What's New in Cocoa

Session 207

Ali Ozer
Daphne Larose

API Refinements

AppKit

Foundation

API Refinements

Ali Ozer, Director, Cocoa

API Refinements

Properties

Nullability

Generics

Swift naming guidelines

String enumerations

Nested types

API Refinements

Properties

Nullability

Generics

Swift naming guidelines

String enumerations

Nested types

String Enumerations and Nested Types

```
// NSSharingService, Swift 3

public let NSSharingServiceNameComposeEmail: String
public let NSSharingServiceNameComposeMessage: String
public let NSSharingServiceNameSendViaAirDrop: String
public let NSSharingServiceNameAddToSafariReadingList: String
...

open class NSSharingService : NSObject {
    ...
    public init?(named serviceName: String)
}
```

String Enumerations and Nested Types

```
// NSSharingService, Swift 3

public let NSSharingServiceNameComposeEmail: String
public let NSSharingServiceNameComposeMessage: String
public let NSSharingServiceNameSendViaAirDrop: String
public let NSSharingServiceNameAddToSafariReadingList: String
...

open class NSSharingService : NSObject {
    ...
    public init?(named serviceName: String)
}
```

String Enumerations and Nested Types

```
// NSSharingService, Swift 3

public let NSSharingServiceNameComposeEmail: String
public let NSSharingServiceNameComposeMessage: String
public let NSSharingServiceNameSendViaAirDrop: String
public let NSSharingServiceNameAddToSafariReadingList: String
...

open class NSSharingService : NSObject {
    ...
    public init?(named serviceName: String)
}
```

```swift
// NSSharingService, Swift 4

open class NSSharingService : NSObject {
    ...
    public struct Name : RawRepresentable, Equatable, Hashable { … }
    ...
    public init?(named serviceName: NSSharingService.Name)
}

extension NSSharingService.Name {
    public static let composeEmail: NSSharingService.Name
    public static let composeMessage: NSSharingService.Name
    public static let sendViaAirDrop: NSSharingService.Name
    public static let addToSafariReadingList: NSSharingService.Name
    ...
}
```

```swift
// NSSharingService, Swift 4

open class NSSharingService : NSObject {
    ...
    public struct Name : RawRepresentable, Equatable, Hashable { … }
    ...
    public init?(named serviceName: NSSharingService.Name)
}

extension NSSharingService.Name {
    public static let composeEmail: NSSharingService.Name
    public static let composeMessage: NSSharingService.Name
    public static let sendViaAirDrop: NSSharingService.Name
    public static let addToSafariReadingList: NSSharingService.Name
    ...
}
```

```
// NSSharingService, Swift 4

open class NSSharingService : NSObject {
    ...
    public struct Name : RawRepresentable, Equatable, Hashable { … }
    ...
    public init?(named serviceName: NSSharingService.Name)
}

extension NSSharingService.Name {
    public static let composeEmail: NSSharingService.Name
    public static let composeMessage: NSSharingService.Name
    public static let sendViaAirDrop: NSSharingService.Name
    public static let addToSafariReadingList: NSSharingService.Name
    ...
}
```

```
// NSSharingService, Swift 4

open class NSSharingService : NSObject {
    ...
    public struct Name : RawRepresentable, Equatable, Hashable { … }
    ...
    public init?(named serviceName: NSSharingService.Name)
}

extension NSSharingService.Name {
    public static let composeEmail: NSSharingService.Name
    public static let composeMessage: NSSharingService.Name
    public static let sendViaAirDrop: NSSharingService.Name
    public static let addToSafariReadingList: NSSharingService.Name
    ...
}
```

```swift
// Swift 3
let s = NSSharingService(named: NSSharingServiceNameComposeEmail)

// Swift 4
let s = NSSharingService(named: .composeEmail)
```

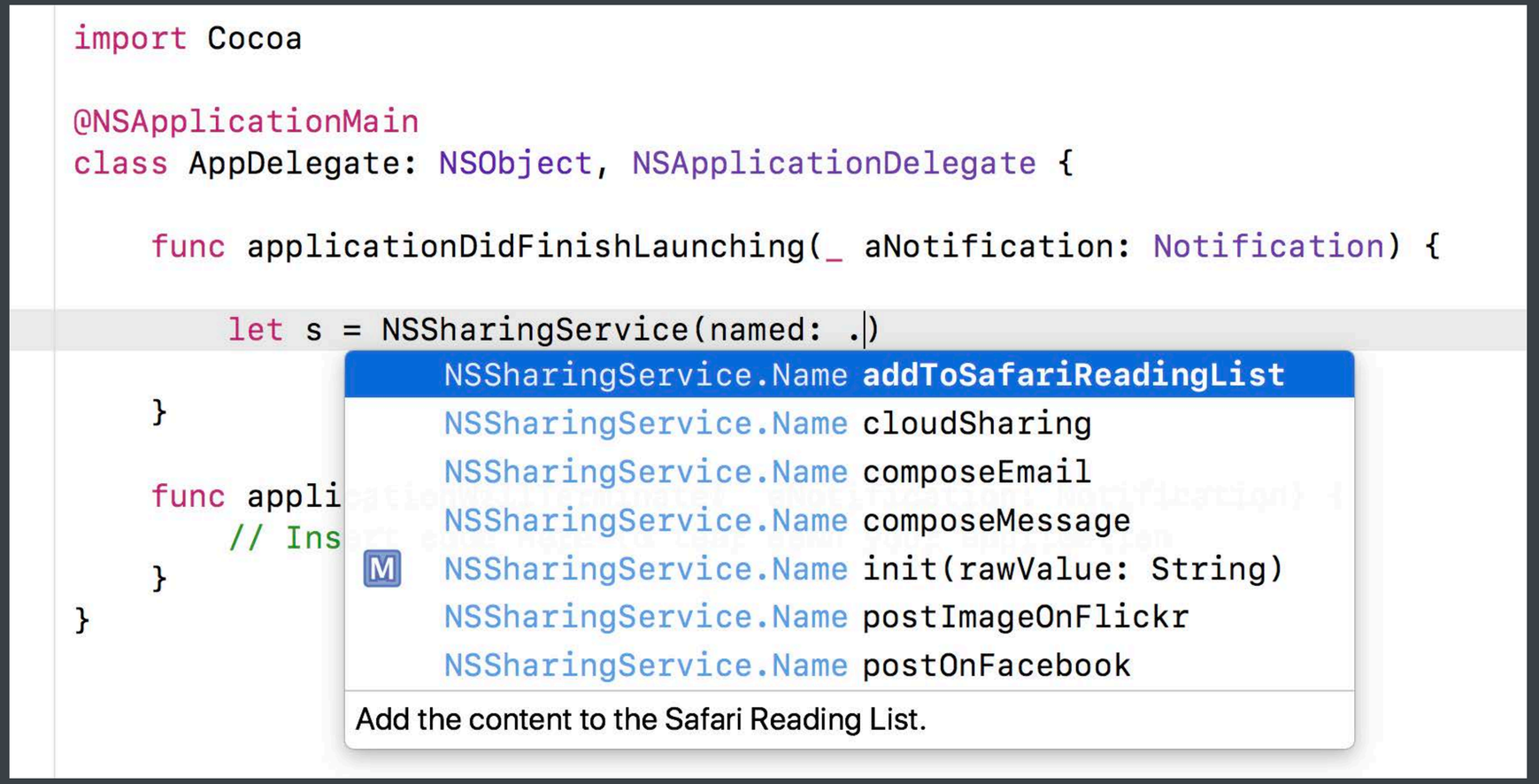
```swift
// Swift 3
let s = NSSharingService(named: NSSharingServiceNameComposeEmail)

// Swift 4
let s = NSSharingService(named: .composeEmail)
```

```swift
// Swift 3
let s = NSSharingService(named: NSSharingServiceNameComposeEmail)

// Swift 4
let s = NSSharingService(named: .composeEmail)
```

```swift
// Swift 3
let s = NSSharingService(named: NSSharingServiceNameComposeEmail)

// Swift 4
let s = NSSharingService(named: .composeEmail)
```

```swift
import Cocoa

@NSApplicationMain
class AppDelegate: NSObject, NSApplicationDelegate {

    func applicationDidFinishLaunching(_ aNotification: Notification) {

        let s = NSSharingService(named: )

    }

    func applicationWillTerminate(_ aNotification: Notification) {
        // Insert code here to tear down your application
    }
}
```

```swift
// Swift 3
let s = NSSharingService(named: NSSharingServiceNameComposeEmail)

// Swift 4
let s = NSSharingService(named: .composeEmail)
```



Source Compatibility

Objective-C

Source Compatibility

Objective-C

```
typedef NSString *NSSharingServiceName NS_EXTENSIBLE_STRING_ENUM;

APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameComposeEmail;
APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameComposeMessage;
APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameSendViaAirDrop;
APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameAddToSafariReadingList;
...
```

Source Compatibility

Objective-C

```
typedef NSString *NSSharingServiceName NS_EXTENSIBLE_STRING_ENUM;

APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameComposeEmail;
APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameComposeMessage;
APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameSendViaAirDrop;
APPKIT_EXTERN NSSharingServiceName const NSSharingServiceNameAddToSafariReadingList;
...
```

Source compatible

Source Compatibility

Swift

Source Compatibility

Swift

```
// Swift 3
public let NSSharingServiceNameComposeEmail: String
...
```

Source Compatibility

Swift

```
// Swift 3
public let NSSharingServiceNameComposeEmail: String
...
```

```
// Swift 4
public static let composeEmail: NSSharingService.Name
...
```

Source Compatibility

Swift

```
// Swift 3
public let NSSharingServiceNameComposeEmail: String
...
```

```
// Swift 4
public static let composeEmail: NSSharingService.Name
...
```

Source compatible in Swift 3 mode

Source Compatibility

Swift

```
// Swift 3
public let NSSharingServiceNameComposeEmail: String
...
```

```
// Swift 4
public static let composeEmail: NSSharingService.Name
...
```

Source compatible in Swift 3 mode

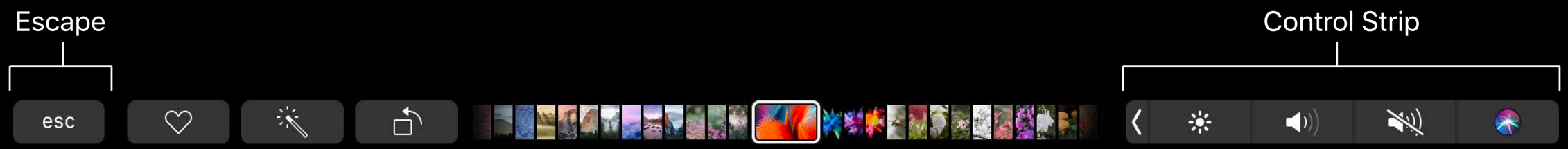
Migrator for moving to Swift 4

AppKit

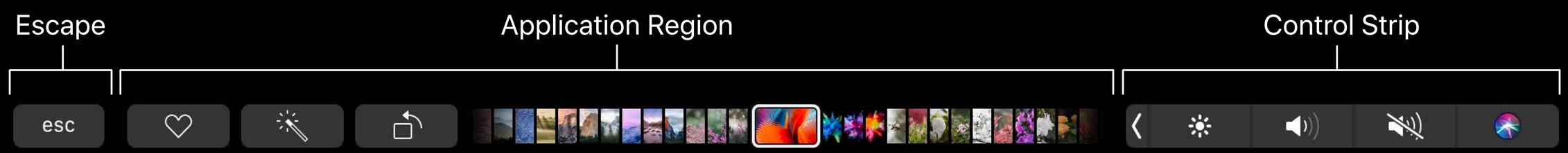

MacBook Pro
esc
delete
tab
caps lock
enter
return
shift
shift
fn

Touch Bar

Touch Bar

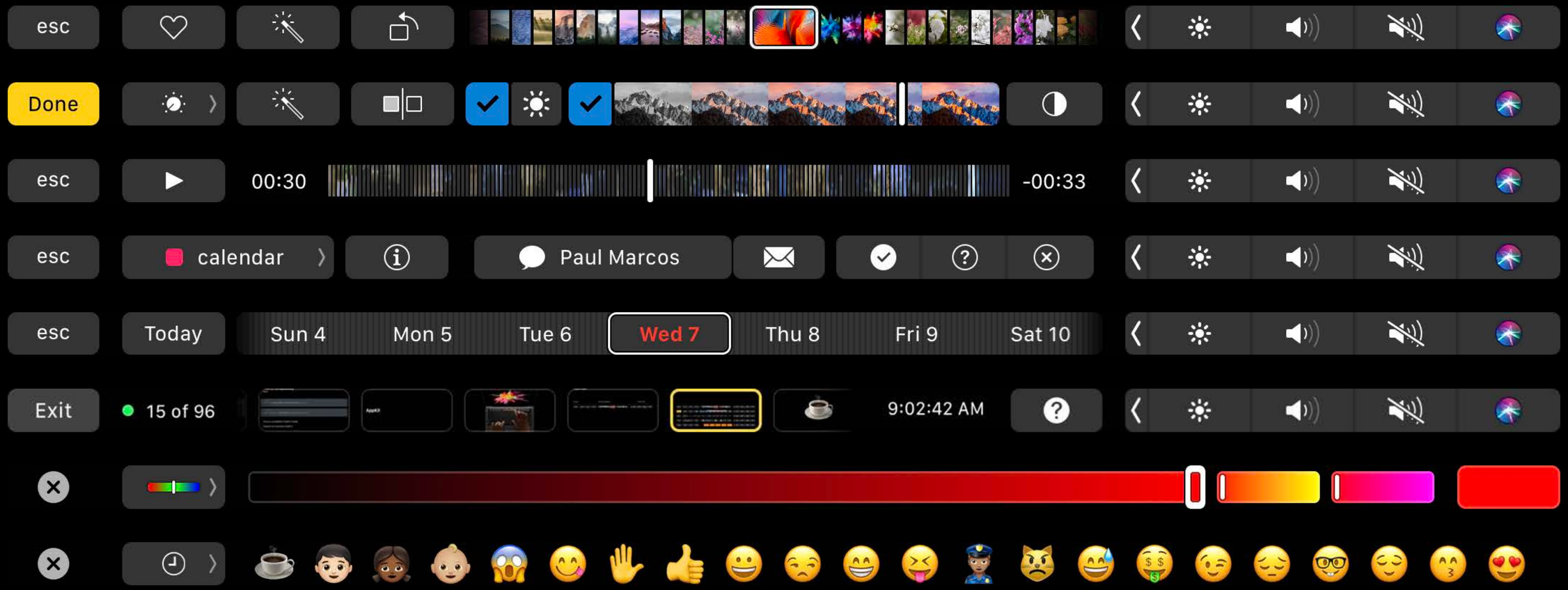


Touch Bar



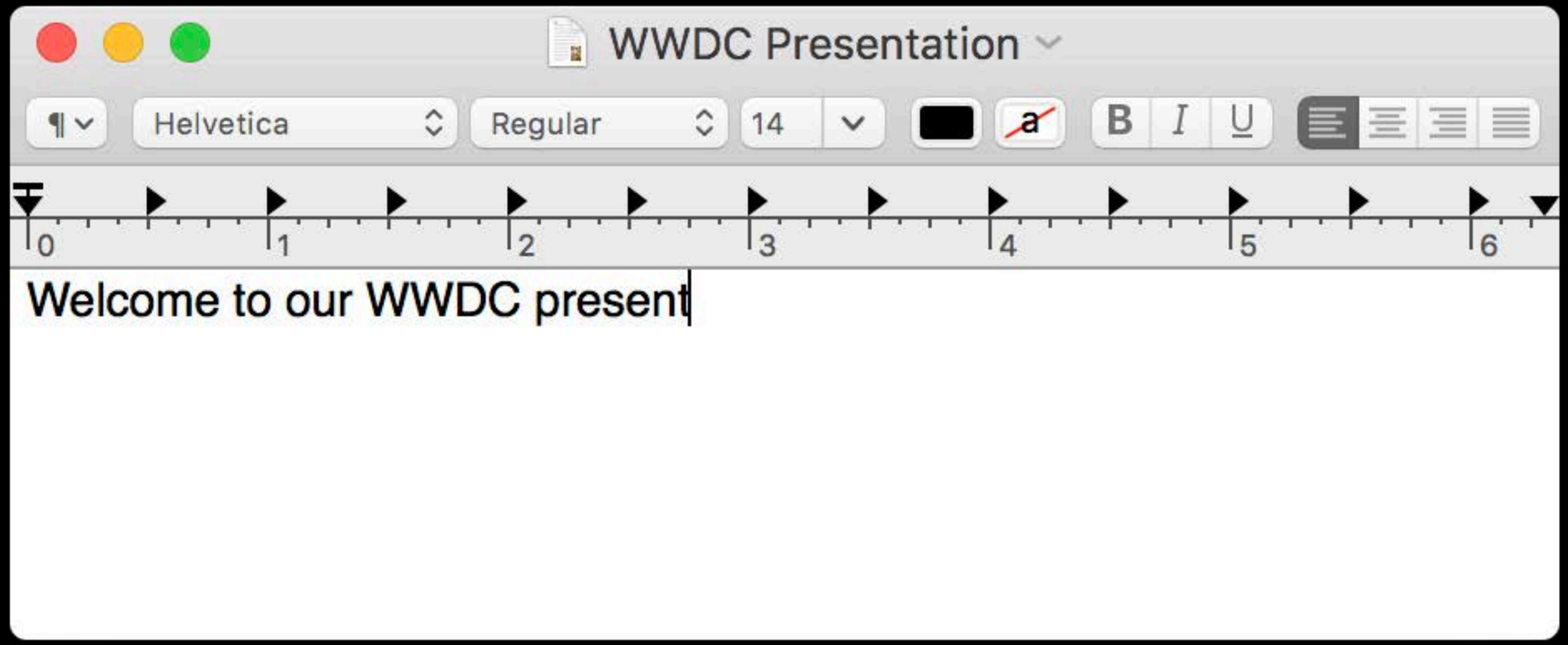
Multi-Touch Input Device with Retina Display

Multi-Touch Input Device with Retina Display

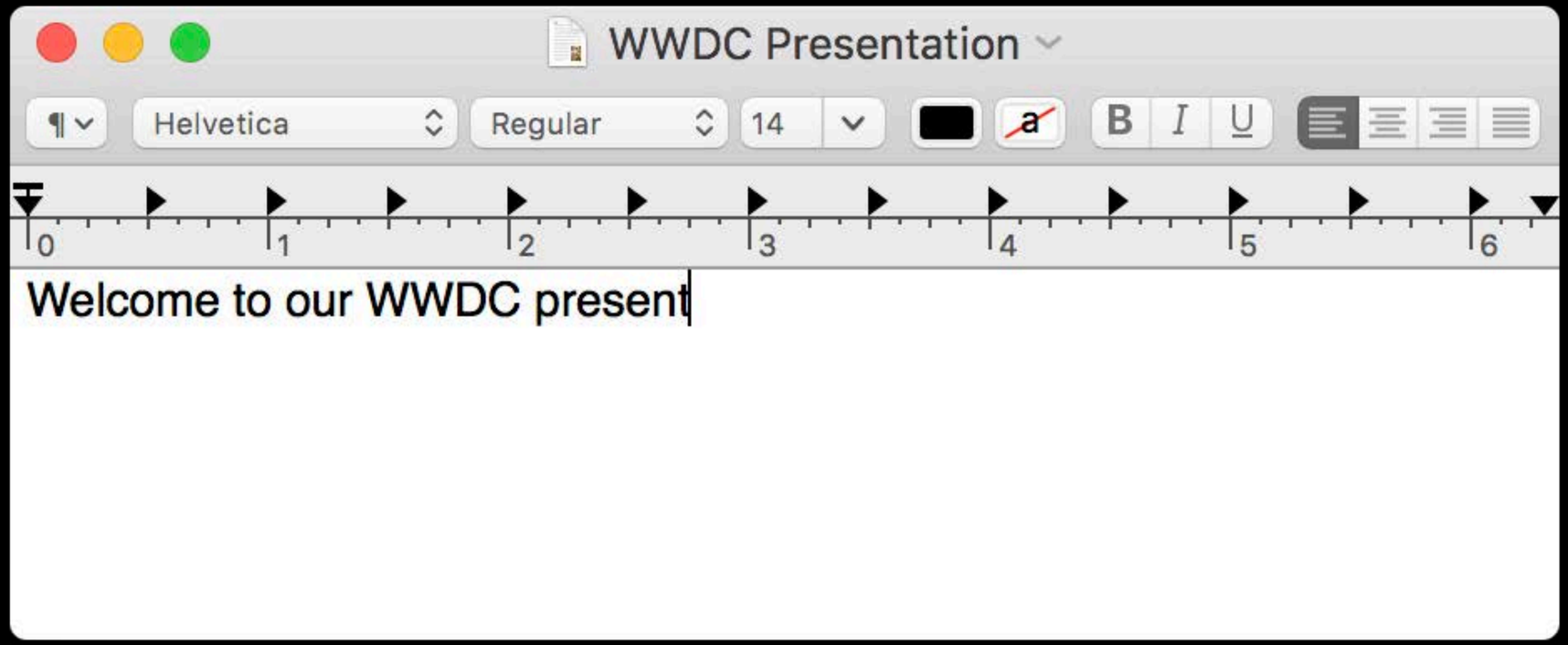


Context Sensitive

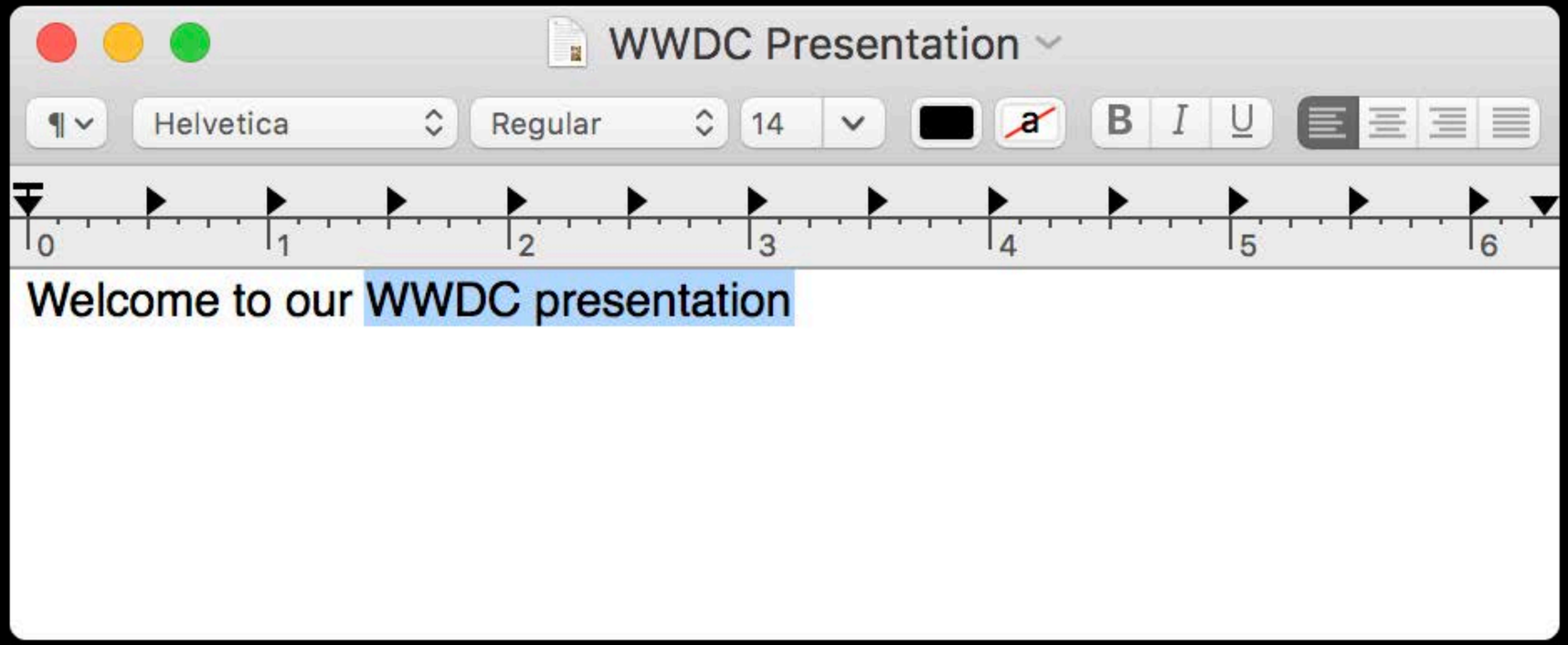
Context Sensitive



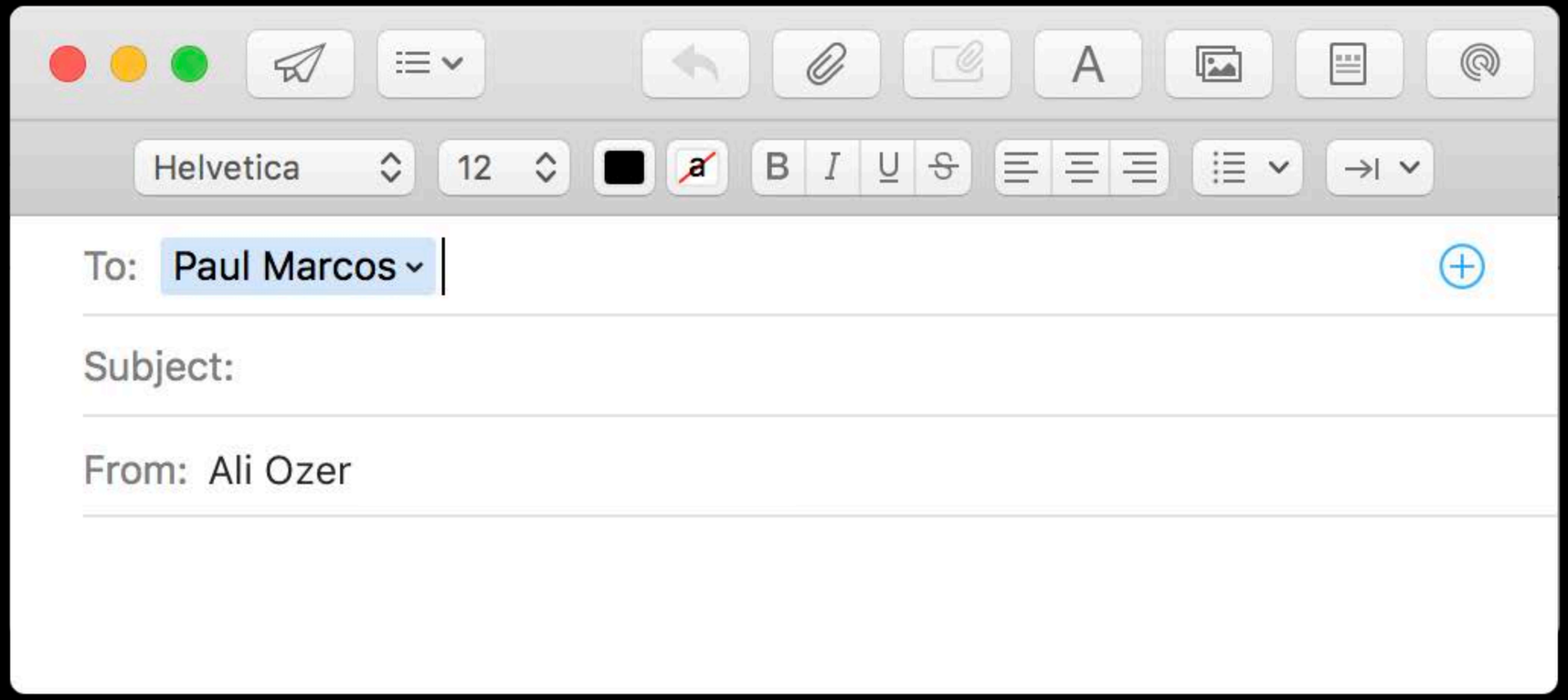
Context Sensitive



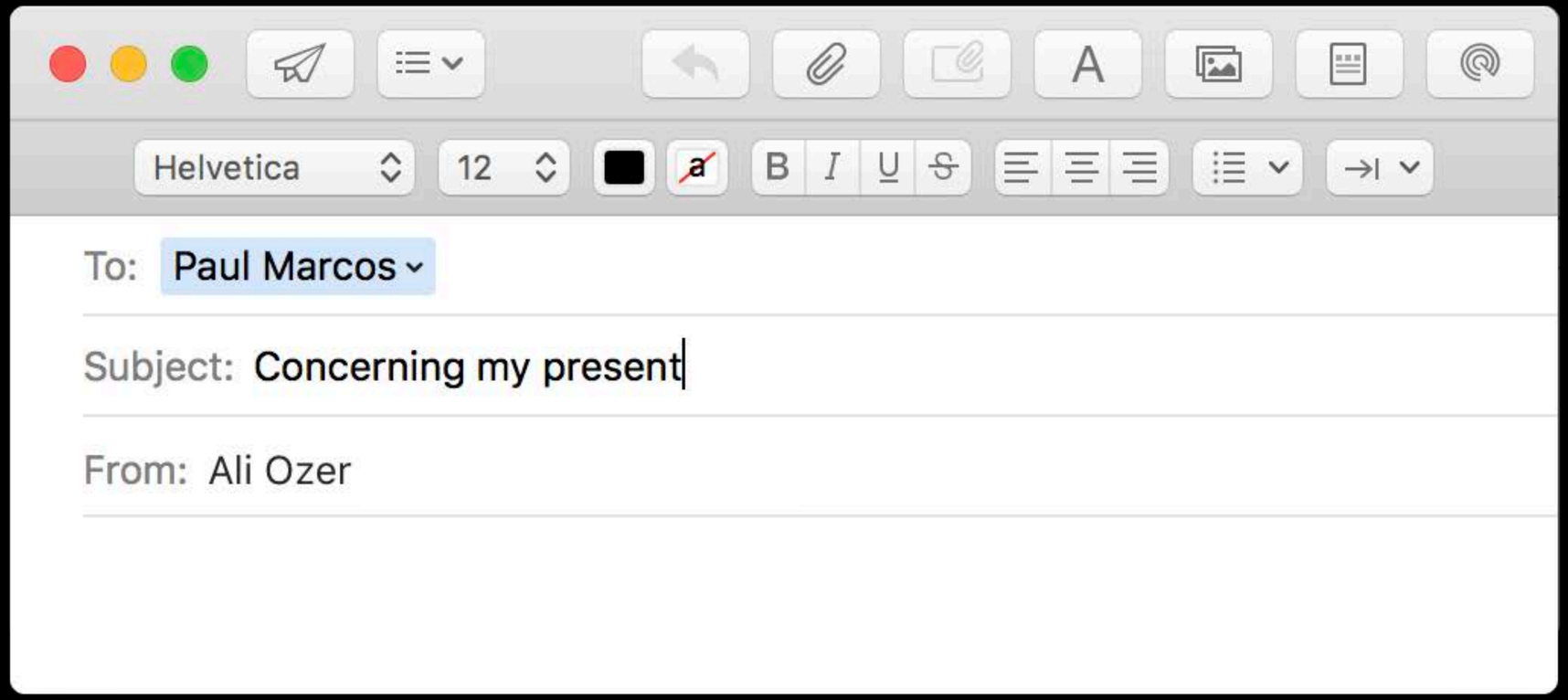
Context Sensitive



Context Sensitive



Context Sensitive



Customizable

Customizable

Customizable

Customizable

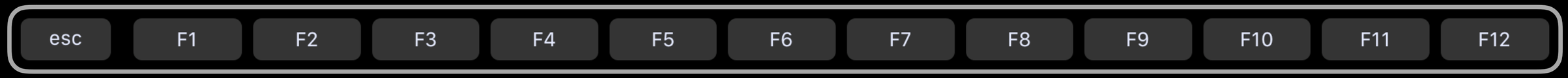
Customizable

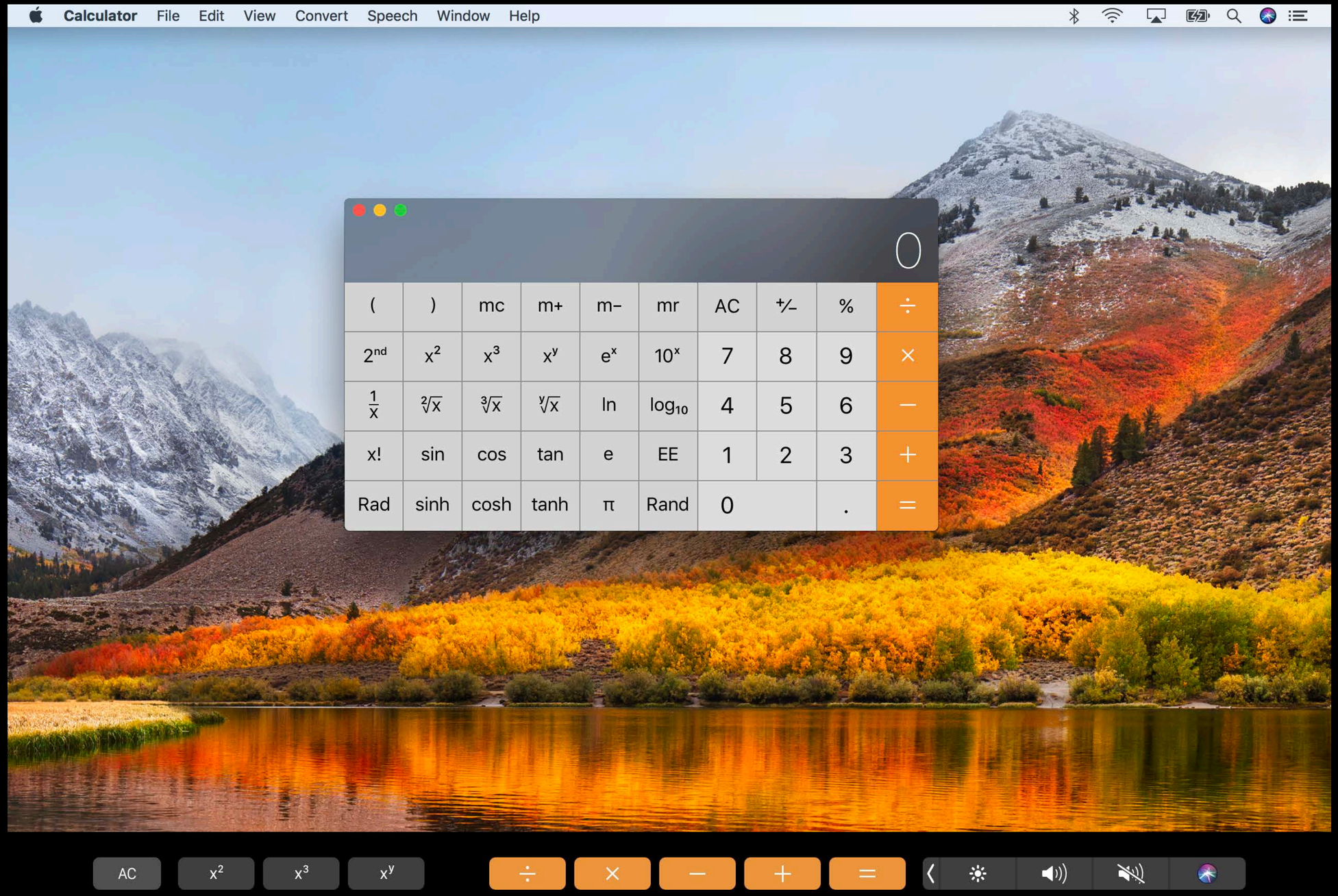
Customizable

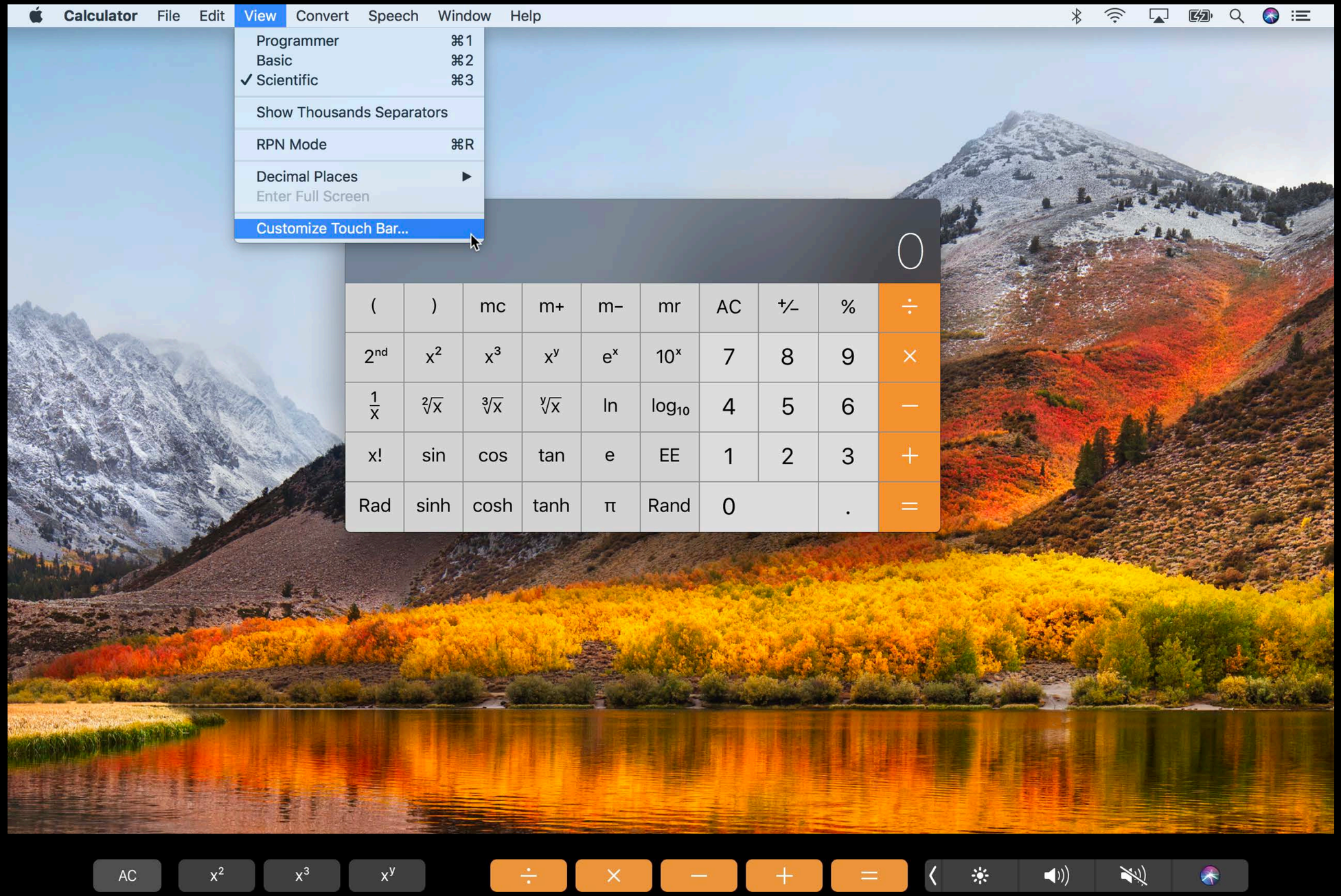
Customizable

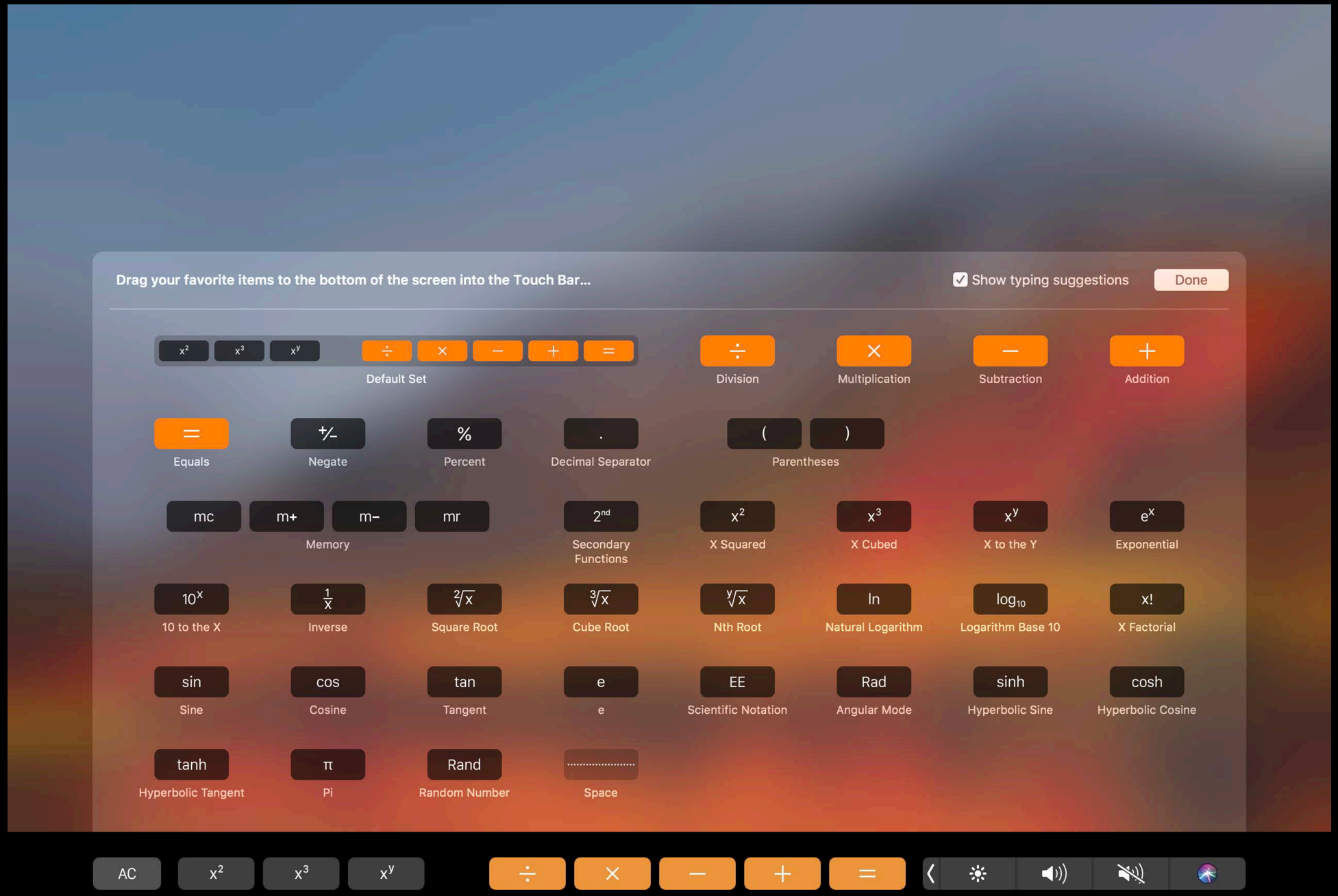
Customizable

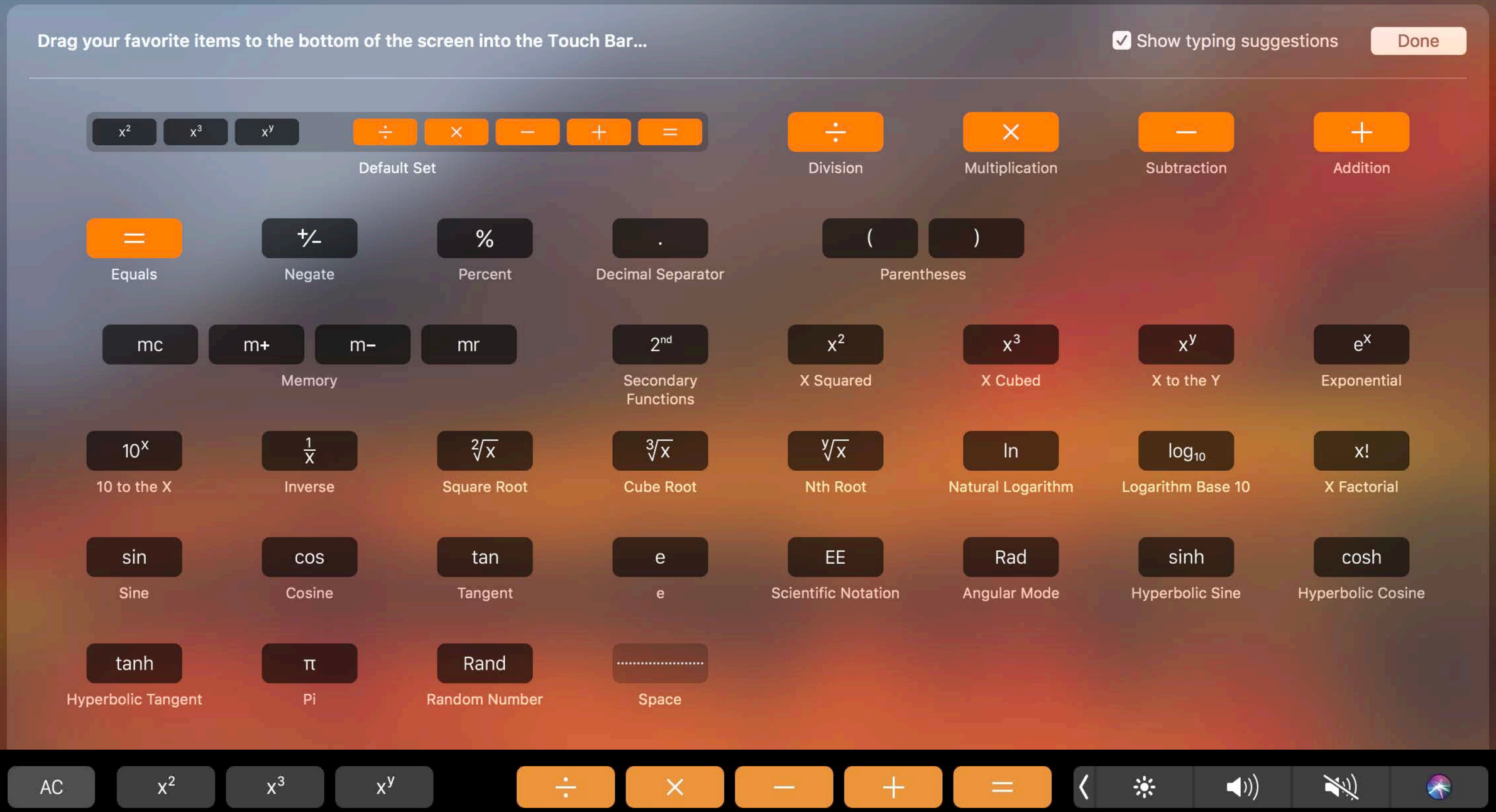
Customizable




Calculator File Edit View Convert Speech Window Help
0
( ) mc m+ m− mr AC +/− % ÷
2nd x² x³ xʸ eˣ 10ˣ 7 8 9 ×
1/x ²√x ³√x ʸ√x ln log₁₀ 4 5 6 −
x! sin cos tan e EE 1 2 3 +
Rad sinh cosh tanh π Rand 0 . =
AC x² x³ xʸ ÷ × − + =


Calculator File Edit View Convert Speech Window Help
Programmer ⌘1
Basic ⌘2
✓ Scientific ⌘3
Show Thousands Separators
RPN Mode ⌘R
Decimal Places ▶
Enter Full Screen
Customize Touch Bar...
0
( ) mc m+ m− mr AC +/− % ÷
2nd x² x³ xʸ eˣ 10ˣ 7 8 9 ×
1/x ²√x ³√x ʸ√x ln log₁₀ 4 5 6 −
x! sin cos tan e EE 1 2 3 +
Rad sinh cosh tanh π Rand 0 . =
AC x² x³ xʸ ÷ × − + =


Drag your favorite items to the bottom of the screen into the Touch Bar...
Show typing suggestions
Done
Default Set
Division
Multiplication
Subtraction
Addition
Equals
Negate
Percent
Decimal Separator
Parentheses
Memory
Secondary Functions
X Squared
X Cubed
X to the Y
Exponential
10 to the X
Inverse
Square Root
Cube Root
Nth Root
Natural Logarithm
Logarithm Base 10
X Factorial
Sine
Cosine
Tangent
e
Scientific Notation
Angular Mode
Hyperbolic Sine
Hyperbolic Cosine
Hyperbolic Tangent
Pi
Random Number
Space
AC


Drag your favorite items to the bottom of the screen into the Touch Bar...
Show typing suggestions
Done
Default Set
Division
Multiplication
Subtraction
Addition
Equals
Negate
Percent
Decimal Separator
Parentheses
Memory
Secondary Functions
X Squared
X Cubed
X to the Y
Exponential
10 to the X
Inverse
Square Root
Cube Root
Nth Root
Natural Logarithm
Logarithm Base 10
X Factorial
Sine
Cosine
Tangent
e
Scientific Notation
Angular Mode
Hyperbolic Sine
Hyperbolic Cosine
Hyperbolic Tangent
Pi
Random Number
Space
AC

Unobstructed Access to Content

Unobstructed Access to Content

Touch Bar Updates

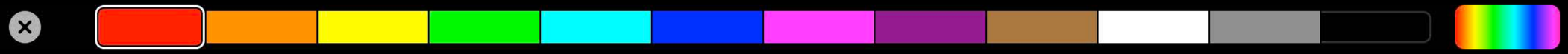
NSColorPickerTouchBarItem

NSGroupTouchBarItem

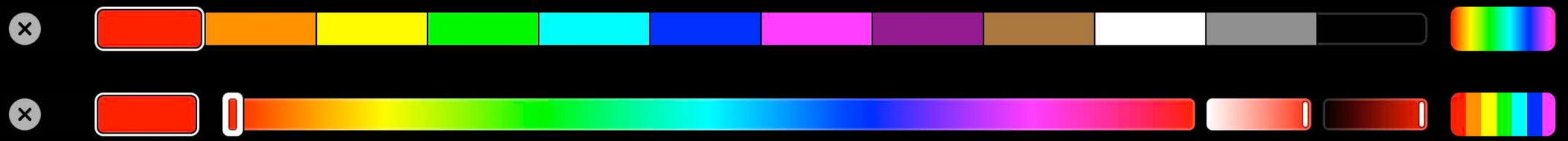
Playground

Touch Bar Color Pickers

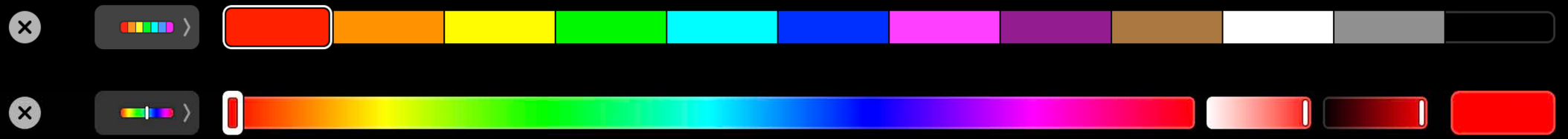
Touch Bar Color Pickers



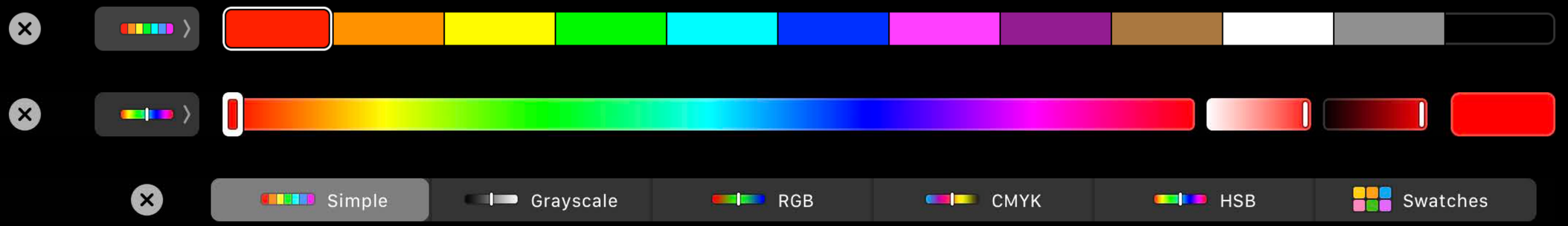
Touch Bar Color Pickers



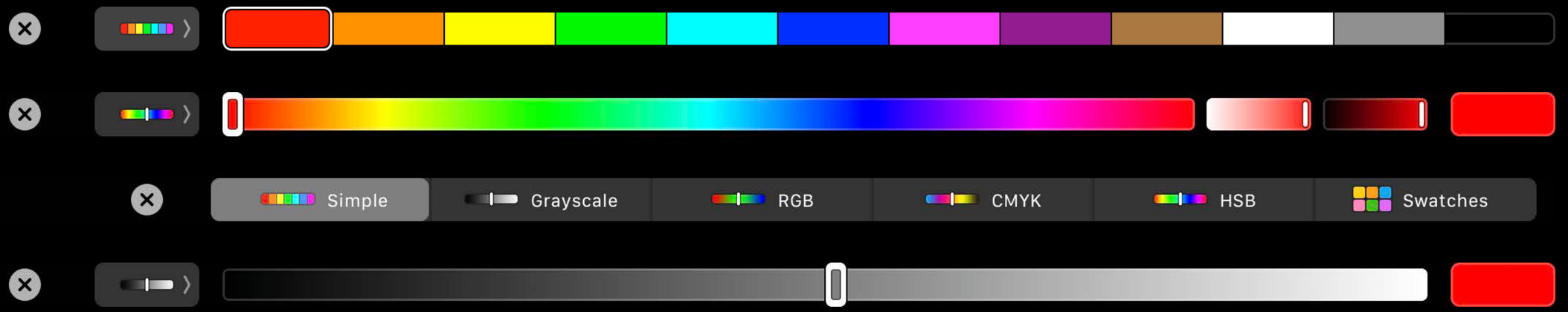
New Color Pickers



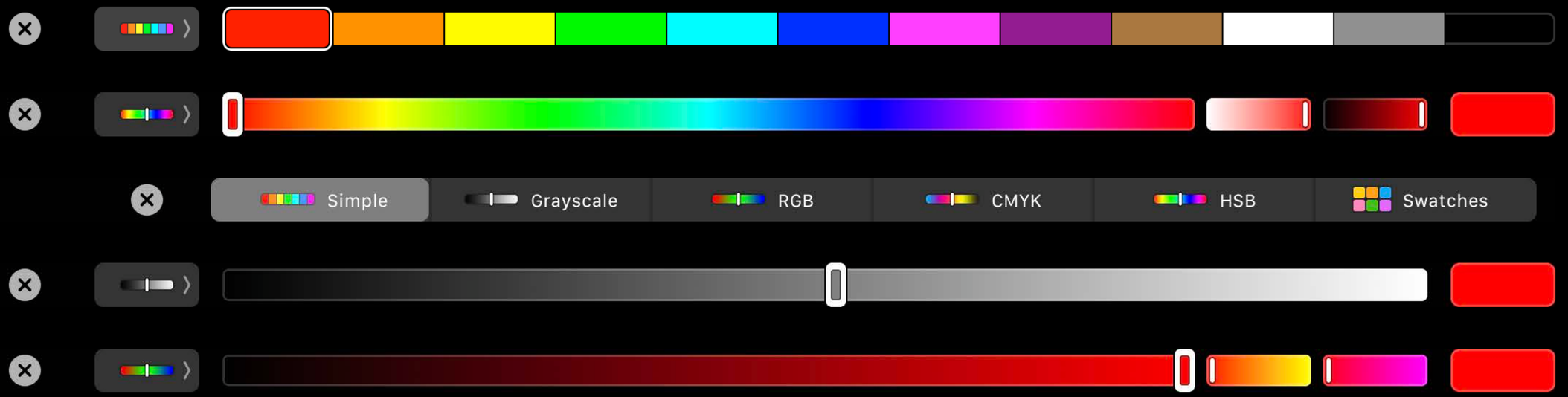
New Color Pickers



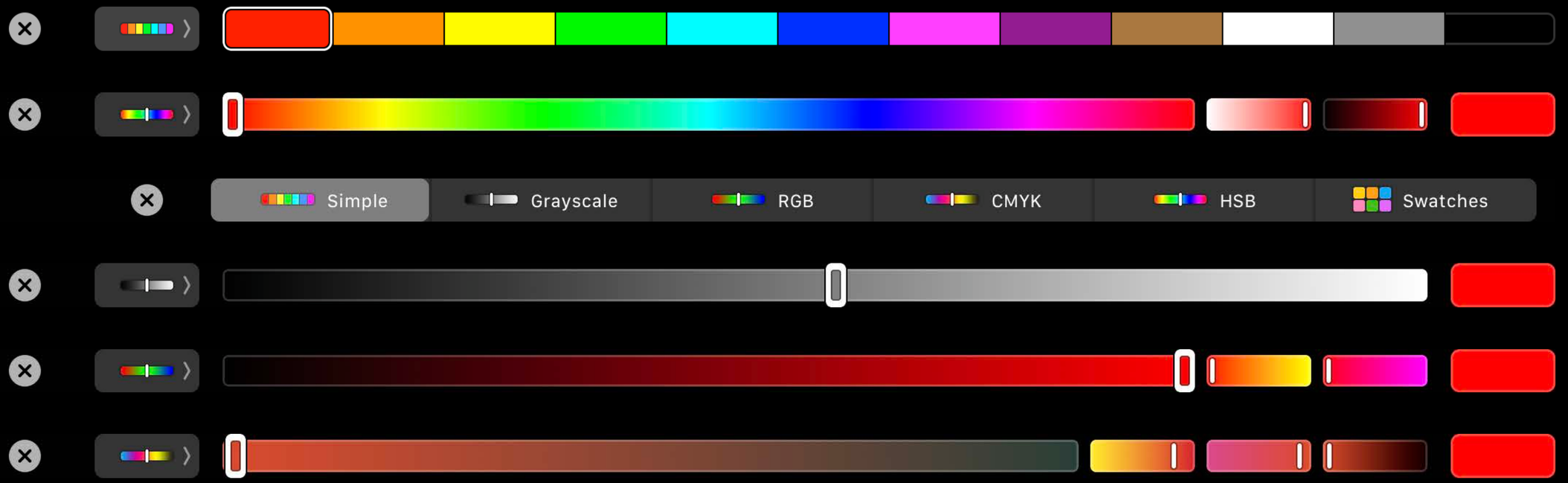
New Color Pickers



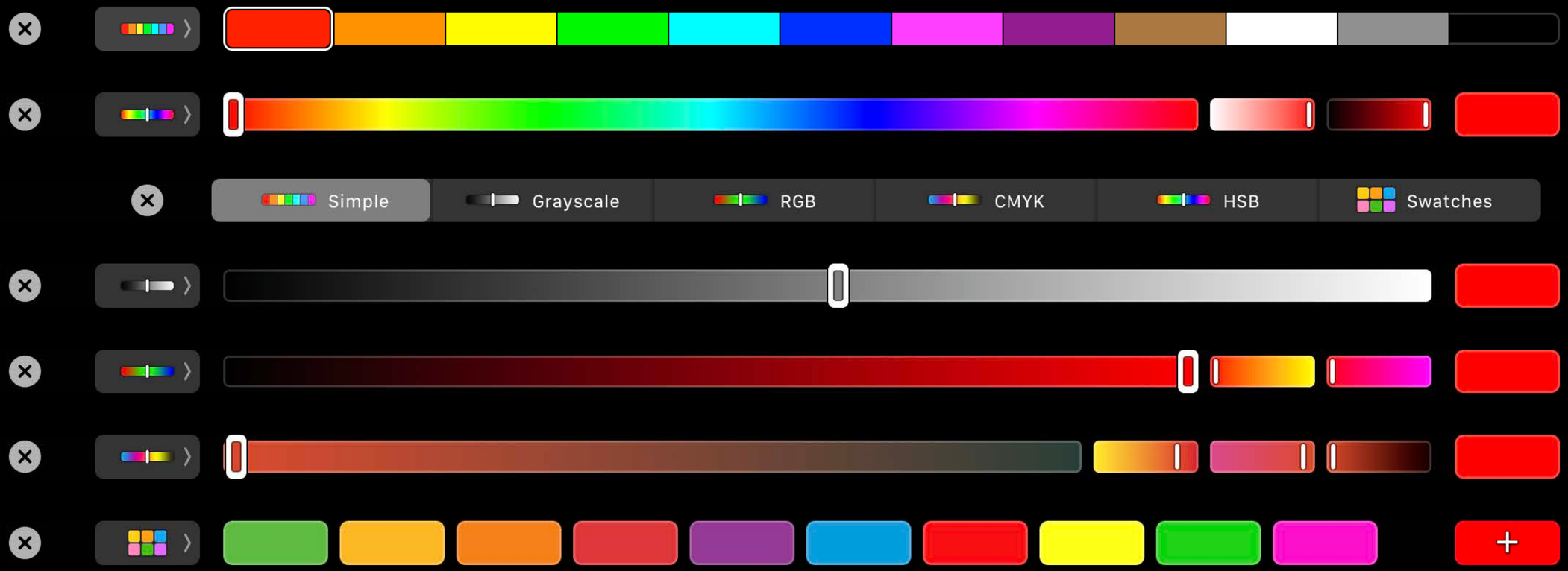
New Color Pickers



New Color Pickers

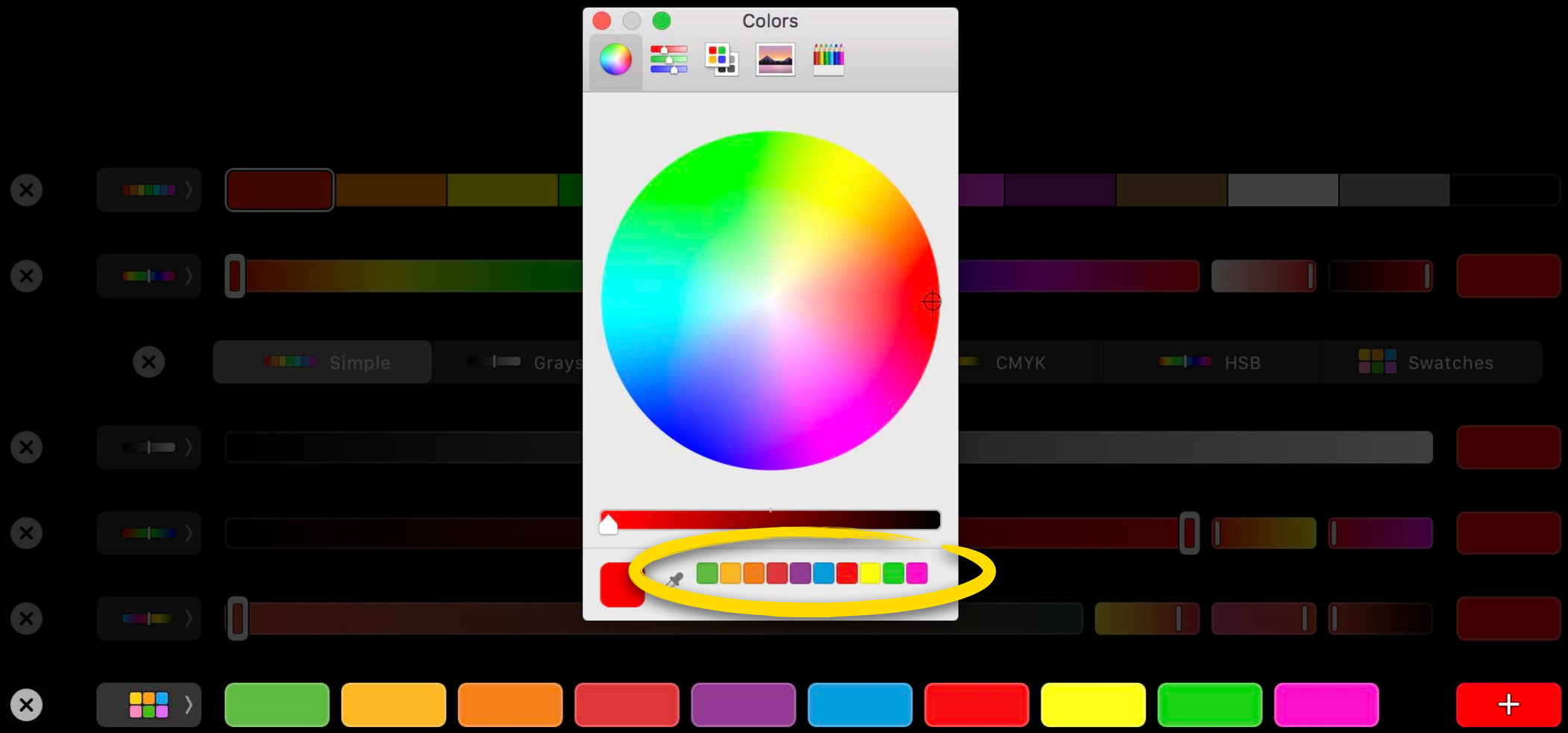


New Color Pickers

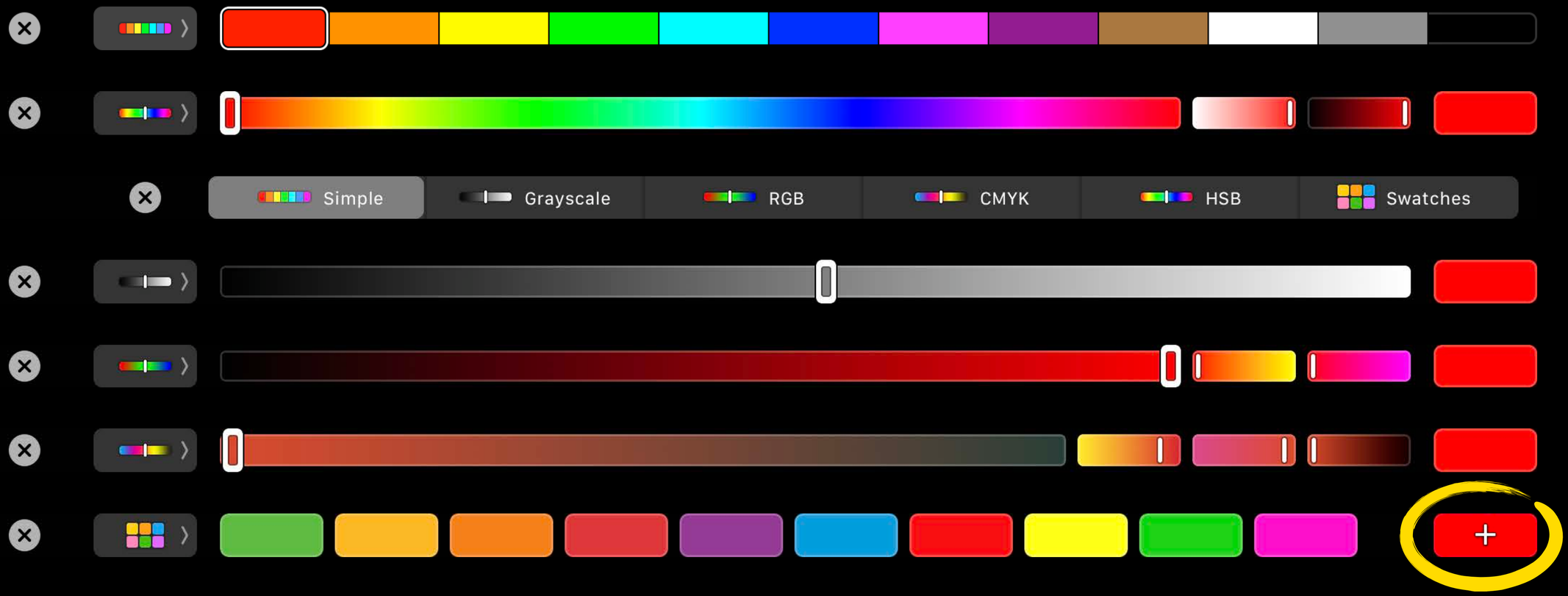


New Color Pickers

New Color Pickers



New Color Pickers



NSColorPickerTouchBarItem

NSColorPickerTouchBarItem

```
open class NSColorPickerTouchBarItem : NSTouchBarItem {
    ...
    open var allowedColorSpaces: [NSColorSpace]?
}
```

NSColorPickerTouchBarItem

NSColorPickerTouchBarItem

```
colorPicker.allowedColorSpaces = [.genericCMYK]
```

NSColorPickerTouchBarItem

```
colorPicker.allowedColorSpaces = [.genericCMYK]
```

```
colorPicker.allowedColorSpaces = [.genericGamma22Gray]
```

NSGroupTouchBarItem

Constructor for “alert” style

NSGroupTouchBarItem

Constructor for “alert” style

NSGroupTouchBarItem

Constructor for “alert” style

NSGroupTouchBarItem

Constructor for “alert” style

```
open class NSGroupTouchBarItem ... {
    ...
    public init(alertStyleGroupItemWithIdentifier: NSTouchBarItem.Identifier)
}
```

NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

NSGroupTouchBarItem

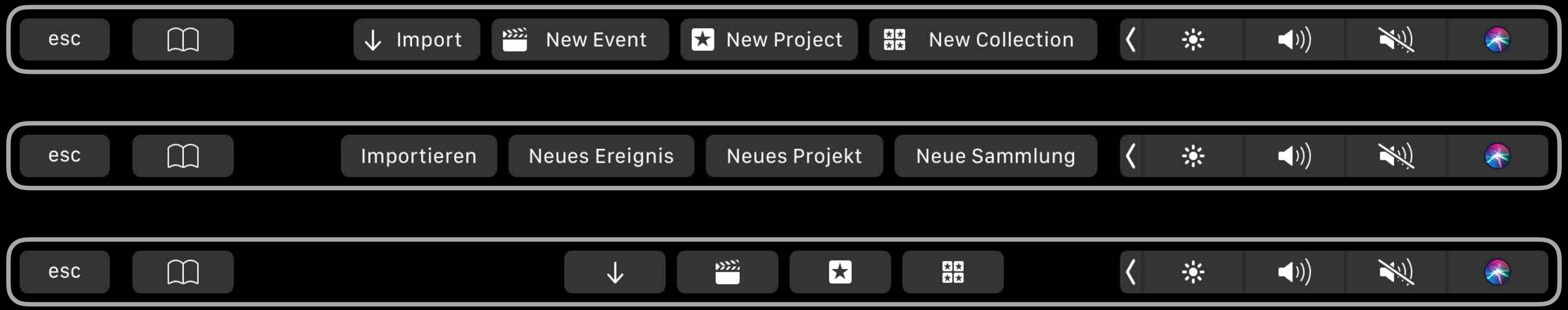
Constructor for “alert” style

Ability to specify compression options

NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options



NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

```
open class NSGroupTouchBarItem ... {
    ...
    open var prioritizedCompressionOptions: [NSUserInterfaceCompressionOptions]
}
```

NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

Support for right-to-left interfaces

NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

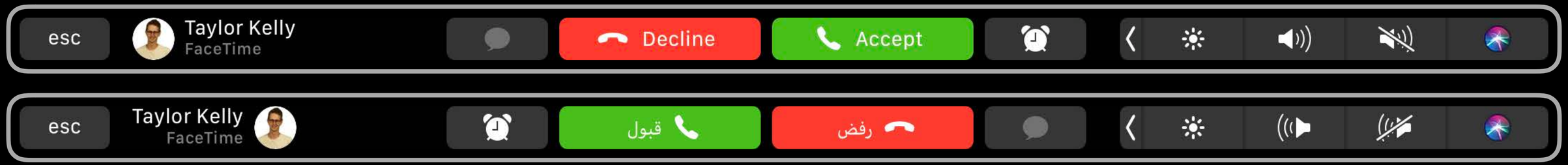
Support for right-to-left interfaces

NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

Support for right-to-left interfaces

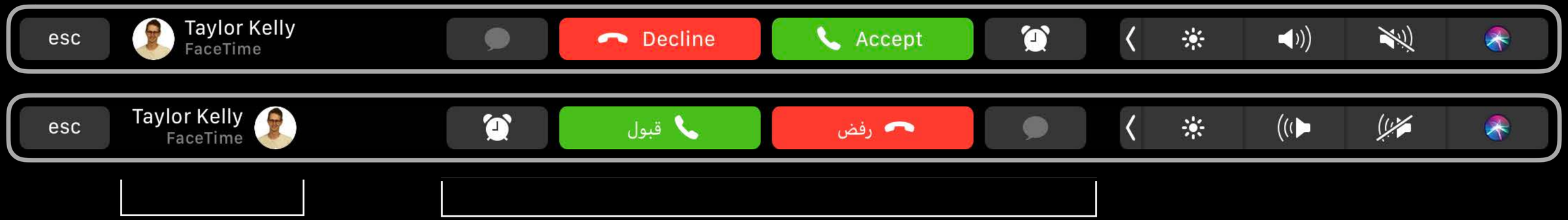


NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

Support for right-to-left interfaces



NSGroupTouchBarItem

Constructor for “alert” style

Ability to specify compression options

Support for right-to-left interfaces

```
open class NSGroupTouchBarItem ... {
    ...
    open var groupUserInterfaceLayoutDirection: NSUserInterfaceLayoutDirection
}
```

Playground Support

Running TouchBarPlayground

TouchBarPlayground

```
import Cocoa

let button = NSButton(title: "Apply", target: nil, action: nil)
let buttonItem = NSCustomTouchBarItem(identifier: NSTouchBarItem.Identifier("applyItem")); buttonItem.view = button
buttonItem.customizationLabel = "Apply"
let touchBar = NSTouchBar()
touchBar.templateItems = [buttonItem]
touchBar.defaultItemIdentifiers = [buttonItem.identifier, .otherItemsProxy]
let colorPicker = NSColorPickerTouchBarItem(identifier: NSTouchBarItem.Identifier("myColorPicker"))

touchBar.templateItems = [colorPicker, buttonItem]
touchBar.defaultItemIdentifiers = [colorPicker.identifier, buttonItem.identifier, .otherItemsProxy]
let sliderItem = NSSliderTouchBarItem(identifier: NSTouchBarItem.Identifier("mySlider"))
sliderItem.slider.widthAnchor.constraint(equalToConstant: 100.0).isActive = true
sliderItem.minimumValueAccessory = NSSliderAccessory(image: NSImage(named: .actionTemplate)!)
sliderItem.maximumValueAccessory = NSSliderAccessory(image: NSImage(named: .actionTemplate)!)

touchBar.templateItems = [colorPicker, buttonItem, sliderItem]
touchBar.defaultItemIdentifiers = [colorPicker.identifier, buttonItem.identifier, sliderItem.identifier, .otherItemsProxy]

touchBar.defaultItemIdentifiers = [colorPicker.identifier, .flexibleSpace, buttonItem.identifier, .flexibleSpace, sliderItem.identifier, .otherItemsProxy]

touchBar.principalItemIdentifier = buttonItem.identifier

let doneButton = NSButton(title: "Done", target: nil, action: nil)
doneButton.keyEquivalent = "\r"
let doneButtonItem = NSCustomTouchBarItem(identifier: NSTouchBarItem.Identifier("done")); doneButtonItem.view = doneButton
touchBar.templateItems = [colorPicker, buttonItem, sliderItem, doneButtonItem]
touchBar.escapeKeyReplacementItemIdentifier = NSTouchBarItem.Identifier("done")
```

Running TouchBarPlayground

TouchBarPlayground

```
import Cocoa

let button = NSButton(title: "Apply", target: nil, action: nil)
let buttonItem = NSCustomTouchBarItem(identifier: NSTouchBarItem.Identifier("applyItem")); buttonItem.view = button
buttonItem.customizationLabel = "Apply"
let touchBar = NSTouchBar()
touchBar.templateItems = [buttonItem]
touchBar.defaultItemIdentifiers = [buttonItem.identifier, .otherItemsProxy]
let colorPicker = NSColorPickerTouchBarItem(identifier: NSTouchBarItem.Identifier("myColorPicker"))

touchBar.templateItems = [colorPicker, buttonItem]
touchBar.defaultItemIdentifiers = [colorPicker.identifier, buttonItem.identifier, .otherItemsProxy]
let sliderItem = NSSliderTouchBarItem(identifier: NSTouchBarItem.Identifier("mySlider"))
sliderItem.slider.widthAnchor.constraint(equalToConstant: 100.0).isActive = true
sliderItem.minimumValueAccessory = NSSliderAccessory(image: NSImage(named: .actionTemplate)!)
sliderItem.maximumValueAccessory = NSSliderAccessory(image: NSImage(named: .actionTemplate)!)

touchBar.templateItems = [colorPicker, buttonItem, sliderItem]
touchBar.defaultItemIdentifiers = [colorPicker.identifier, buttonItem.identifier, sliderItem.identifier, .otherItemsProxy]

touchBar.defaultItemIdentifiers = [colorPicker.identifier, .flexibleSpace, buttonItem.identifier, .flexibleSpace, sliderItem.identifier, .otherItemsProxy]

touchBar.principalItemIdentifier = buttonItem.identifier

let doneButton = NSButton(title: "Done", target: nil, action: nil)
doneButton.keyEquivalent = "\r"
let doneButtonItem = NSCustomTouchBarItem(identifier: NSTouchBarItem.Identifier("done")); doneButtonItem.view = doneButton
touchBar.templateItems = [colorPicker, buttonItem, sliderItem, doneButtonItem]
touchBar.escapeKeyReplacementItemIdentifier = NSTouchBarItem.Identifier("done")
```

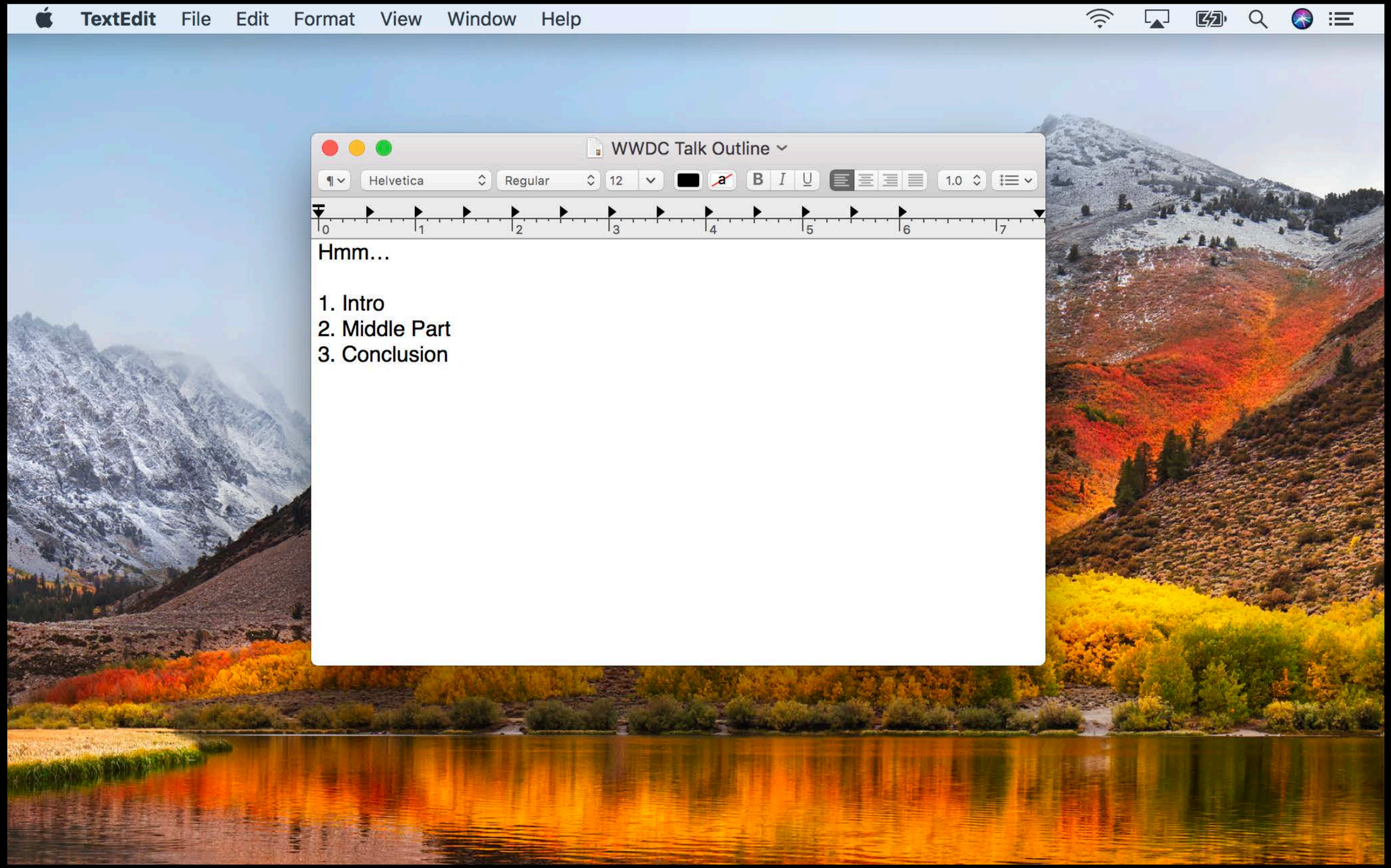
NSTouchBar Sessions and Lab

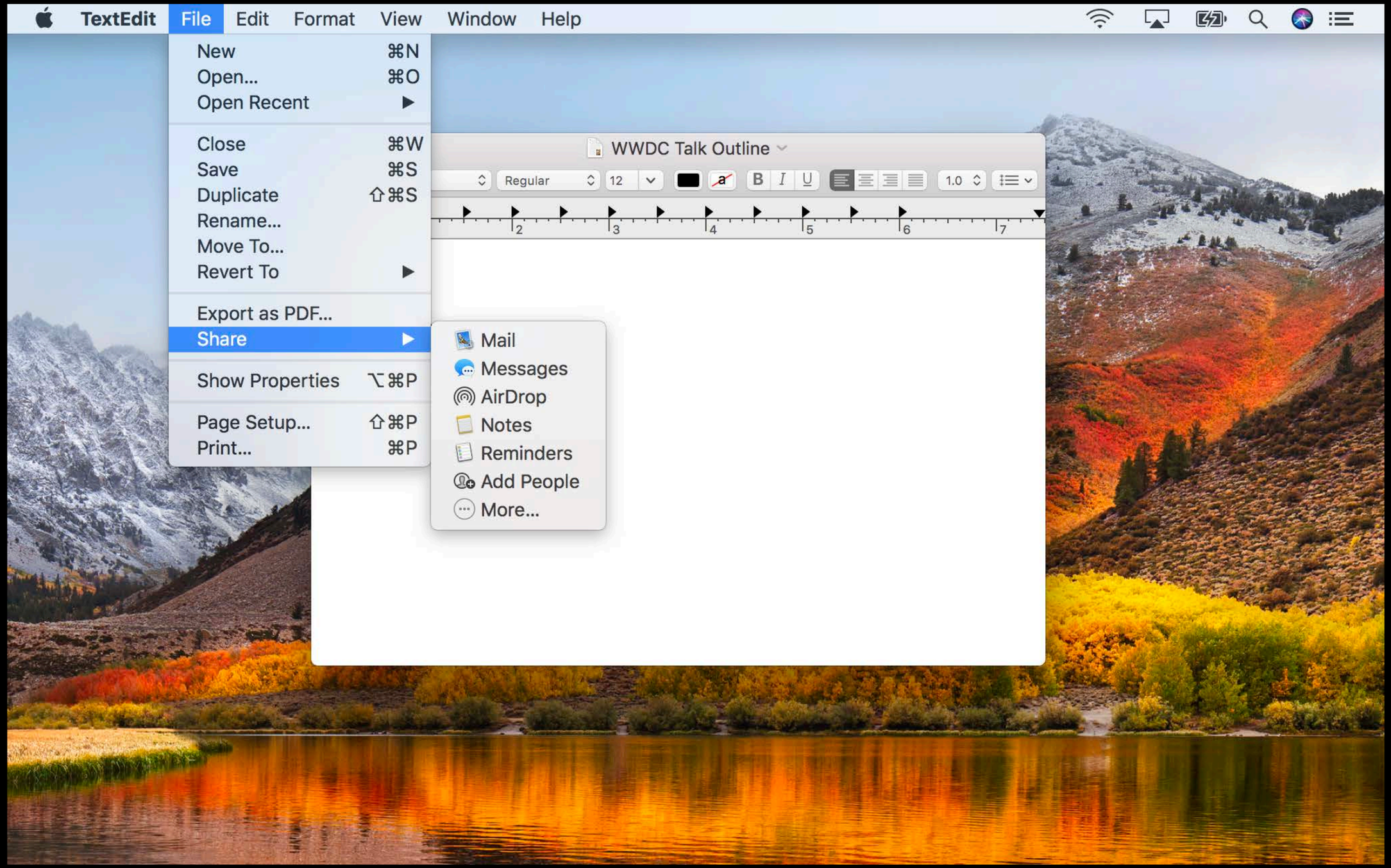
Touch Bar Fundamentals	Grand Ballroom A	Wednesday 10:00AM
Advanced Touch Bar	Grand Ballroom A	Wednesday 5:10PM
Cocoa Touch Bar Lab	Technology Lab C	Thursday 9:00-11:00AM

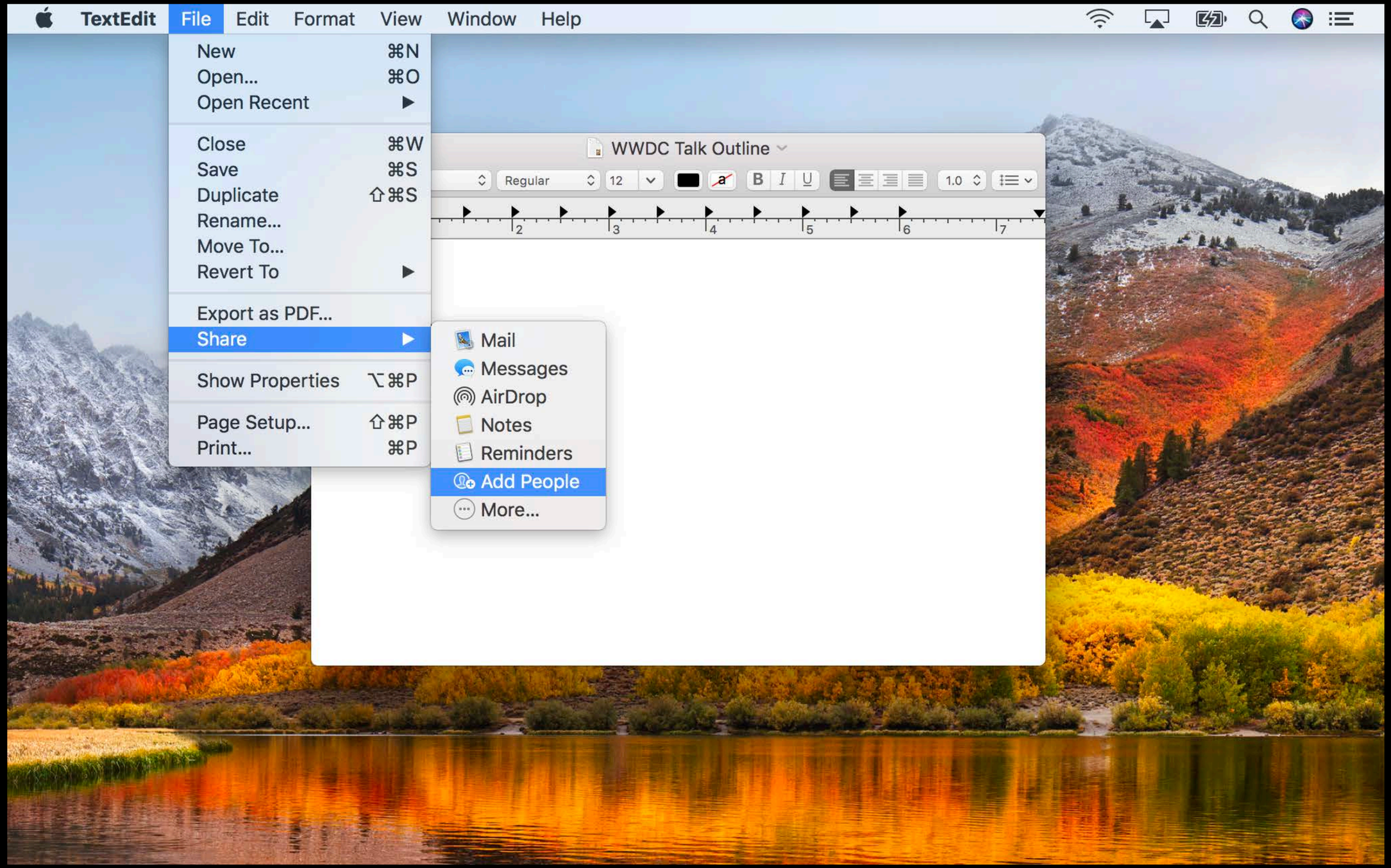
Document Sharing

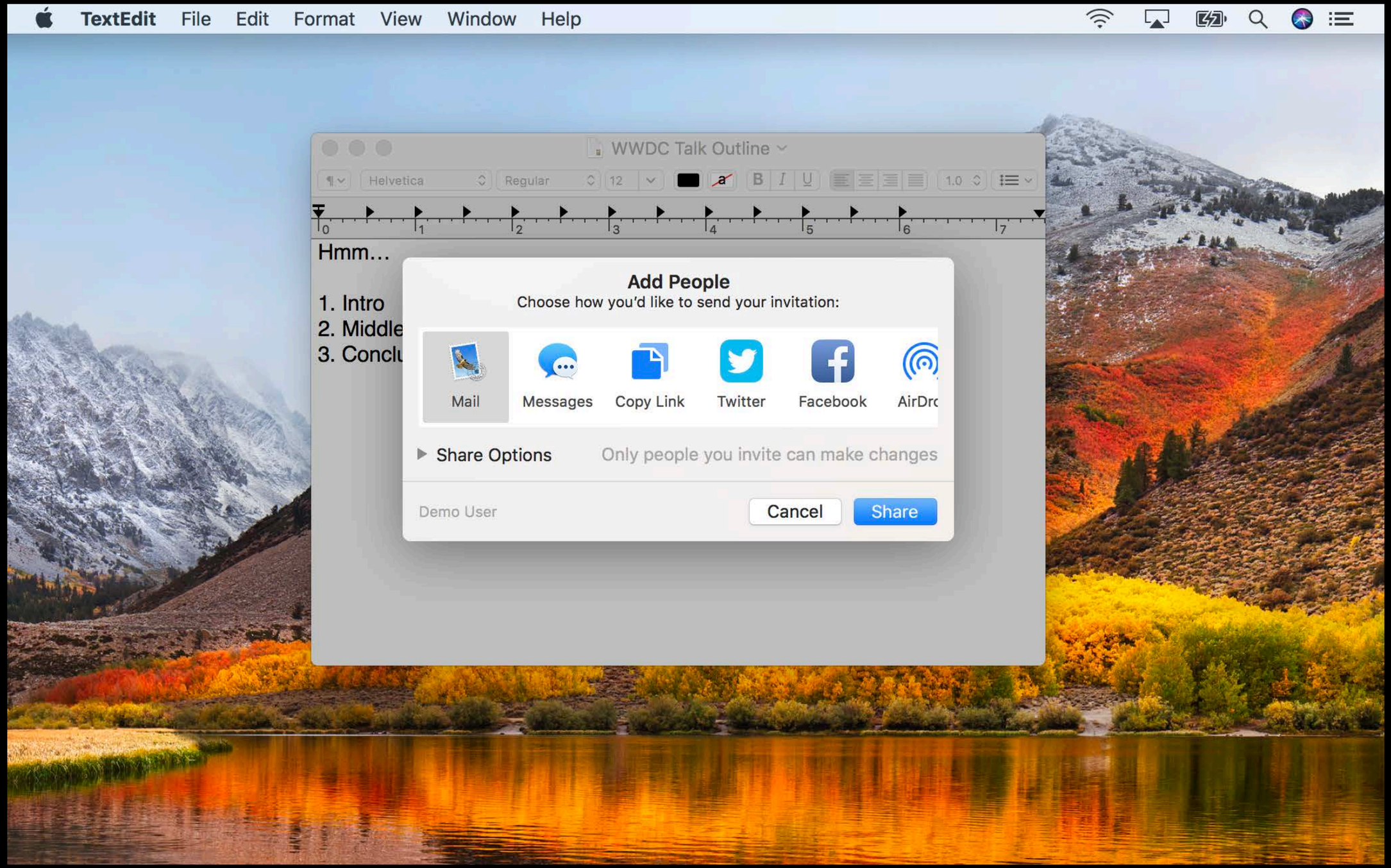
In Sierra, we introduced:

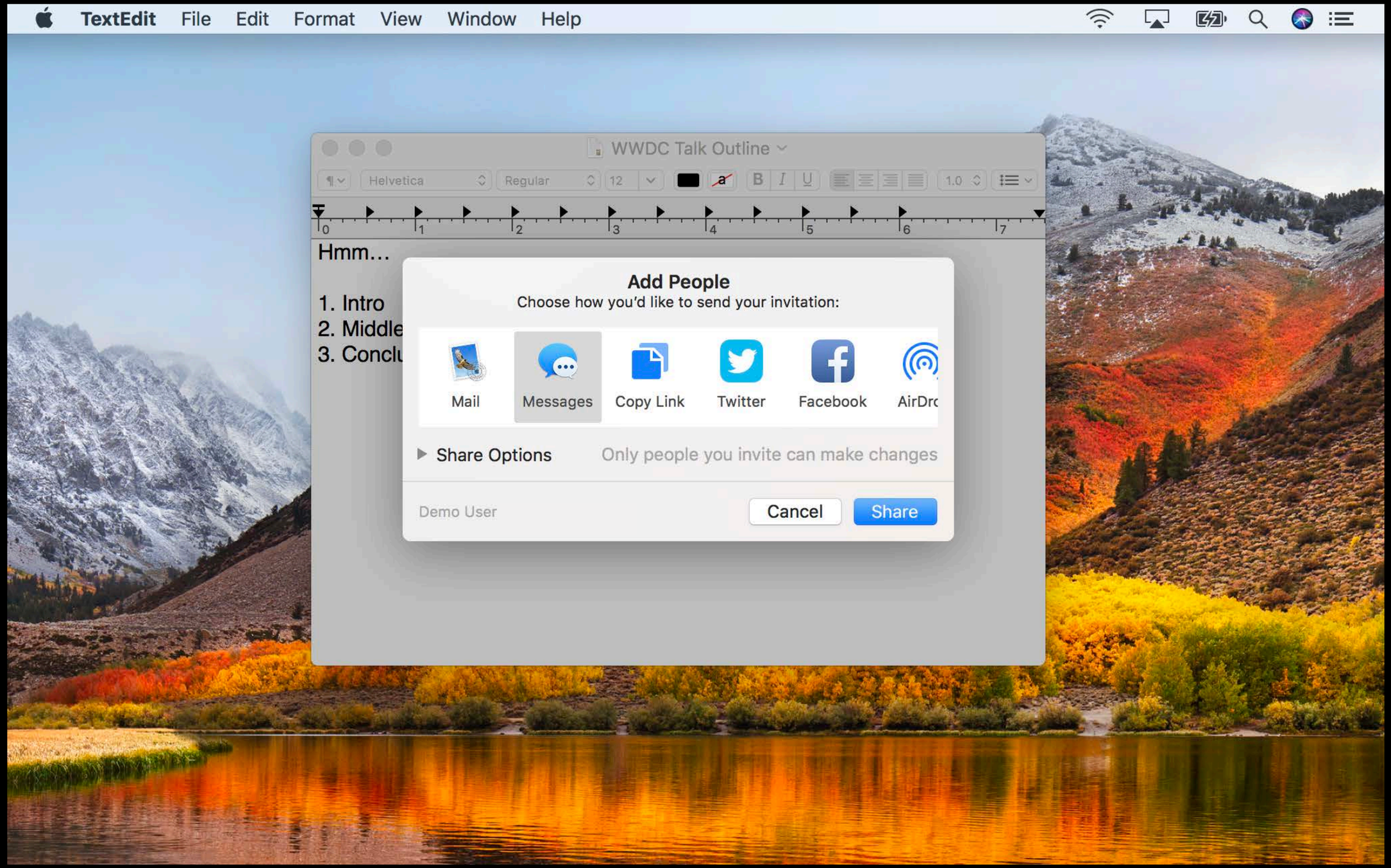
```
public static let cloudSharing: NSSharingService.Name
```

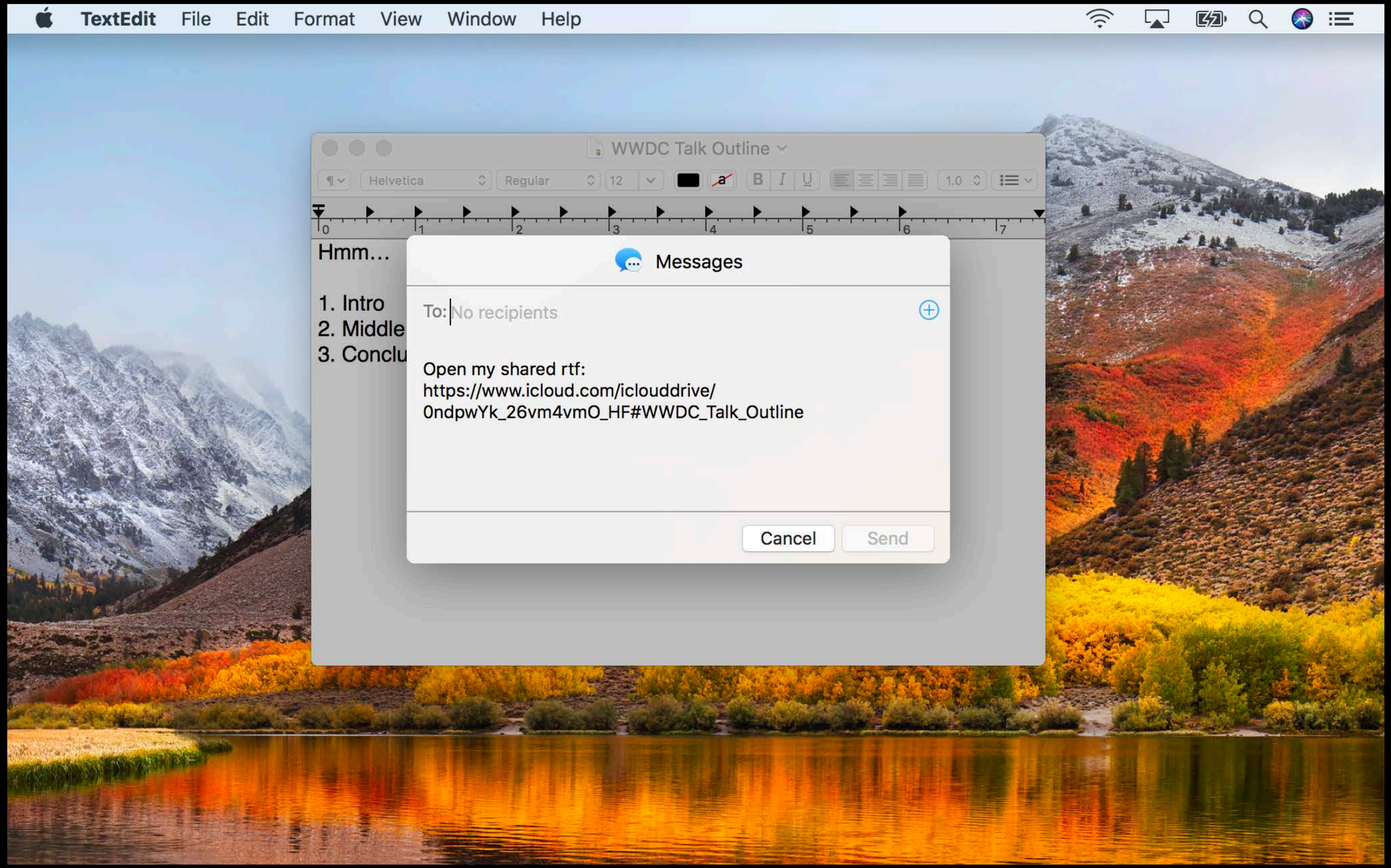

TextEdit File Edit Format View Window Help
WWDC Talk Outline
Helvetica
Regular
12
1.0
Hmm…
1. Intro
2. Middle Part
3. Conclusion

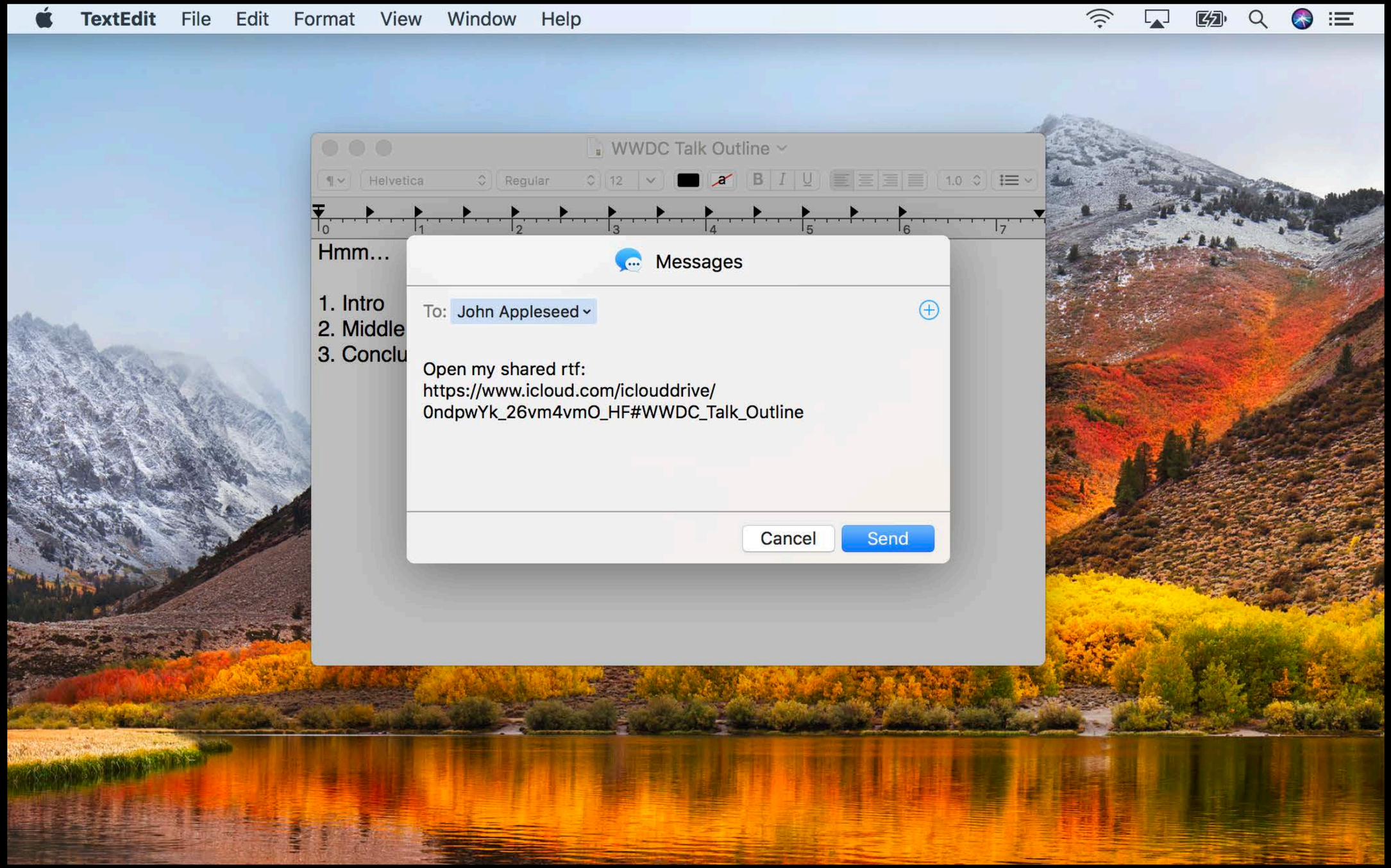

TextEdit
File
Edit
Format
View
Window
Help
New ⌘N
Open... ⌘O
Open Recent
Close ⌘W
Save ⌘S
Duplicate ⇧⌘S
Rename...
Move To...
Revert To
Export as PDF...
Share
Show Properties ⌥⌘P
Page Setup... ⇧⌘P
Print... ⌘P
Mail
Messages
AirDrop
Notes
Reminders
Add People
More...
WWDC Talk Outline
Regular
12
1.0

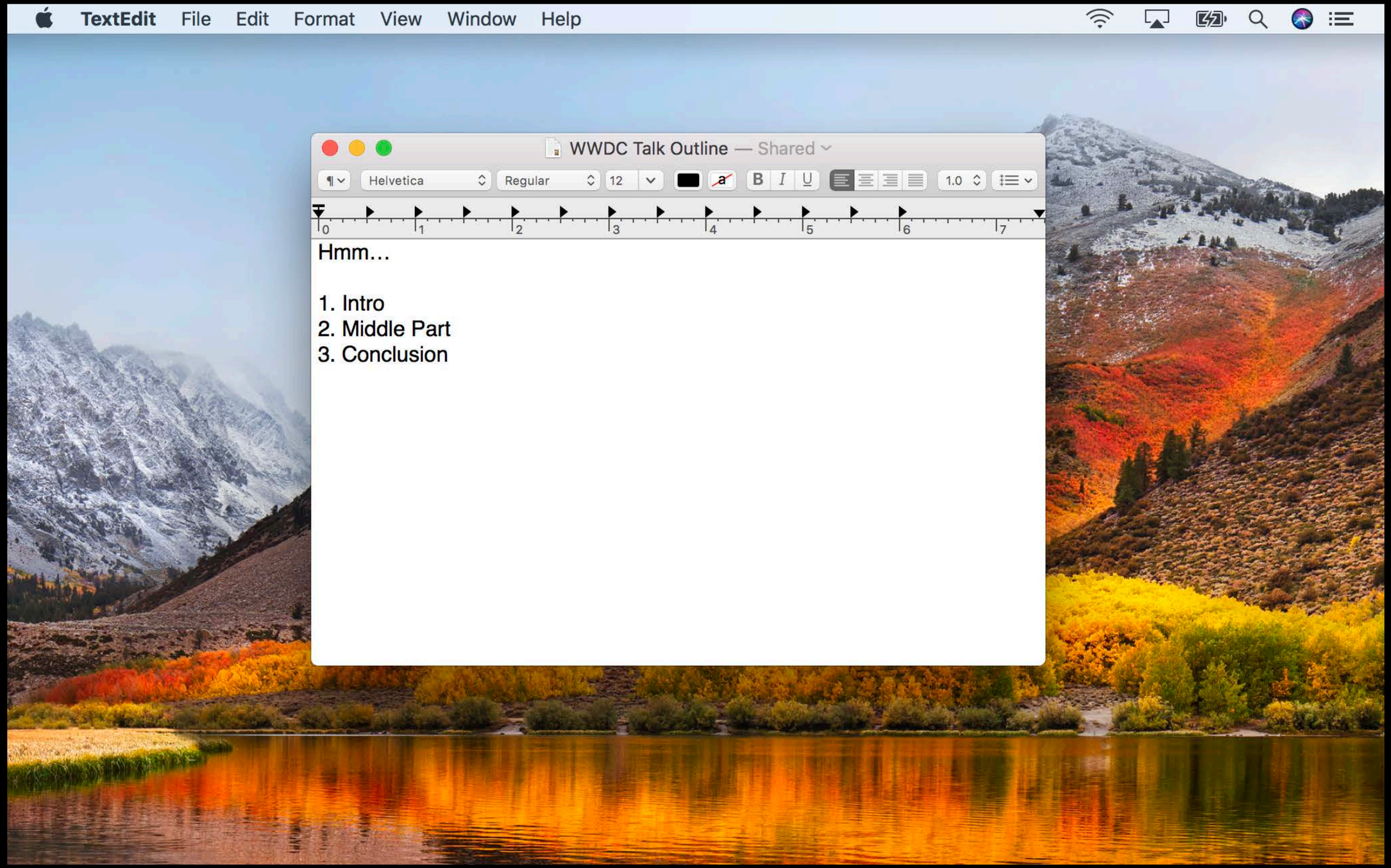

TextEdit
File
Edit
Format
View
Window
Help
New ⌘N
Open... ⌘O
Open Recent
Close ⌘W
Save ⌘S
Duplicate ⇧⌘S
Rename...
Move To...
Revert To
Export as PDF...
Share
Show Properties ⌥⌘P
Page Setup... ⇧⌘P
Print... ⌘P
Mail
Messages
AirDrop
Notes
Reminders
Add People
More...
WWDC Talk Outline
Regular
12
1.0

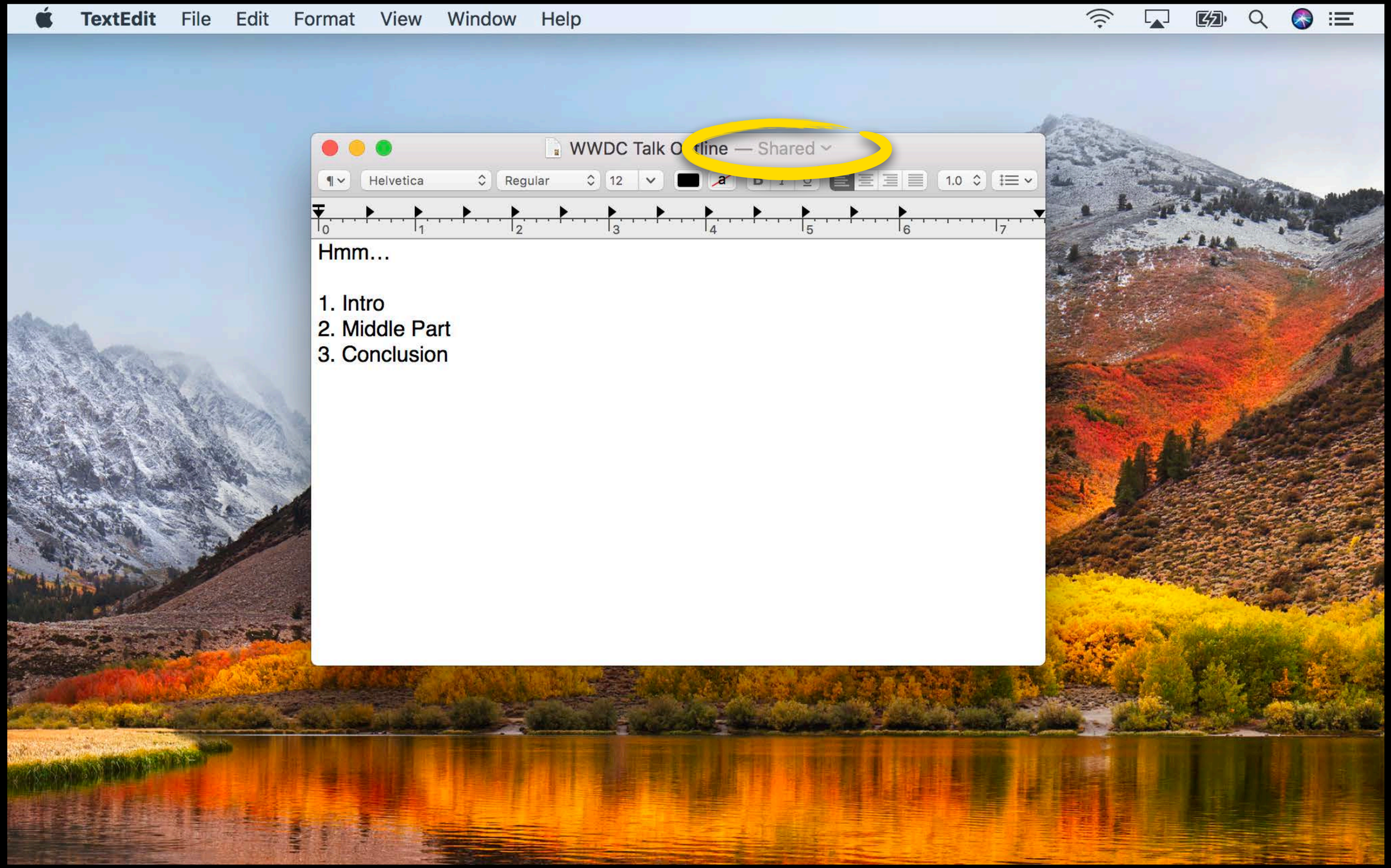

TextEdit File Edit Format View Window Help
WWDC Talk Outline
Helvetica Regular 12
Hmm…
1. Intro
2. Middle
3. Conclu
Add People
Choose how you'd like to send your invitation:
Mail
Messages
Copy Link
Twitter
Facebook
AirDrc
Share Options
Only people you invite can make changes
Demo User
Cancel
Share

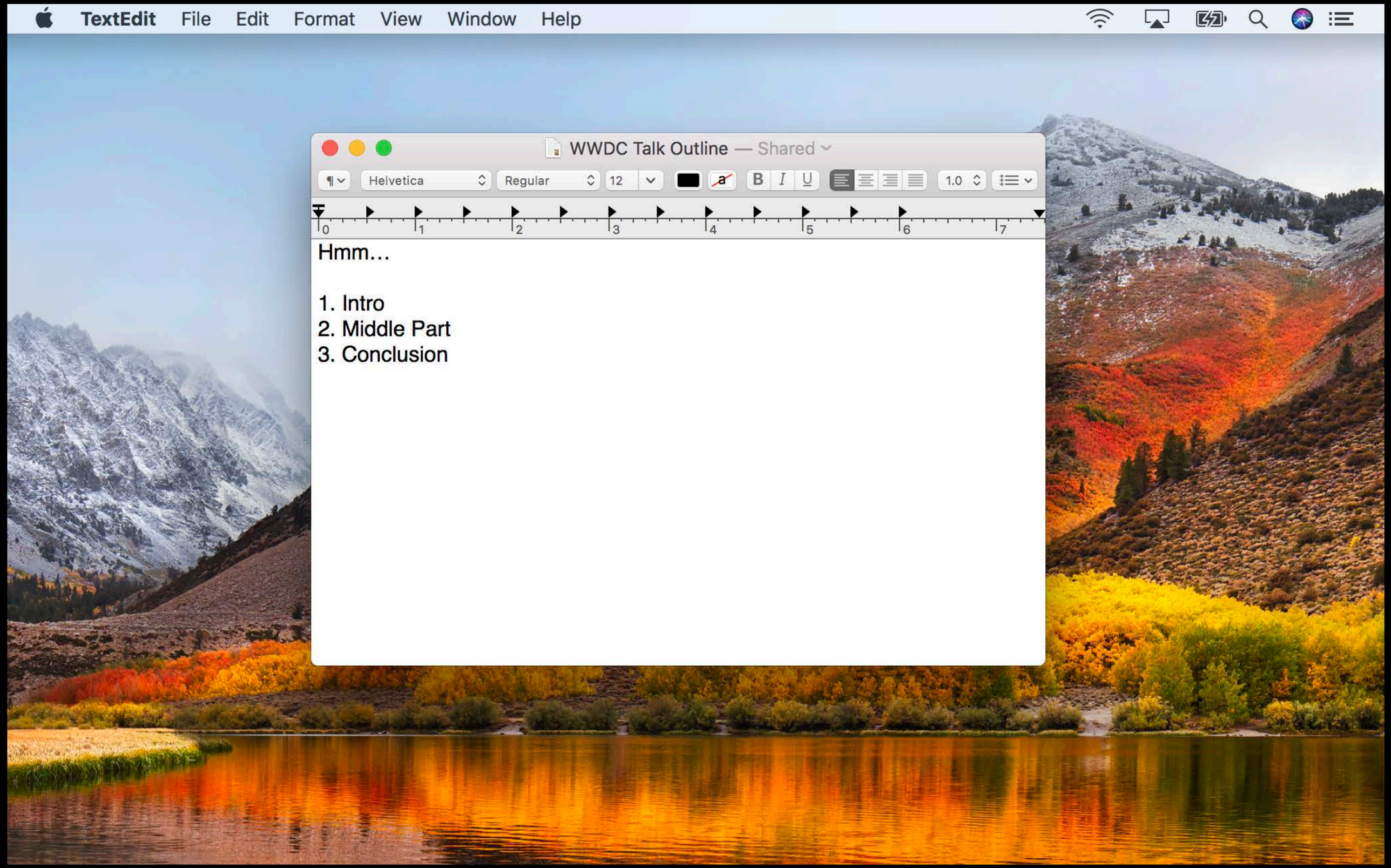

TextEdit File Edit Format View Window Help
WWDC Talk Outline
Helvetica Regular 12
Hmm…
1. Intro
2. Middle
3. Conclu
Add People
Choose how you'd like to send your invitation:
Mail
Messages
Copy Link
Twitter
Facebook
AirDrc
Share Options
Only people you invite can make changes
Demo User
Cancel
Share


TextEdit File Edit Format View Window Help
WWDC Talk Outline
Helvetica
Regular
12
1.0
Hmm…
1. Intro
2. Middle
3. Conclu
Messages
To: No recipients
Open my shared rtf:
https://www.icloud.com/iclouddrive/
0ndpwYk_26vm4vmO_HF#WWDC_Talk_Outline
Cancel
Send


TextEdit File Edit Format View Window Help
WWDC Talk Outline
Helvetica
Regular
12
Hmm…
1. Intro
2. Middle
3. Conclu
Messages
To: John Appleseed
Open my shared rtf:
https://www.icloud.com/iclouddrive/
0ndpwYk_26vm4vmO_HF#WWDC_Talk_Outline
Cancel
Send


TextEdit File Edit Format View Window Help
WWDC Talk Outline — Shared
Helvetica
Regular
12
1.0
Hmm…
1. Intro
2. Middle Part
3. Conclusion


TextEdit File Edit Format View Window Help
WWDC Talk O line — Shared
Helvetica
Regular
12
1.0
Hmm…
1. Intro
2. Middle Part
3. Conclusion


TextEdit File Edit Format View Window Help
WWDC Talk Outline — Shared
Helvetica
Regular
12
1.0
Hmm…
1. Intro
2. Middle Part
3. Conclusion

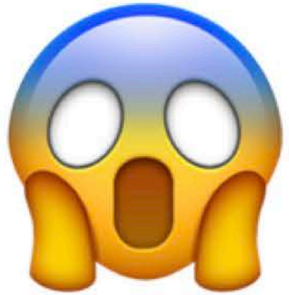
TextEdit File Edit Format View Window Help

WWDC Talk Outline — Shared

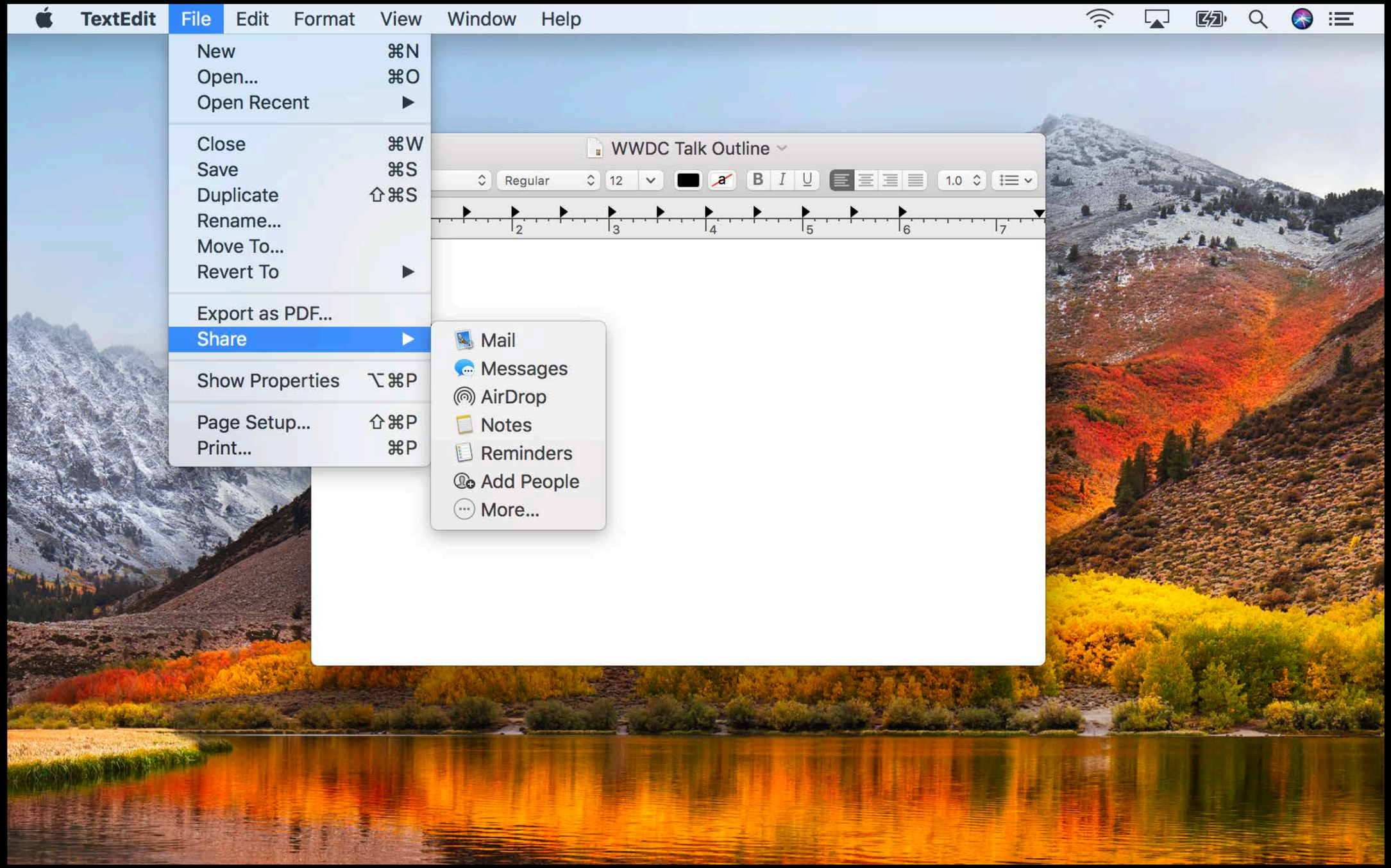
Helvetica Regular 16 1.0

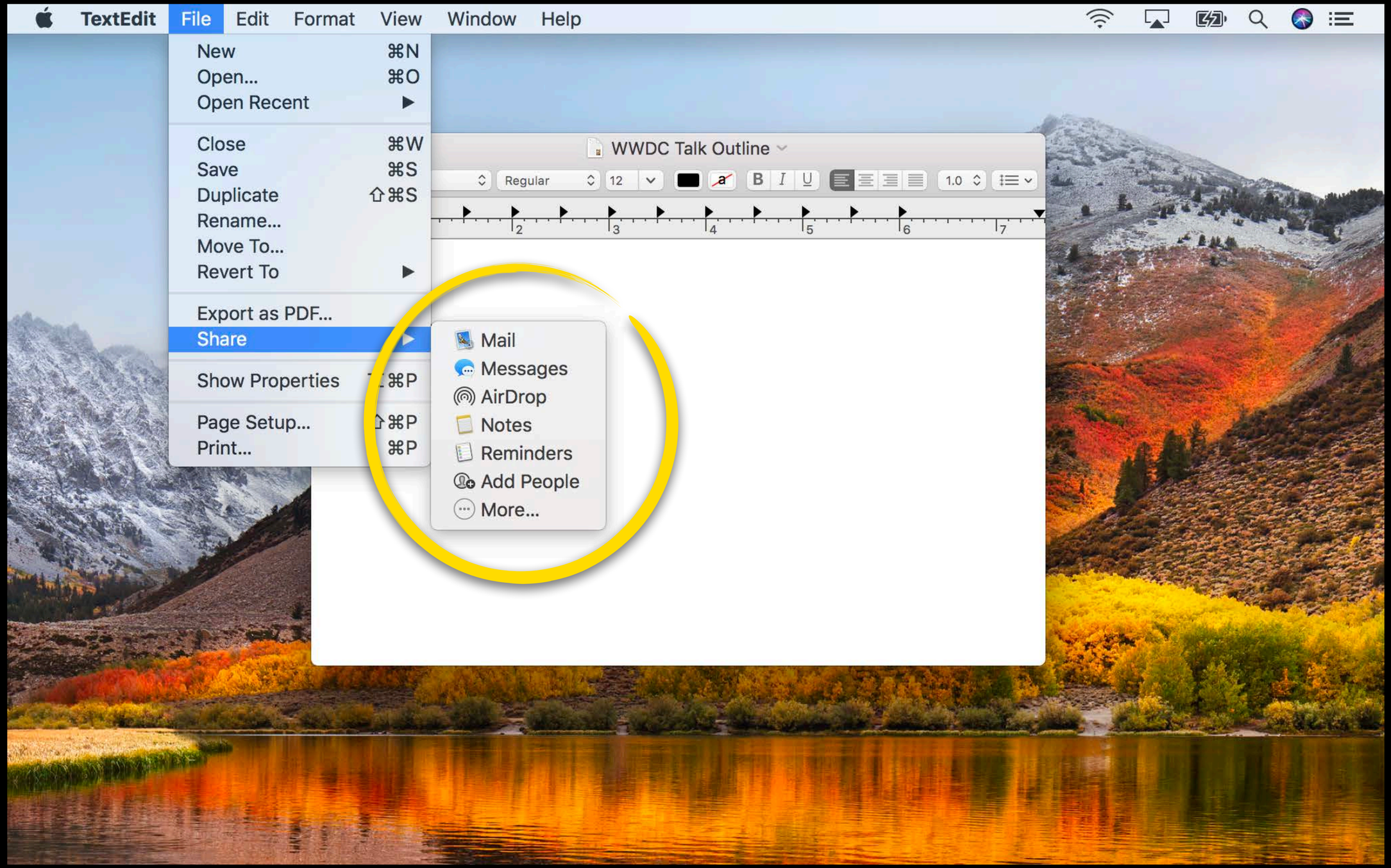
Hmm…

1. Intro
2. Middle Part
3. Conclusion



Zero


TextEdit
File
Edit
Format
View
Window
Help
New ⌘N
Open... ⌘O
Open Recent
Close ⌘W
Save ⌘S
Duplicate ⇧⌘S
Rename...
Move To...
Revert To
Export as PDF...
Share
Show Properties ⌥⌘P
Page Setup... ⇧⌘P
Print... ⌘P
Mail
Messages
AirDrop
Notes
Reminders
Add People
More...
WWDC Talk Outline
Regular
12
1.0


TextEdit
File
Edit
Format
View
Window
Help
New ⌘N
Open... ⌘O
Open Recent
Close ⌘W
Save ⌘S
Duplicate ⇧⌘S
Rename...
Move To...
Revert To
Export as PDF...
Share
Show Properties ⌥⌘P
Page Setup... ⇧⌘P
Print... ⌘P
Mail
Messages
AirDrop
Notes
Reminders
Add People
More...
WWDC Talk Outline
Regular
12
1.0

Document Sharing

Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

```
open class NSDocumentController ... {
    ...
    open var allowsAutomaticShareMenu: Bool { get }
    open func standardShareMenuItem() -> NSMenuItem
}
```

Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

```
open class NSDocumentController ... {
    ...
    open var allowsAutomaticShareMenu: Bool { get }
    open func standardShareMenuItem() -> NSMenuItem
}
```

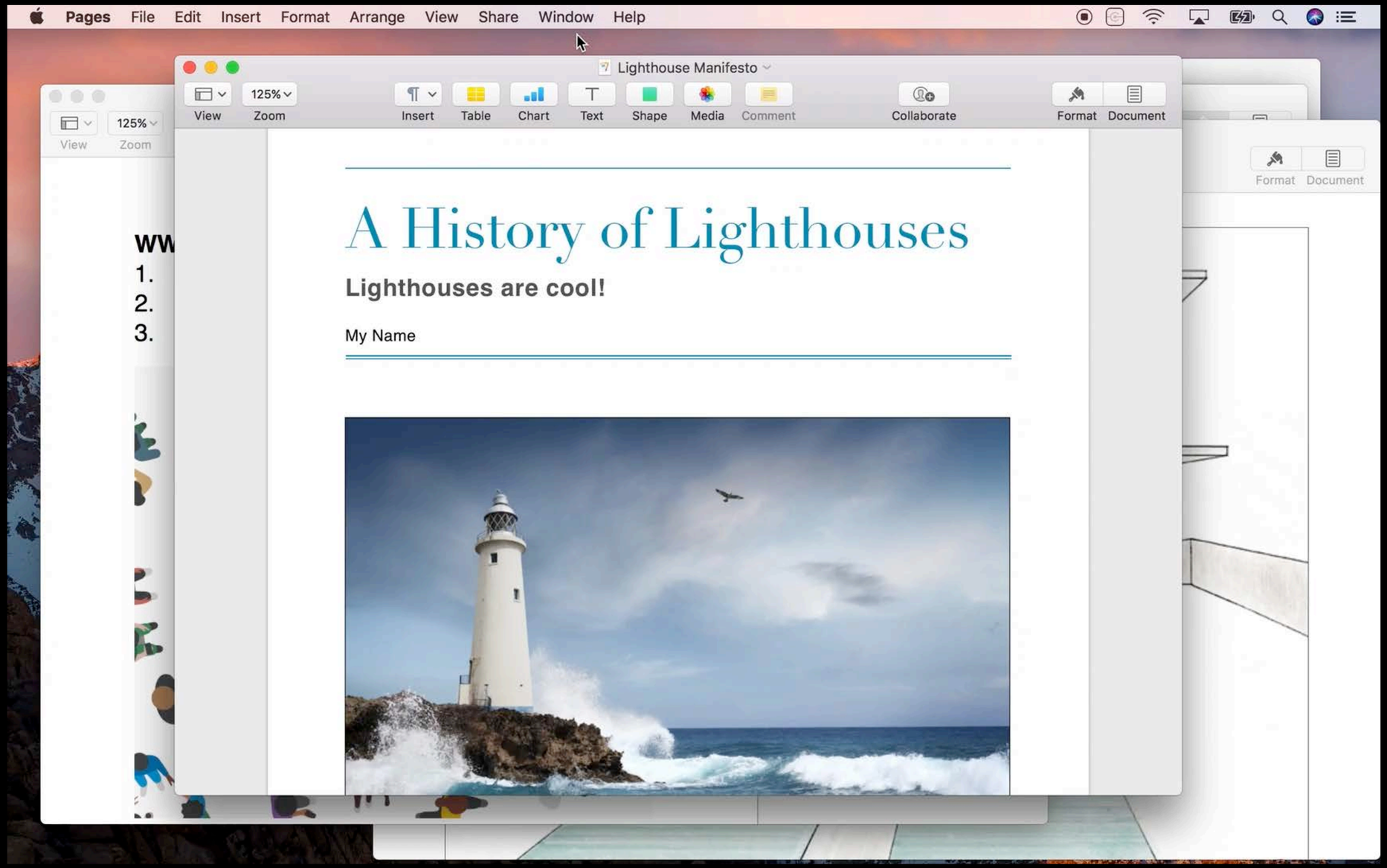
Document Sharing

```
open class NSDocument ... {
    ...
    open var allowsDocumentSharing: Bool { get }
    open func share(with sharingService: NSSharingService,
                    completionHandler: ((Bool) -> Void)? = nil)
    open func prepare(_ sharingServicePicker: NSSharingServicePicker)
}
```

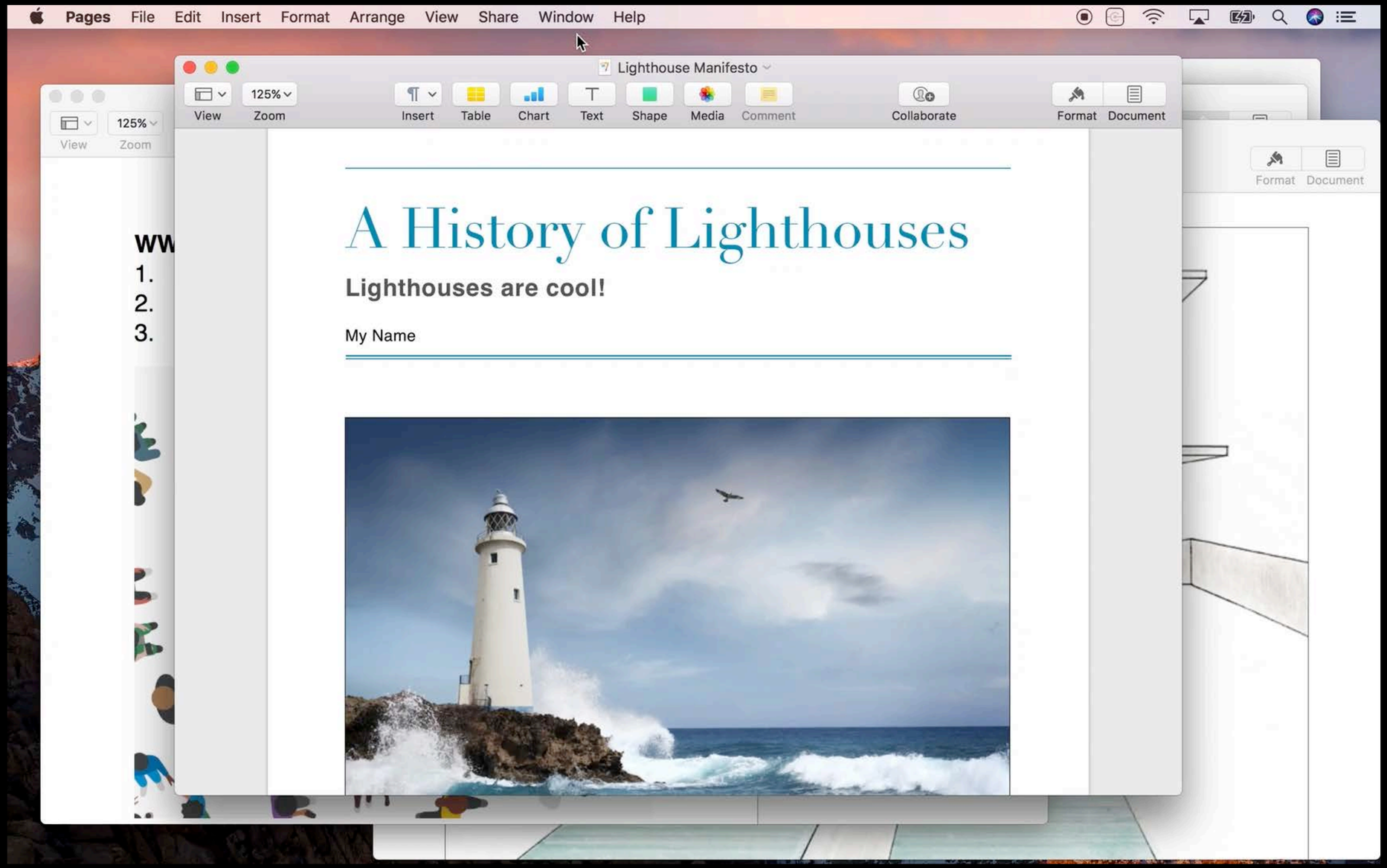
```
open class NSDocumentController ... {
    ...
    open var allowsAutomaticShareMenu: Bool { get }
    open func standardShareMenuItem() -> NSMenuItem
}
```

Tabbed Windows

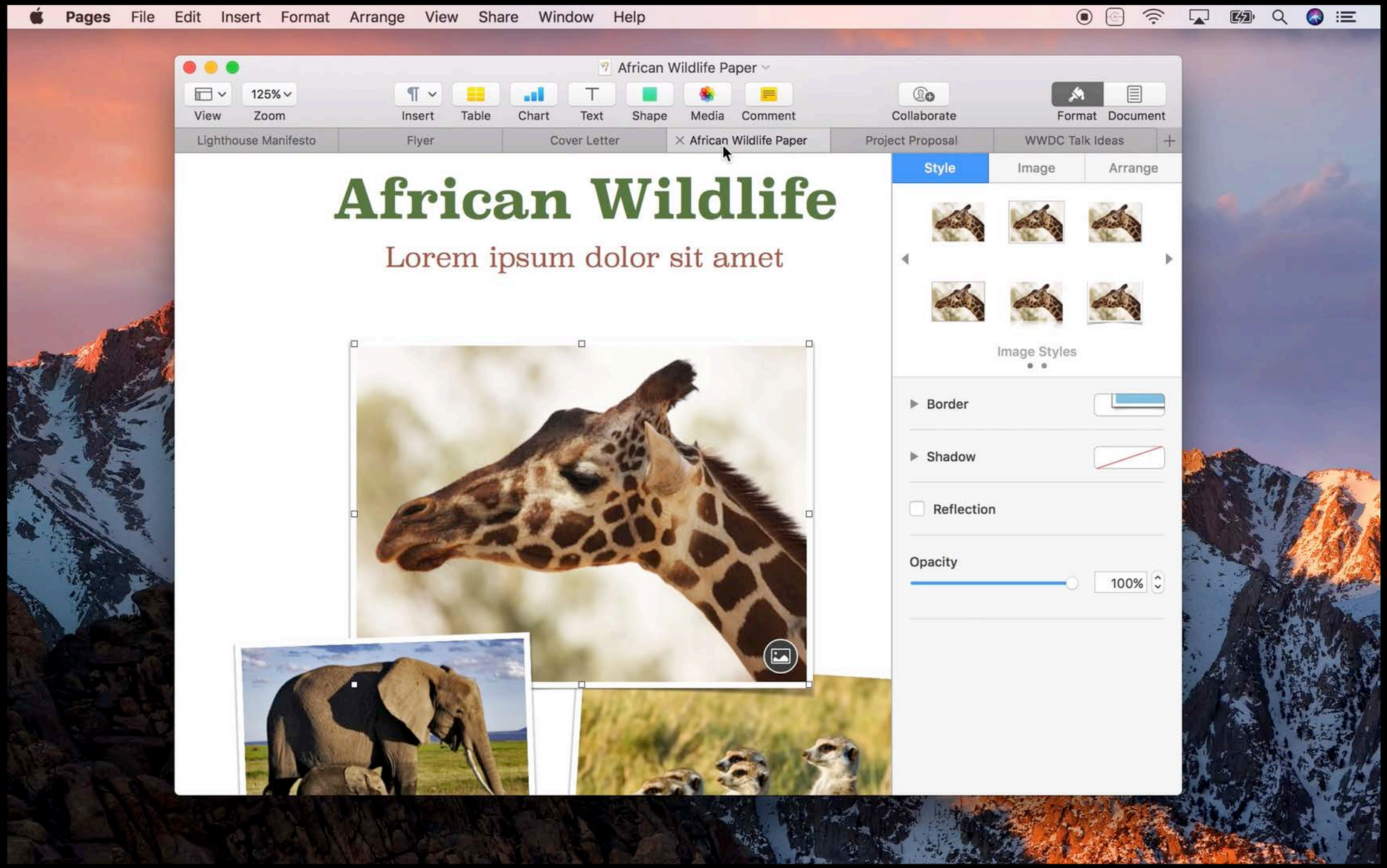
Tabbed Windows



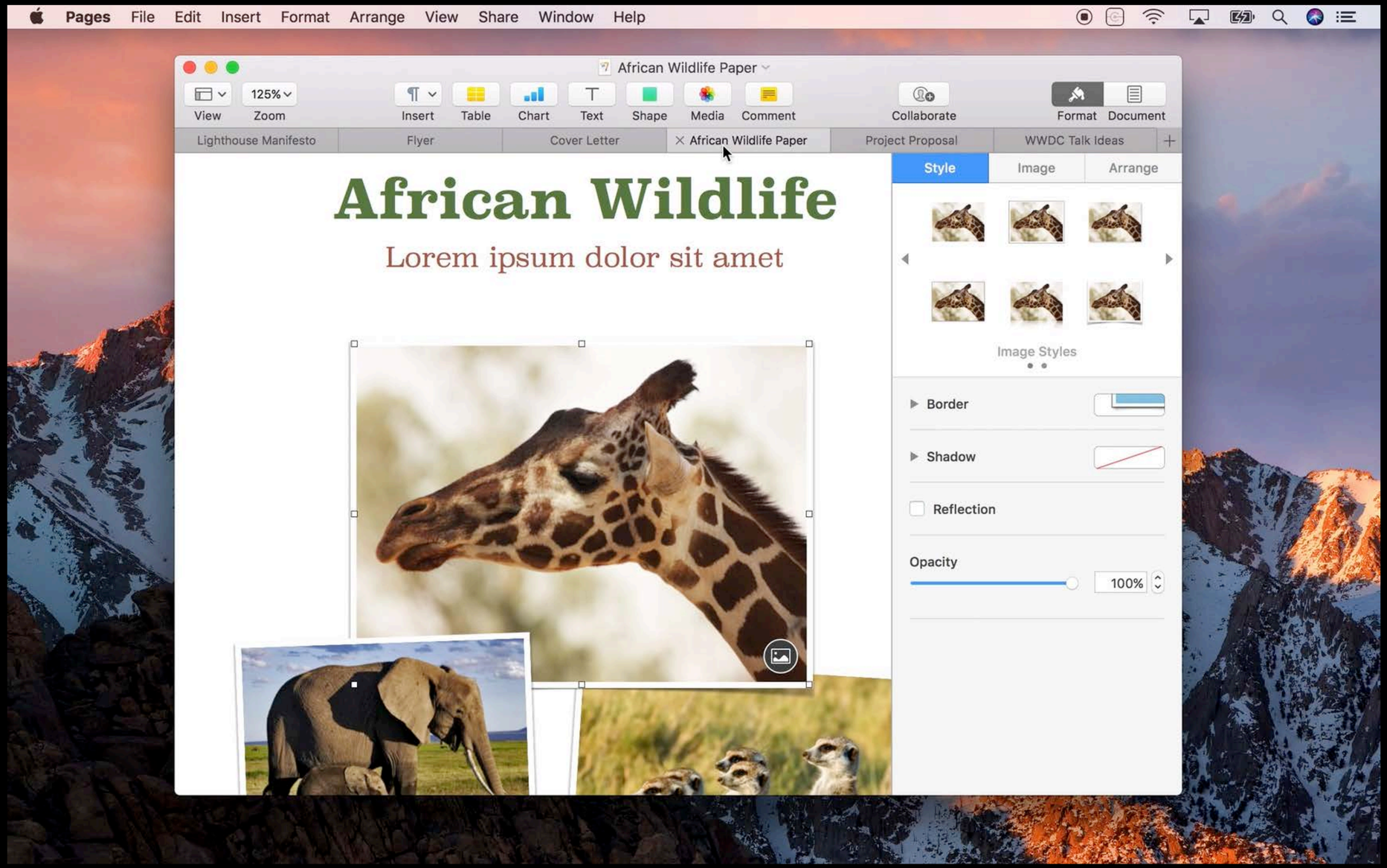
Tabbed Windows



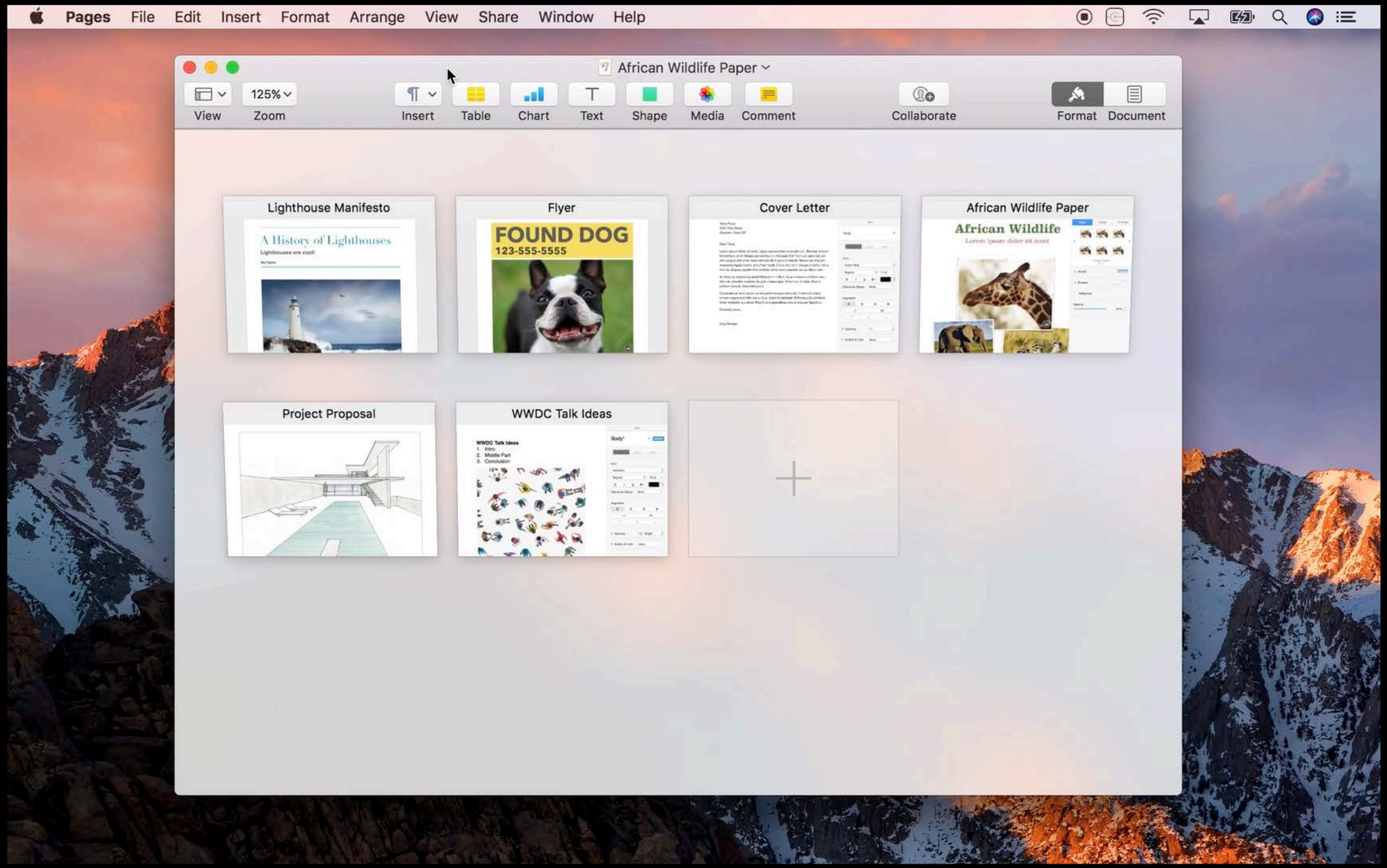
Tab Overview



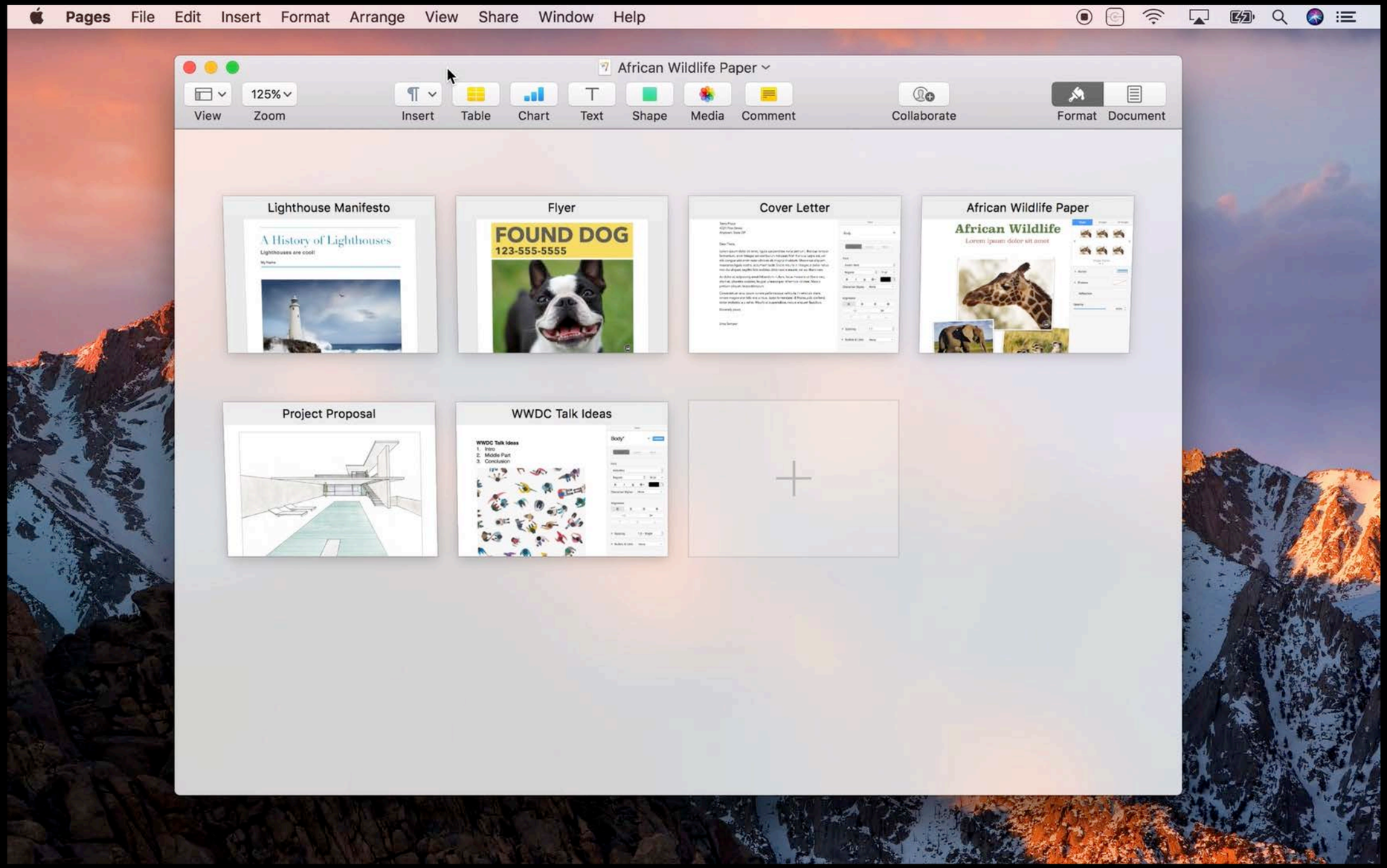
Tab Overview



Tab Overview



Tab Overview

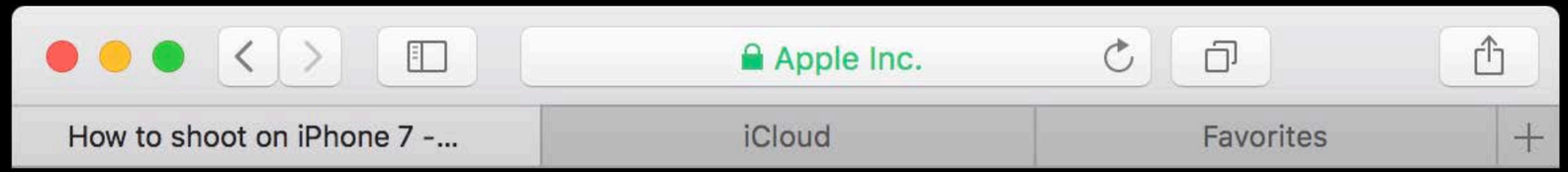


NSWindowTab

```
open class NSWindowTab : NSObject {
    open var title: String!
    @NSCopying open var attributedTitle: NSAttributedString?
    open var toolTip: String!
    open var accessoryView: NSView?
}
```

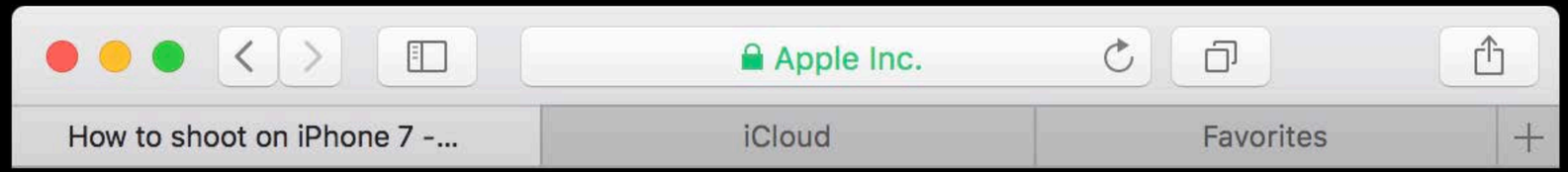
NSWindowTab

```
open class NSWindowTab : NSObject {
    open var title: String!
    @NSCopying open var attributedTitle: NSAttributedString?
    open var toolTip: String!
    open var accessoryView: NSView?
}
```



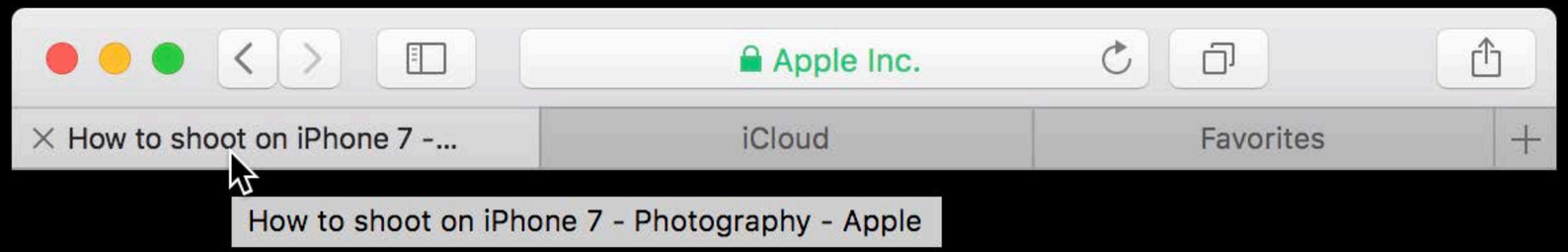
NSWindowTab

```
open class NSWindowTab : NSObject {
    open var title: String!
    @NSCopying open var attributedTitle: NSAttributedString?
    open var toolTip: String!
    open var accessoryView: NSView?
}
```



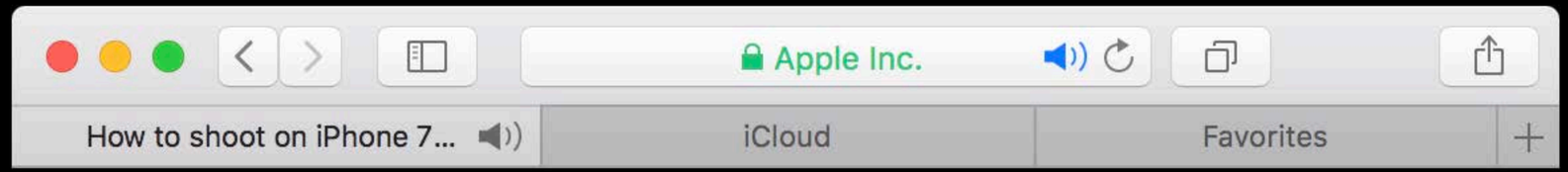
NSWindowTab

```
open class NSWindowTab : NSObject {
    open var title: String!
    @NSCopying open var attributedTitle: NSAttributedString?
    open var toolTip: String!
    open var accessoryView: NSView?
}
```



NSWindowTab

```
open class NSWindowTab : NSObject {
    open var title: String!
    @NSCopying open var attributedTitle: NSAttributedString?
    open var toolTip: String!
    open var accessoryView: NSView?
}
```



NSWindowTabGroup

```
open class NSWindowTabGroup : NSObject {
    open var identifier: NSWindow.TabbingIdentifier { get }
    open var isOverviewVisible: Bool
    open var isTabBarVisible: Bool { get }

    open var windows: [NSWindow] { get }
    weak open var selectedWindow: NSWindow?

    open func addWindow(_ window: NSWindow)
    open func insertWindow(_ window: NSWindow, at index: Int)
    open func removeWindow(_ window: NSWindow)
}
```

NSWindowTabGroup

```
open class NSWindowTabGroup : NSObject {
    open var identifier: NSWindow.TabbingIdentifier { get }
    open var isOverviewVisible: Bool
    open var isTabBarVisible: Bool { get }

    open var windows: [NSWindow] { get }
    weak open var selectedWindow: NSWindow?

    open func addWindow(_ window: NSWindow)
    open func insertWindow(_ window: NSWindow, at index: Int)
    open func removeWindow(_ window: NSWindow)
}
```

NSWindowTabGroup

```
open class NSWindowTabGroup : NSObject {
    open var identifier: NSWindow.TabbingIdentifier { get }
    open var isOverviewVisible: Bool
    open var isTabBarVisible: Bool { get }

    open var windows: [NSWindow] { get }
    weak open var selectedWindow: NSWindow?

    open func addWindow(_ window: NSWindow)
    open func insertWindow(_ window: NSWindow, at index: Int)
    open func removeWindow(_ window: NSWindow)
}
```

NSWindowTabGroup

```
open class NSWindowTabGroup : NSObject {
    open var identifier: NSWindow.TabbingIdentifier { get }
    open var isOverviewVisible: Bool
    open var isTabBarVisible: Bool { get }

    open var windows: [NSWindow] { get }
    weak open var selectedWindow: NSWindow?

    open func addWindow(_ window: NSWindow)
    open func insertWindow(_ window: NSWindow, at index: Int)
    open func removeWindow(_ window: NSWindow)
}
```

Opening URLs

Opening URLs

```
public protocol NSApplicationDelegate {
    ...
    optional func application(_ application: NSApplication, open urls: [URL])
}
```

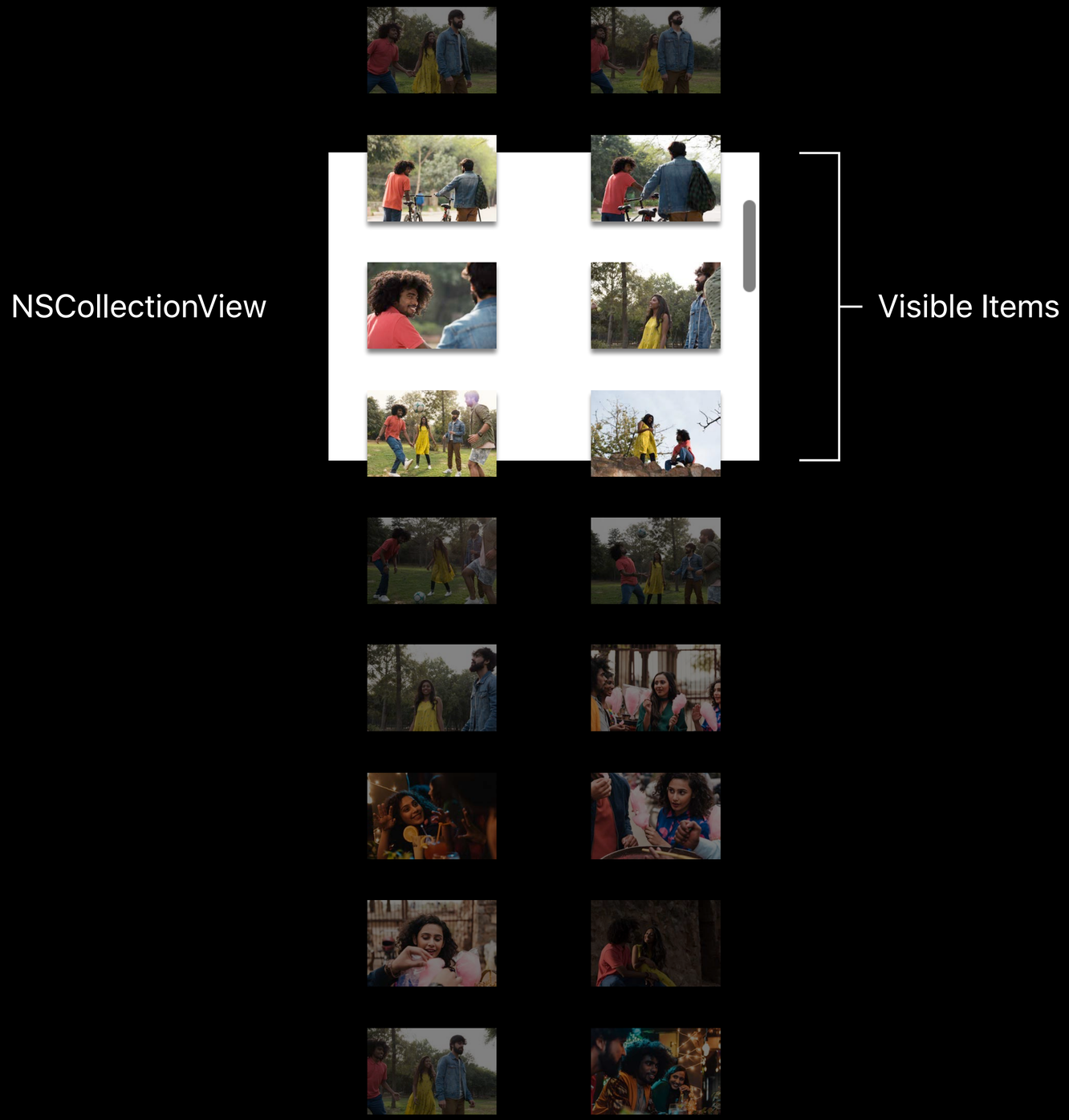
NSCollectionView Prefetching

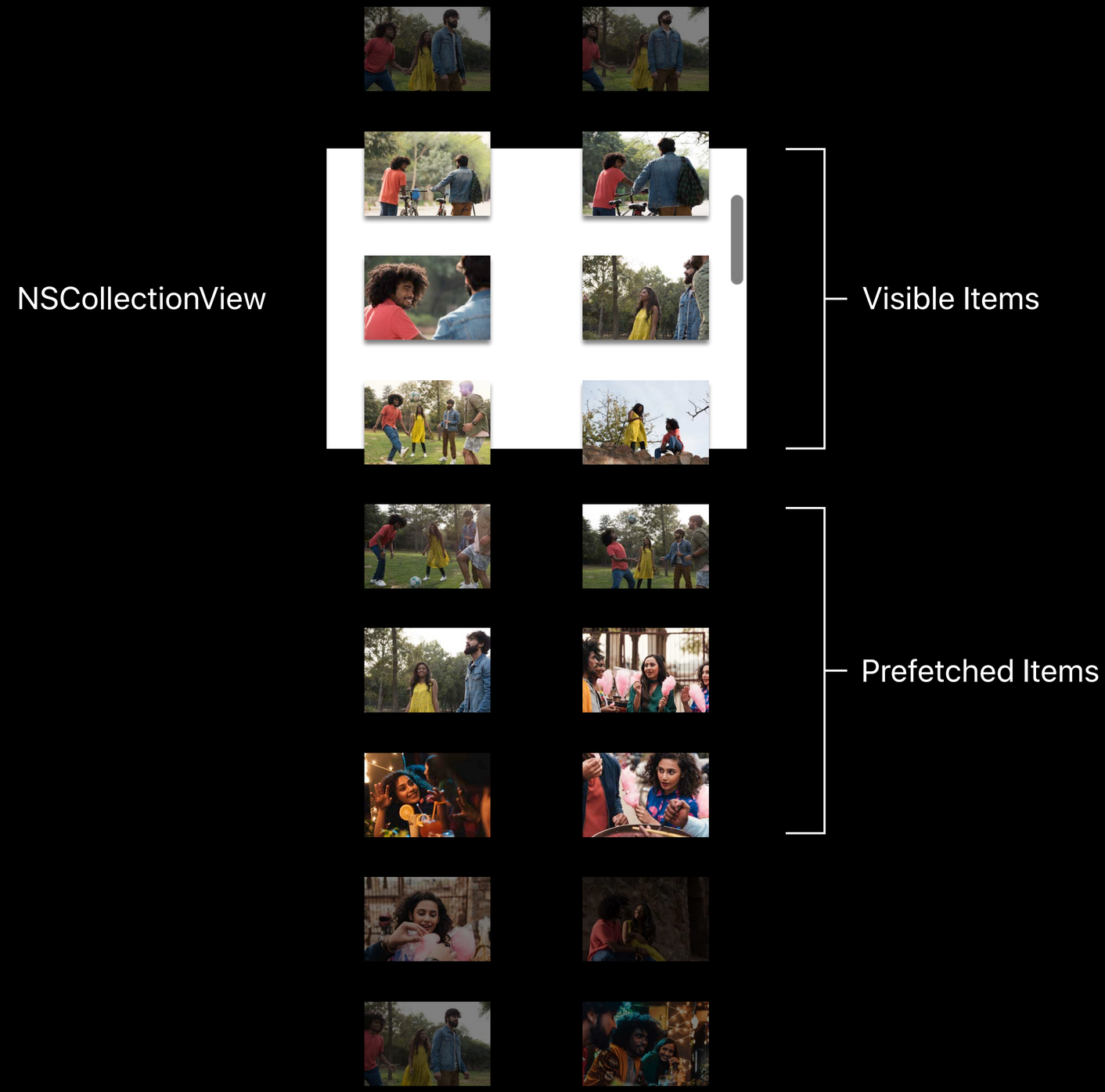
NSCollectionView

NSCollectionView

Visible Items


NSCollectionView
Visible Items


NSCollectionView
Visible Items


NSCollectionView
Visible Items
Prefetched Items

NSCollectionView Prefetching

```
open class NSCollectionView ... {
    ...
    weak open var prefetchDataSource: NSCollectionViewPrefetching?
}
```

NSCollectionView Prefetching

```
open class NSCollectionView ... {
    ...
    weak open var prefetchDataSource: NSCollectionViewPrefetching?
}
```

```
public protocol NSCollectionViewPrefetching ... {
    public func collectionView(_ collectionView: NSCollectionView,
                               prefetchItemsAt: [IndexPath])
    optional public func collectionView(_ collectionView: NSCollectionView,
                                cancelPrefetchingForItemsAt: [IndexPath])
}
```

NSCollectionView Prefetching

```
open class NSCollectionView ... {
    ...
    weak open var prefetchDataSource: NSCollectionViewPrefetching?
}
```

```
public protocol NSCollectionViewPrefetching ... {
    public func collectionView(_ collectionView: NSCollectionView,
                               prefetchItemsAt: [IndexPath])
    optional public func collectionView(_ collectionView: NSCollectionView,
                                cancelPrefetchingForItemsAt: [IndexPath])
}
```

NSCollectionView Prefetching

```
open class NSCollectionView ... {
    ...
    weak open var prefetchDataSource: NSCollectionViewPrefetching?
}
```

```
public protocol NSCollectionViewPrefetching ... {
    public func collectionView(_ collectionView: NSCollectionView,
                               prefetchItemsAt: [IndexPath])
    optional public func collectionView(_ collectionView: NSCollectionView,
                                cancelPrefetchingForItemsAt: [IndexPath])
}
```

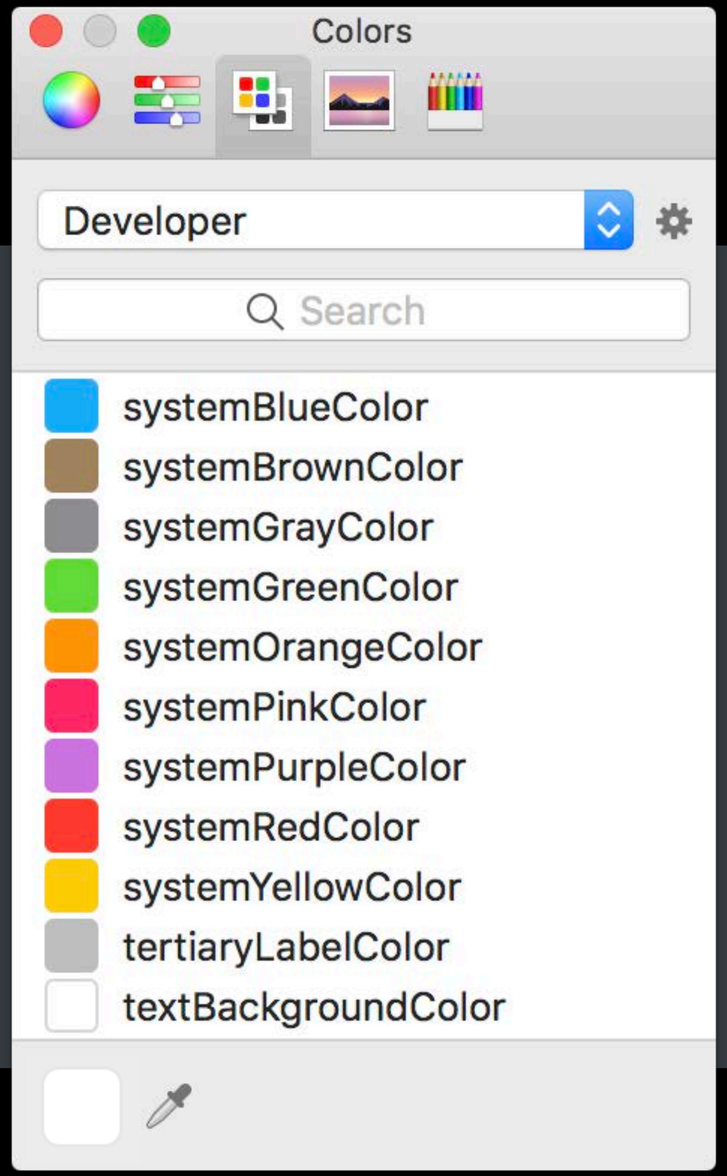
System Colors

System Colors

```
open class NSColor ... {
    ...
    open class var systemBlue:   NSColor { get }
    open class var systemBrown:  NSColor { get }
    open class var systemGray:   NSColor { get }
    open class var systemGreen:  NSColor { get }
    open class var systemOrange: NSColor { get }
    open class var systemPink:   NSColor { get }
    open class var systemPurple: NSColor { get }
    open class var systemRed:    NSColor { get }
    open class var systemYellow: NSColor { get }
}
```

System Colors

```
open class NSColor ... {
    ...
    open class var systemBlue:   NSColor { get }
    open class var systemBrown:  NSColor { get }
    open class var systemGray:   NSColor { get }
    open class var systemGreen:  NSColor { get }
    open class var systemOrange: NSColor { get }
    open class var systemPink:   NSColor { get }
    open class var systemPurple: NSColor { get }
    open class var systemRed:    NSColor { get }
    open class var systemYellow: NSColor { get }
}
```



Older “Standard” Colors Now in sRGB

Older “Standard” Colors Now in sRGB

```
open class NSColor ... {
    open class var red:    NSColor { get }  /* 1.0, 0.0, 0.0 RGB */
    open class var green:  NSColor { get }  /* 0.0, 1.0, 0.0 RGB */
    open class var yellow: NSColor { get }  /* 1.0, 1.0, 0.0 RGB */
    ...
    open class var gray:      NSColor { get }  /* 0.5 white   */
    open class var lightGray: NSColor { get }  /* 0.667 white */
    open class var white:     NSColor { get }  /* 1.0 white   */
}
```

Older “Standard” Colors Now in sRGB

```
open class NSColor ... {
    open class var red:    NSColor { get }  /* 1.0, 0.0, 0.0 RGB */
    open class var green:  NSColor { get }  /* 0.0, 1.0, 0.0 RGB */
    open class var yellow: NSColor { get }  /* 1.0, 1.0, 0.0 RGB */
    ...
    open class var gray:      NSColor { get }  /* 0.5 white   */
    open class var lightGray: NSColor { get }  /* 0.667 white */
    open class var white:     NSColor { get }  /* 1.0 white   */
}
```

These now use sRGB or genericGamma22Gray

Older “Standard” Colors Now in sRGB

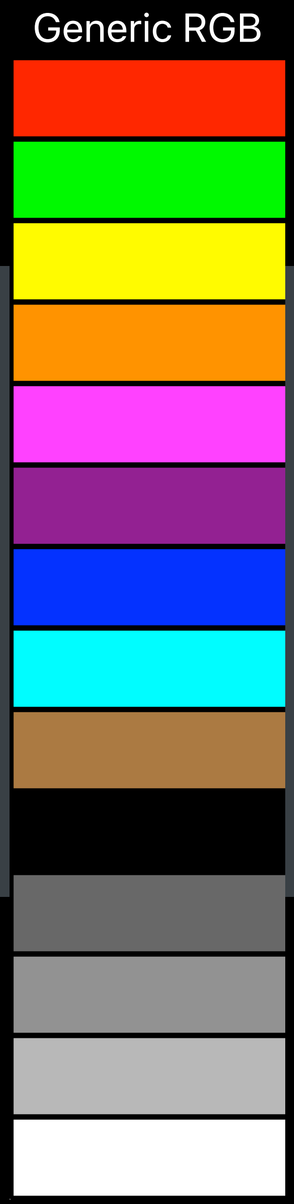
```
open class NSColor ... {
    open class var red:    NSColor { get }  /* 1.0, 0.0, 0.0 RGB */
    open class var green:  NSColor { get }  /* 0.0, 1.0, 0.0 RGB */
    open class var yellow: NSColor { get }  /* 1.0, 1.0, 0.0 RGB */
    ...
    open class var gray:      NSColor { get }  /* 0.5 white   */
    open class var lightGray: NSColor { get }  /* 0.667 white */
    open class var white:     NSColor { get }  /* 1.0 white   */
}
```

These now use sRGB or genericGamma22Gray

For apps linked against the 10.13 SDK or later

Older “Standard” Colors Now in sRGB

```
open class NSColor ... {
    open class var red:    NSColor { get }  /* 1.0, 0.0, 0.0 RGB */
    open class var green:  NSColor { get }  /* 0.0, 1.0, 0.0 RGB */
    open class var yellow: NSColor { get }  /* 1.0, 1.0, 0.0 RGB */
    ...
    open class var gray:      NSColor { get }  /* 0.5 white   */
    open class var lightGray: NSColor { get }  /* 0.667 white */
    open class var white:     NSColor { get }  /* 1.0 white   */
}
```

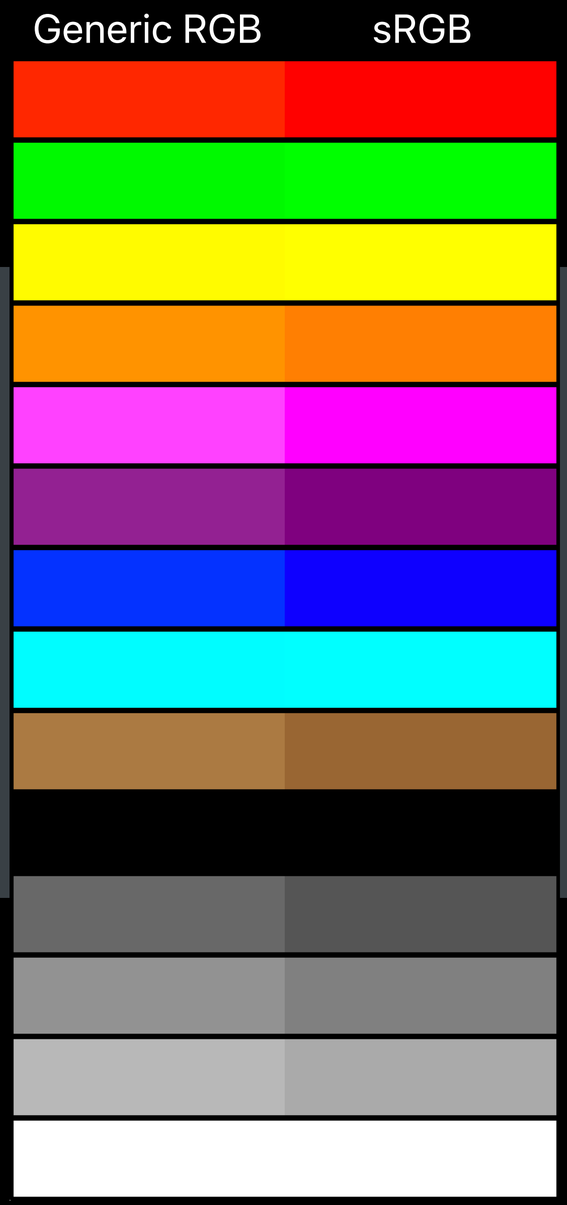


These now use sRGB or genericGamma22Gray

For apps linked against the 10.13 SDK or later

Older “Standard” Colors Now in sRGB

```
open class NSColor ... {
    open class var red:    NSColor { get }  /* 1.0, 0.0, 0.0 RGB */
    open class var green:  NSColor { get }  /* 0.0, 1.0, 0.0 RGB */
    open class var yellow: NSColor { get }  /* 1.0, 1.0, 0.0 RGB */
    ...
    open class var gray:      NSColor { get }  /* 0.5 white   */
    open class var lightGray: NSColor { get }  /* 0.667 white */
    open class var white:     NSColor { get }  /* 1.0 white   */
}
```

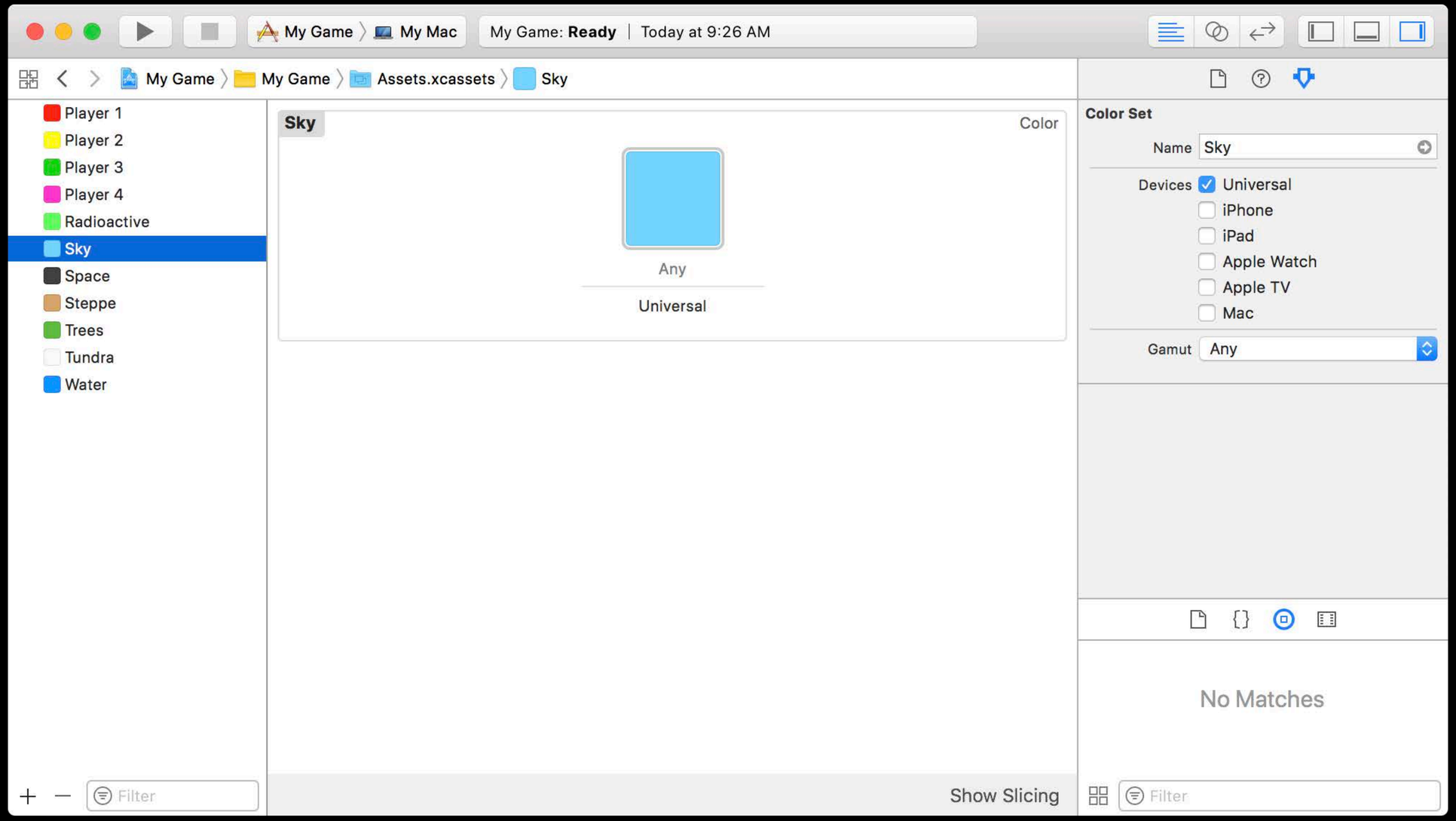


These now use sRGB or genericGamma22Gray

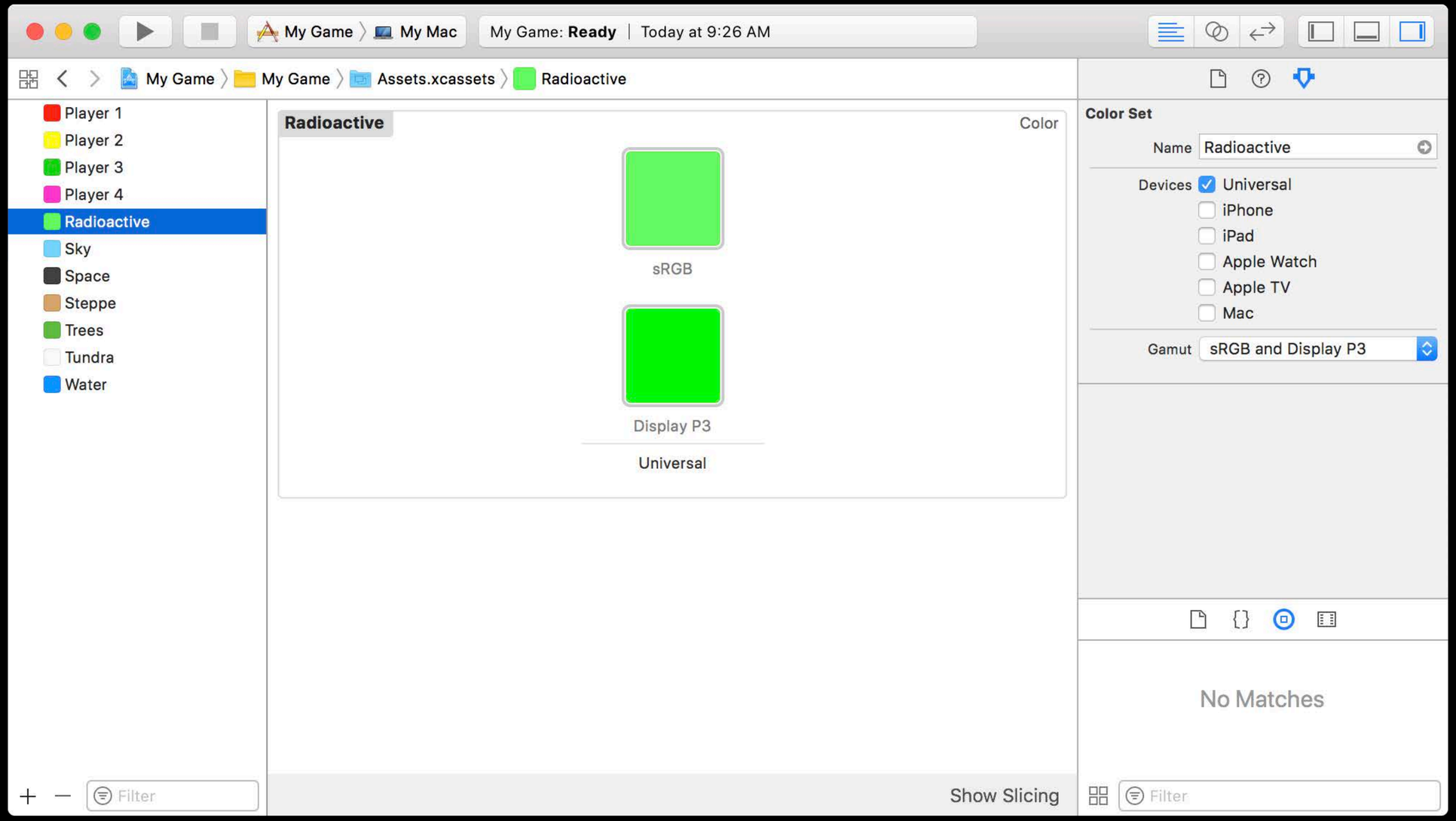
For apps linked against the 10.13 SDK or later

Colors in Asset Catalogs

Colors in Asset Catalogs



Colors in Asset Catalogs

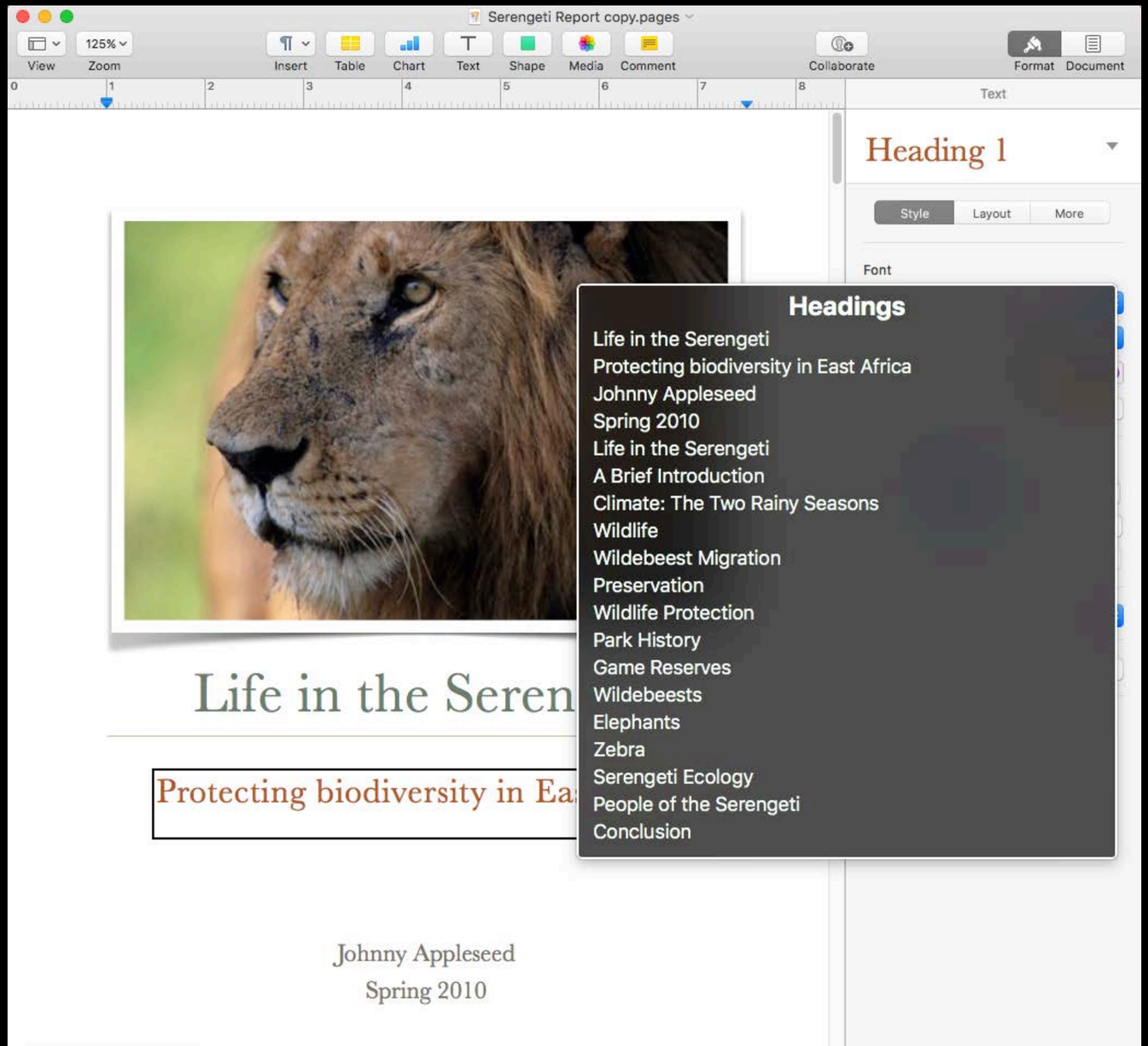


Colors in Asset Catalogs

```
open class NSColor ... {
    ...
    public init?(named name: NSColor.Name)
    public init?(named name: NSColor.Name, bundle: NSBundle?)
}
```

NSAccessibilityCustomRotor

NSAccessibilityCustomRotor



NSAccessibilityCustomRotor

```
public protocol NSAccessibility ... {
    ...
    func accessibilityCustomRotors() -> [NSAccessibilityCustomRotor]
}
```

Property Clean-Up

Property Clean-Up

```
// 10.12
// NSWindow
unowned(unsafe) open var firstResponder: NSResponder { get }
// NSBox
unowned(unsafe) open var contentView: NSView?
...
```

Property Clean-Up

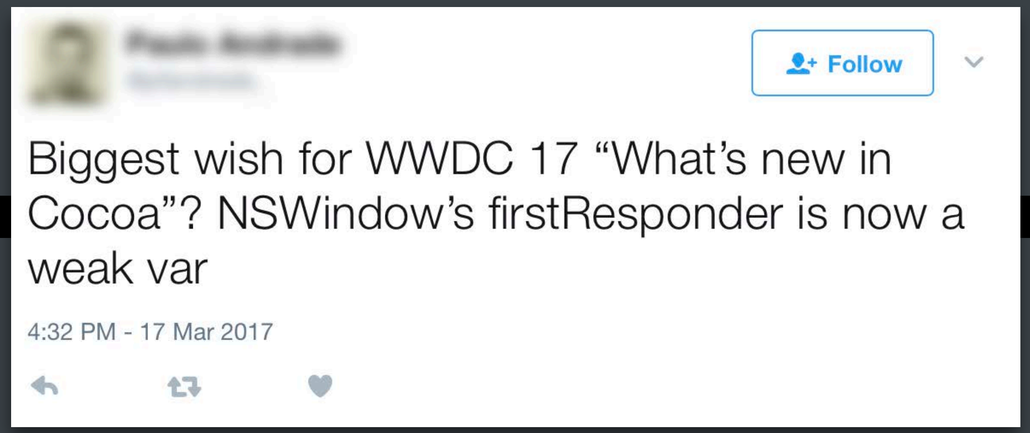
```
// 10.12
// NSWindow
unowned(unsafe) open var firstResponder: NSResponder { get }
// NSBox
unowned(unsafe) open var contentView: NSView?
...
```

```
// 10.13
// NSWindow
weak open var firstResponder: NSResponder? { get }
// NSBox
open var contentView: NSView?
...
```

Property Clean-Up

```
// 10.12
// NSWindow
unowned(unsafe) open var firstResponder: NSResponder { get }
// NSBox
unowned(unsafe) open var contentView: NSView?
...
```

```
// 10.13
// NSWindow
weak open var firstResponder: NSResponder? { get }
// NSBox
open var contentView: NSView?
...
```



Text

Text

Improved handling of orphan text in NSTextField

Text

Improved handling of orphan text in NSTextField

CGGlyph-based APIs in NSFont, NSBezierPath

Text

Improved handling of orphan text in NSTextField

CGGlyph-based APIs in NSFont, NSBezierPath

NSFontAssetRequest for downloading fonts

Text

Improved handling of orphan text in NSTextField

CGGlyph-based APIs in NSFont, NSBezierPath

NSFontAssetRequest for downloading fonts

Nastaliq script

Text

Improved handling of orphan text in NSTextField

CGGlyph-based APIs in NSFont, NSBezierPath

NSFontAssetRequest for downloading fonts

Nastaliq script

Text

Improved handling of orphan text in NSTextField

CGGlyph-based APIs in NSFont, NSBezierPath

NSFontAssetRequest for downloading fonts

Nastaliq script

Honorable Mentions

Honorable Mentions

NSSegmentedControl alignment and distribution properties

Honorable Mentions

NSSegmentedControl alignment and distribution properties

NSLevelIndicator new look and API refinements

Honorable Mentions

NSSegmentedControl alignment and distribution properties

NSLevelIndicator new look and API refinements

NSMenuItem allowsKeyEquivalentWhenHidden

Honorable Mentions

NSSegmentedControl alignment and distribution properties

NSLevelIndicator new look and API refinements

NSMenuItem allowsKeyEquivalentWhenHidden

NSTableView automatic variable row heights

Honorable Mentions

NSSegmentedControl alignment and distribution properties

NSLevelIndicator new look and API refinements

NSMenuItem allowsKeyEquivalentWhenHidden

NSTableView automatic variable row heights

Asynchronous restorable state encoding

Honorable Mentions

NSSegmentedControl alignment and distribution properties

NSLevelIndicator new look and API refinements

NSMenuItem allowsKeyEquivalentWhenHidden

NSTableView automatic variable row heights

Asynchronous restorable state encoding

Improved handling of large items during dragging

Honorable Mentions

NSSegmentedControl alignment and distribution properties

NSLevelIndicator new look and API refinements

NSMenuItem allowsKeyEquivalentWhenHidden

NSTableView automatic variable row heights

Asynchronous restorable state encoding

Improved handling of large items during dragging

Drawers - drop them

Container Views in Cocoa

Container Views in Cocoa

NSBrowser

NSTableView

NSOutlineView

NSCollectionView

NSStackView

NSGridView

Container Views in Cocoa

NSBrowser

NSTableView

NSOutlineView

NSCollectionView

NSStackView

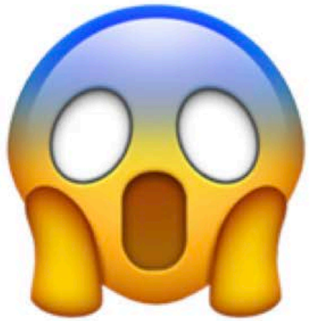
NSGridView

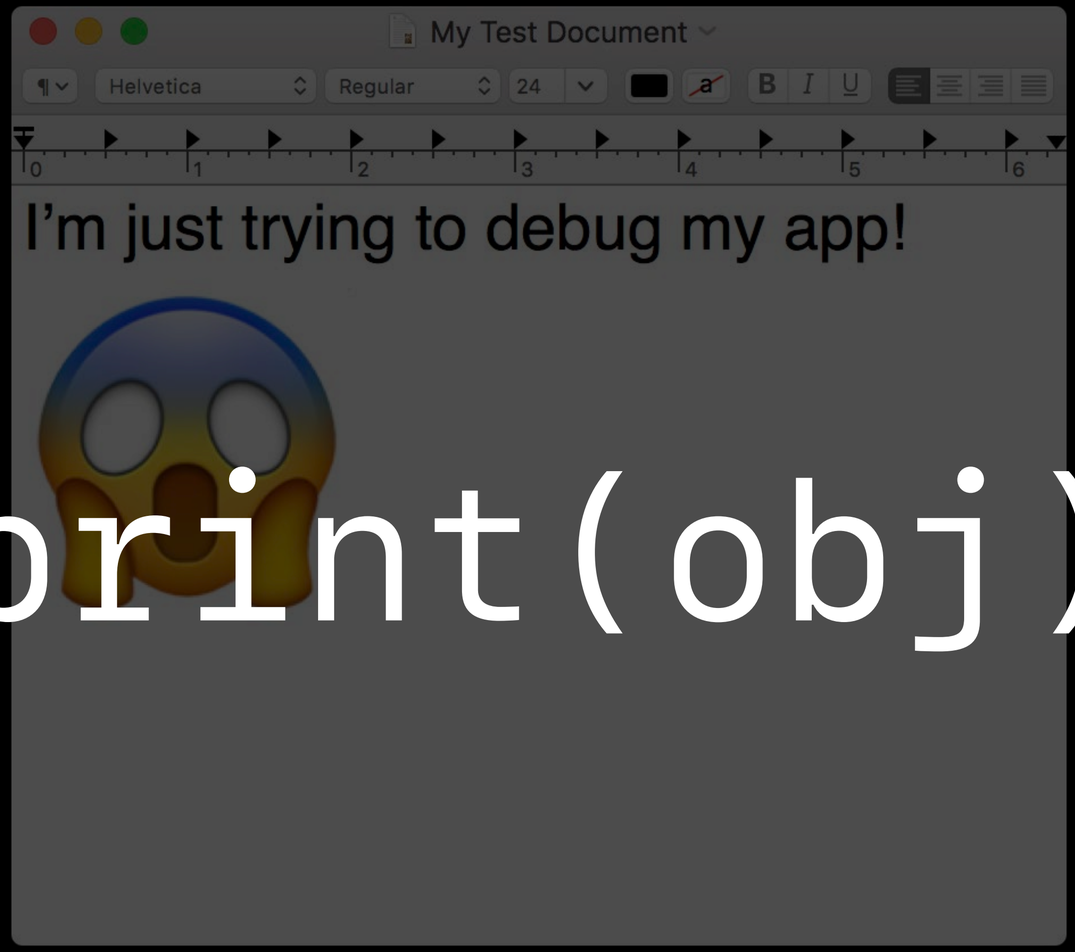
Choosing the Right Cocoa Container View | Grand Ballroom A | Wednesday 3:10PM

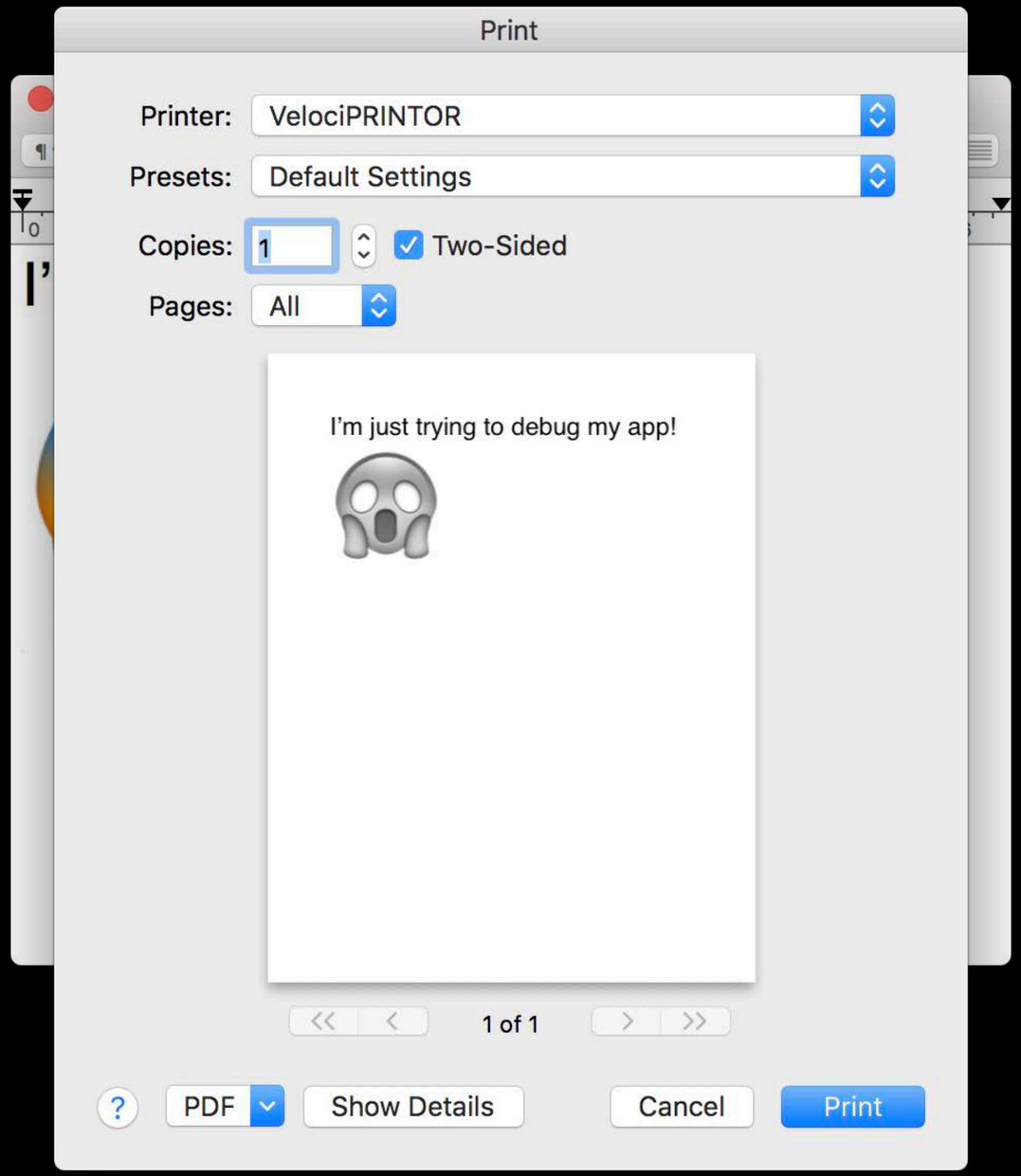
My Test Document

Helvetica | Regular | 24

I’m just trying to debug my app!




My Test Document
Helvetica
Regular
24
I’m just trying to debug my app!
print(obj)


Print
Printer:
VelociPRINTOR
Presets:
Default Settings
Copies:
1
Two-Sided
Pages:
All
I'm just trying to debug my app!
1 of 1
PDF
Show Details
Cancel
Print

NSView and NSWindow.print() Renamed

```
open class NSView ... {
    ...
    open func printView(_ sender: Any?)
}
```

```
open class NSWindow ... {
    ...
    open func printWindow(_ sender: Any?)
}
```

NSView and NSWindow.print() Renamed

```
open class NSView ... {
    ...
    open func printView(_ sender: Any?)
}
```

```
open class NSWindow ... {
    ...
    open func printWindow(_ sender: Any?)
}
```

Tweet Your Tips!

Cocoa Development Tips	Hall 2	Friday 9:00AM

Tweet Your Tips!

Use `#WWDC17` and `#cocoatip`

Foundation

Daphne Larose, Foundation

What’s New in Foundation

What’s New in Foundation

Better support for key paths

What’s New in Foundation

Better support for key paths

Encoding and Decoding in Swift

Improved Key Paths

Improved Key Paths

New literal syntax

Improved Key Paths

New literal syntax

Type-safe

Improved Key Paths

New literal syntax

Type-safe

Performant

Improved Key Paths

New literal syntax

Type-safe

Performant

```
#keyPath(BaseType.propertyName)
```

Improved Key Paths

New literal syntax

Type-safe

Performant

```
#keyPath(BaseType.propertyName)
```

```
\BaseType.propertyName
```

Block-based Key-Value Observing

```
@objc class DogOwner : NSObject {
  @objc dynamic var name: String?
  @objc dynamic var dog: Dog?
}

var daphne = DogOwner(name: "Daphne")
let observation = daphne.observe(\DogOwner.dog) { (observed, change) in
  print("Congrats on your new puppy \(observed.dogName)!")
}
```

Block-based Key-Value Observing

```
@objc class DogOwner : NSObject {
  @objc dynamic var name: String?
  @objc dynamic var dog: Dog?
}

var daphne = DogOwner(name: "Daphne")
let observation = daphne.observe(\DogOwner.dog) { (observed, change) in
  print("Congrats on your new puppy \(observed.dogName)!")
}
```

Encoding and Decoding in Swift

Encoding and Decoding in Swift

Swift types ↔ JSON, Property lists, etc.

Encoding and Decoding in Swift

Swift types ↔ JSON, Property lists, etc.

Deeply customizable

Encoding and Decoding in Swift

Swift types ↔ JSON, Property lists, etc.

Deeply customizable

Easy to use

Encoding and Decoding in Swift

Swift types ↔ JSON, Property lists, etc.

Deeply customizable

Easy to use

```
struct Dog : Codable {
  let dogName: String
  let breed: BreedType
}
```

Encoding and Decoding in Swift

Swift types ↔ JSON, Property lists, etc.

Deeply customizable

Easy to use

```
struct Dog : Codable {
  let dogName: String
  let breed: BreedType
}
```

Related Session

What’s New in Foundation	Hall 2	Wednesday 11:00AM

What’s New in Foundation

What’s New in Foundation | Hall 2 | Wednesday 11:00AM

What Else is New in Foundation

What Else is New in Foundation

New and enhanced API

Available on both macOS and iOS

NSXPCConnection

Can advertise intent to publish progress

NSXPCConnection

Can advertise intent to publish progress

```
@protocol MyServerProtocol
-(void)performRequestWithReply:(void(^)(BOOL success))reply;
@end
```

NSXPCConnection

Can advertise intent to publish progress

```
@protocol MyServerProtocol
-(NSProgress *)performRequestWithReply:(void(^)(BOOL success))reply;
@end
```

NSXPCConnection

Can advertise intent to publish progress

```
@protocol MyServerProtocol
-(NSProgress *)performRequestWithReply:(void(^)(BOOL success))reply;
@end
```

NSXPCConnection

Can advertise intent to publish progress

```
@protocol MyServerProtocol
-(NSProgress *)performRequestWithReply:(void(^)(BOOL success))reply;
@end
```

Return progress before you reply

NSXPCConnection

Can advertise intent to publish progress

```
@protocol MyServerProtocol
-(NSProgress *)performRequestWithReply:(void(^)(BOOL success))reply;
@end
```

Return progress before you reply

Updates performed asynchronously

URLSession

URLSession

```
var sessionConfig = URLSessionConfiguration.default
sessionConfig.waitsForConnectivity = true
```

URLSession

```
var sessionConfig = URLSessionConfiguration.default
sessionConfig.waitsForConnectivity = true
```

Multipath TCP now available on iOS

URLSession

```
var sessionConfig = URLSessionConfiguration.default
sessionConfig.waitsForConnectivity = true
```

Multipath TCP now available on iOS

URLSessionTask supports NSProgressReporting protocol

URLSession

```
var sessionConfig = URLSessionConfiguration.default
sessionConfig.waitsForConnectivity = true
```

Multipath TCP now available on iOS

URLSessionTask supports NSProgressReporting protocol

NSFileProviderService

NSFileProviderService

Apps ↔ File Providers

NSFileProviderService

Apps ↔ File Providers

Discover file providers for any URL with NSFileManager

NSFileProviderService

Apps ↔ File Providers

Discover file providers for any URL with NSFileManager

Apps can use file provider’s specialized services

NSFileProviderService

Apps ↔ File Providers

Discover file providers for any URL with NSFileManager

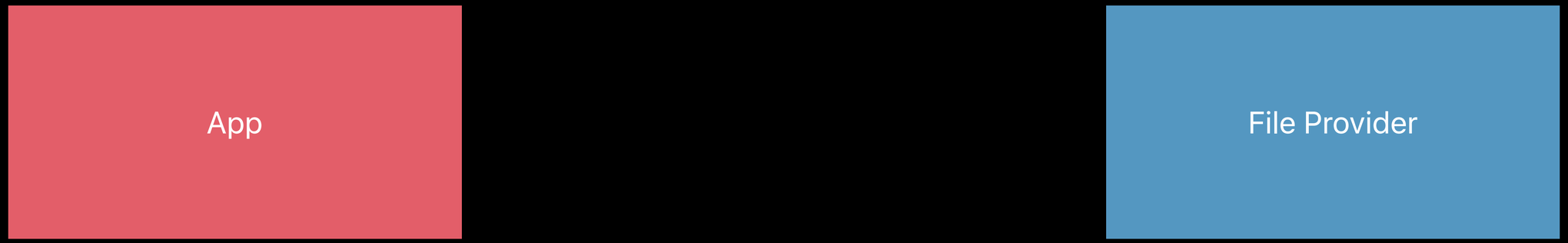
Apps can use file provider’s specialized services

File Provider Communication

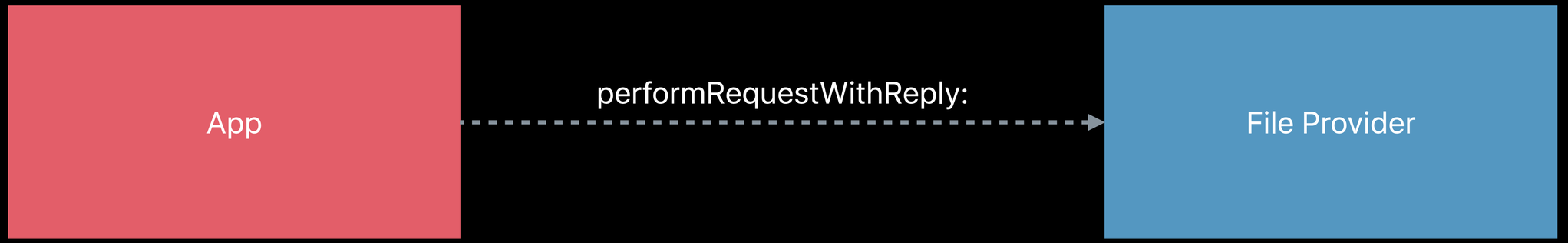
File Provider Communication

```
@protocol MyServerProtocol
-(NSProgress *)performRequestWithReply:(void(^)(BOOL success))reply;
@end
```

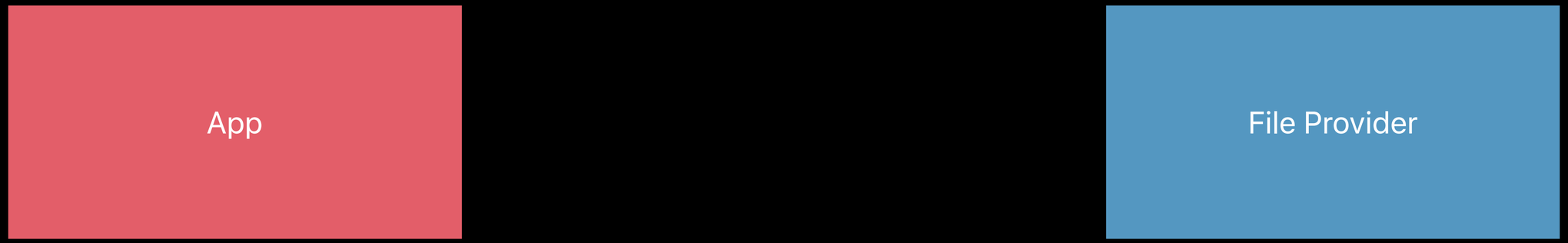
File Provider Communication



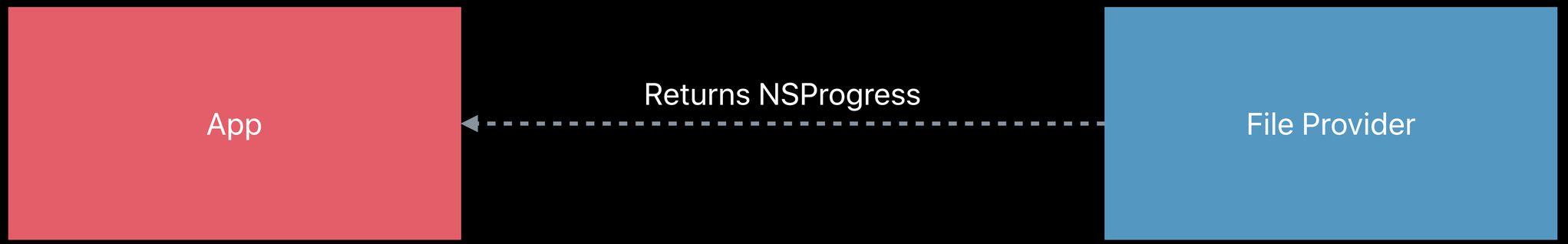
File Provider Communication



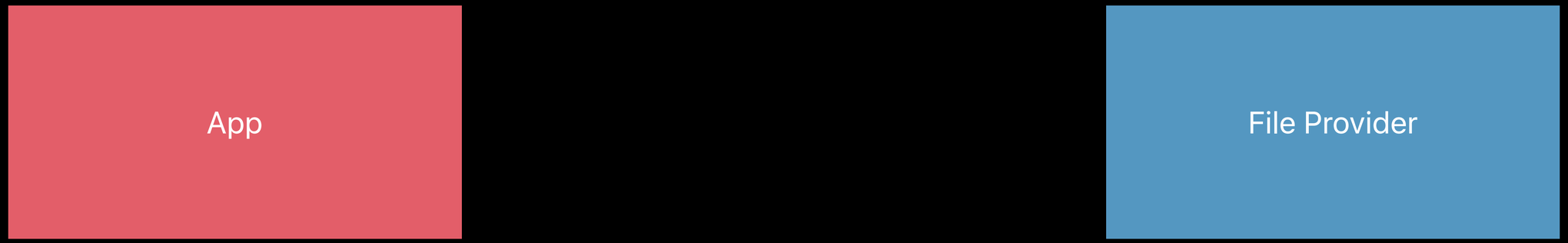
File Provider Communication



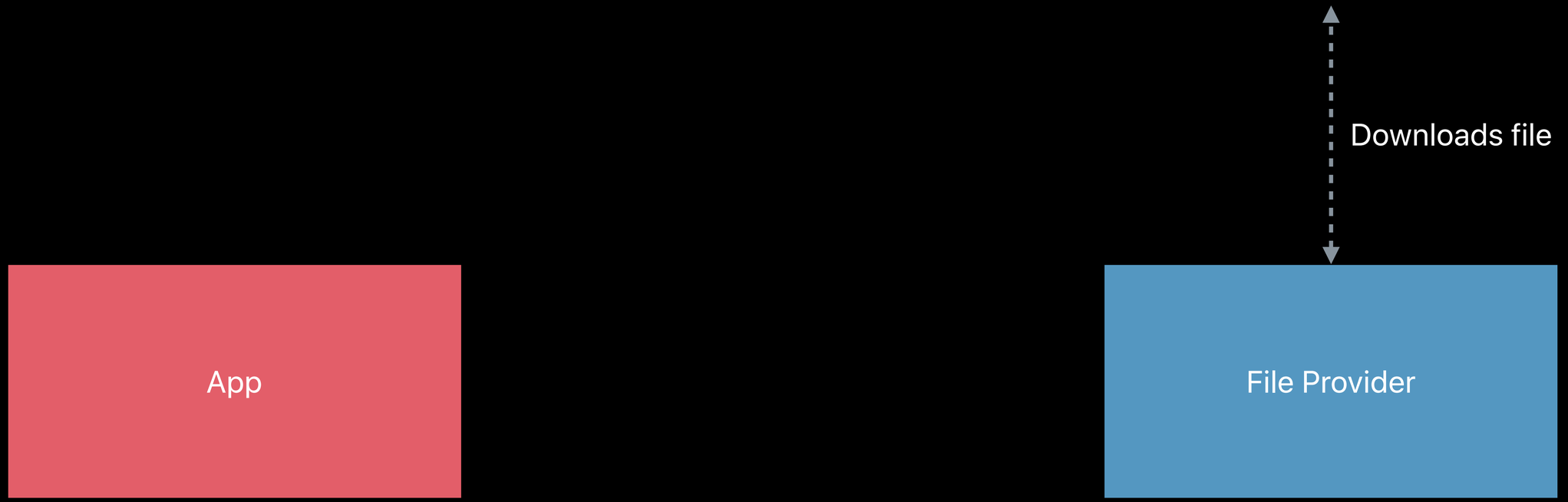
File Provider Communication



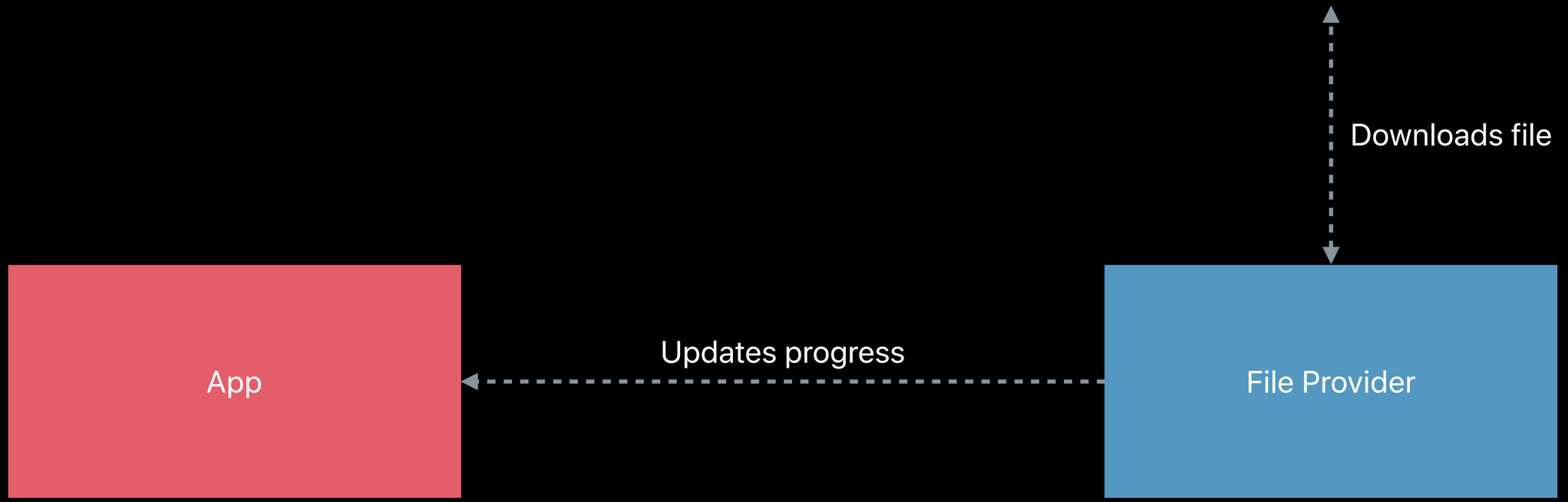
File Provider Communication



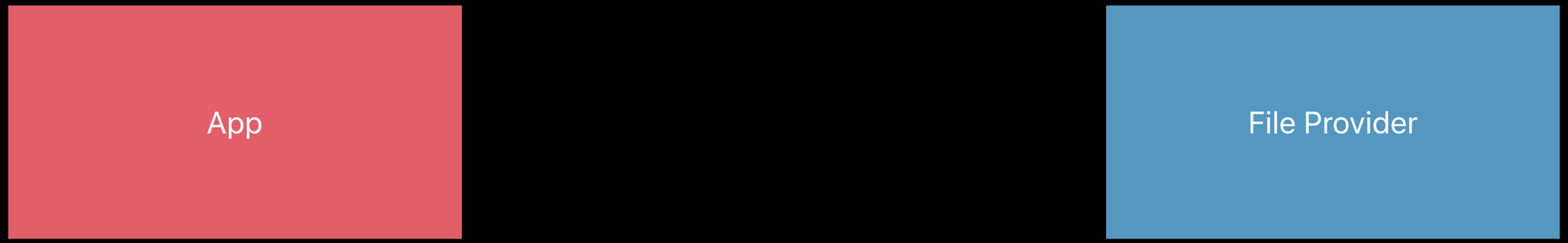
File Provider Communication



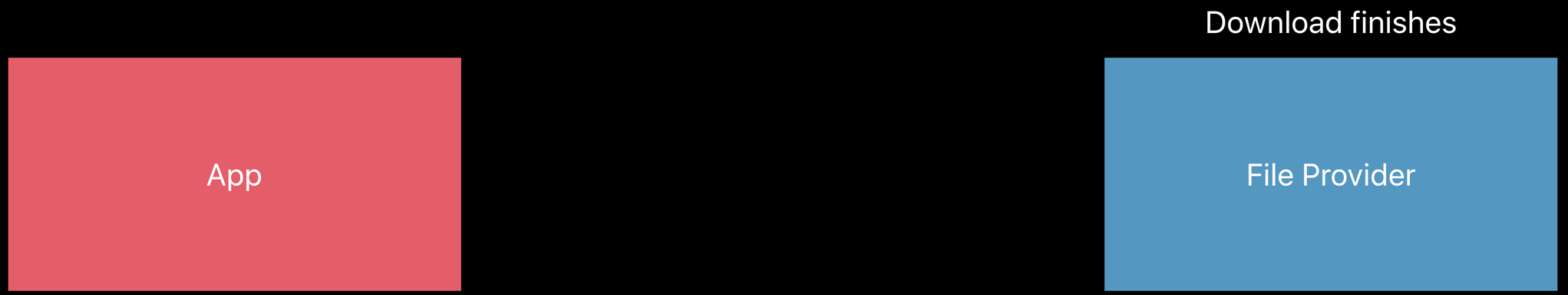
File Provider Communication



File Provider Communication

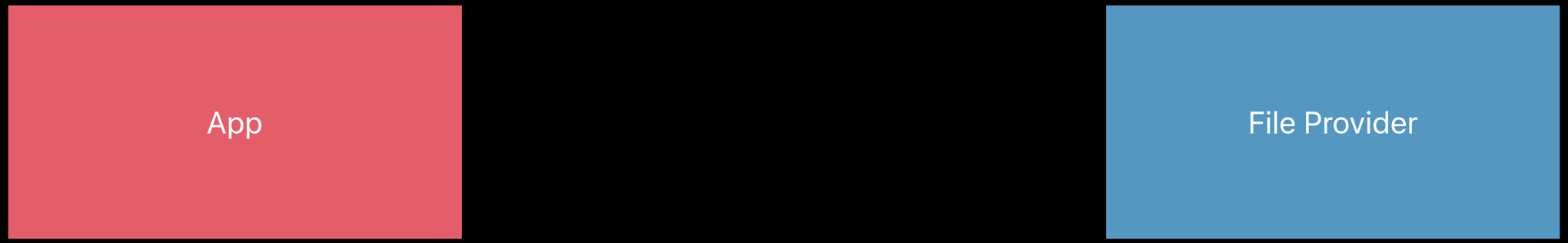


File Provider Communication

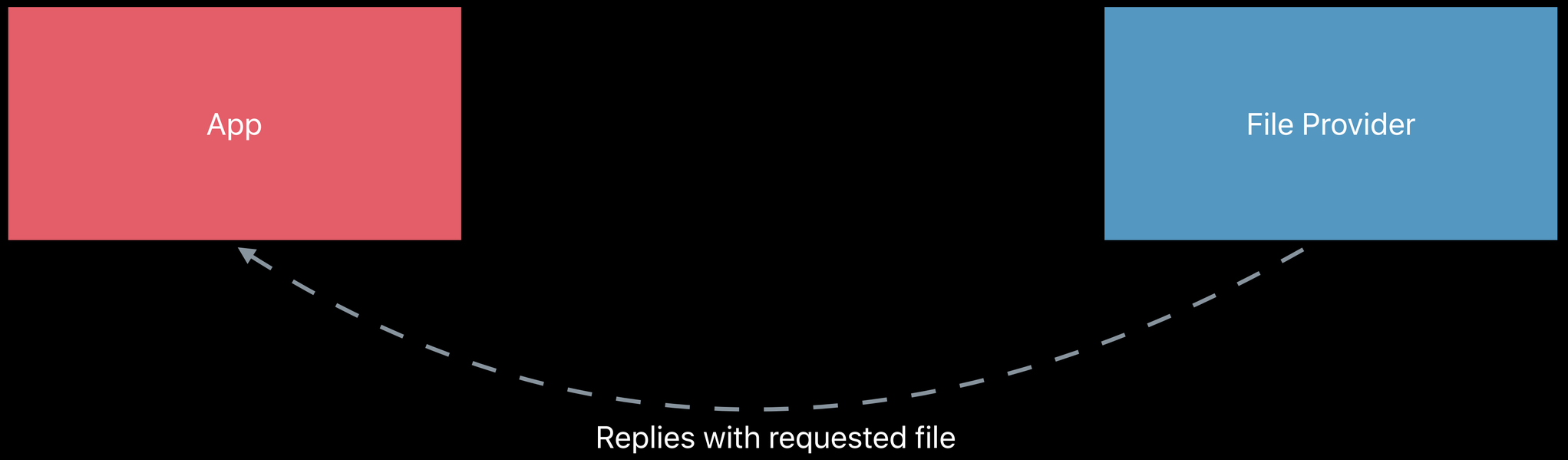


File Provider Communication

File Provider Communication



File Provider Communication



Improved Available Storage Space API

Check for available space more accurately

```
public struct URLResourceValues {
  public var volumeAvailableCapacityForImportantUsage: Int64? { get }
  public var volumeAvailableCapacityForOpportunisticUsage: Int64? { get }
}
```

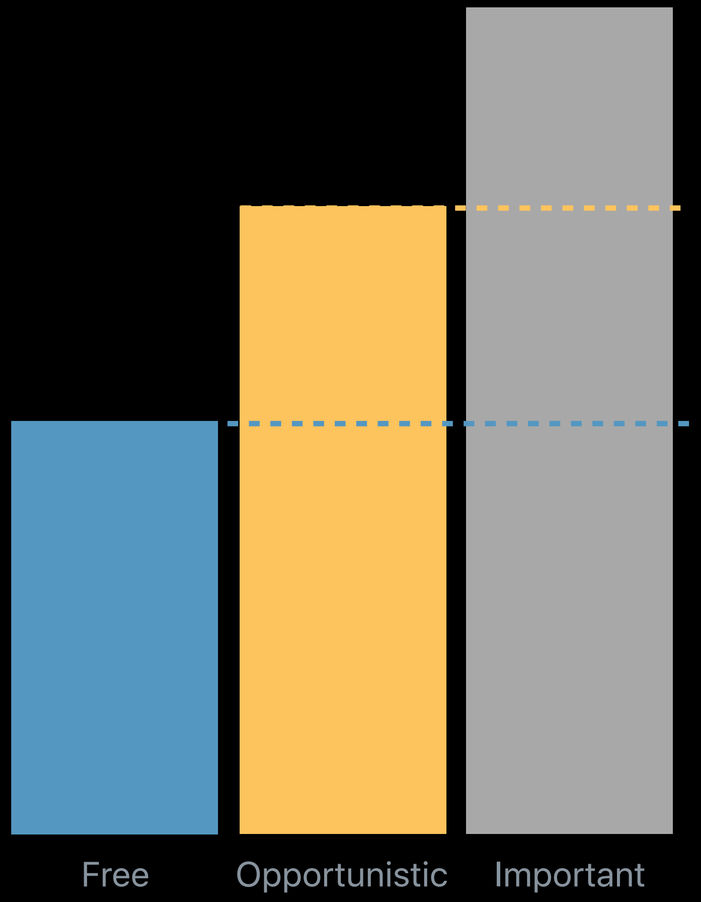
Improved Available Storage Space API

Free Opportunistic Important

Improved Available Storage Space API

Improved Available Storage Space API

Improved Available Storage Space API



Improved Available Storage Space API

Preflight check

```
let fileURL = URL(fileURLWithPath: "/Path/to/file")
let set: Set<URLResourceKey> = [URLResourceKey.volumeAvailableCapacityForOpportunisticUsageKey]
if let fileResourceValues = try? fileURL.resourceValues(forKeys: set) {
  if (fileResourceValues.volumeAvailableCapacityForOpportunisticUsageKey > 50000000) {
    // download additional file
  }
}
```

Improved Available Storage Space API

Preflight check

```
let fileURL = URL(fileURLWithPath: "/Path/to/file")
let set: Set<URLResourceKey> = [URLResourceKey.volumeAvailableCapacityForOpportunisticUsageKey]
if let fileResourceValues = try? fileURL.resourceValues(forKeys: set) {
  if (fileResourceValues.volumeAvailableCapacityForOpportunisticUsageKey > 50000000) {
    // download additional file
  }
}
```

NSLinguisticTagger

NSLinguisticTagger

Tagging a unit

NSLinguisticTagger

Tagging a unit

```
"Foundation is really cool. We write lots of code.\n"
```

NSLinguisticTagger

Tagging a unit

```
"Foundation is really cool. We write lots of code.\n"
```

Word

NSLinguisticTagger

Tagging a unit

```
"Foundation is really cool. We write lots of code.\n"
```

Sentence

NSLinguisticTagger

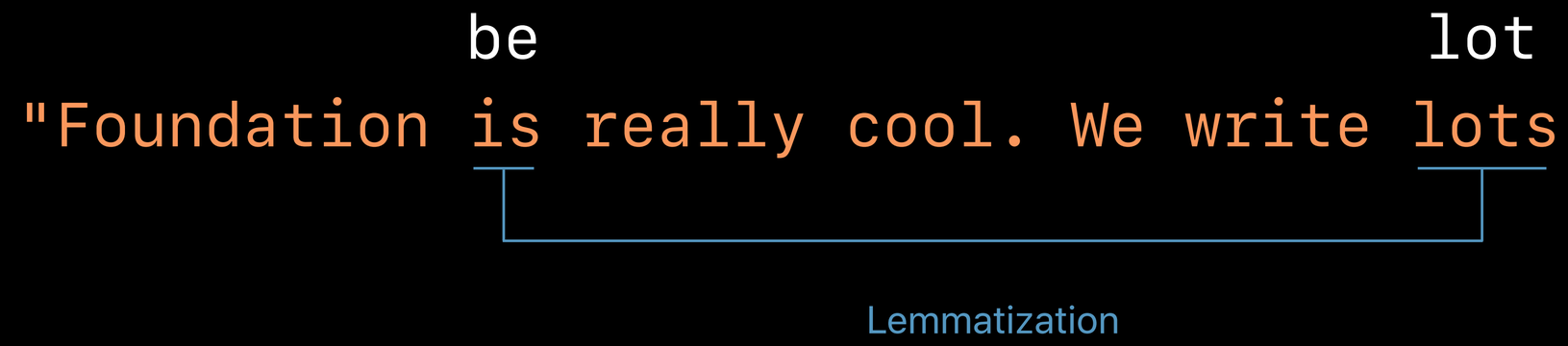
Tagging a unit

```
"Foundation is really cool. We write lots of code.\n"
```

Paragraph

NSLinguisticTagger

Tagging a unit



NSLinguisticTagger

NSLinguisticTagger

Redone and improved

NSLinguisticTagger

Redone and improved

Common case convenience methods added

NSLinguisticTagger

Redone and improved

Common case convenience methods added

```
let text = "Ich weiß nicht was soll es bedeuten."
let language = NSLinguisticTagger.dominantLanguage(for: text)  // "de"
```

NSLinguisticTagger

Redone and improved

Common case convenience methods added

```
let text = "Ich weiß nicht was soll es bedeuten."
let language = NSLinguisticTagger.dominantLanguage(for: text)  // "de"
```

JSONSerialization

Supports sorted keys

JSONSerialization

Supports sorted keys

```
let dict = ["a": 1, "b": 2, "c": 3, "d": 4]
```

JSONSerialization

Supports sorted keys

```
let dict = ["a": 1, "b": 2, "c": 3, "d": 4]

var data = try JSONSerialization.data(withJSONObject: dict, options: [])
// {"b":2,"a":1,"c":3,"d":4}
```

JSONSerialization

Supports sorted keys

```
let dict = ["a": 1, "b": 2, "c": 3, "d": 4]

var data = try JSONSerialization.data(withJSONObject: dict, options: [])
// {"b":2,"a":1,"c":3,"d":4}

data = try JSONSerialization.data(withJSONObject: dict, options: .sortedKeys)
```

JSONSerialization

Supports sorted keys

```
let dict = ["a": 1, "b": 2, "c": 3, "d": 4]

var data = try JSONSerialization.data(withJSONObject: dict, options: [])
// {"b":2,"a":1,"c":3,"d":4}

data = try JSONSerialization.data(withJSONObject: dict, options: .sortedKeys)
// {"a":1,"b":3,"c":4,"d":2}
```

NSItemProvider

More explicit operations

Supports progress and cancellation

Enhance your custom classes to work with multiple representations

NSItemProvider

More explicit operations

Supports progress and cancellation

Enhance your custom classes to work with multiple representations

NSUserActivity

```
open class NSUserActivity : NSObject {
  open var webpageURL: URL?
```

NSUserActivity

```
open class NSUserActivity : NSObject {
  open var webpageURL: URL?
  open var referrerURL: URL? // Set URL of webpage that referred webpageURL
}
```

Failable APIs

Better error handling

Additional convenience methods that take URLs

Process Launching

```
// NSTask
+(nullable NSTask *)launchedTaskWithExecutableURL:(NSURL *)url
                                        arguments:(NSArray<NSString *> *)arguments
                                            error:(out NSError ** _Nullable)error
                                terminationHandler:(void(^_Nullable)(NSTask *))terminationHandler;
// Process, Swift 4
open class func run(_ url: URL,
                    arguments: [String],
                    terminationHandler: ((Process) -> Void)? = nil) throws -> Process
```

NSDictionary and NSArray

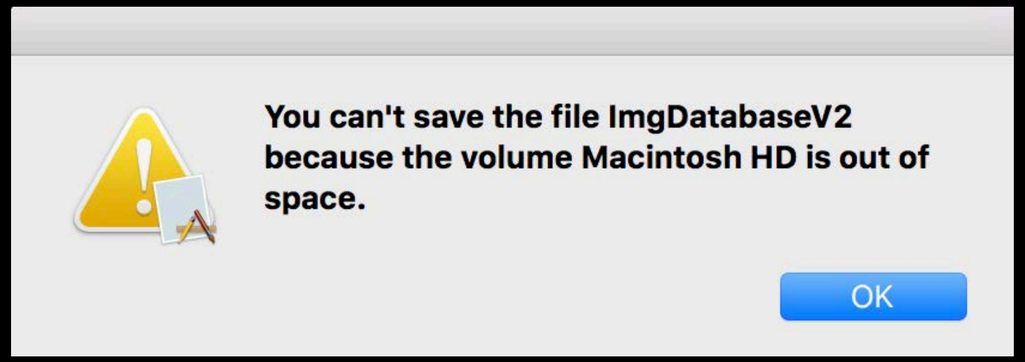
```
+(nullable NSDictionary<NSString*, ObjectType>*)dictionaryWithContentsOfURL:(NSURL *)url
                                                                      error:(NSError **)error;
+(nullable NSArray<ObjectType>*)arrayWithContentsOfURL:(NSURL *)url
                                                 error:(NSError **)error;

-(BOOL)writeToURL:(NSURL *)url error:(NSError **)error;
```

NSLocalizedFailureErrorKey

Customize “what failed” while keeping “why it failed”

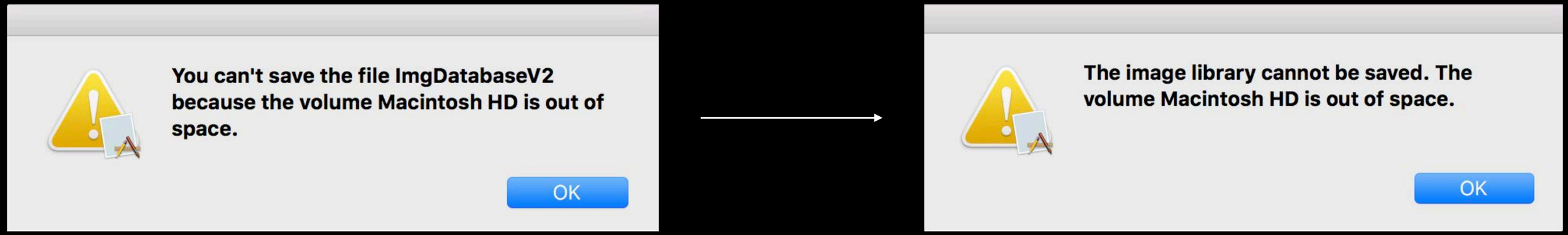
NSLocalizedFailureErrorKey

Customize “what failed” while keeping “why it failed”



NSLocalizedFailureErrorKey

Customize “what failed” while keeping “why it failed”



NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""
```

NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""

let nsrange = NSRange(string.indices, in: string)
for match in re.matches(in: string, range: nsrange) {
```

NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""

let nsrange = NSRange(string.indices, in: string)
for match in re.matches(in: string, range: nsrange) {
```

NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""

let nsrange = NSRange(string.indices, in: string)
for match in re.matches(in: string, range: nsrange) {
```

NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""

let nsrange = NSRange(string.indices, in: string)
for match in re.matches(in: string, range: nsrange) {
```

NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""

let nsrange = NSRange(string.indices, in: string)
for match in re.matches(in: string, range: nsrange) {
```

NSRange ↔ Range

```
let greeting: String = "hello there swift"
let re = NSRegularExpression(pattern: "[sw]ift")
var found: String = ""

let nsrange = NSRange(string.indices, in: string)
for match in re.matches(in: string, range: nsrange) {
  found.append(Range(match.rangeAt(1), in: string)!)
}
```

Performance Improvements

Performance Improvements

Copy-on-write NSArray, NSDictionary, NSSet

Performance Improvements

Copy-on-write NSArray, NSDictionary, NSSet

struct Data inlining

Performance Improvements

Copy-on-write NSArray, NSDictionary, NSSet

struct Data inlining

Faster calendrical calculations with lower peak memory

Performance Improvements

Copy-on-write NSArray, NSDictionary, NSSet

struct Data inlining

Faster calendrical calculations with lower peak memory

Faster bridging of NSNumber to and from Swift

Performance Improvements

Copy-on-write NSArray, NSDictionary, NSSet

struct Data inlining

Faster calendrical calculations with lower peak memory

Faster bridging of NSNumber to and from Swift

NSArchiver and NSUnarchiver

NSArchiver and NSUnarchiver replaced in 10.4

Now formally deprecated

Old formats still supported

Use NSKeyedArchiver 🙏

Additional Mentions

Additional Mentions

Core Data highlights

Additional Mentions

Core Data highlights

Core Spotlight available on macOS

Additional Mentions

Core Data highlights

Core Spotlight available on macOS

Thermal Notifications available on iOS

Additional Mentions

Core Data highlights

Core Spotlight available on macOS

Thermal Notifications available on iOS

What’s New in Cocoa Touch	Hall 3	Tuesday 10:20AM
What’s New in Core Data	Grand Ballroom B	Wednesday 10:00AM
What's New in Core Spotlight for iOS and macOS	Grand Ballroom B	Thursday 4:10PM

Wrap-Up

API Refinements

AppKit

Foundation

Wrap-Up

API Refinements

AppKit

Foundation

Release Notes

More Information

https://developer.apple.com/wwdc17/207

Reminder

Tweet your Cocoa development tips!

Use `#WWDC17` and `#cocoatip`

Cocoa Development Tips	Hall 2	Friday 9:00AM

Related Sessions

What's New in Swift	Hall 3	Tuesday 1:50PM
Natural Language Processing and your Apps	Hall 3	Wednesday 9:00AM
What's New in Core Data	Grand Ballroom B	Wednesday 10:00AM
Touch Bar Fundamentals	Grand Ballroom A	Wednesday 10:00AM
What's New in Foundation	Hall 2	Wednesday 11:00AM
Choosing the Right Cocoa Container View	Grand Ballroom A	Wednesday 3:10PM
Advances in Networking, Part 2	Executive Ballroom	Wednesday 4:10PM

Related Sessions

Advanced Touch Bar	Grand Ballroom A	Wednesday 5:10PM
What's New in Core Spotlight for iOS and macOS	Grand Ballroom B	Thursday 4:10PM
Cocoa Development Tips	Hall 2	Friday 9:00AM
File Provider Enhancements	Hall 3	Friday 11:00AM
Efficient Interactions with Frameworks	Hall 2	Friday 1:50PM

Labs

Cocoa Lab	Technology Lab C	Wed 11:00AM-1:00PM
Foundation Lab	Technology Lab C	Wed 1:00-3:10PM
Core Data Lab	Technology Lab C	Wed 3:10-5:30PM
Cocoa Touch Bar Lab	Technology Lab C	Thu 9:00-11:00AM
Text and Fonts Lab	Technology Lab H	Thu 1:50-3:50PM
Core Data Lab	Technology Lab H	Thu 4:10-6:00PM
Cocoa Lab	Technology Lab B	Fri 1:50-3:30PM


WWDC17