# Accessibility for OS X

## Beautiful apps usable by everyone!

Session 203

Dr. Gregory Hughes

These are confidential sessions—please refrain from streaming, blogging, or taking pictures

## Focus Areas

## Focus Areas

Technologies

## Focus Areas

Technologies

## Focus Areas

Technologies

API

## Focus Areas

Technologies

API

## Focus Areas

Technologies

API

Custom UI

## Focus Areas

Technologies

API

Custom UI

## Technologies

LED Flash for Alerts

Mouse Keys

Contrast Enhancements

Speak Selection

Hearing-Aid Mode

Sticky Keys

Slow Keys

Mono Audio

Invert Colors

Zoom

Guided Access

Large Text

Zoom Window

Subtitles

Assistive Touch

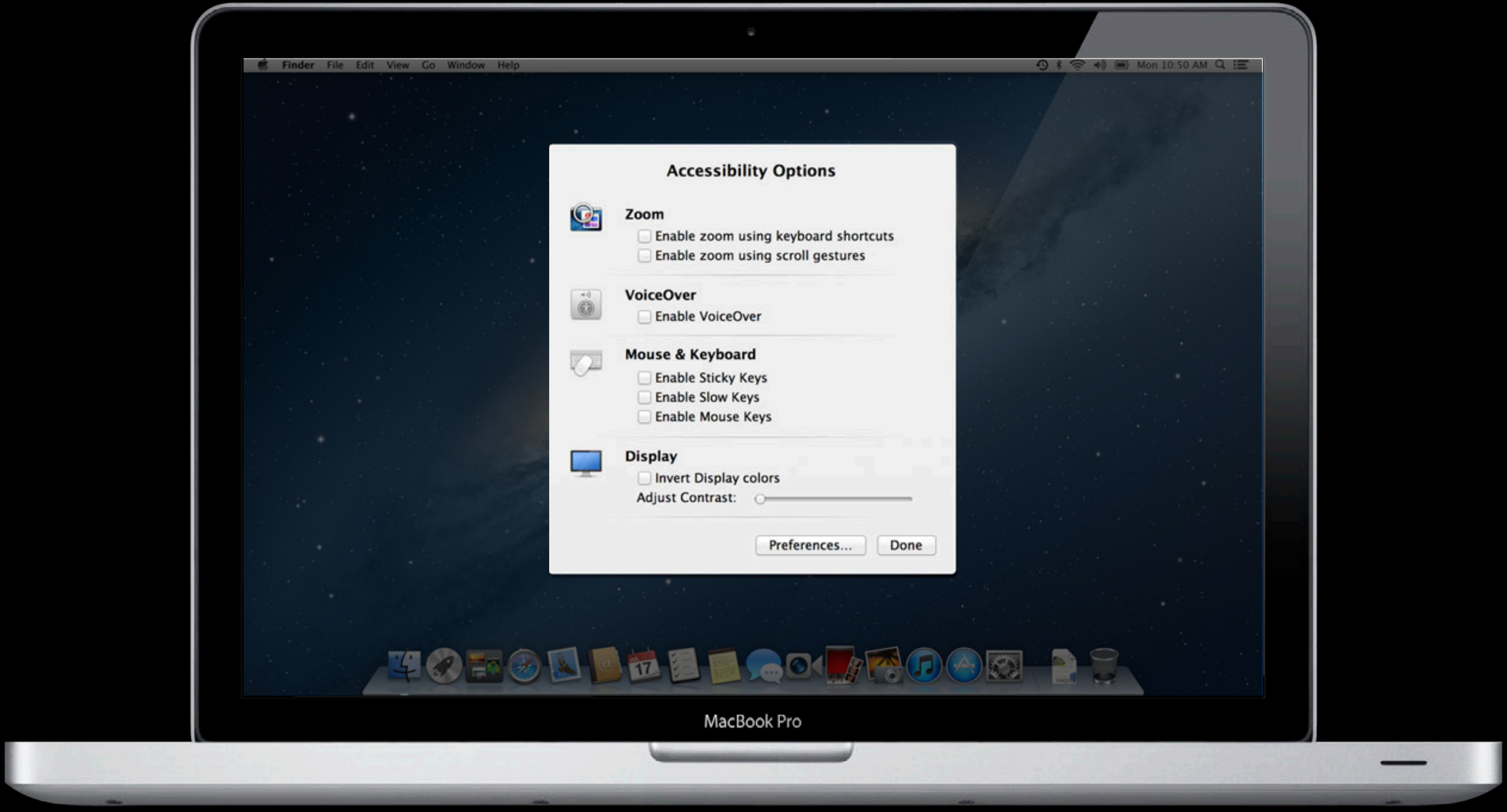
Speak Auto-text

VoiceOver

Closed Captioning

Cursor Scaling

Finder File Edit View Go Window Help
Mon 10:50 AM
MacBook Pro



Accessibility Options

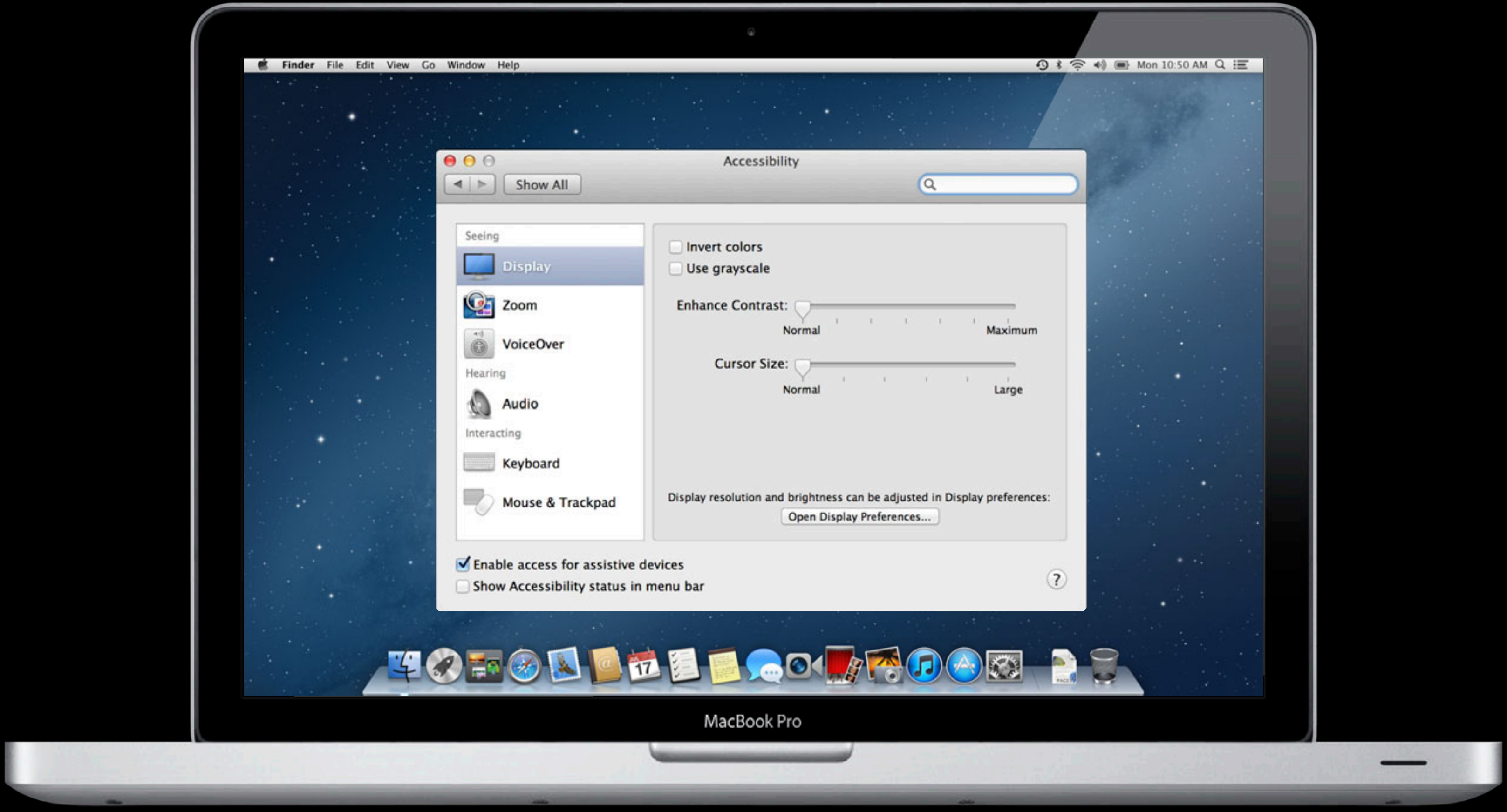
**Zoom**
- [ ] Enable zoom using keyboard shortcuts
- [ ] Enable zoom using scroll gestures

**VoiceOver**
- [ ] Enable VoiceOver

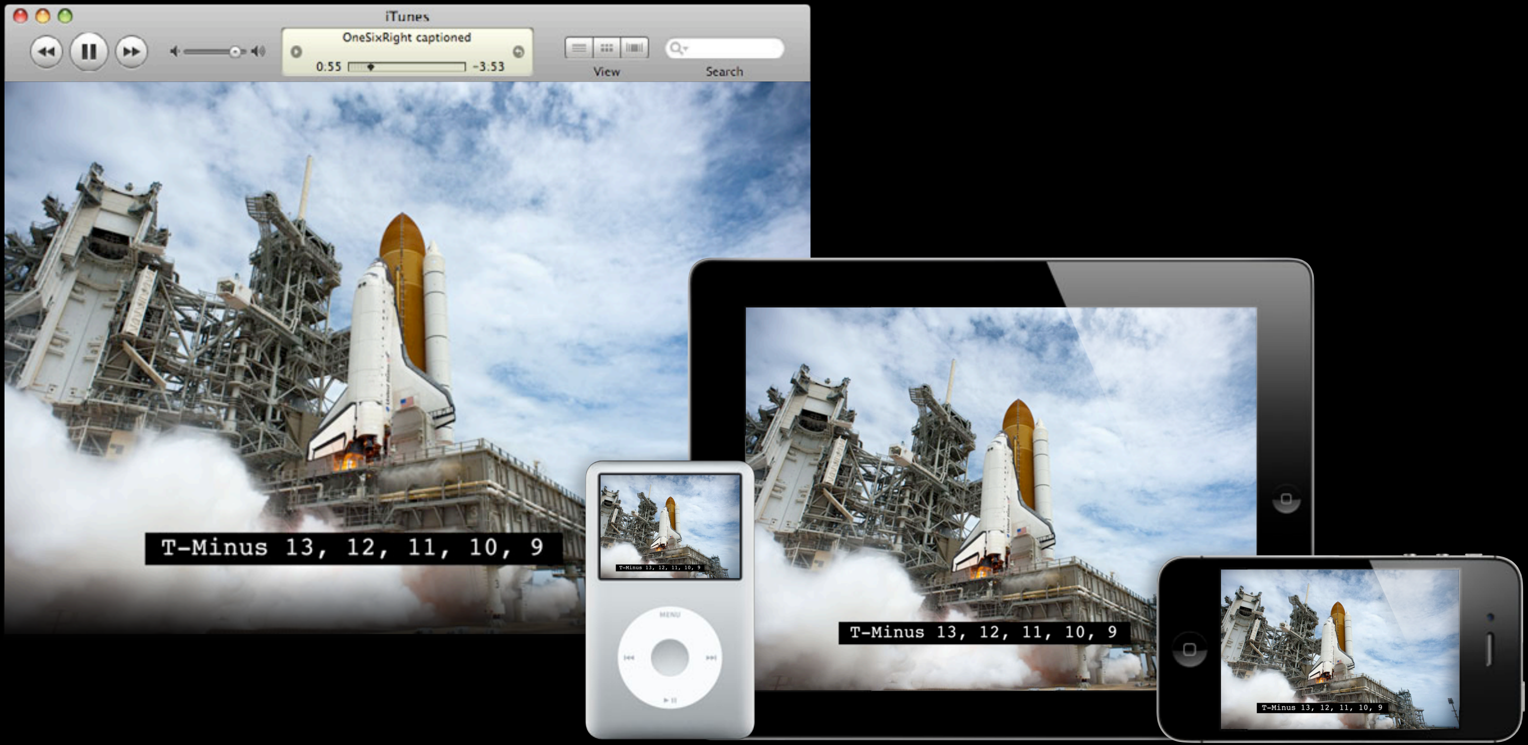
**Mouse & Keyboard**
- [ ] Enable Sticky Keys
- [ ] Enable Slow Keys
- [ ] Enable Mouse Keys

**Display**
- [ ] Invert Display colors

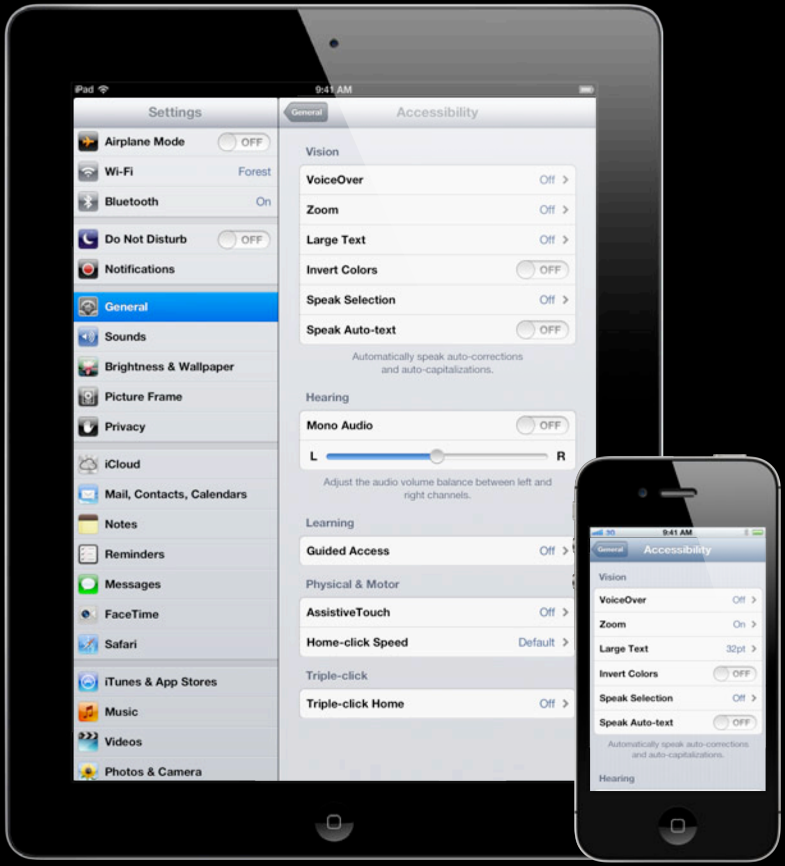
Adjust Contrast:



Invert colors
Use grayscale
Enhance Contrast: Normal — Maximum
Cursor Size: Normal — Large
Display resolution and brightness can be adjusted in Display preferences:
[✓] Enable access for assistive devices
Show Accessibility status in menu bar


iTunes
OneSixRight captioned
0:55
-3:53
View
Search
T-Minus 13, 12, 11, 10, 9
MENU

[redacted]



iPad 9:41 AM

**Settings**

| | |
|---|---|
| Airplane Mode | OFF |
| Wi-Fi | Forest |
| Bluetooth | On |
| Do Not Disturb | OFF |
| Notifications | |
| General | |
| Sounds | |
| Brightness & Wallpaper | |
| Picture Frame | |
| Privacy | |
| iCloud | |
| Mail, Contacts, Calendars | |
| Notes | |
| Reminders | |
| Messages | |
| FaceTime | |
| Safari | |
| iTunes & App Stores | |
| Music | |
| Videos | |
| Photos & Camera | |

General **Accessibility**

Vision

| | |
|---|---|
| VoiceOver | Off |
| Zoom | Off |
| Large Text | Off |
| Invert Colors | OFF |
| Speak Selection | Off |
| Speak Auto-text | OFF |

Automatically speak auto-corrections and auto-capitalizations.

Hearing

| | |
|---|---|
| Mono Audio | OFF |

L R

Adjust the audio volume balance between left and right channels.

Learning

| | |
|---|---|
| Guided Access | Off |

Physical & Motor

| | |
|---|---|
| AssistiveTouch | Off |
| Home-click Speed | Default |

Triple-click

| | |
|---|---|
| Triple-click Home | Off |

9:41 AM

General **Accessibility**

Vision

| | |
|---|---|
| VoiceOver | Off |
| Zoom | On |
| Large Text | 32pt |
| Invert Colors | OFF |
| Speak Selection | Off |
| Speak Auto-text | OFF |

Automatically speak auto-corrections and auto-capitalizations.

Hearing

## VoiceOver

MacBook Pro
tv

[redacted]

6 years

# Millions

Macs with voiceover

## VoiceOver

## Importance of Accessibility

## Why Invest Time in Accessibility

## Why Invest Time in Accessibility

- Access to everyone

## Why Invest Time in Accessibility

- Access to everyone
- Expand your user base

[redacted]

# 10 million

Americans are visually impaired

www.afb.org/info_documents.asp?collectionid=15 [Mar 25, 2008]

[redacted]

## 31 million

Americans are hearing impaired

S. Kochkin. Marketrak vii: Hearing loss population tops 31 million. The Hearing Review, (2005), July 2005.

[redacted]

# 12 million

Americans have a learning disability

C. Smith and L. Strick. Learning Disabilities: A to Z. The Free Press, 1997.

[redacted]

## 1 in 5

Americans have a disability

http://www.census.gov/Press-Release/www/releases/archives/facts_for_features_special_editions/001823.html

## 50 million

Americans have a disability

http://www.census.gov/Press-Release/www/releases/archives/facts_for_features_special_editions/001823.html

## Why Invest Time in Accessibility

- Access to everyone
- Expand your user base

## Why Invest Time in Accessibility

- Access to everyone
- Expand your user base
- Comply with market regulations

## Why Invest Time in Accessibility

- Access to everyone
- Expand your user base
- Comply with market regulations
- Gain a competitive edge

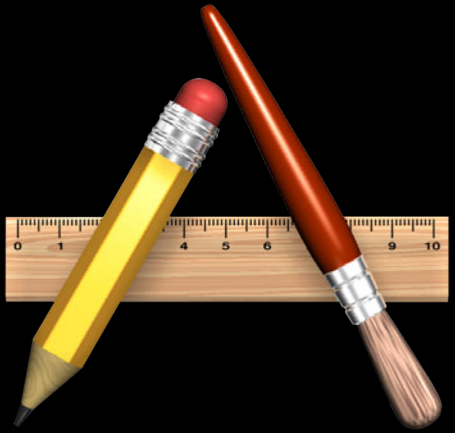
## Accessibility API

## *Demo*

VoiceOver in action!

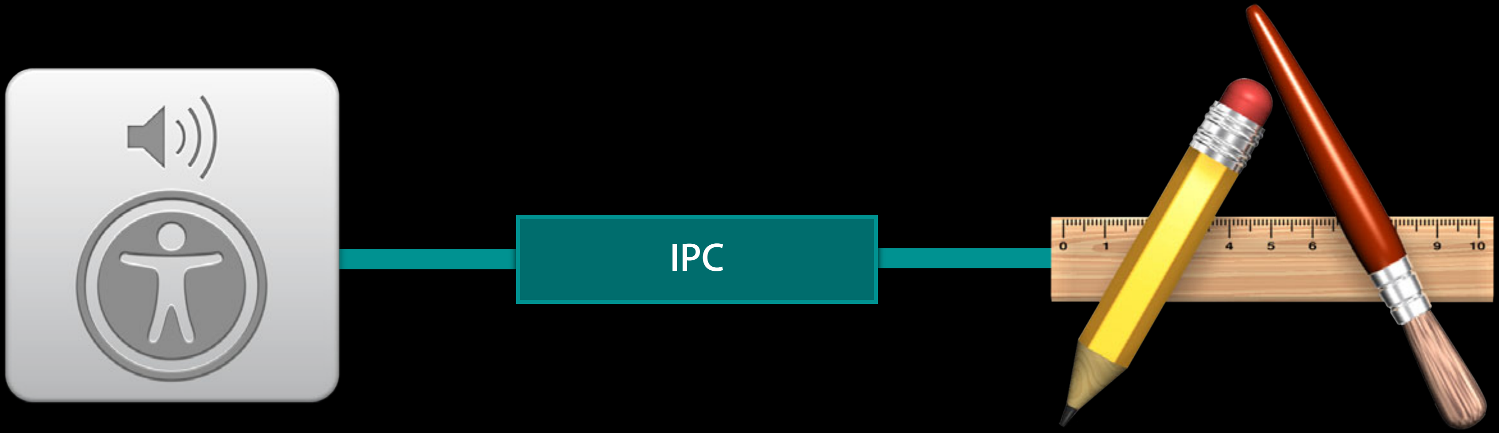
## Providing Accessibility Information

## Providing Accessibility Information

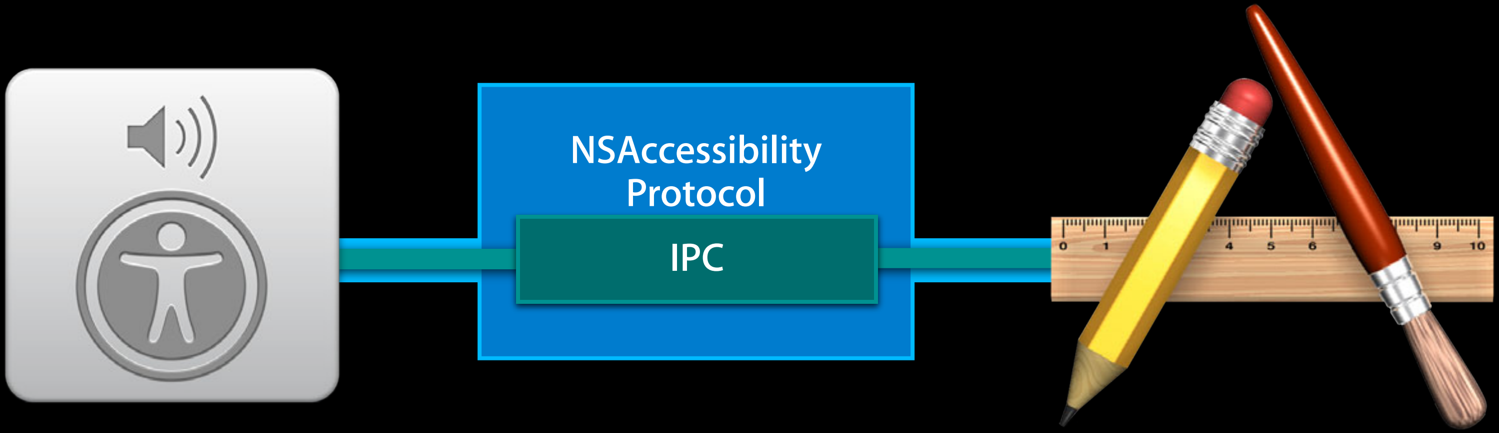


## Providing Accessibility Information

## Providing Accessibility Information

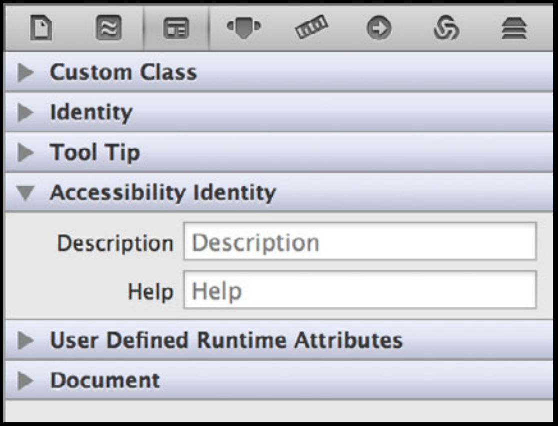


## Providing Accessibility Information



## Providing Accessibility Information

## Providing Accessibility Information


Custom Class

Identity

Tool Tip

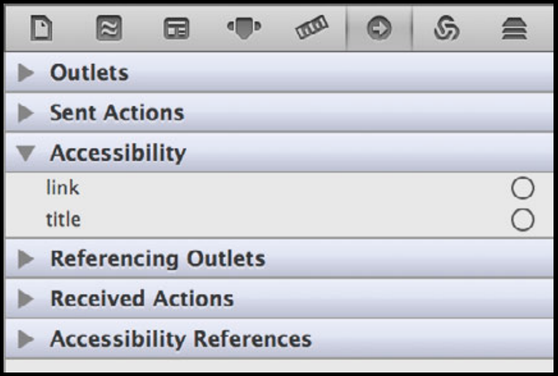
Accessibility Identity

Description: Description

Help: Help

User Defined Runtime Attributes

Document


Outlets

Sent Actions

Accessibility

link

title

Referencing Outlets

Received Actions

Accessibility References

[redacted]

Always use standard
Cocoa controls when possible!

## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
4. Test with Accessibility Inspector and VoiceOver

## Accessibility Recipe for Custom UI

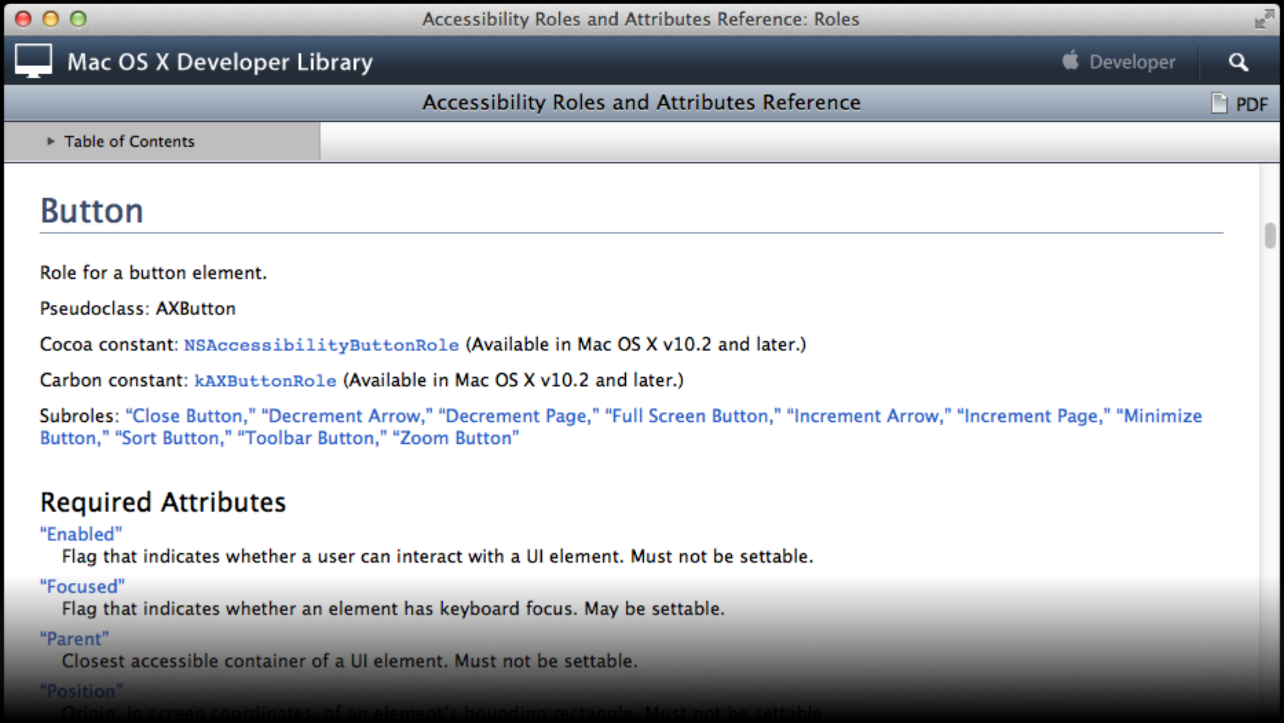
1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
4. Test with Accessibility Inspector and VoiceOver

## Determine Required Attributes

Use Apple documentation
“Accessibility Roles and Attribute Reference”

## Determine Required Attributes

### Use Apple documentation "Accessibility Roles and Attribute Reference"


Accessibility Roles and Attributes Reference: Roles
Mac OS X Developer Library
Developer
Accessibility Roles and Attributes Reference
PDF
Table of Contents

Button

Role for a button element.

Pseudoclass: AXButton

Cocoa constant: `NSAccessibilityButtonRole` (Available in Mac OS X v10.2 and later.)

Carbon constant: `kAXButtonRole` (Available in Mac OS X v10.2 and later.)

Subroles: "Close Button," "Decrement Arrow," "Decrement Page," "Full Screen Button," "Increment Arrow," "Increment Page," "Minimize Button," "Sort Button," "Toolbar Button," "Zoom Button"

Required Attributes

"Enabled"
Flag that indicates whether a user can interact with a UI element. Must not be settable.

"Focused"
Flag that indicates whether an element has keyboard focus. May be settable.

"Parent"
Closest accessible container of a UI element. Must not be settable.

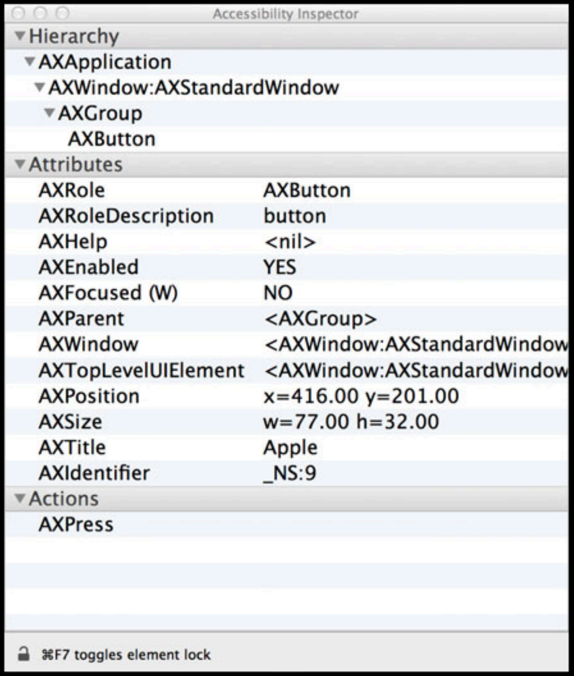
"Position"
Origin, in screen coordinates, of an element's bounding rectangle. Must not be settable.

## Button Accessibility

| Attributes | Actions |
|---|---|
| Role | Press |
| Role Description | |
| Size | |
| Position | |
| Enabled | |
| Focused | |
| Parent | |
| Top-Level UIElement | |
| Title or Description | |
| Window | |

## Determine Required Attributes

## Determine Required Attributes



▼Hierarchy
▼AXApplication
▼AXWindow:AXStandardWindow
▼AXGroup
AXButton

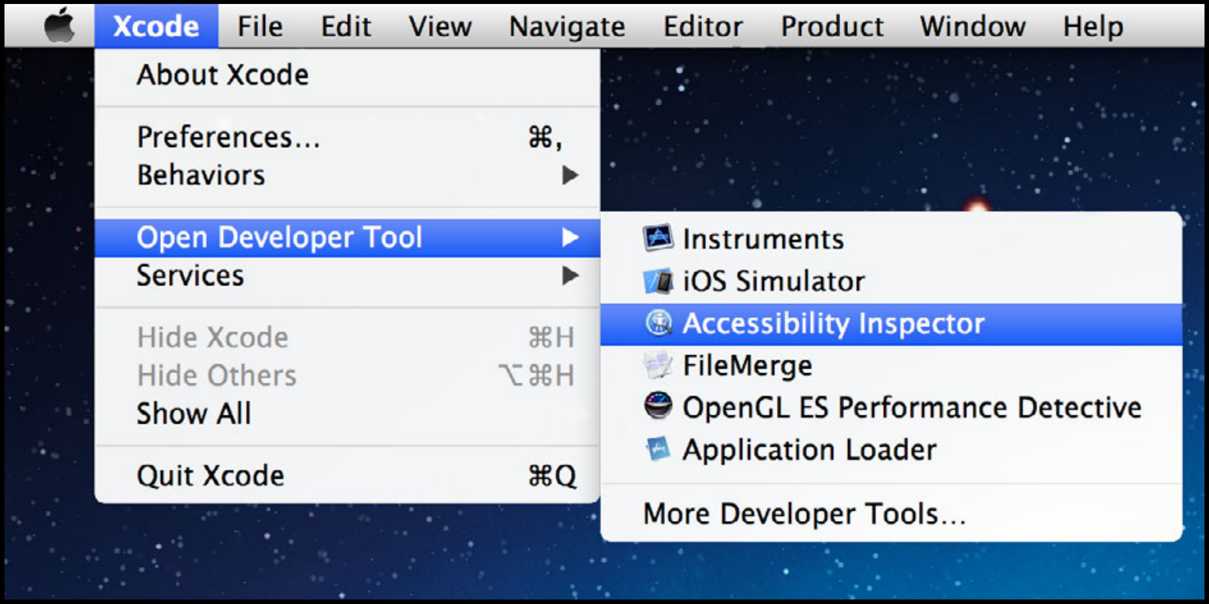
▼Attributes

| | |
|---|---|
| AXRole | AXButton |
| AXRoleDescription | button |
| AXHelp | <nil> |
| AXEnabled | YES |
| AXFocused (W) | NO |
| AXParent | <AXGroup> |
| AXWindow | <AXWindow:AXStandardWindow |
| AXTopLevelUIElement | <AXWindow:AXStandardWindow |
| AXPosition | x=416.00 y=201.00 |
| AXSize | w=77.00 h=32.00 |
| AXTitle | Apple |
| AXIdentifier | _NS:9 |

▼Actions
AXPress

⌘F7 toggles element lock

## Determine Required Attributes



## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
4. Test with Accessibility Inspector and VoiceOver

## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
4. Test with Accessibility Inspector and VoiceOver

## NSAccessibility API

### Overview

## NSAccessibility API

### Overview

- Is ignored

## NSAccessibility API

### Overview

- Is ignored
- Attributes (title, description, etc.)
  - Supported, getter and setter

## NSAccessibility API

### Overview

- Is ignored
- Attributes (title, description, etc.)
  - Supported, getter and setter
- Parameterized attributes (string for range, line for index, etc.)
  - Supported and getter

## NSAccessibility API

### Overview

- Is ignored
- Attributes (title, description, etc.)
  - Supported, getter and setter
- Parameterized attributes (string for range, line for index, etc.)
  - Supported and getter
- Actions (press, increment, etc.)
  - Supported and performer

## NSAccessibility API

### Overview

- Is ignored
- Attributes (title, description, etc.)
  - Supported, getter and setter
- Parameterized attributes (string for range, line for index, etc.)
  - Supported and getter
- Actions (press, increment, etc.)
  - Supported and performer
- Hit testing and focus testing

## NSAccessibility API

### Overview

- Is ignored
- Attributes (title, description, etc.)
  - Supported, getter and setter
- Parameterized attributes (string for range, line for index, etc.)
  - Supported and getter
- Actions (press, increment, etc.)
  - Supported and performer
- Hit testing and focus testing
- Notifications

## NSAccessibility Protocol

### Is ignored?

```
- (BOOL)accessibilityIsIgnored;
```

## NSAccessibility Protocol

### Getting and setting attributes

```
- (NSArray *)accessibilityAttributeNames;

- (id)accessibilityAttributeValue:(NSString *)attribute;
```

## NSAccessibility Protocol

### Getting and setting attributes

```
- (NSArray *)accessibilityAttributeNames;

- (id)accessibilityAttributeValue:(NSString *)attribute;

- (BOOL)accessibilityIsAttributeSettable:(NSString *)attribute;

- (void)accessibilitySetValue:(id)value
                 forAttribute:(NSString *)attribute;
```

## NSAccessibility Protocol

### Getting and setting parameterized attributes

```
- (NSArray *)accessibilityParameterizedAttributeNames;

- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter;
```

## NSAccessibility Protocol

### Actions

```
- (NSArray *)accessibilityActionNames;

- (NSString *)accessibilityActionDescription:(NSString *)action;

- (void)accessibilityPerformAction:(NSString *)action;
```

- Example Actions

```
AXPressAction                  AXCancelAction
AXIncrementAction              AXRaiseAction
AXDecrementAction              AXShowMenuAction
AXConfirmAction
```

## NSAccessibility Protocol

### Hit testing and focus testing

```
- (id)accessibilityHitTest:(NSPoint)point;

- (id)accessibilityFocusedUIElement;
```

## NSAccessibility Protocol

### Sending notifications

```
NSAccessibilityPostNotification(id element, NSString *notification)
```

- Example Notifications

```
AXFocusedUIElementChangedNotification
AXValueChangedNotification
AXUIElementDestroyedNotification
AXWindowCreatedNotification
others…
```

## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
4. Test with Accessibility Inspector and VoiceOver

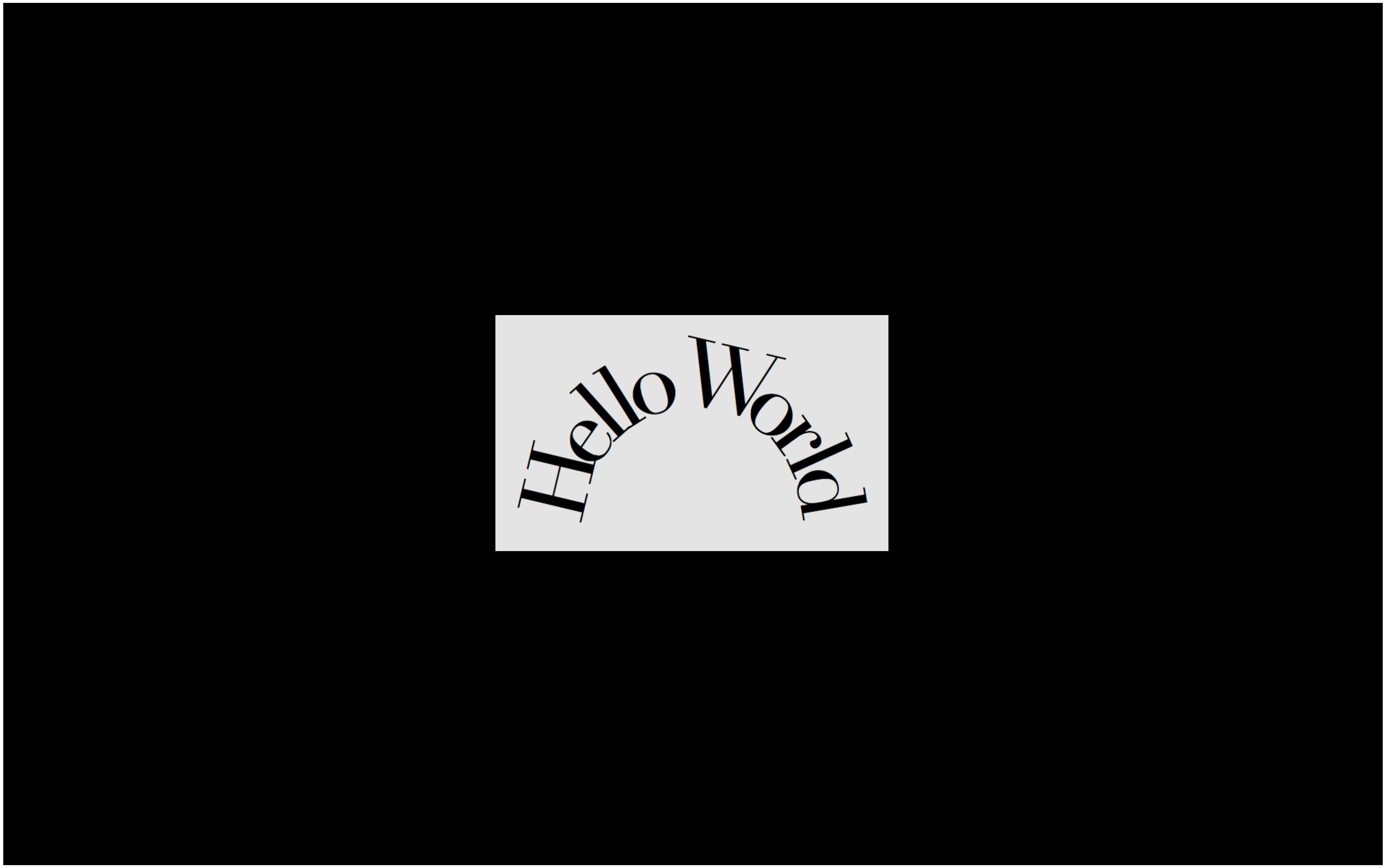
## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
    a) Always remember to call super!
4. Test with Accessibility Inspector and VoiceOver

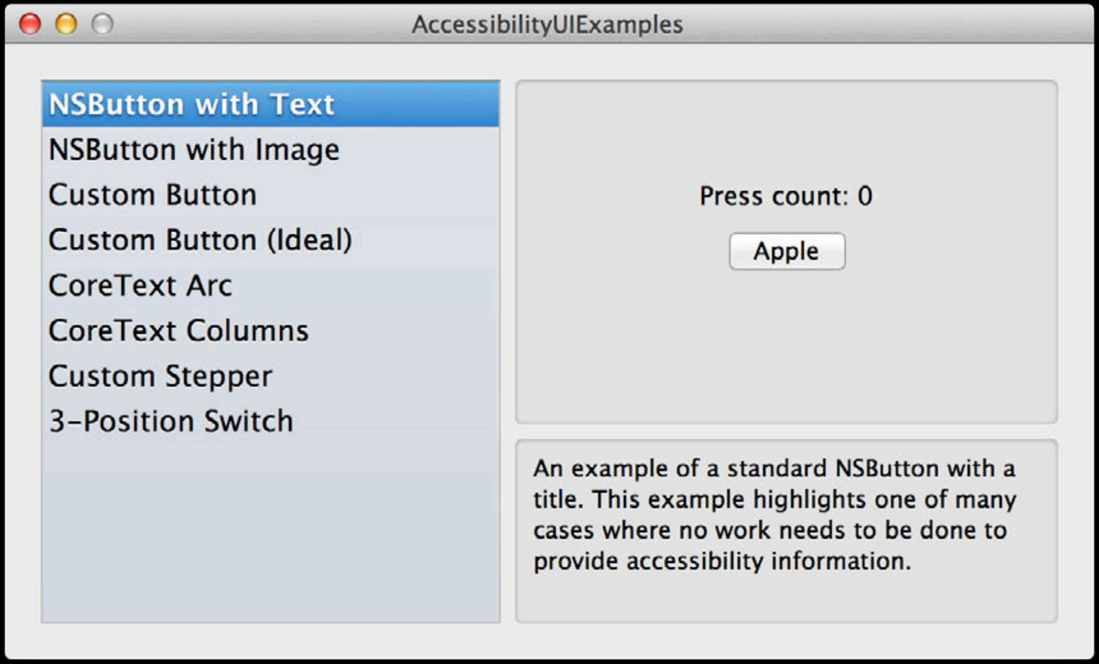
## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine required attributes
3. Implement required NSAccessibility methods
    a) Always remember to call super!
4. Test with Accessibility Inspector and VoiceOver

## Accessible Custom UI


Hello World

Apple designs Macs, the best personal computers in the world, along with OS X, iLife, iWork and professional software. Apple leads the digital music revolution with its iPods and iTunes online store. Apple has reinvented the mobile phone with its revolutionary iPhone and App Store, and is defining the future of mobile media and computing devices with iPad.

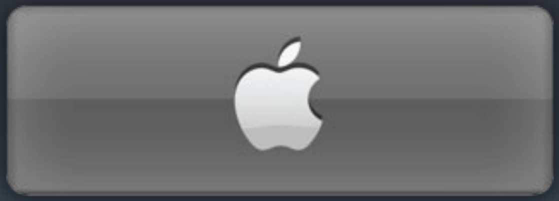


An example of a standard NSButton with a title. This example highlights one of many cases where no work needs to be done to provide accessibility information.

## *Demo*

AccessibilityUIExamples

## Custom Button



## Accessibility Recipe for Custom UI

1. Subclass appropriately
2. Determine attributes
3. Implement NSAccessibility
4. Test

## Accessibility Recipe for Custom UI

1. ~~Subclass appropriately~~ (skip for demonstration)
2. Determine attributes
3. Implement NSAccessibility
4. Test

## Button Accessibility

| Attributes | Actions |
|---|---|
| Role | Press |
| Role Description | |
| Size | |
| Position | |
| Enabled | |
| Focused | |
| Parent | |
| Top-Level UIElement | |
| Title or Description | |
| Window | |

## Button Accessibility

| Attributes | Actions |
|---|---|
| Role | Press |
| Role Description | |
| Size | |
| Position | |
| Enabled | |
| Focused | |
| Parent | |
| Top-Level UIElement | |
| Title or Description | |
| Window | |

## Button Accessibility

| Attributes | Actions |
|---|---|
| Role | Press |
| Title or Description | |

## Not Ignored

```
- (BOOL)accessibilityIsIgnored {
    return NO;
}
```

## Supported Attributes

```
- (NSArray *)accessibilityAttributeNames {
    static NSMutableArray *attributes = nil;

    if ( attributes == nil ) {

        attributes = [[super accessibilityAttributeNames] mutableCopy];

        NSArray *appendAttributes = @[NSAccessibilityDescriptionAttribute,
                                      NSAccessibilityRoleAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![attributes containsObject:attribute] ) {
                [attributes addObject:attribute];
            }
        }

    }
    return attributes;
}
```

## Supported Attributes

```
- (NSArray *)accessibilityAttributeNames {
    static NSMutableArray *attributes = nil;

    if ( attributes == nil ) {

        attributes = [[super accessibilityAttributeNames] mutableCopy];

        NSArray *appendAttributes = @[NSAccessibilityDescriptionAttribute,
                                      NSAccessibilityRoleAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![attributes containsObject:attribute] ) {
                [attributes addObject:attribute];
            }
        }

    }
    return attributes;
}
```

## Supported Attributes

```
- (NSArray *)accessibilityAttributeNames {
    static NSMutableArray *attributes = nil;

    if ( attributes == nil ) {

        attributes = [[super accessibilityAttributeNames] mutableCopy];

        NSArray *appendAttributes = @[NSAccessibilityDescriptionAttribute,
                                      NSAccessibilityRoleAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![attributes containsObject:attribute] ) {
                [attributes addObject:attribute];
            }
        }

    }
    return attributes;
}
```

## Supported Attributes

```
- (NSArray *)accessibilityAttributeNames {
    static NSMutableArray *attributes = nil;

    if ( attributes == nil ) {

        attributes = [[super accessibilityAttributeNames] mutableCopy];

        NSArray *appendAttributes = @[NSAccessibilityDescriptionAttribute,
                                      NSAccessibilityRoleAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![attributes containsObject:attribute] ) {
                [attributes addObject:attribute];
            }
        }

    }
    return attributes;
}
```

## Value For Attribute

```
- (id)accessibilityAttributeValue:(NSString *)attribute {

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        return NSAccessibilityButtonRole;
    }

    else if ( [attribute
               isEqualToString:NSAccessibilityDescriptionAttribute] ) {
        return NSLocalizedString(@"Apple", @"Apple button.");
    }

    return [super accessibilityAttributeValue:attribute];
}
```

## Value For Attribute

### Role

```
- (id)accessibilityAttributeValue:(NSString *)attribute {

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        return NSAccessibilityButtonRole;
    }

    else if ( [attribute
                isEqualToString:NSAccessibilityDescriptionAttribute] ) {
        return NSLocalizedString(@"Apple", @"Apple button.");
    }

    return [super accessibilityAttributeValue:attribute];
}
```

## Value For Attribute

### Description

```
- (id)accessibilityAttributeValue:(NSString *)attribute {

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        return NSAccessibilityButtonRole;
    }

    else if ( [attribute
                isEqualToString:NSAccessibilityDescriptionAttribute] ) {
        return NSLocalizedString(@"Apple", @"Apple button.");
    }

    return [super accessibilityAttributeValue:attribute];
}
```

## Value For Attribute

### Call Super!

```
- (id)accessibilityAttributeValue:(NSString *)attribute {

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        return NSAccessibilityButtonRole;
    }

    else if ( [attribute
               isEqualToString:NSAccessibilityDescriptionAttribute] ) {
        return NSLocalizedString(@"Apple", @"Apple button.");
    }

    return [super accessibilityAttributeValue:attribute];
}
```

## Supported Actions

```
- (NSArray *)accessibilityActionNames {
    static NSMutableArray *actions = nil;
    if ( actions == nil ) {
        actions = [[super accessibilityActionNames] mutableCopy];
        if ( ![actions containsObject:NSAccessibilityPressAction] )
            [actions addObject:NSAccessibilityPressAction];
    }
    return actions;
}
```

## Perform Action

```
- (void)accessibilityPerformAction:(NSString *)action {

    if ([action isEqualToString:NSAccessibilityPressAction]) {
        [self performPress];
    }

    else {
        [super accessibilityPerformAction:action];
    }

}
```

## Perform Action

```
- (void)accessibilityPerformAction:(NSString *)action {

    if ([action isEqualToString:NSAccessibilityPressAction]) {
        [self performPress];
    }

    else {
        [super accessibilityPerformAction:action];
    }

}
```

## Perform Action

```
- (void)accessibilityPerformAction:(NSString *)action {

    if ([action isEqualToString:NSAccessibilityPressAction]) {
        [self performPress];
    }

    else {
        [super accessibilityPerformAction:action];
    }

}
```

## Simple Text Field

Hello World

## Text Field Accessibility

**Attributes**

| | |
|---|---|
| Role | Top-Level UIElement |
| Role Description | Value |
| Size | Window |
| Position | Number Of Characters |
| Enabled | Selected Text |
| Focused | Selected Text Range |
| Parent | Visible Character Range |

## Text Field Accessibility

**Attributes**

| | |
|---|---|
| Role | Top-Level UIElement |
| Role Description | Value |
| Size | Window |
| Position | Number Of Characters |
| Enabled | Selected Text |
| Focused | Selected Text Range |
| Parent | Visible Character Range |

## Text Field Accessibility

| Attributes |
|---|
| Role |
| Enabled |
| Value |
| Number Of Characters |
| Selected Text |
| Selected Text Range |
| Visible Character Range |

## Text Field Accessibility

| Attributes |
|---|
| Role |
| Enabled |
| Value |
| Number Of Characters |
| Selected Text |
| Selected Text Range |
| Visible Character Range |

## Text Field Accessibility

| Attributes |
|---|
| Role |
| Value |
| Number Of Characters |
| Visible Character Range |

## Not Ignored

```
- (BOOL)accessibilityIsIgnored {
    return NO;
}
```

## Supported Attributes

```
- (NSArray *)accessibilityAttributeNames {
    static NSMutableArray *attributes = nil;
    if ( attributes == nil ) {
        attributes = [[super accessibilityAttributeNames] mutableCopy];

        NSArray *appendAttributes = @[NSAccessibilityRoleAttribute,
                     NSAccessibilityValueAttribute,
                     NSAccessibilityNumberOfCharactersAttribute,
                     NSAccessibilityVisibleCharacterRangeAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![attributes containsObject:attribute] ) {
                [attributes addObject:attribute];
            }
        }
    }
    return attributes;
}
```

## Value For Attribute

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...







    ...
    return value;
}
```

## Value For Attribute

### Role

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        value = NSAccessibilityStaticTextRole;
    }





    ...
    return value;
}
```

## Value For Attribute

### Role

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        value = NSAccessibilityStaticTextRole;
    }





    ...
    return value;
}
```

## Value For Attribute

### Value

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:NSAccessibilityValueAttribute] ) {
        value = self.string;
    }




    ...
    return value;
}
```

## Value For Attribute

### Number of characters

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                NSAccessibilityNumberOfCharactersAttribute] ) {
        value = [NSNumber numberWithUnsignedInteger:self.string.length];
    }



    ...
    return value;
}
```

## Value For Attribute

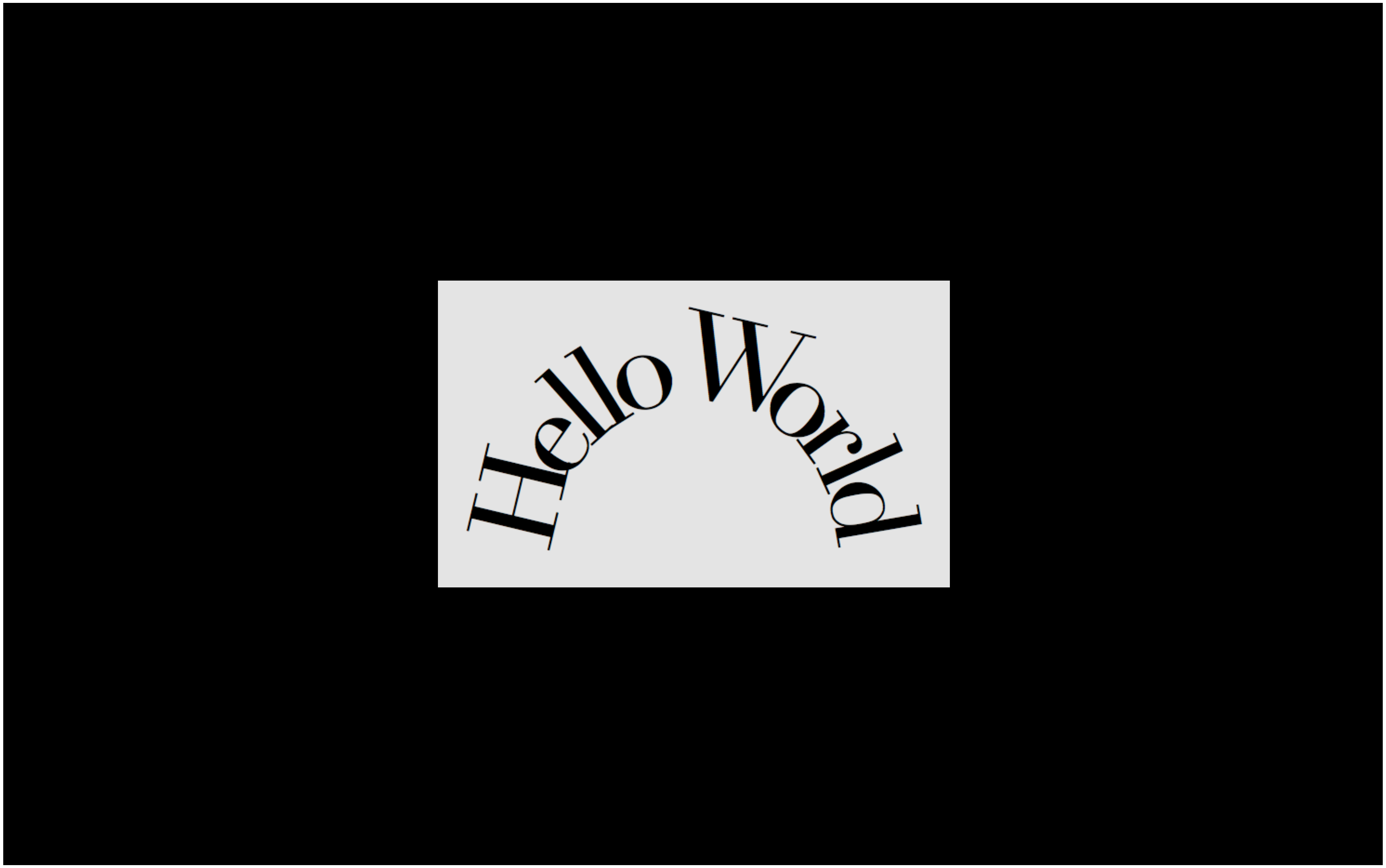
### Visible character range

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                  NSAccessibilityVisibleCharacterRangeAttribute] ) {
        value = [NSValue valueWithRange:
                    NSMakeRange(0, self.string.length)];
    }



    ...
    return value;
}
```

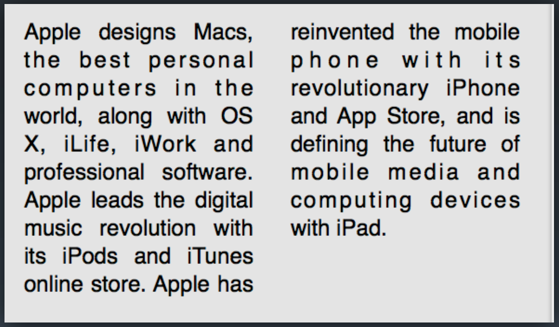
## Value For Attribute

### Call super!

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else {
        // Fetch remaining attribute values from the parent class, NSView.
        value = [super accessibilityAttributeValue:attribute];
    }




    ...
    return value;
}
```


Hello World

## Multi-line Text Field



Apple designs Macs, the best personal computers in the world, along with OS X, iLife, iWork and professional software. Apple leads the digital music revolution with its iPods and iTunes online store. Apple has reinvented the mobile phone with its revolutionary iPhone and App Store, and is defining the future of mobile media and computing devices with iPad.

## Text Field Accessibility

| Attributes |
|---|
| Role |
| Value |
| Number Of Characters |
| Visible Character Range |

# Text Field Accessibility

| Parameterized Attributes |
|---|
| Line For Index |
| Range For Line |
| String For Range |
| Attributed String For Range |
| Bounds For Range |

## Text Field Accessibility

| Parameterized Attributes |
|---|
| Line For Index |
| Range For Line |
| String For Range |
| Attributed String For Range |
| Bounds For Range |

**Enables line-by-line navigation for VoiceOver!**

## Value for Parameterized Attribute

### Supported parameterized attributes

```
- (NSArray *)accessibilityParameterizedAttributeNames {
    static NSMutableArray *pAttributes = nil;
    if ( pAttributes == nil ) {
        pAttributes = [super accessibilityParameterizedAttributeNames];
        pAttributes = [pAttributes mutableCopy];

        NSArray *appendAttributes =
          @[NSAccessibilityLineForIndexParameterizedAttribute,
            NSAccessibilityRangeForLineParameterizedAttribute,
            NSAccessibilityStringForRangeParameterizedAttribute,
            NSAccessibilityAttributedStringForRangeParameterizedAttribute,
            NSAccessibilityBoundsForRangeParameterizedAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![pAttributes containsObject:attribute] )
                [pAttributes addObject:attribute];
        }
    }
    return pAttributes;
}
```

## Value for Parameterized Attribute

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...









    ...
    return value;
}
```

## Value for Parameterized Attribute

### String for range

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...

    if ( [attribute isEqualToString:
          NSAccessibilityStringForRangeParameterizedAttribute] ) {

        if ( [parameter isKindOfClass:[NSValue class]] ) {
            NSRange *range = [(NSValue *)parameter rangeValue];
            value = [self stringForRange:range];
        }
    }


    ...
    return value;
}
```

## Value for Parameterized Attribute

### String for range

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...

    if ( [attribute isEqualToString:
           NSAccessibilityStringForRangeParameterizedAttribute] ) {

        if ( [parameter isKindOfClass:[NSValue class]] ) {
             NSRange *range = [(NSValue *)parameter rangeValue];
             value = [self stringForRange:range];
        }
    }


    ...
    return value;
}
```

## Value for Parameterized Attribute

### Attributed string for range

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...

    else if ([attribute isEqualToString:
          NSAccessibilityAttributedStringForRangeParameterizedAttribute]){

        if ( [parameter isKindOfClass:[NSValue class]] ) {
             NSRange *range = [(NSValue *)parameter rangeValue];
             value = [self attributedStringForRange:range];
        }
    }


    ...
    return value;
}
```

## Value for Parameterized Attribute

### Line for index

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                NSAccessibilityLineForIndexParameterizedAttribute] ) {

        if ( [parameter isKindOfClass:[NSValue class]] ) {
             NSUInteger i = [(NSValue *)parameter unsignedIntegerValue];
             NSUInteger line = [self lineForIndex:i];
             value = [NSNumber numberWithUnsignedInteger:line];
        }
    }

    ...
    return value;
}
```

## Value for Parameterized Attribute

### Range for line

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                NSAccessibilityRangeForLineParameterizedAttribute] ) {

        if ( [parameter isKindOfClass:[NSValue class]] ) {
             NSUInteger i = [(NSValue *)parameter unsignedIntegerValue];
             NSRange range = [self rangeForLine:i];
             value = [NSValue valueWithRange:rangeForLine];
        }
    }

    ...
    return value;
}
```

## Value for Parameterized Attribute

### Bounds for range

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                NSAccessibilityBoundsForRangeParameterizedAttribute] ) {

        if ( [parameter isKindOfClass:[NSValue class]] ) {
             NSRange range = [parameter rangeValue];
             NSRect bounds = [self boundsForRange:range];
             value = [NSValue valueWithRect:bounds];
        }
    }

    ...
    return value;
}
```

## Value for Parameterized Attribute

### Call super!

```
- (id)accessibilityAttributeValue:(NSString *)attribute
                     forParameter:(id)parameter {
    id value = nil;
    ...
    else {
        value = [super accessibilityAttributeValue:attribute
                                      forParameter:parameter];
    }




    ...
    return value;
}
```

Apple designs Macs, the best personal computers in the world, along with OS X, iLife, iWork and professional software. Apple leads the digital music revolution with its iPods and iTunes online store. Apple has reinvented the mobile phone with its revolutionary iPhone and App Store, and is defining the future of mobile media and computing devices with iPad.

## Custom Stepper

## Incrementor Accessibility

| Attributes | Actions |
|---|---|
| Role | Increment |
| Role Description | Decrement |
| Size | |
| Position | |
| Enabled | |
| Focused | |
| Parent | |
| Top-Level UIElement | |
| Title or Description | |
| Window | |
| Children | |
| Increment Button | |
| Decrement Button | |

## Incrementor Accessibility

| Attributes | Actions |
|---|---|
| Role | Increment |
| Role Description | Decrement |
| Size | |
| Position | |
| Enabled | |
| Focused | |
| Parent | |
| Top-Level UIElement | |
| Title or Description | |
| Window | |
| Children | |
| Increment Button | |
| Decrement Button | |

## Incrementor Accessibility

| Attributes | Actions |
|---|---|
| Role | Increment |
| Enabled | Decrement |
| Title or Description | |
| Children | |
| Increment Button | |
| Decrement Button | |

## Create Accessibility Children

```
- (NSArray *)myAccessibilityChildren {
    static NSArray *children = nil;
    if  (children == nil) {

        FauxUIElement *up = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityIncrementArrowSubrole parent:self];
        up.tag = kCustomStepperUpButtonTag;

        FauxUIElement *down = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityDecrementArrowSubrole parent:self];
        down = kCustomStepperDownButtonTag;

        children = [[NSArray alloc] initWithObjects:up, down, nil];
    }
    return children;
}
```

## Create Accessibility Children

```
- (NSArray *)myAccessibilityChildren {
    static NSArray *children = nil;
    if  (children == nil) {

        FauxUIElement *up = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityIncrementArrowSubrole parent:self];
        up.tag = kCustomStepperUpButtonTag;

        FauxUIElement *down = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityDecrementArrowSubrole parent:self];
        down = kCustomStepperDownButtonTag;

        children = [[NSArray alloc] initWithObjects:up, down, nil];
    }
    return children;
}
```

## Create Accessibility Children

```
- (NSArray *)myAccessibilityChildren {
    static NSArray *children = nil;
    if  (children == nil) {

        FauxUIElement *up = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityIncrementArrowSubrole parent:self];
        up.tag = kCustomStepperUpButtonTag;

        FauxUIElement *down = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityDecrementArrowSubrole parent:self];
        down = kCustomStepperDownButtonTag;

        children = [[NSArray alloc] initWithObjects:up, down, nil];
    }
    return children;
}
```

## Create Accessibility Children

```
- (NSArray *)myAccessibilityChildren {
    static NSArray *children = nil;
    if  (children == nil) {

        FauxUIElement *up = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityIncrementArrowSubrole parent:self];
        up.tag = kCustomStepperUpButtonTag;

        FauxUIElement *down = [FauxUIElement elementWithRole:
                NSAccessibilityButtonRole
                subrole:NSAccessibilityDecrementArrowSubrole parent:self];
        down = kCustomStepperDownButtonTag;

        children = [[NSArray alloc] initWithObjects:up, down, nil];
    }
    return children;
}
```

## Not Ignored

```
- (BOOL)accessibilityIsIgnored {
    return NO;
}
```

## Supported Attributes

```
- (NSArray *)accessibilityAttributeNames {
    static NSMutableArray *attributes = nil;
    if ( attributes == nil ) {
        attributes = [[super accessibilityAttributeNames] mutableCopy];
        NSArray *appendAttributes = @[NSAccessibilityRoleAttribute,
                                      NSAccessibilityChildrenAttribute,
                                      NSAccessibilityIncrementButtonAttribute,
                                      NSAccessibilityDecrementButtonAttribute,
                                      NSAccessibilityDescriptionAttribute];

        for ( NSString *attribute in appendAttributes ) {
            if ( ![attributes containsObject:attribute] )
                [attributes addObject:attribute];
        }
    }
    return attributes;
}
```

## Value For Attribute

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...







    ...
    return value;
}
```

## Value For Attribute

### Role

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        value = NSAccessibilityIncrementorRole;
    }



    ...
    return value;
}
```

## Value For Attribute

### Role

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    if ( [attribute isEqualToString:NSAccessibilityRoleAttribute] ) {
        value = NSAccessibilityIncrementorRole;
    }



    ...
    return value;
}
```

## Value For Attribute

### Children

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                 NSAccessibilityChildrenAttribute] ) {
        value = [self myAccessibilityChildren];
    }


    ...
    return value;
}
```

## Value For Attribute

### Increment button

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                  NSAccessibilityIncrementButtonAttribute] ) {
        return [[self myAccessibilityChildren] objectAtIndex:0];
    }


    ...
    return value;
}
```

## Value For Attribute

### Decrement button

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                NSAccessibilityDecrementButtonAttribute] ) {
        return [[self myAccessibilityChildren] objectAtIndex:1];
    }


    ...
    return value;
}
```

## Value For Attribute

### Description

```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else if ( [attribute isEqualToString:
                NSAccessibilityDescriptionAttribute] ) {
        return NSLocalizedString(@"volume", @”Volume stepper”);
    }


    ...
    return value;
}
```

## Value For Attribute

### Call super!

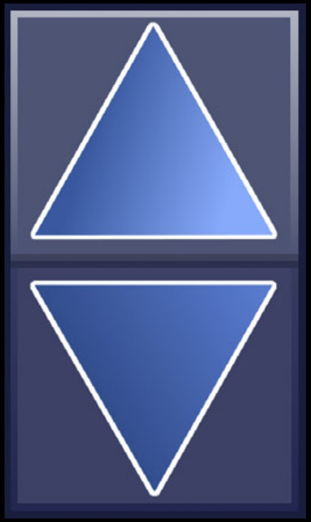
```
- (id)accessibilityAttributeValue:(NSString *)attribute {
    id value = nil;
    ...

    else {
        return [super accessibilityAttributeValue:attribute];
    }



    ...
    return value;
}
```

## Supported Actions

```
- (NSArray *)accessibilityActionNames {
    static NSMutableArray *actions = nil;
    if ( actions == nil ) {
        actions = [[super accessibilityActionNames] mutableCopy];
        if ( ![actions containsObject:NSAccessibilityIncrementAction] )
            [actions addObject:NSAccessibilityIncrementAction];
        if ( ![actions containsObject:NSAccessibilityDecrementAction] )
            [actions addObject:NSAccessibilityDecrementAction];
    }
    return actions;
}
```

## Perform Action

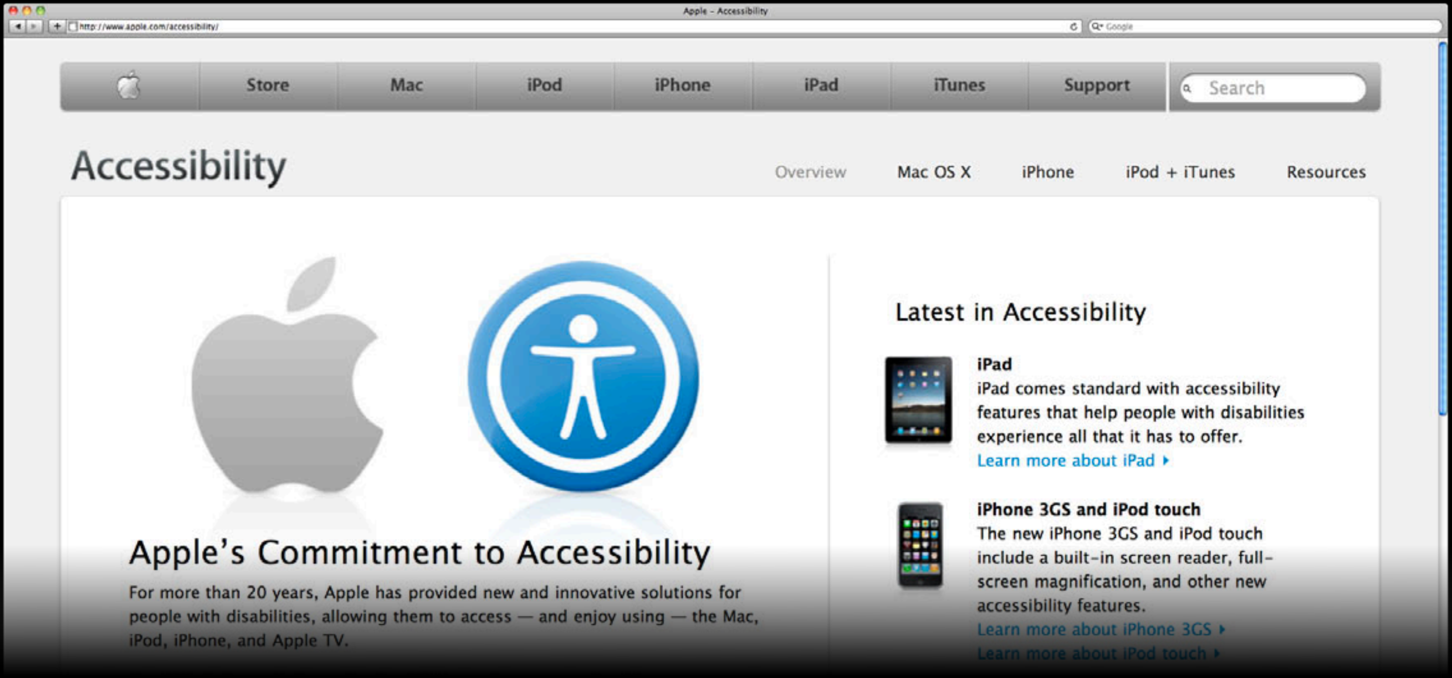
```
- (void)accessibilityPerformAction:(NSString *)action {

    if ([action isEqualToString: NSAccessibilityIncrementAction]) {
        [self performIncrement];
    }
    else if ([action isEqualToString: NSAccessibilityDecrementAction]) {
        [self performDecrement];
    }
    else {
        [super accessibilityPerformAction:action];
    }
}
```



## *Demo*

## Accessibility Web Page

http://www.apple.com/accessibility



Accessibility

Apple's Commitment to Accessibility

For more than 20 years, Apple has provided new and innovative solutions for people with disabilities, allowing them to access — and enjoy using — the Mac, iPod, iPhone, and Apple TV.

Latest in Accessibility

**iPad**
iPad comes standard with accessibility features that help people with disabilities experience all that it has to offer.
Learn more about iPad ›

**iPhone 3GS and iPod touch**
The new iPhone 3GS and iPod touch include a built-in screen reader, full-screen magnification, and other new accessibility features.
Learn more about iPhone 3GS ›
Learn more about iPod touch ›

## More Information

**Bill Dudney**
Application Technologies Evangelist
dudney@apple.com

**Accessibility Mailing List**
Public Developer List
accessibility-dev@lists.apple.com

**Documentation**
http://developer.apple.com/wwdc

**Apple Developer Forums**
http://devforums.apple.com

## Related Sessions

| | |
|---|---|
| Accessibility for iOS | Russian Hill<br>Wednesday 9:00AM |
| Improving Accessibility in Books | Russian Hill<br>Thursday 9:00AM |

## Labs

| | |
|---|---|
| Accessibility and Speech Lab | App Services Lab B<br>Wednesday 11:30AM |

## Summary

Technologies

API

Custom UI

WWDC2012