该无障碍元素应该引起自动翻页，当 VoiceOver 阅读完成当页的文本时。

使用该特性描述一个无障碍元素的特征，该元素在一系列页面内容中呈现一个页面，例如呈现一本书中的一页。当 VoiceOver 阅读完成当前页面的内容，调用具有 `UIAccessibilityScrollDirectionNext` 属性的 `accessibilityScroll` 方法滚动到下一内容页。如果 VoiceOver 检测到当前页面与先前的页面无差异，停止滚动。

在 iOS5.0 及更高的版本中可获得。

- **UIAccessibilityTraitHeader**

该无障碍元素是一个将内容划分为章节的标题，例如导航栏的标题。

在 iOS6.0 及更高的版本中可获得。

1.3.2 通知字典关键值（Notification Dictionary Keys）

被用在通知的 `userInfo` 参数字典的关键值。

声明

SWIFT

```
let UIAccessibilityAnnouncementKeyStringValue: String
```

```
let UIAccessibilityAnnouncementKeyWasSuccessful: String
```

OBJECTIVE-C

```
NSString *const UIAccessibilityAnnouncementKeyStringValue;
```

```
NSString *const UIAccessibilityAnnouncementKeyWasSuccessful;
```

常量

- UIAccessibilityAnnouncementKeyStringValue

已完成通知的文本。

在 iOS6.0 及更高的版本中可获得。

- UIAccessibilityAnnouncementKeyWasSuccessful

标识通知是否发送成功。

该关键值是个被解释为布尔值的 `NSNumber` 对象。

在 iOS6.0 及更高的版本中可获得。

1.3.3 属性字符串的语音属性 (Speech Attributes for Attributed Strings)

开发者可以将该属性应用到到属性字符串文本, 用来修改文本是怎样发声的。

声明

SWIFT

```
let UIAccessibilitySpeechAttributePunctuation: String
```

```
let UIAccessibilitySpeechAttributeLanguage: String
```

```
let UIAccessibilitySpeechAttributePitch: String
```

OBJECTIVE-C

```
NSString *const UIAccessibilitySpeechAttributePunctuation;
```

```
NSString *const UIAccessibilitySpeechAttributeLanguage;
```

```
NSString *const UIAccessibilitySpeechAttributePitch;
```

常量

- UIAccessibilitySpeechAttributePunctuation

该关键值是个被解释为布尔值的 **NSNumber** 对象。当该值为 YES 时，文本中所有标点符号都会被朗读出来。开发者可以在代码区，或其他与标点有关的地方，使用该属性。

在 iOS7.0 及更高的版本中可获得。

- UIAccessibilitySpeechAttributeLanguage

该关键值是一个包含 BCP 47 语言编码的 **NSString** 对象。当将该值应用于字符串文本，特定语言的规则会控制字符串是怎样被发音的。

在 iOS7.0 及更高的版本中可获得。

- UIAccessibilitySpeechAttributePitch

该关键值是一个包含范围 0.0 到 2.0 的浮点型数值的 **NSNumber** 对象。该值标明文本是否应该使用较高或较低的音调读出，而不是使用默认音调。该属性的默认值是 1.0，代表一个正常音调。值在 0.0 到 1.0 之间时使用较低的音调，值在 1.0 到 2.0 之间时使用较高的音调。

在 iOS7.0 及更高的版本中可获得。

1.3.4 辅助技术标识（Assistive Technology Identifiers）

当暂停或重启辅助技术时，可以使用该标识。

声明

SWIFT

```
let UIAccessibilityNotificationSwitchControlIdentifier: String
```

OBJECTIVE-C

```
NSString *const UIAccessibilityNotificationSwitchControlIdentifier;
```

常量

- UIAccessibilityNotificationSwitchControlIdentifier

开关控制技术。

该技术允许运动障碍用户使用独立物理按钮访问应用。当启用该技术，iOS 光标在屏幕上元素间循环。用户点击开关可以操作光标下的元素。

在 iOS8.0 及更高的版本中可获得。

1.3.5 UIAccessibilityNavigationStyle

描述一个对象元素应该怎样被辅助技术导航的常量。

声明

SWIFT

```
enum UIAccessibilityNavigationStyle : Int {  
  
    case Automatic  
  
    case Separate  
  
    case Combined  
  
}
```

OBJECTIVE-C

```
typedef enum UIAccessibilityNavigationStyle : NSInteger {  
  
    UIAccessibilityNavigationStyleAutomatic = 0,  
  
    UIAccessibilityNavigationStyleSeparate = 1,  
  
    UIAccessibilityNavigationStyleCombined = 2,  
  
} UIAccessibilityNavigationStyle;
```

常量

- **UIAccessibilityNavigationStyleAutomatic**

辅助技术自动判定接收者的元素应该怎样被导航。

此为默认值。

在 iOS8.0 及更高的版本中可获得。

- **UIAccessibilityNavigationStyleSeparate**

接收者的元素应该作为独立项目被导航。

在 iOS8.0 及更高的版本中可获得。

- **UIAccessibilityNavigationStyleCombined**

接收者的元素应该被组合且被作为一个独立项目进行导航。

在 iOS8.0 及更高的版本中可获得。

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4 通知

1.4.1 UIAccessibilityAnnouncementNotification

当通知需要被传递到辅助技术时，由应用发送的通知。

该通知包含一个参数，该参数是个包含通知的 `NSString` 对象。一个辅助技术输出的通知文本被包含在该参数中。

使用该通知提供关于事件的无障碍信息，该事件不会更新应用用户界面或只简单更新用户界面。

使用 `UIAccessibilityPostNotification` 发送该通知。

声明

SWIFT

```
var UIAccessibilityAnnouncementNotification: UIAccessibilityNotifications
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS4.0 及更高的版本中可获得。

1.4.2 UIAccessibilityAnnouncementDidFinishNotification

当系统阅读完成一个通知的时候，由 UIKit 发送的通知。

该参数是个具有两个关键值的字典，参数为 `UIAccessibilityAnnouncementKeyStringValue` 和 `UIAccessibilityAnnouncementKeyWasSuccessful`。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityAnnouncementDidFinishNotification: String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS6.0 及更高的版本中可获得。

1.4.3 UIAccessibilityBoldTextStatusDidChangeNotification

当系统粗体文本设置改变时，由 UIKit 发送的通知。

该通知不包含参数，使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityBoldTextStatusDidChangeNotification: String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.4

UIAccessibilityClosedCaptioningStatusDidChangeNotification

当隐藏式字幕设置改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityClosedCaptioningStatusDidChangeNotification: String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS5.0 及更高的版本中可获得。

1.4.5

UIAccessibilityDarkerSystemColorsStatusDidChangeNotificatio

n

当系统的加深颜色设置改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityDarkerSystemColorsStatusDidChangeNotification:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```


可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.6 UIAccessibilityGrayscaleStatusDidChangeNotification

当系统的灰度设置改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityGrayscaleStatusDidChangeNotification:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.7 UIAccessibilityGuidedAccessStatusDidChangeNotification

当系统的引导式访问设置改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

使用 `UIAccessibilityIsGuidedAccessEnabled` 函数判定当前引导式访问是否可用。

声明

SWIFT

```
let UIAccessibilityGuidedAccessStatusDidChangeNotification:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS6.0 及更高的版本中可获得。

1.4.8 UIAccessibilityInvertColorsStatusDidChangeNotification

当反转颜色设置被改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

使用 [UIAccessibilityIsInvertColorsEnabled](#) 函数判定当前是不是反转色。

声明

SWIFT

```
let UIAccessibilityInvertColorsStatusDidChangeNotification:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS6.0 及更高的版本中可获得。

1.4.9 UIAccessibilityLayoutChangedNotification

当屏幕布局改变时，例如当一个元素出现或消失时，由应用发送的通知。

该通知包含一个参数，该参数是一个 VoiceOver 朗读的 NSString 对象，或者是一个 VoiceOver 将要移至的无障碍元素。使用 [UIAccessibilityPostNotification](#) 函数发送该通知。

声明

SWIFT

```
var UIAccessibilityLayoutChangedNotification: UIAccessibilityNotifications
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS3.0 及更高的版本中可获得。

1.4.10 UIAccessibilityMonoAudioStatusDidChangeNotification

当系统音频从立体声变为单声道时，由 UIKit 发送的通知。

该通知不包含参数，使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityMonoAudioStatusDidChangeNotification: String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS5.0 及更高的版本中可获得。

1.4.11 UIAccessibilityPageScrolledNotification

当滚动操作已经完成并且 `accessibilityScroll:` 方法被调用时，由应用发送的通知。

该通知包含一个参数，该参数是一个包含新滚动位置描述的 NSString 对象。辅助技术输出的是被包含在该参数中的描述字符串。

在用户施行 VoiceOver 滚动手势之后，使用该通知提供屏幕内容的自定义信息。例如，一个基于 tab 的应用可能提供一个字符串，例如“Tab 3 of 5”，或一个在多个页面中展示信息的应用可能提供一个字符串，例如“Page 19 of 27”。

当重复接收相同滚动位置字符串时，辅助技术告知用户已经到达边框或边界。

使用 `UIAccessibilityPostNotification` 函数发送该通知。

声明

SWIFT

```
var UIAccessibilityPageScrolledNotification: UIAccessibilityNotifications
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS4.2 及更高的版本中可获得。

1.4.12 UIAccessibilityPauseAssistiveTechnologyNotification

当想要暂停辅助技术操作时，发送该通知。

发送该通知时，使用参数指定要暂停的辅助技术。例如，当 APP 在播放动画时，可能要在开关控件中暂停扫描。开发者必须发送一个 `UIAccessibilityResumeAssistiveTechnologyNotification` 通知恢复辅助技术的操作，平衡该通知。使用 `UIAccessibilityPostNotification` 函数发送该通知。

声明

SWIFT

```
var UIAccessibilityPauseAssistiveTechnologyNotification:  
UIAccessibilityNotifications
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.13

UIAccessibilityReduceMotionStatusDidChangeNotification

当系统减少动态效果设置改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityReduceMotionStatusDidChangeNotification: String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.14

UIAccessibilityReduceTransparencyStatusDidChangeNotification

n

当系统的降低透明度设置发生改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

```
SWIFT
```

```
let UIAccessibilityReduceTransparencyStatusDidChangeNotification: String
```

导入语句

```
OBJECTIVE-C
```

```
@import UIKit;
```

```
SWIFT
```

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.15 UIAccessibilityResumeAssistiveTechnologyNotification

发送该通知暂时恢复辅助技术的操作。

发送该通知时，使用参数指定要恢复的辅助技术。开发者必须发送该通知平衡先前发送的 [UIAccessibilityPauseAssistiveTechnologyNotification](#) 通知。使用 [UIAccessibilityPostNotification](#) 函数发送该通知。

声明

SWIFT

```
var UIAccessibilityResumeAssistiveTechnologyNotification :  
UIAccessibilityNotifications
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.16 UIAccessibilityScreenChangedNotification

当一个新的视图出现，并覆盖屏幕的主要部分时，由应用发送的通知。

该通知包含一个参数，该参数是一个 VoiceOver 可以读出的 NSString 对象，或 VoiceOver 将要移动至的一个无障碍元素。使用 [UIAccessibilityPostNotification](#) 函数发送该通知。

声明

SWIFT

```
var UIAccessibilityScreenChangedNotification: UIAccessibilityNotifications
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```



```
SWIFT
```

```
import UIKit
```

可获得性

在 iOS3.0 及更高的版本中可获得。

1.4.17 UIAccessibilitySpeakScreenStatusDidChangeNotification

当系统朗读屏幕设置发生改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

```
SWIFT
```

```
let UIAccessibilitySpeakScreenStatusDidChangeNotification: String
```

导入语句

```
OBJECTIVE-C
```

```
@import UIKit;
```

```
SWIFT
```

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.18

UIAccessibilitySpeakSelectionStatusDidChangeNotification

当系统朗读所选项设置发生改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilitySpeakSelectionStatusDidChangeNotification:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.19

UIAccessibilitySwitchControlStatusDidChangeNotification

当系统切换控制设置发生改变时，由 UIKit 发送的通知。

该通知不包含参数。使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilitySwitchControlStatusDidChangeNotification:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS8.0 及更高的版本中可获得。

1.4.20 UIAccessibilityVoiceOverStatusChanged

当 VoiceOver 启动或停止时，由 UIKit 发送的通知。该通知不包含参数。

使用该通知为 VoiceOver 用户自定义应用用户界面。例如，如果要展示一个 UI 元素，且该元素简单覆盖了用户界面的其他部分，为 VoiceOver 用户要提供一致性的呈现，但对于非 VoiceOver 用户，允许它消失。开发者可以使用 [UIAccessibilityIsVoiceOverRunning](#) 函数判定当前 VoiceOver 是否在运行。

使用默认通知中心观察该通知。

声明

SWIFT

```
let UIAccessibilityVoiceOverStatusChanged:String
```

导入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS4.0 及更高的版本中可获得。

2. UIAccessibilityAction

继承自: Not Applicable;

遵循: Not Applicable;

导入语句:

```
OBJECTIVE-C
```

```
@import UIKit;
```

可获得性: 在 iOS4.0 及更高版本中可获得;

UIAccessibilityAction 非正式协议为无障碍元素提供一种方式，以支持特定操作，例如在范围内选择值，或在屏幕上滚动信息。例如，为了响应滚动手势，开发者需要实现 `accessibilityScroll:` 方法并发送具有新页面状态的 `UIAccessibilityPageScrolledNotification`（例如“第三页共九页（Page 3 of 9）”）。或者，为了实现一个元素可访问，例如滑动条或选择器视图，首先需要包含 `UIAccessibilityTraitAdjustable` 特性来描述该元素的特征。然后，开发者必须实现 `accessibilityIncrement` 和 `accessibilityDecrement` 方法。当做完这些，辅助技术用户可以使用辅助技术特殊手势调整该元素。

2.1 执行操作

2.1.1 – accessibilityActivate

告知元素激活自己，并报告操作成功或失败。

声明

SWIFT

```
func accessibilityActivate() -> Bool
```

OBJECTIVE-C

```
-(BOOL)accessibilityActivate
```

返回值

如果元素被激活，返回 YES，如果未被激活，返回 NO。

简介

开发者可以使用该方法让用户更容易的访问复杂控件。当 VoiceOver 用户双击选定元素时，无障碍系统调用该方法。实现该方法应该可以激活元素并执行其认为合理的其他任务。例如，开发者可能使用该方法激活一个控件，该控件激活需要一个复杂手势，且 VoiceOver 用户很难完成，可能是因为当 VoiceOver 运行时，该手势有不一样的含义。

执行任务之后，返回一个适当的布尔值标识操作成功或失败。

可获得性

在 iOS7.0 及更高的版本中可获得。

2.1.2 – accessibilityIncrement

告知无障碍元素增加其内容的值。

声明

SWIFT

```
func accessibilityIncrement()
```

OBJECTIVE-C

```
-(void)accessibilityIncrement
```

简介

如果元素具有 `UIAccessibilityTraitAdjustable` 特性，开发者必须实现该方法。使用该方法增加元素的值。例如，一个 `UISlider` 对象使用该方法以合适的数量增加该元素的值。

可获得性

在 iOS4.0 及更高的版本中可获得。

2.1.3 – accessibilityDecrement

告知无障碍元素减少其内容的值。

声明

SWIFT

```
func accessibilityDecrement()
```

OBJECTIVE-C

```
-(void)accessibilityDecrement
```

简介

如果元素具有 `UIAccessibilityTraitAdjustable` 特性，开发者必须实现该方法。使用该方法减少元素的值。例如，一个 `UISlider` 对象使用该方法以合适的数量减少该元素的值。

可获得性

在 iOS4.0 及更高的版本中可获得。

2.1.4 – accessibilityScroll:

以一种应用指定的方式滚动屏幕内容，并返回操作成功或失败。

声明

SWIFT

```
func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool
```

OBJECTIVE-C

```
-(BOOL)accessibilityScroll:(UIAccessibilityScrollDirection)direction
```

参数

direction : 指定滚动操作方向的常量。有效常量的描述，详见 [UIAccessibilityScrollDirection](#)。

返回值

如果滚动操作成功，返回 YES；否则，返回 NO。该方法默认返回 NO。

简介

如果一个在视图层次中的视图支持页面滚动操作，需要实现该方法。

- 如果特定方向的滚动操作成功，返回 YES 并发送 [UIAccessibilityPageScrolledNotification](#) 通知。
- 如果滚动操作失败，在视图层次中由父视图调用 `accessibilityScroll:`。

可获得性

在 iOS4.2 及更高的版本中可获得。

2.1.5 – accessibilityPerformEscape

关闭一个模态视图，并返回该操作的成功或失败。

声明

SWIFT

```
func accessibilityPerformEscape() -> Bool
```

OBJECTIVE-C

```
-(BOOL)accessibilityPerformEscape
```

返回值

如果模态视图被成功关闭，返回 YES；否则，返回 NO。该方法默认返回 NO。

简介

在一个元素上、或在可以模态出现或视图层次中的包含视图上，实现该方法。当 VoiceOver 用户执行关闭操作，该方法关闭视图。例如，为了给用户一个从容的关闭操作来执行关闭浮窗，开发者可能需要为该浮窗实现该方法。

可获得性

在 iOS5.0 及更高的版本中可获得。

2.1.6 – accessibilityPerformMagicTap

执行一个突出的操作。

声明

SWIFT

```
func accessibilityPerformMagicTap() -> Bool
```

OBJECTIVE-C

```
-(BOOL)accessibilityPerformMagicTap
```

返回值

如果魔法轻拍操作成功，返回 YES；否则，返回 NO。该方法默认返回 NO。

简介

该方法执行的精确操作来自应用，一般会弹出应用最重要的状态。例如，在电话应用中，魔法轻拍执行应答和结束通话；在音乐应用中，魔法轻拍执行播放和暂停音乐；在计时应用中，魔法轻拍执行启动和停止计时器；在相机应用中，执行拍照操作。

可获得性

在 iOS6.0 及更高的版本中可获得。

2.2 访问自定义操作

2.2.1 accessibilityCustomActions 属性

与内置操作一起呈现的自定义操作数组。

声明

SWIFT

```
var accessibilityCustomActions: [UIAccessibilityCustomAction]?
```

OBJECTIVE-C

```
@property(nonatomic, strong) NSArray <UIAccessibilityCustomAction *>  
*accessibilityCustomActions
```

简介

该数组包含一个或多个定义已支持操作的 `UIAccessibilityCustomAction` 对象。辅助技术，例如 VoiceOver，会在合适的时间呈现自定义操作。

可获得性

在 iOS8.0 及更高的版本中可获得。

2.3 数据类型

2.3.1 UIAccessibilityScrollDirection

一个滚动操作的方向。

声明

SWIFT

```
enum UIAccessibilityScrollDirection : Int {  
  
    case Right  
  
    case Left  
  
    case Up  
  
    case Down  
  
    case Next  
  
    case Previous  
  
}
```

OBJECTIVE-C

```
typedef enum {  
  
    UIAccessibilityScrollDirectionRight = 1,  
  
    UIAccessibilityScrollDirectionLeft,  
  
    UIAccessibilityScrollDirectionUp,  
  
    UIAccessibilityScrollDirectionDown,  
  
    UIAccessibilityScrollDirectionNext,  
  
    UIAccessibilityScrollDirectionPrevious  
  
} UIAccessibilityScrollDirection;
```

常量

- **UIAccessibilityScrollDirectionRight**
用户执行向右的滚动操作。
在 iOS4.2 及更高的版本中可获得。
- **UIAccessibilityScrollDirectionLeft**
用户执行向左的滚动操作。
在 iOS4.2 及更高的版本中可获得。
- **UIAccessibilityScrollDirectionUp**
用户执行向上的滚动操作。
在 iOS4.2 及更高的版本中可获得。
- **UIAccessibilityScrollDirectionDown**
用户执行向下的滚动操作。
在 iOS4.2 及更高的版本中可获得。
- **UIAccessibilityScrollDirectionNext**
用户以有序视图集的顺序执行滚动到下一个视图的操作。
在 iOS5.0 及更高的版本中可获得。
- **UIAccessibilityScrollDirectionPrevious**
用户以有序视图集的顺序执行滚动到上一个视图的操作。
在 iOS5.0 及更高的版本中可获得。

引入语句

OBJECTIVE-C

```
@import UIKit;
```

SWIFT

```
import UIKit
```

可获得性

在 iOS4.2 及更高的版本中可获得。

3. UIAccessibilityContainer

继承自: Not Applicable;

遵循: Not Applicable;

导入语句:

```
OBJECTIVE-C
```

```
@import UIKit;
```

可获得性: 在 iOS3.0 及更高版本中可获得;

UIAccessibilityContainer 非正式协议为 UIView 子类提供一种方式让可选择组件像独立元素一样可访问。例如, 一个视图可能包含多个图标或者图像文本, 对终端用户来说, 这些图标或图像文本是作为独立元素出现和起作用的。但是因为这些组件未实现 UIView 实例, 它们不会自动对残障用户无障碍。因此, 这样的一个容器视图, 应该实现 UIAccessibilityContainer 方法来向辅助应用提供这些组件的无障碍信息, 例如 VoiceOver。

实现 UIAccessibilityContainer 非正式协议的视图使用 UIAccessibilityElement 方法 `initWithAccessibilityContainer:` 创建一个无障碍元素, 该无障碍元素用来呈现需要对残障用户可访问的每个无视图组件。注意, 但是, 容器视图本身不是个无障碍元素, 因为用户是与容器中的内容进行交互, 不是容器本身。这就意味着一个实现 UIAccessibilityContainer 方法的容器视图, 必须将 UIAccessibility 非正式协议的 `isAccessibilityElement` 属性设置为 NO。

容器视图内无障碍元素的顺序应该与其呈现给用户的顺序相同, 从左到右, 从上到下。

3.1 提供关于无障碍元素的信息

3.1.1 - accessibilityElementCount

返回容器内无障碍元素的数目。

声明

SWIFT

```
func accessibilityElementCount() -> Int
```

OBJECTIVE-C

```
- (NSInteger)accessibilityElementCount
```

返回值

容器内无障碍元素的数目。该方法默认返回 0。

可获得性

在 iOS3.0 及更高的版本中可获得。

3.1.2 - accessibilityElementAtIndex:

返回指定索引的无障碍元素。

声明

SWIFT

```
func accessibilityElementAtIndex(_ index: Int) -> AnyObject?
```

OBJECTIVE-C

```
- (id)accessibilityElementAtIndex:(NSInteger)index
```

参数

index : 无障碍元素的索引。

返回值

返回指定索引的无障碍元素，或，如果指定索引下不存在无障碍元素，返回 nil。

可获得性

在 iOS3.0 及更高的版本中可获得。

参见

- [indexOfAccessibilityElement:](#)

3.1.3 - `indexOfAccessibilityElement:`

返回指定无障碍元素的索引。

声明

SWIFT

```
func indexOfAccessibilityElement(_ element: AnyObject) -> Int
```

OBJECTIVE-C

```
- (NSInteger)indexOfAccessibilityElement:(id)element
```

参数

element : 无障碍元素。

返回值

指定无障碍元素的索引，或，如果该元素不存在，返回 NSNotFound。

可获得性

在 iOS3.0 及更高的版本中可获得。

参见

- [accessibilityElementAtIndex:](#)

3.1.4 accessibilityElements 属性

容器内无障碍元素的数组。

声明

SWIFT

```
var accessibilityElements: [AnyObject]?
```

OBJECTIVE-C

```
@property(nonatomic, strong) NSArray *accessibilityElements
```

简介

容器可以实现该属性代替动态方法来支持被包含元素的检索。该属性的默认值为 nil。

可获得性

在 iOS8.0 及更高的版本可获得。

4. UIAccessibilityFocus

继承自: Not Applicable;

遵循: Not Applicable;

导入语句:

```
OBJECTIVE-C
```

```
@import UIKit;
```

可获得性: 在 iOS4.0 及更高版本中可获得;

UIAccessibilityFocus 非正式协议提供了一个方法查明辅助技术, 例如 VoiceOver, 是否可以聚焦在一个无障碍元素上。

VoiceOver 和其他辅助技术在元素上放置一个视觉焦点, 这允许用户查看元素而不激活元素。如果已知视觉焦点的当前位置, 开发者可以优化辅助技术用户的用户体验。例如, 如果应用想要用户点击一次来选择对象, 然后双击激活对象, VoiceOver 用户在点击选择之前, 必须要做额外的点击来将 VoiceOver 聚焦到对象上。为了提升 VoiceOver 的用户体验, 开发者可以在选择元素的同时将 VoiceOver 聚焦到元素上。使用这种方法, 用户可以激活元素, 而不用再次点击选择元素。

4.1 获得焦点信息

4.1.1 – accessibilityElementDidBecomeFocused

当辅助技术将视觉焦点设置在无障碍元素上后，发送。

声明

SWIFT

```
func accessibilityElementDidBecomeFocused()
```

OBJECTIVE-C

```
-(void)accessibilityElementDidBecomeFocused
```

简介

如果需要知道辅助技术什么时候在无障碍元素上设置了视觉焦点，重写 `accessibilityElementDidBecomeFocused`。

可获得性

在 iOS4.0 及更高的版本中可获得。

4.1.2 – accessibilityElementDidLoseFocus

当辅助技术从一个无障碍元素移开其视觉焦点后，发送。

声明

SWIFT

```
func accessibilityElementDidLoseFocus()
```

OBJECTIVE-C

```
-(void)accessibilityElementDidLoseFocus
```

讨论

如果需要知道辅助技术什么时候从无障碍元素移开其视觉焦点，重写 `accessibilityElementDidLoseFocus` 。 注 意 ， 在 `accessibilityElementDidBecomeFocused` 前，发送 `accessibilityElementDidLoseFocus`。

可获得性

在 iOS4.0 及更高的版本中可获得。

4.1.3 – `accessibilityElementIsFocused`

返回一个布尔值，标识辅助技术是否已聚焦在无障碍元素上。

声明

SWIFT

```
func accessibilityElementIsFocused() -> Bool
```

OBJECTIVE-C

```
-(BOOL)accessibilityElementIsFocused
```

返回值

如果辅助技术真的已聚焦在元素上，返回 YES，否则，返回 NO。

可获得性

在 iOS4.0 及更高的版本中可获得。

5. UIAccessibilityIdentification

继承自: [NSObject](#);

遵循: Not Applicable;

导入语句:

```
OBJECTIVE-C
```

```
@import UIKit;
```

可获得性: 在 iOS5.0 及更高版本中可获得;

UIAccessibilityIdentification 协议被用来在用户界面中关联元素的唯一标识符。开发者可以在 UI 自动化脚本中使用定义的标识符, 因为 `accessibilityIdentifier` 的值对应的是 [UIAElement name](#) 方法的返回值。

5.1 访问一个元素的标识符

5.1.1 accessibilityIdentifier 必填属性

标识元素的字符串。

声明

SWIFT

```
var accessibilityIdentifier: String? { get set }
```

OBJECTIVE-C

```
@property(nonatomic, copy) NSString *accessibilityIdentifier
```

简介

一个标识符可以被用来在开发者编写的脚本中唯一标识元素，该脚本被用在 UI 自动化界面中。使用标识符允许开发者避免不适当的设置，或访问一个元素的无障碍标签。

可访问性

在 iOS5.0 及更高的版本中可获得。

6. UIAccessibilityReadingContent

继承自: Not Applicable;

遵循: Not Applicable;

导入语句:

```
OBJECTIVE-C
```

```
@import UIKit;
```

可获得性: 在 iOS5.0 及更高版本中可获得;

UIAccessibilityReadingContent 协议可以被实施在呈现想要用户阅读内容的对象, 例如一本书或一篇文章。为了给 VoiceOver 用户一个卓越、连续的阅读体验, 开发者可以在这样的元素上实现该协议, 使用 UIAccessibilityTraitCausesPageTurn 特性描述其特征, 并使用 UIAccessibilityScrollDirectionNext 和 UIAccessibilityScrollDirectionPrevious 常量实现翻页。

6.1 访问页面上的内容

6.1.1 - `accessibilityLineNumberForPoint`: 必填

返回包含指定点的行数。

声明

SWIFT

```
func accessibilityLineNumberForPoint(_ point: CGPoint) -> Int
```

OBJECTIVE-C

```
-(NSInteger)accessibilityLineNumberForPoint:(CGPoint)point
```

参数

`point` : 在屏幕坐标中，接收者视图空间边界中的点。也就是，`[self pointInside:point withEvent:event] == YES` 的点。

返回值

包含指定点的行数，或，如果点指向接收者矩形中的空白区域，返回 `NSNotFound`。该方法默认返回 `NSNotFound`。

简介

当 `point` 位于视图或元素边界内时，才调用该方法。

可获得性

在 iOS5.0 及更高的版本中可获得。

参见

[UIAccessibilityConvertFrameToScreenCoordinates](#)

6.1.2 - accessibilityContentForLineNumber:必填

返回与指定行数相关联的文本。

声明

SWIFT

```
func accessibilityContentForLineNumber(_ lineNumber: Int) -> String?
```

OBJECTIVE-C

```
-(NSString *)accessibilityContentForLineNumber:(NSInteger)lineNumber
```

参数

lineNumber : 接收者内容中的行数。

返回值

包含指定行数相关联文本的字符串，或，如果行数无效，返回 nil。该函数默认返回 nil。

可获得性

在 iOS5.0 及更高的版本中获得。

6.1.3 - accessibilityFrameForLineNumber:必填

返回与指定行数相关联的屏幕上框架。

声明

SWIFT

```
func accessibilityFrameForLineNumber(_ lineNumber: Int) -> CGRect
```

OBJECTIVE-C

```
-(CGRect)accessibilityFrameForLineNumber:(NSInteger)lineNumber
```

参数

lineNumber : 行数。

返回值

在屏幕坐标中,包含指定行数的接收者的框架。该方法默认返回 `CGRectZero`。

简介

为了判定一行的屏幕上矩形 (或框架), 开发者可以使用如下的代码:

OBJECTIVE-C

```
1   CGRect lineBounds = // 在视图空间中线的边界。  
2   UIView *view = // 相关联视图。  
3   return UIAccessibilityConvertFrameToScreenCoordinates(lineBounds, view);
```

SWIFT

```
1   let lineBounds: CGRect = //在视图空间中线的边界。  
2   let view: UIView = //相关联视图。  
3   return UIAccessibilityConvertFrameToScreenCoordinates(lineBounds, view)
```

可获得性

在 iOS5.0 及更高的版本中可获得。

参见

[UIAccessibilityConvertFrameToScreenCoordinates](#)

6.1.4 - accessibilityPageContent 必填

返回当前页面中展示的文本。

声明

SWIFT

```
func accessibilityPageContent() -> String?
```

OBJECTIVE-C

```
-(NSString *)accessibilityPageContent
```

返回值

包含当前页面所展示内容的字符串。

可获得性

在 iOS5.0 及更高的版本中可获得。