Frameworks

#WWDC14

Adopting Handoff on iOS and OS X

Session 219

Michael Jurewitz
Engineering

Vince Spader
Cocoa Frameworks Engineer

Keith Stattenfield
CoreFrameworks Engineer

X

iOS

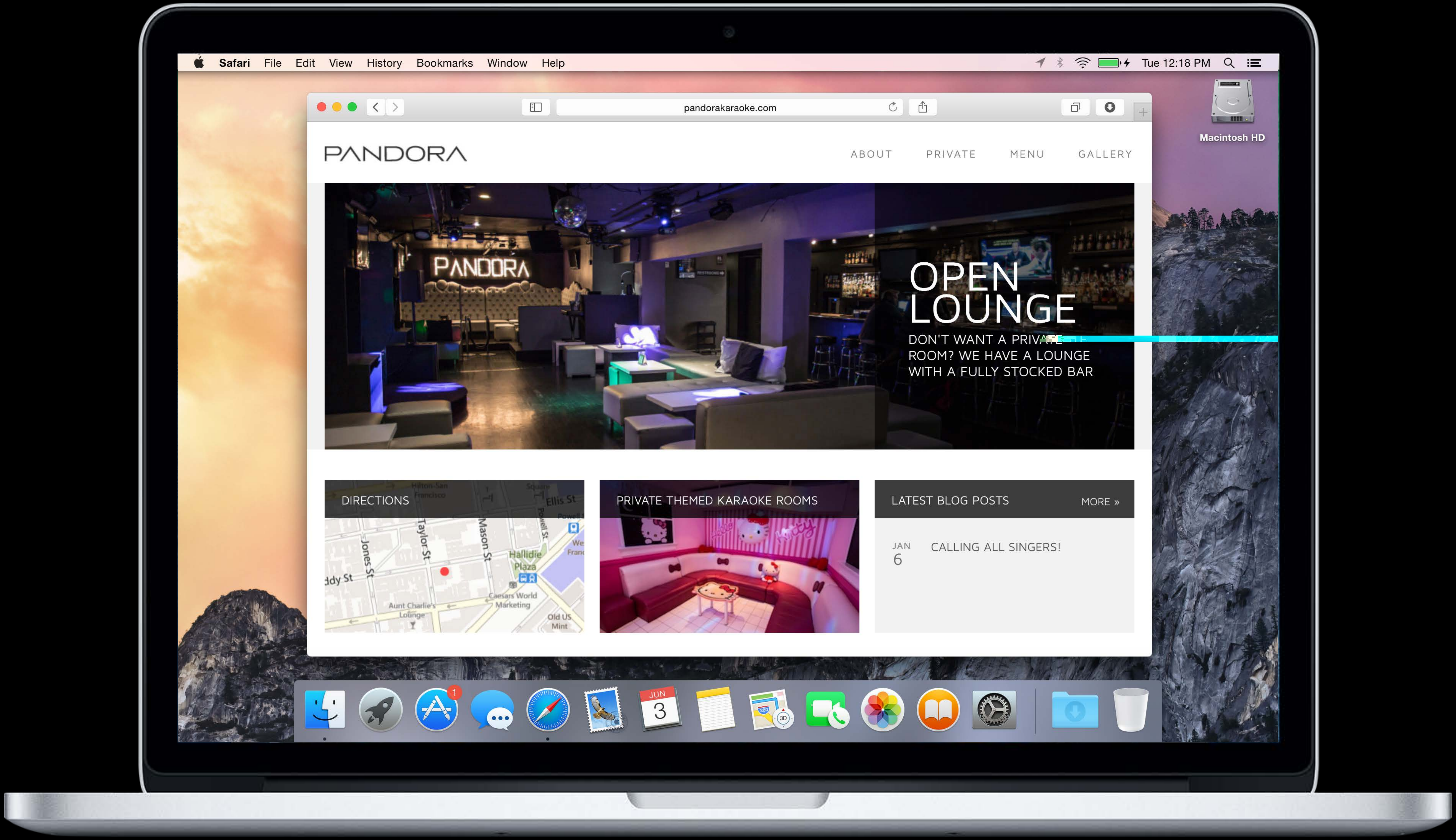
What You Will Learn

What is Handoff

Adopting Handoff in your app

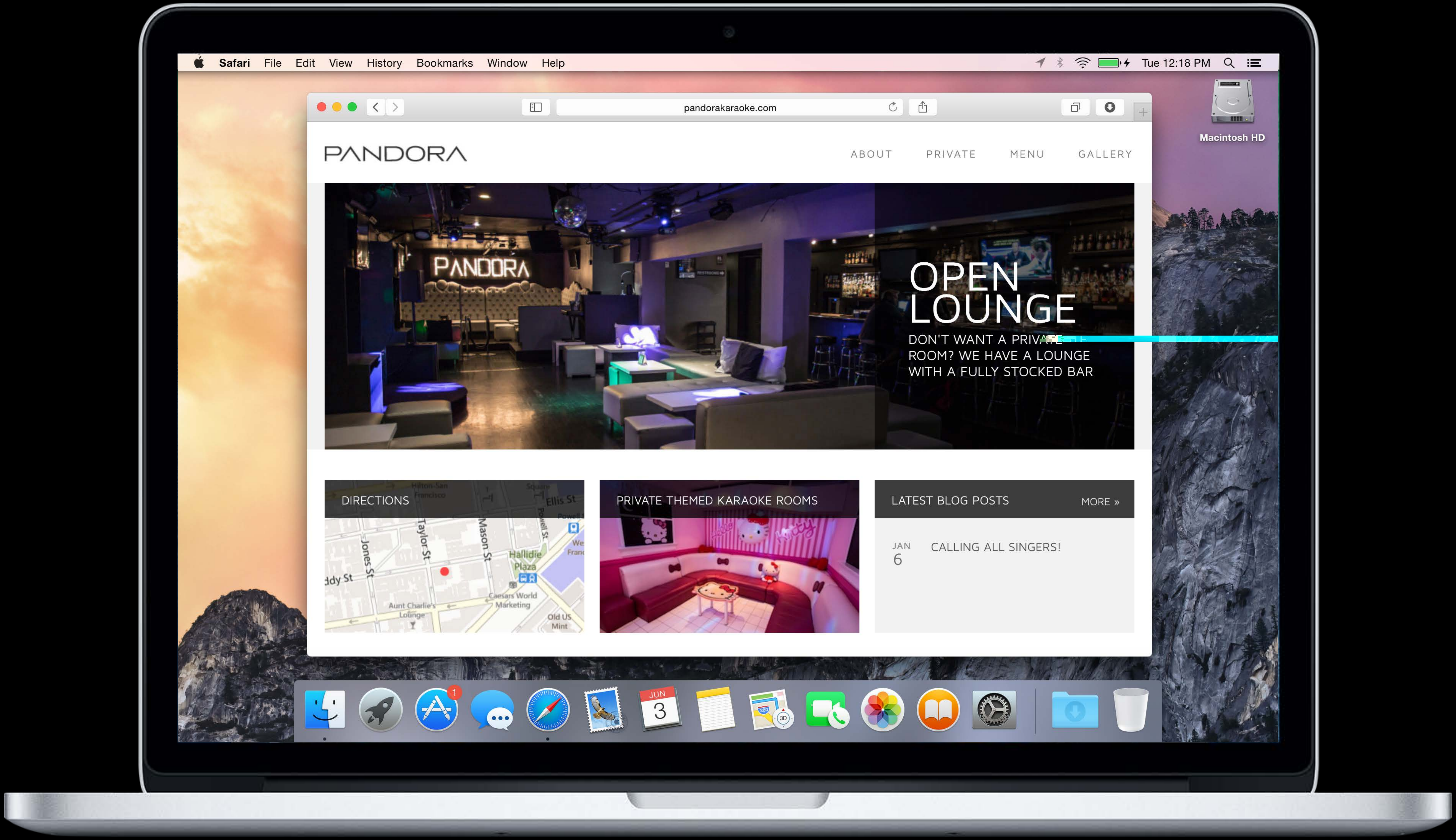
In-depth Handoff adoption

What is Handoff?



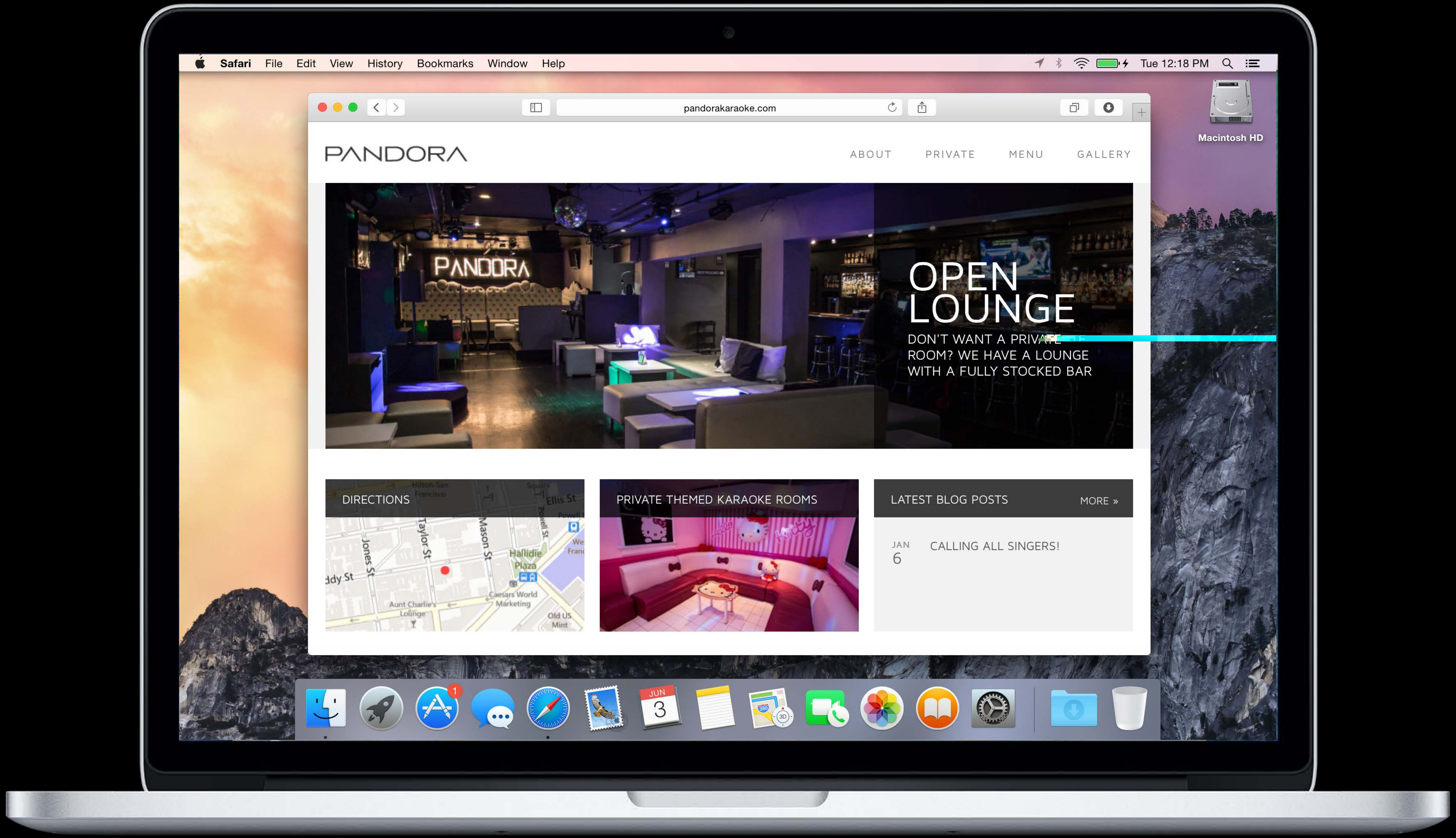
OPEN LOUNGE

DON'T WANT A PRIVATE ROOM? WE HAVE A LOUNGE WITH A FULLY STOCKED BAR

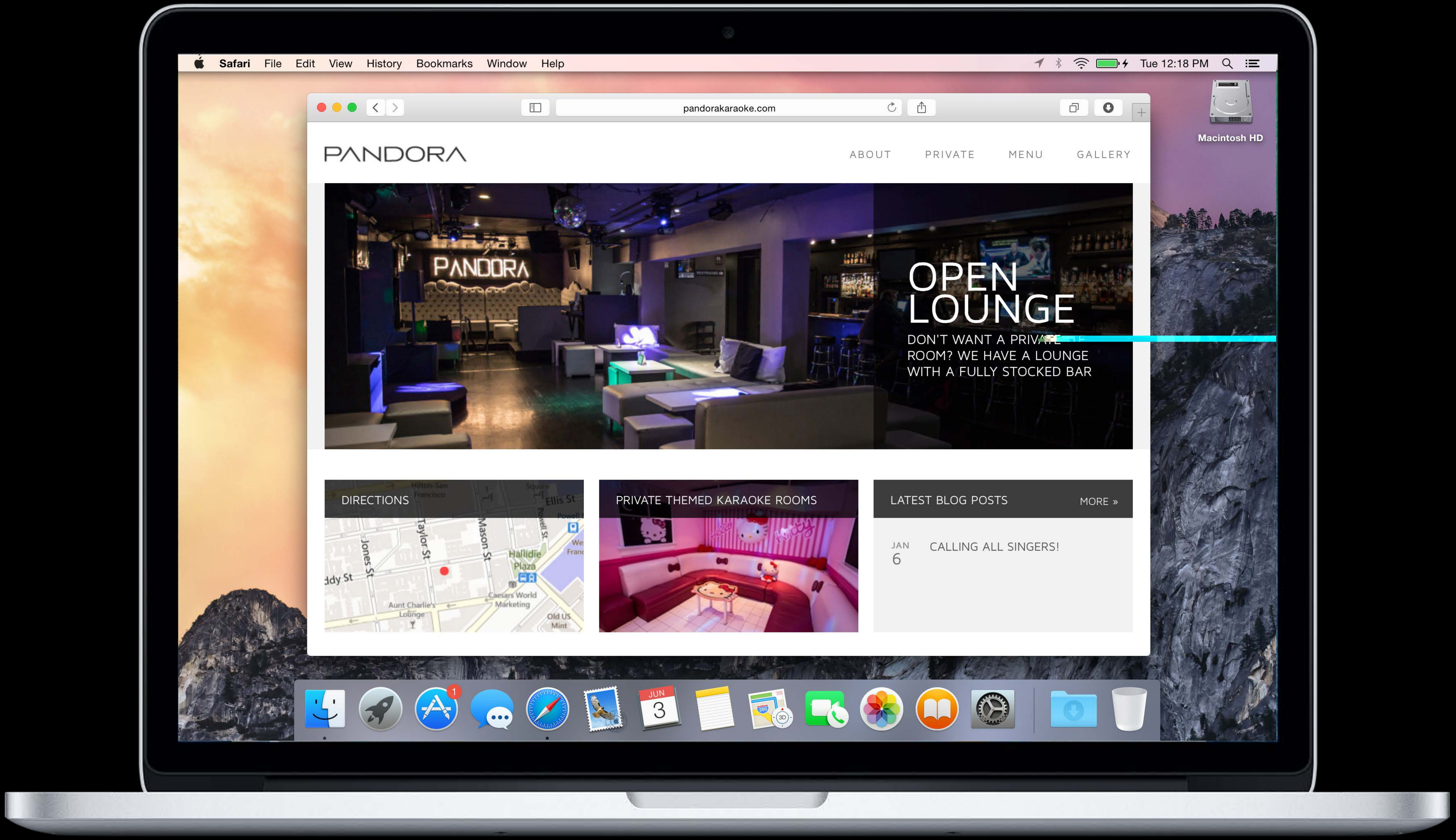


OPEN LOUNGE

DON'T WANT A PRIVATE ROOM? WE HAVE A LOUNGE WITH A FULLY STOCKED BAR



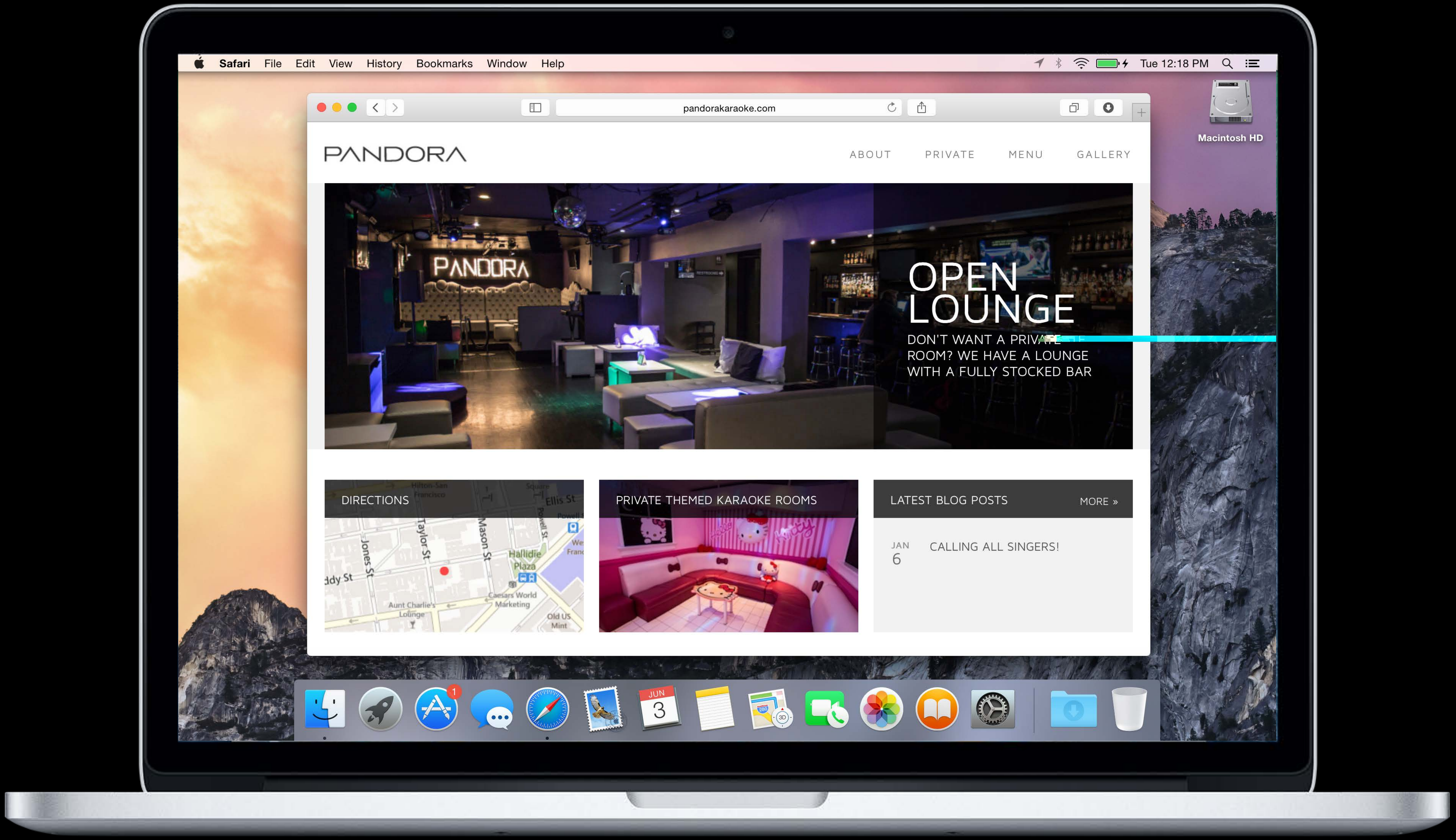
DON'T WANT A PRIVATE ROOM? WE HAVE A LOUNGE WITH A FULLY STOCKED BAR

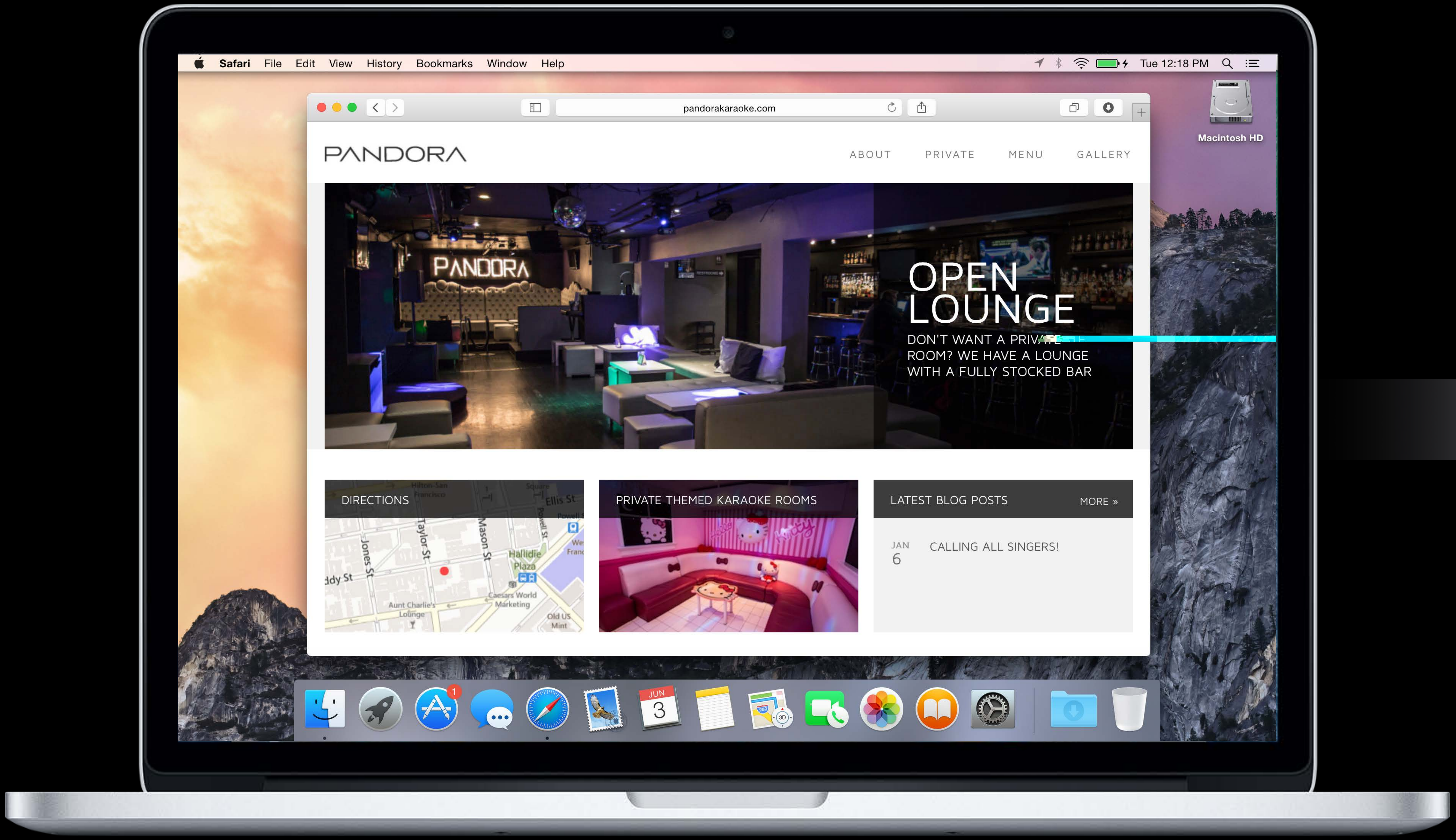


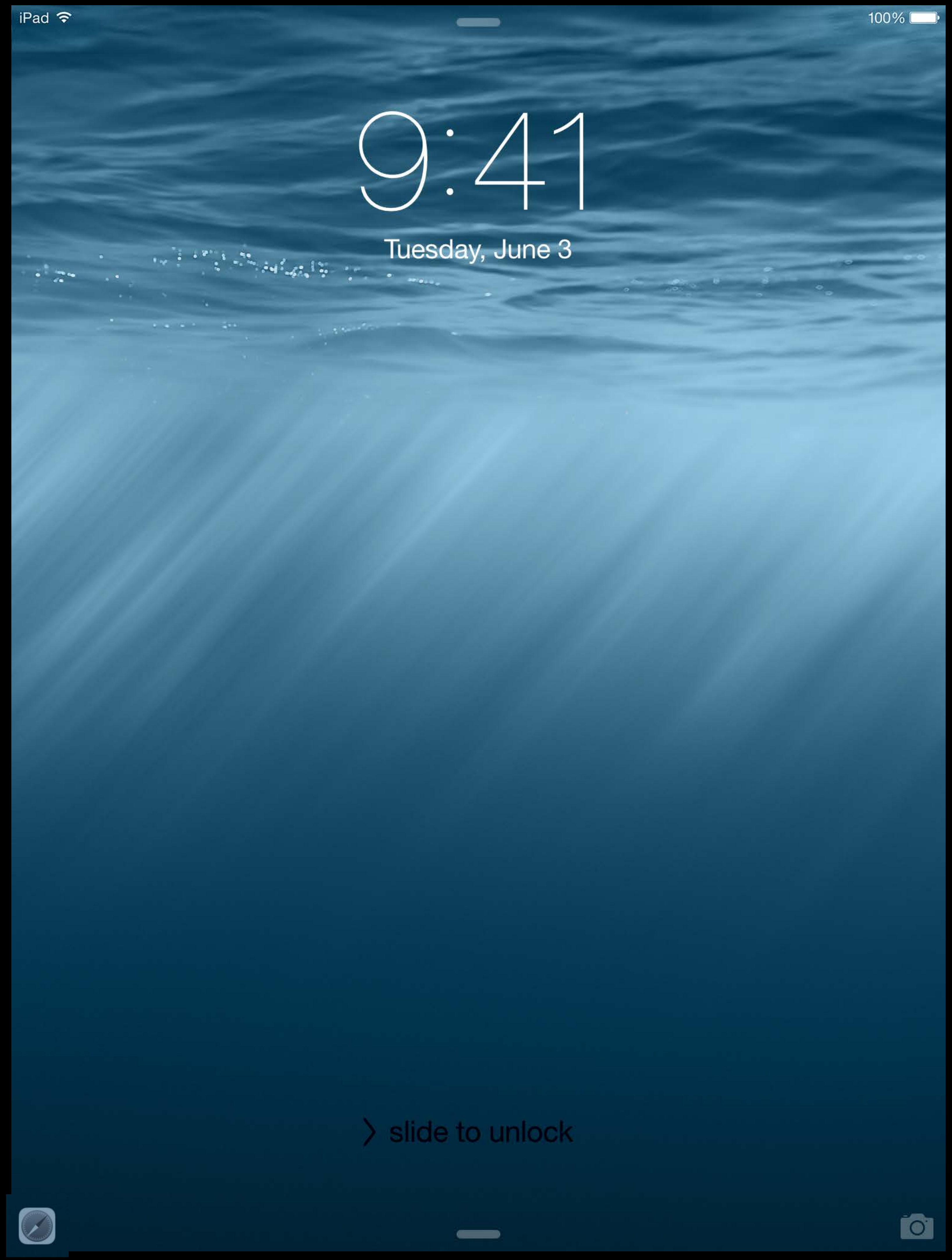
OPEN LOUNGE

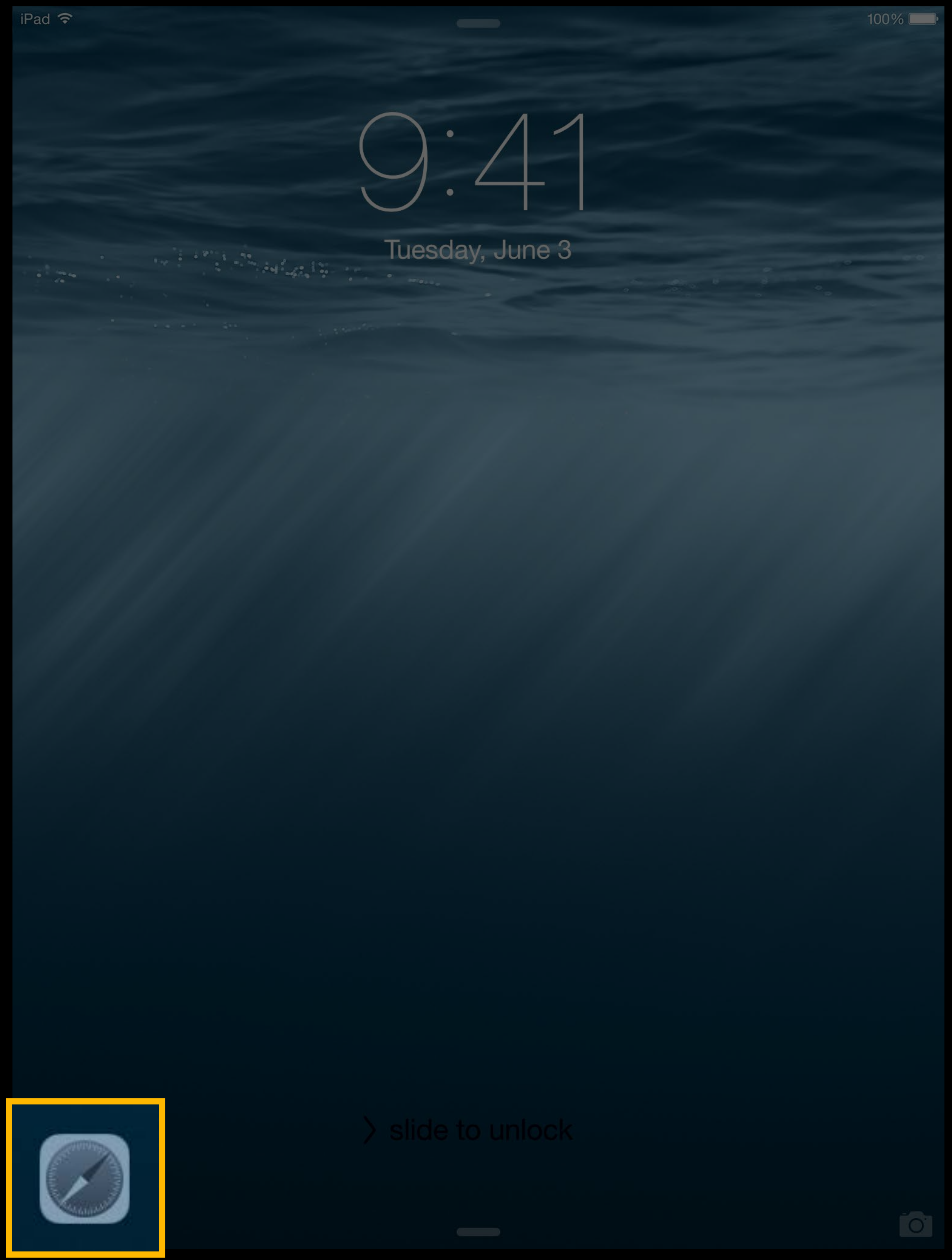
DON'T WANT A PRIVATE ROOM? WE HAVE A LOUNGE WITH A FULLY STOCKED BAR

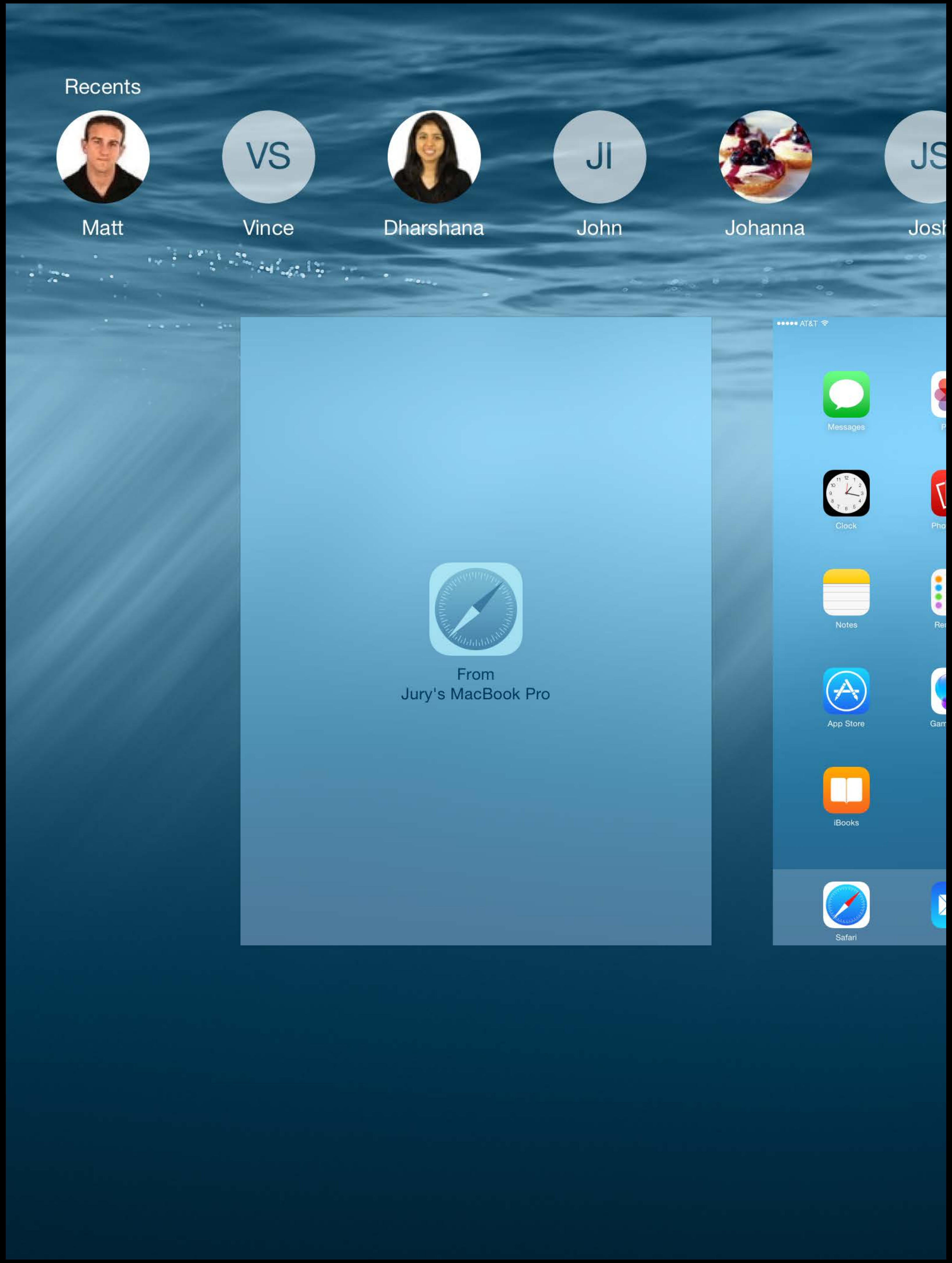
JAN 6 CALLING ALL SINGERS!


Safari File Edit View History Bookmarks Window Help
Tue 12:18 PM
Macintosh HD
pandorakaraoke.com
PANDORA
ABOUT
PRIVATE
MENU
GALLERY
OPEN LOUNGE
DON'T WANT A PRIVATE ROOM? WE HAVE A LOUNGE WITH A FULLY STOCKED BAR
DIRECTIONS
PRIVATE THEMED KARAOKE ROOMS
LATEST BLOG POSTS
MORE »
JAN 6
CALLING ALL SINGERS!


Safari File Edit View History Bookmarks Window Help
Tue 12:18 PM
Macintosh HD
pandorakaraoke.com
PANDORA
ABOUT
PRIVATE
MENU
GALLERY
OPEN LOUNGE
DON'T WANT A PRIVATE ROOM? WE HAVE A LOUNGE WITH A FULLY STOCKED BAR
DIRECTIONS
PRIVATE THEMED KARAOKE ROOMS
LATEST BLOG POSTS
MORE »
JAN 6
CALLING ALL SINGERS!


iPad
100%
9:41
Tuesday, June 3
slide to unlock


iPad
100%
9:41
Tuesday, June 3
slide to unlock


Recents
Matt
Vince
Dharshana
John
Johanna
Josh
From
Jury's MacBook Pro
Messages
Clock
Notes
App Store
iBooks
Safari

Mail from Jury's iPad

Adopting Handoff

Decide which activities to support in your app

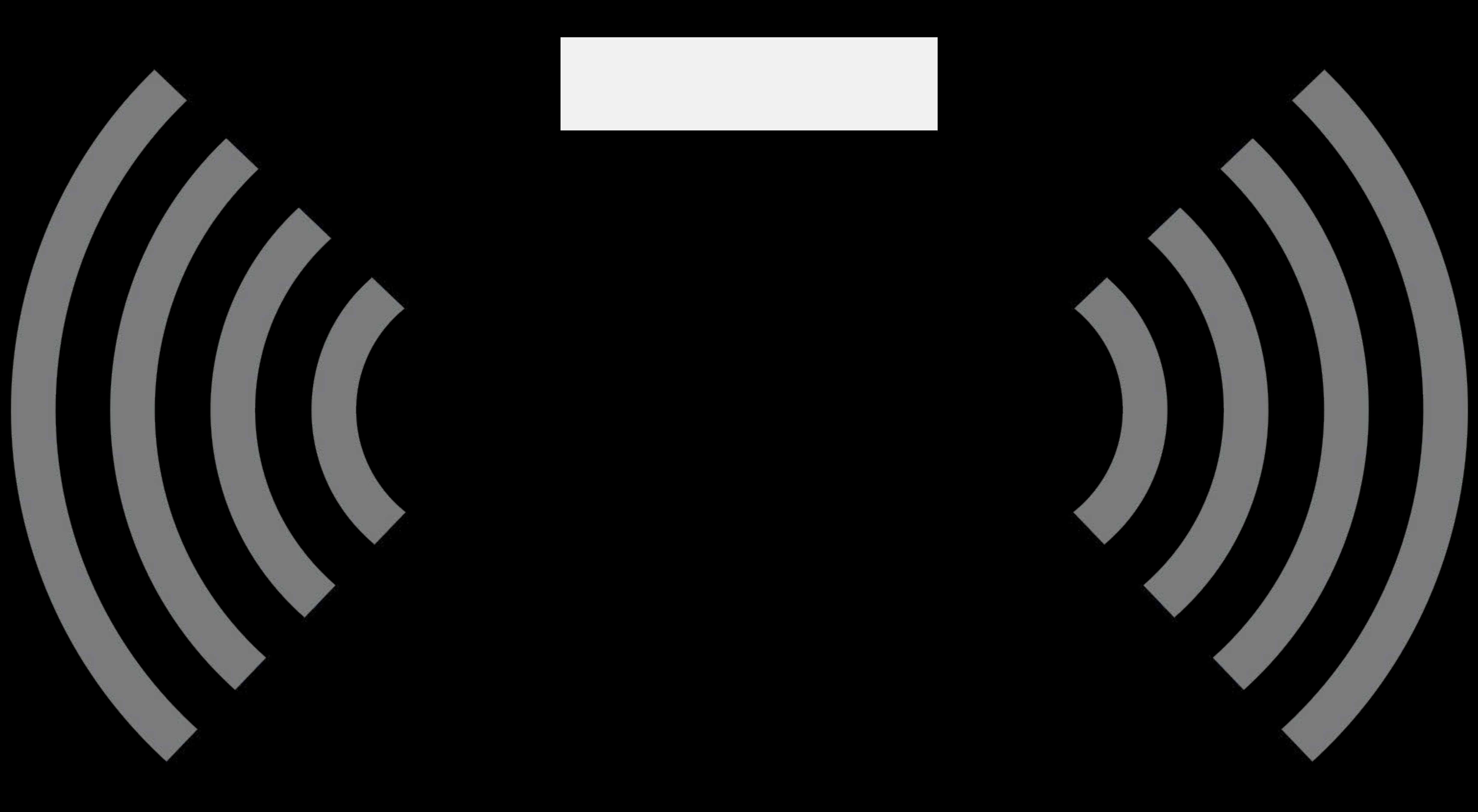
Create activities in specific parts of your app

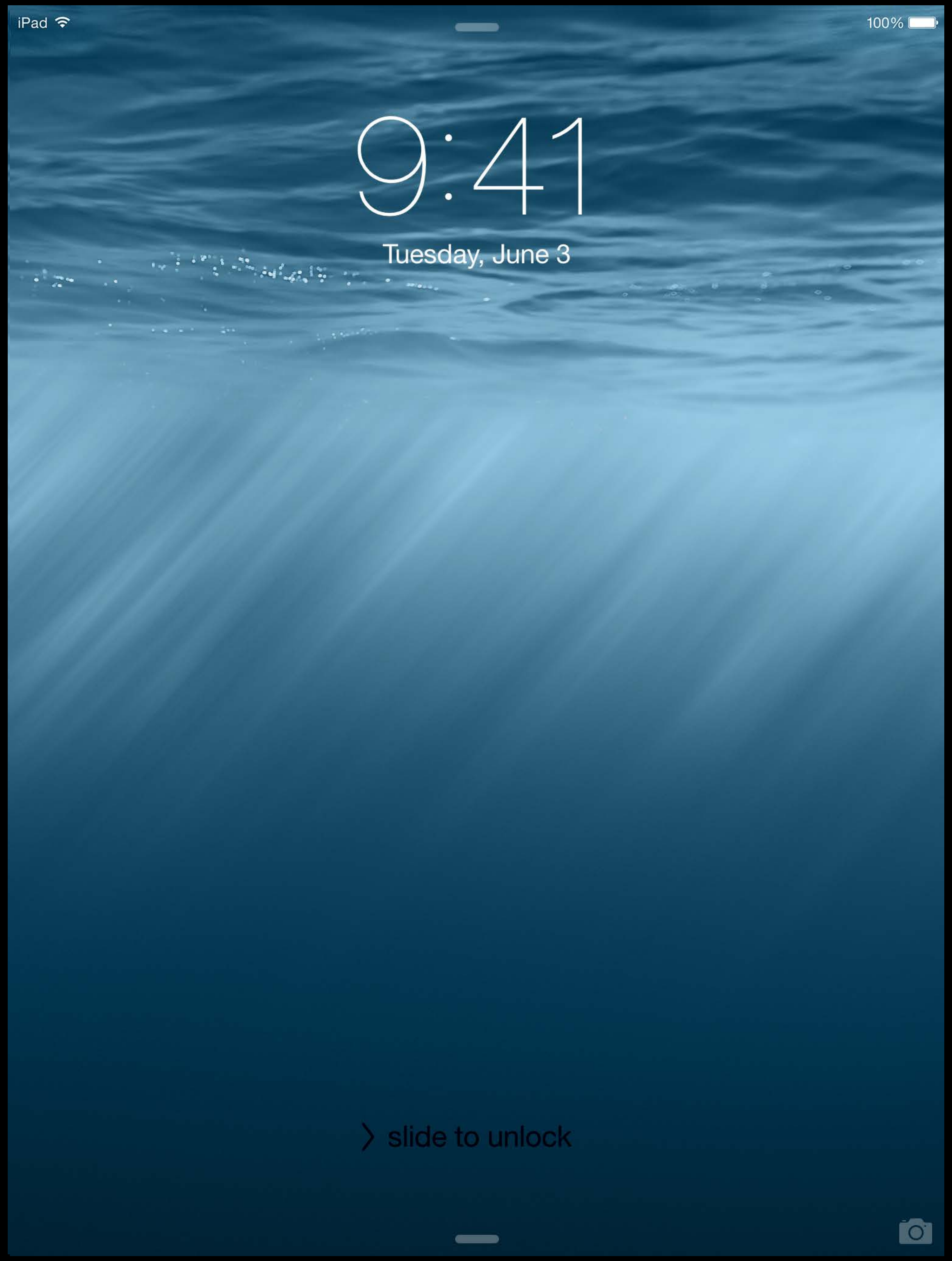
Handle continuing incoming activities in your app

Activity

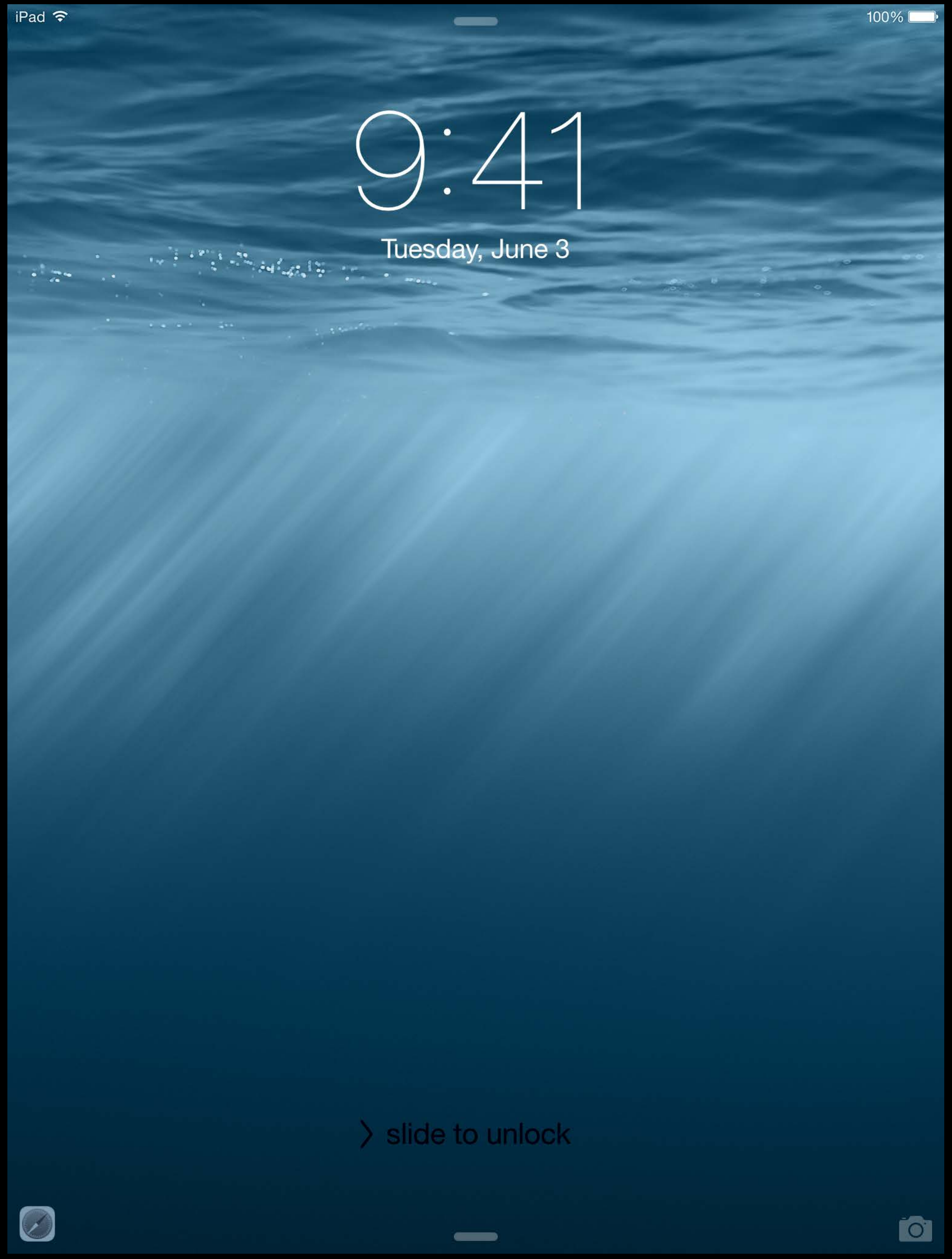
NSUserActivity

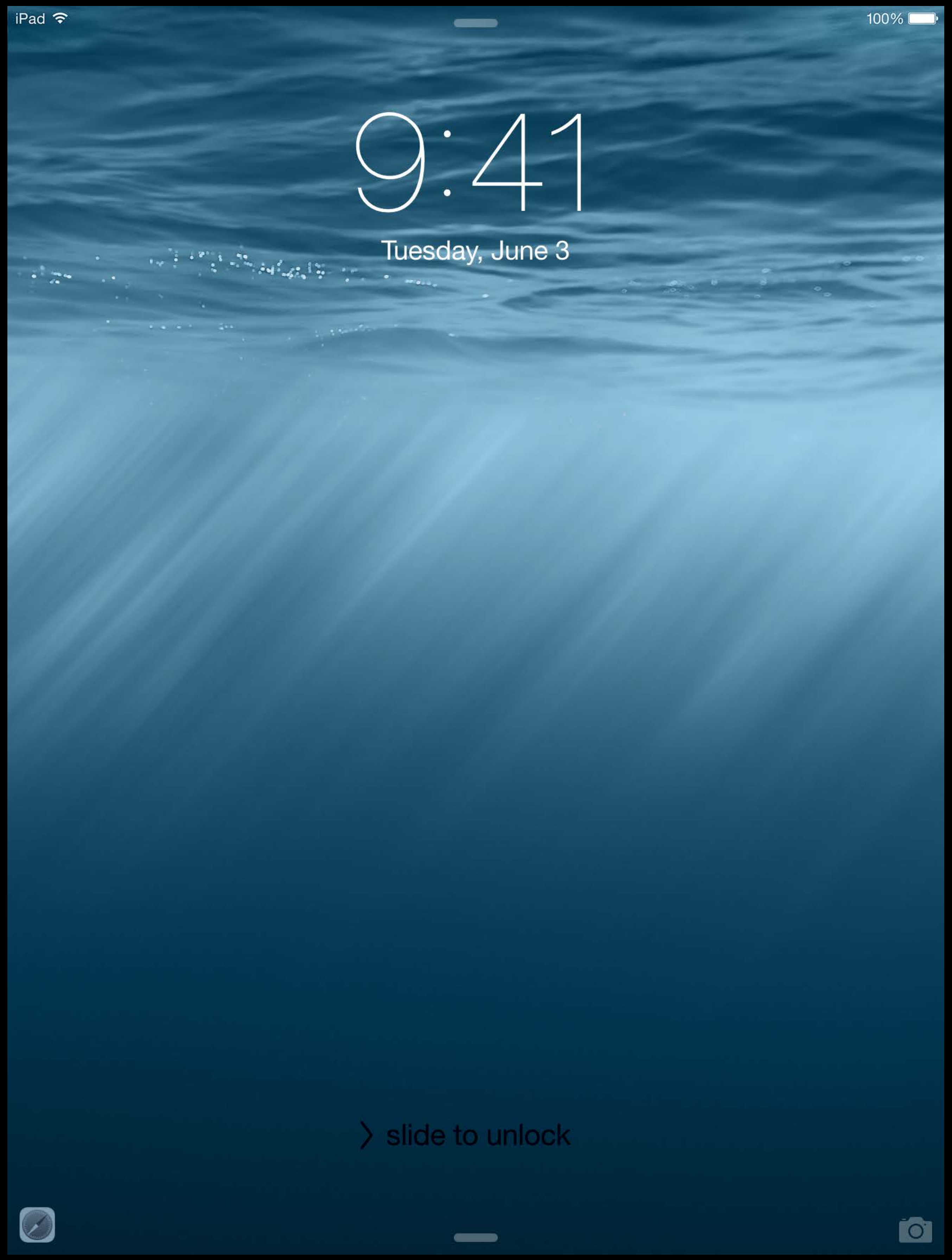
NSUserActivity

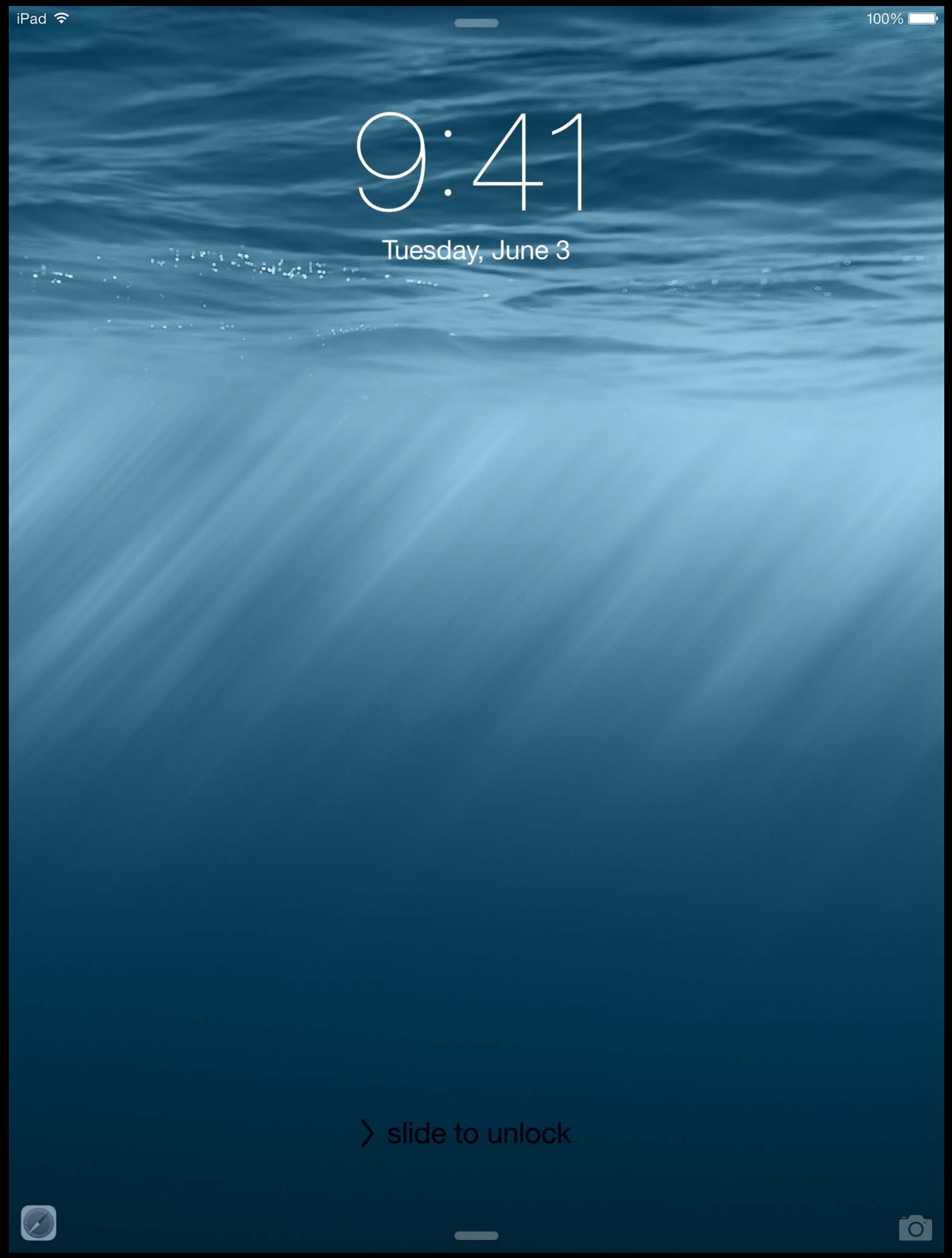


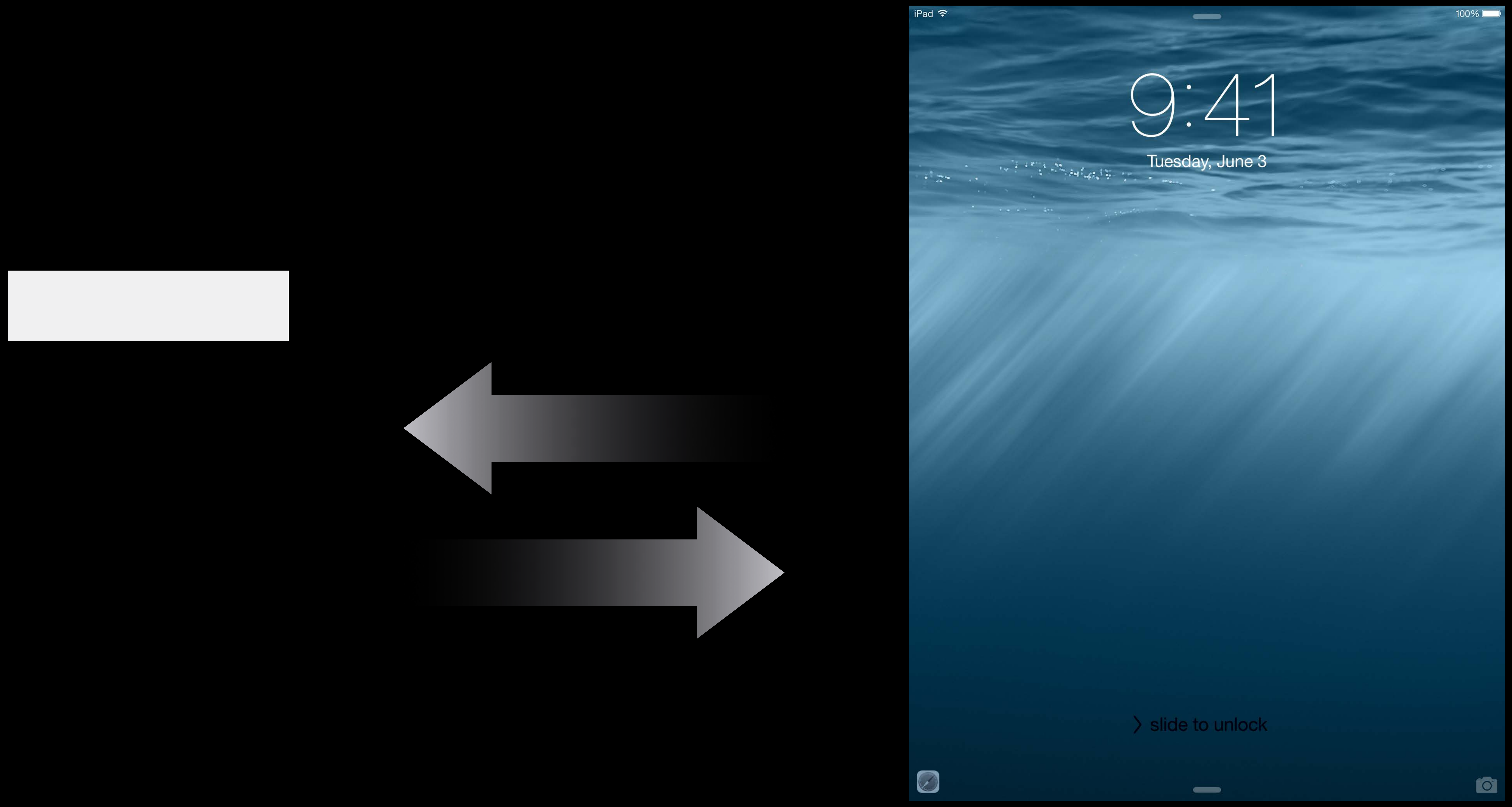

iPad
100%
9:41
Tuesday, June 3
slide to unlock

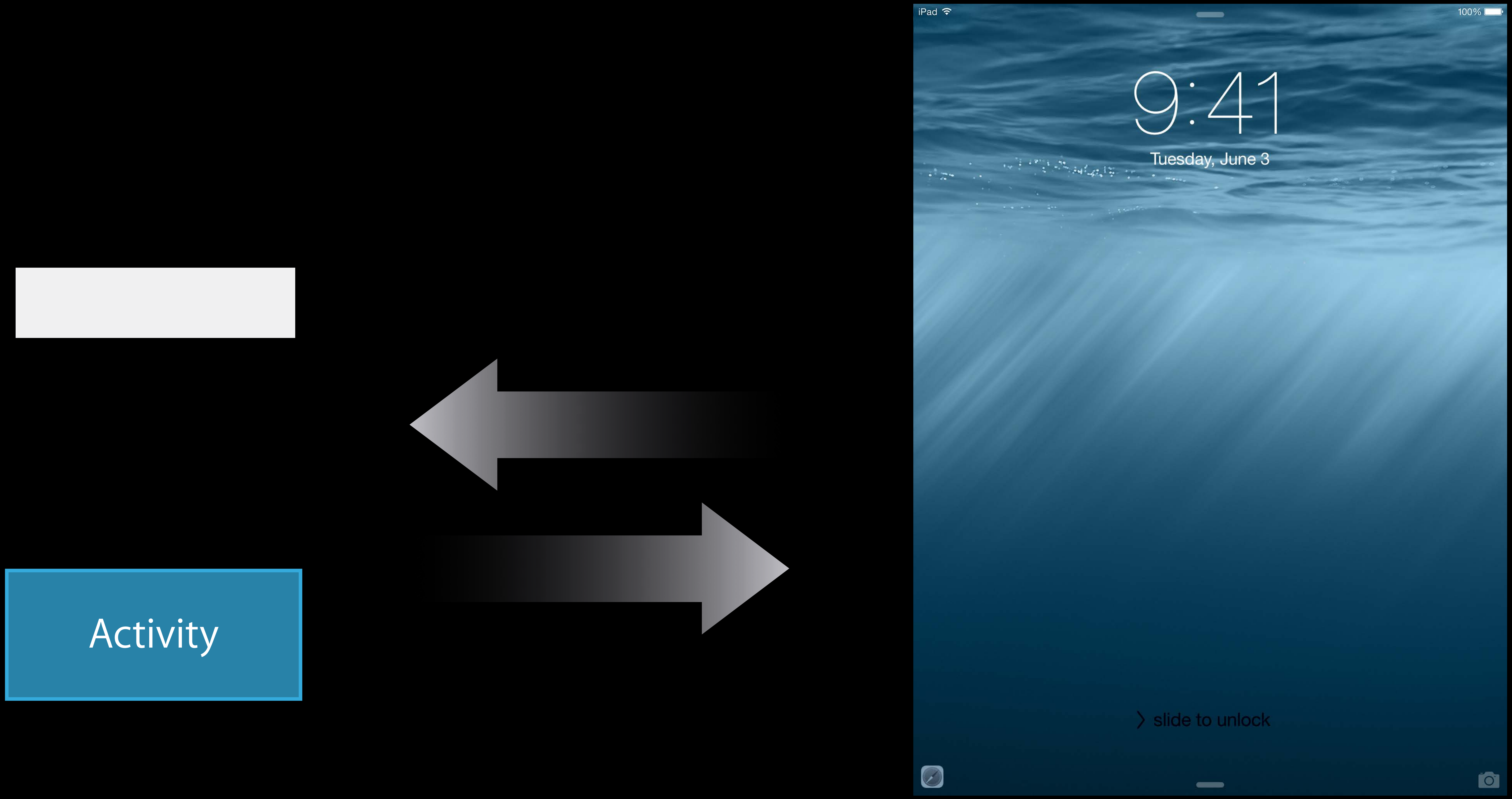
iPad
100%
9:41
Tuesday, June 3
slide to unlock

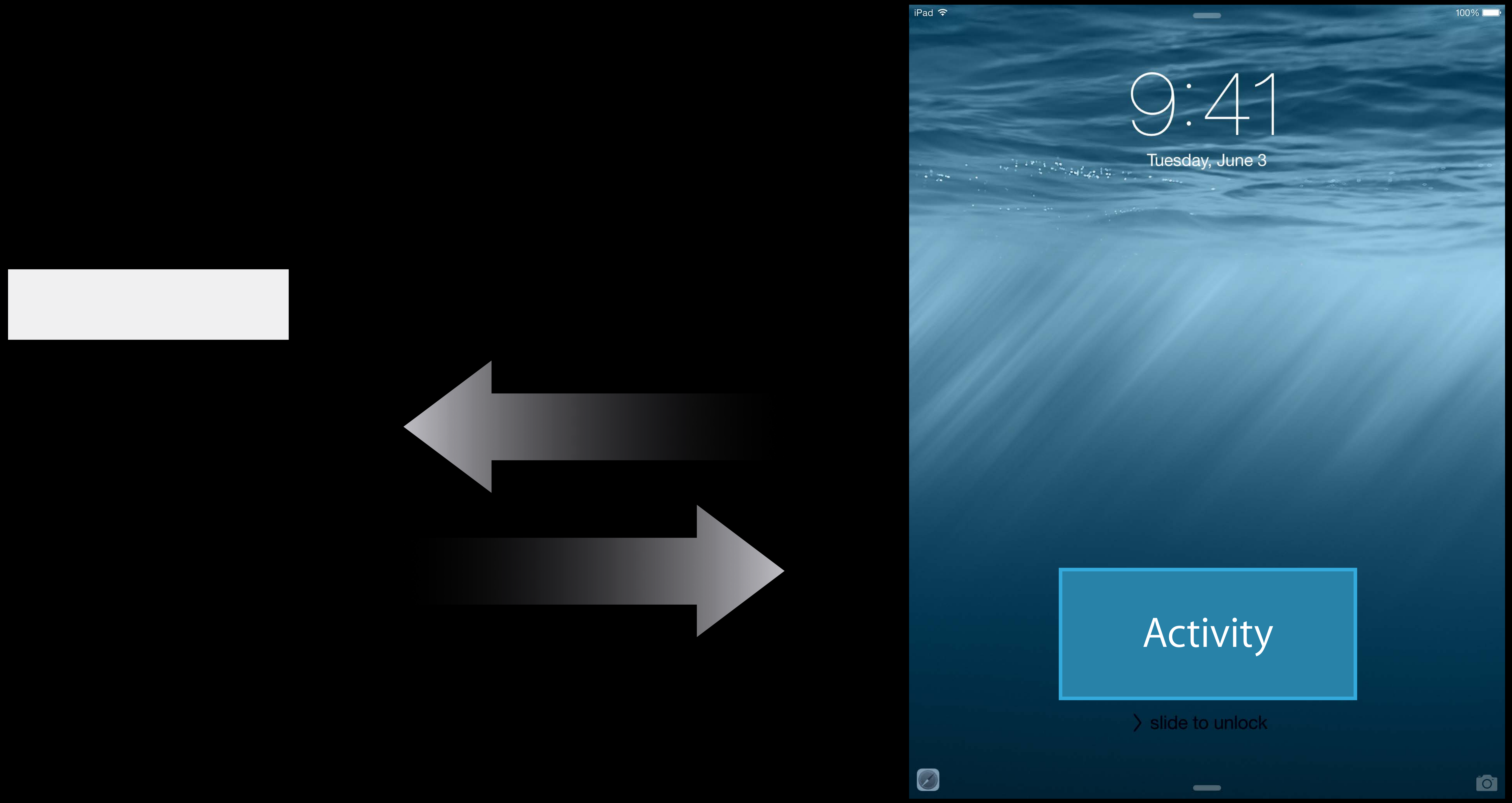

iPad
100%
9:41
Tuesday, June 3
slide to unlock

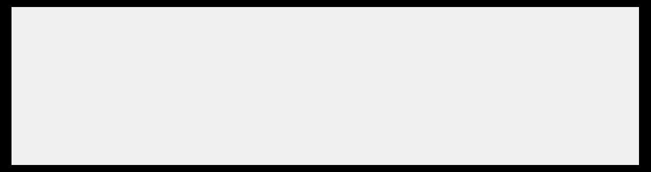

iPad
100%
9:41
Tuesday, June 3
slide to unlock

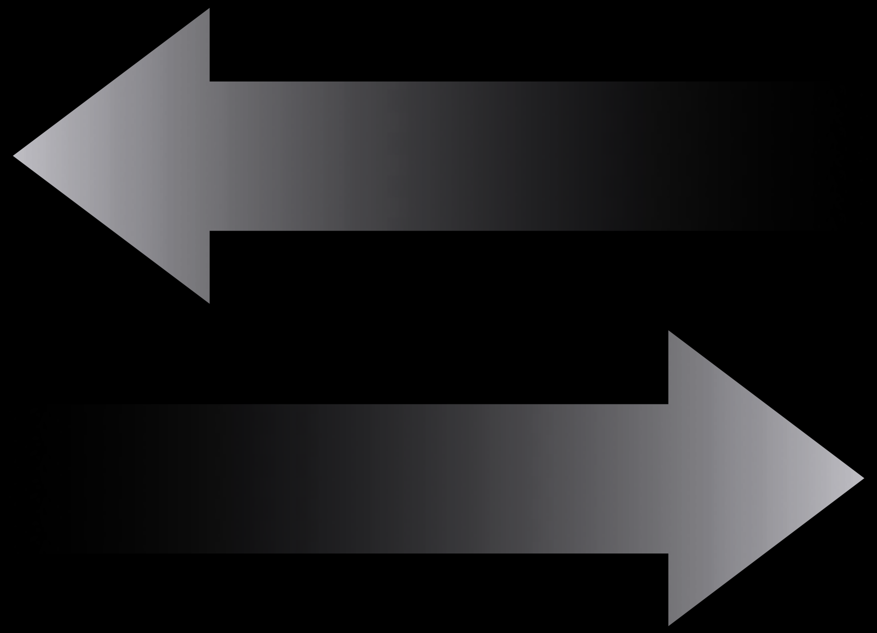

iPad
100%
9:41
Tuesday, June 3
slide to unlock

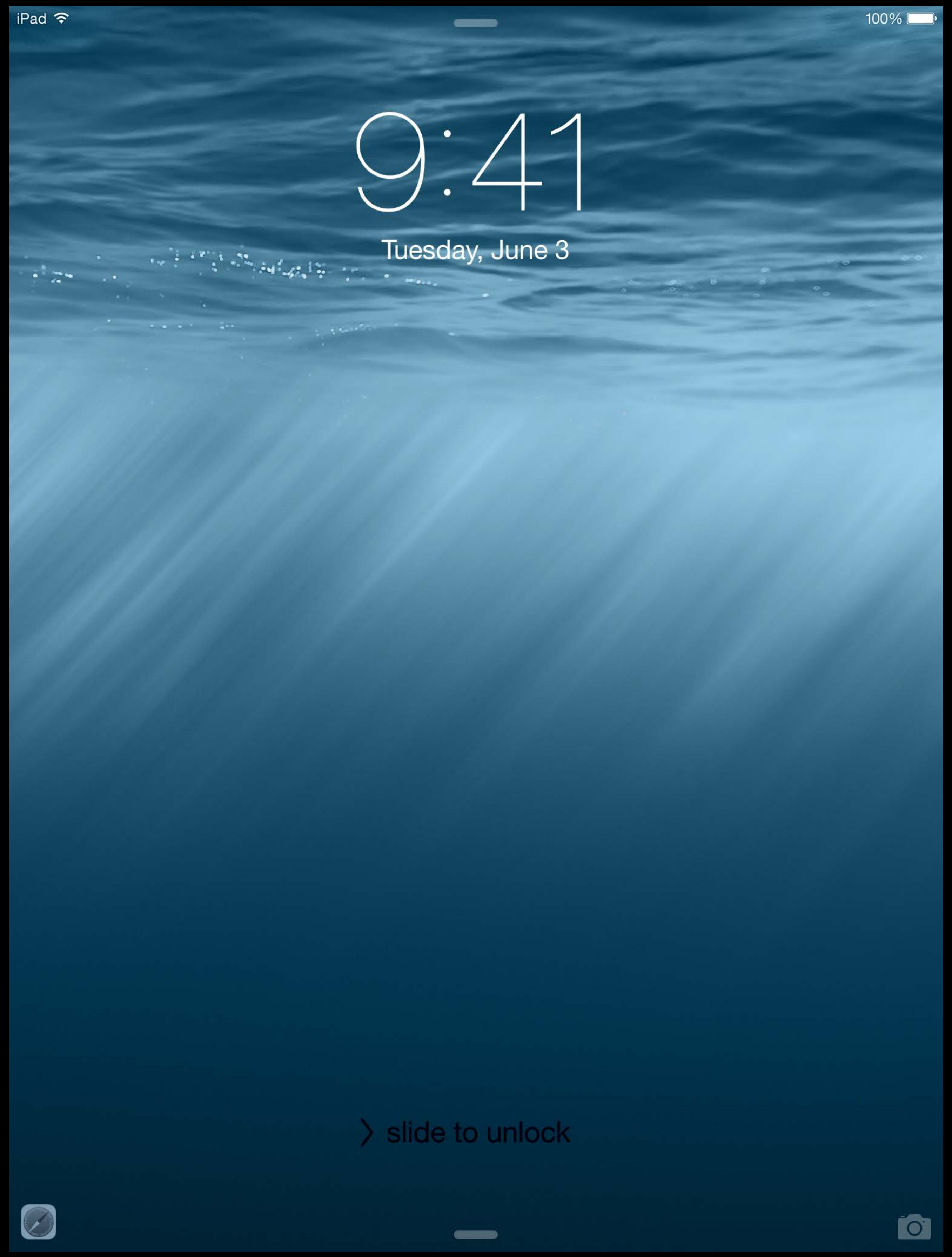

iPad
100%
9:41
Tuesday, June 3
slide to unlock


Activity
iPad
100%
9:41
Tuesday, June 3
slide to unlock


iPad
100%
9:41
Tuesday, June 3
Activity
slide to unlock






iPad
100%
9:41
Tuesday, June 3
slide to unlock

Additional Handoff Support

Streams between applications in two devices

Handoff between native app and website you own

Agenda

AppKit and UIKit support for adopting Handoff

Working with NSUserActivity directly

Native app to website Handoff

Using continuation streams between apps

Adopting Handoff in Your App

Vince Spader
Cocoa Frameworks Engineer

Adopting Handoff in Your App

AppKit/UIKit support for Handoff

Creating

Updating

Continuing

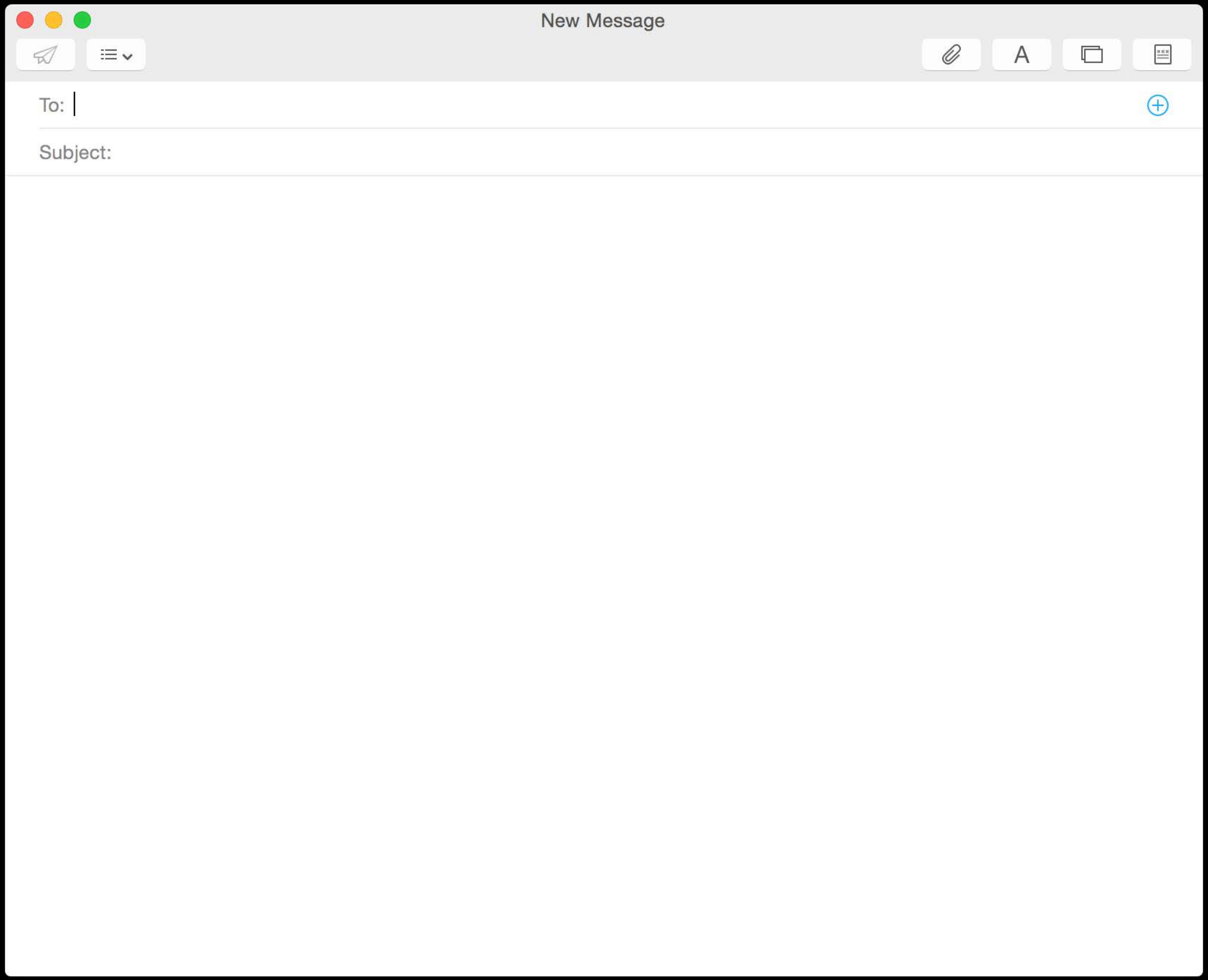
Creating User Activities

Creating User Activities

What do users do in your app?

Creating User Activities

What do users do in your app?



Creating User Activities

What do users do in your app?

Creating User Activities

What do users do in your app?

Reading messages

Picking an item from a list

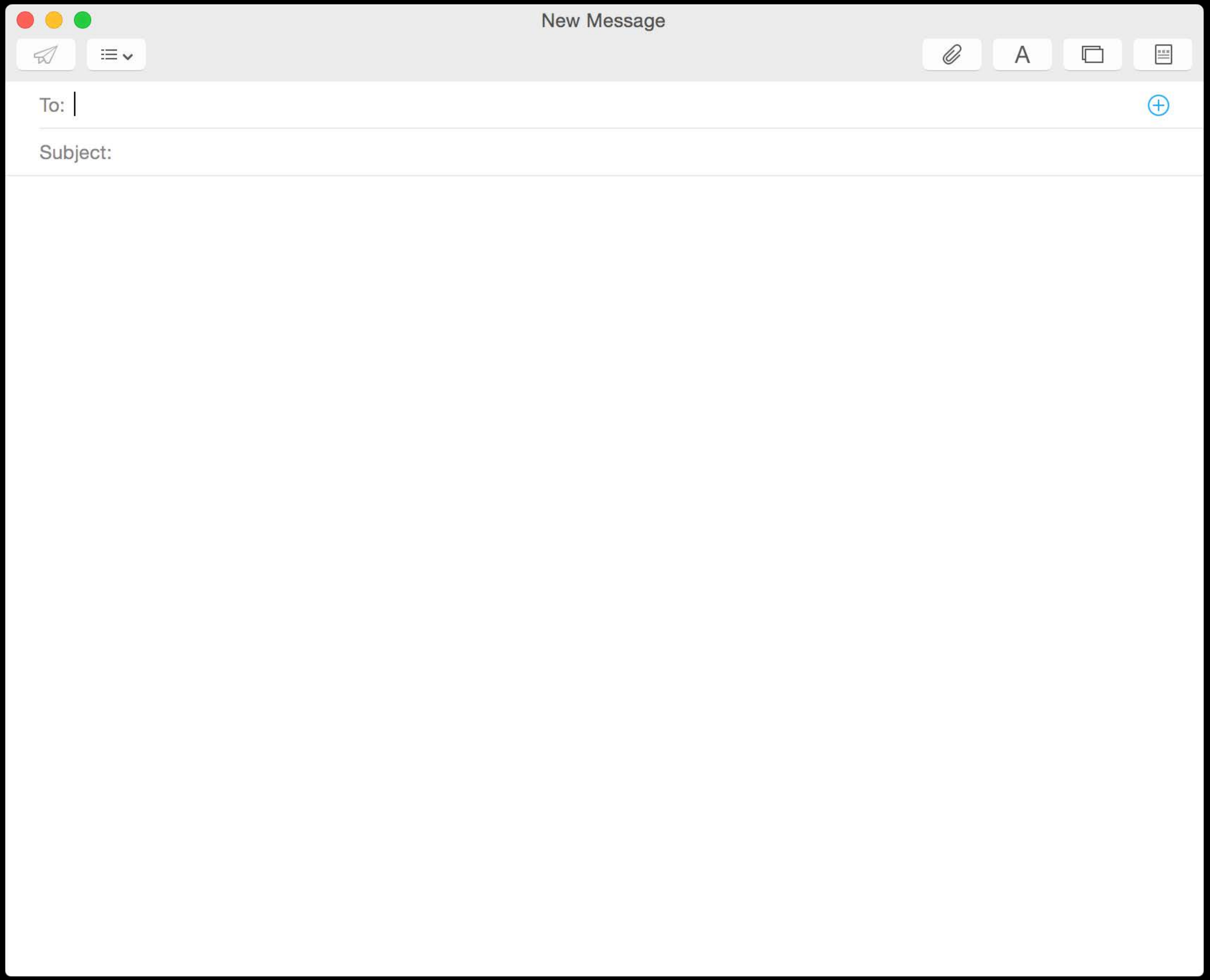
Editing a document

Creating User Activities

What do users do in your app?

Creating User Activities

What do users do in your app?



Creating User Activities

Documents and Responders

Creating User Activities

Documents and Responders

NSDocument, UIDocument, NSResponder and UIResponder now have:

```
@property (strong) NSUserActivity *userActivity;
```

Creating User Activities

Documents and Responders

NSDocument, UIDocument, NSResponder and UIResponder now have:

```
@property (strong) NSUserActivity *userActivity;
```

You can set it like this:

```
NSUserActivity *userActivity = [[NSUserActivity alloc]
  initWithActivityType:@“com.company.viewing-message”];
userActivity.title = @“Viewing Message”;
document.userActivity = userActivity;
```

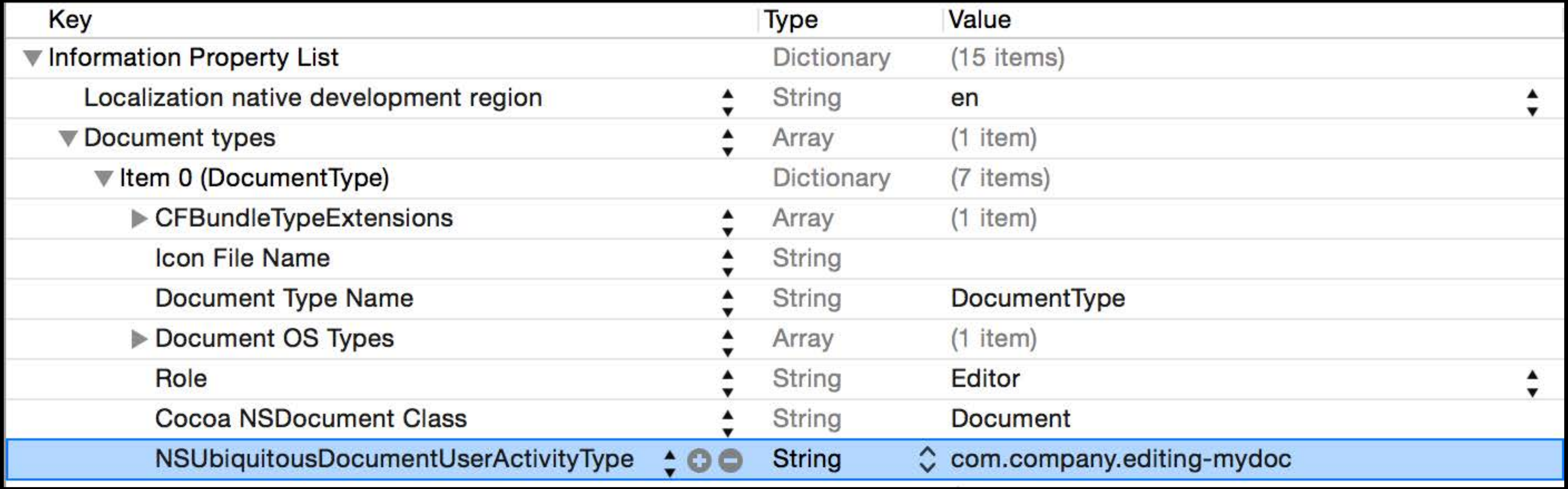
Creating User Activities

Document-based apps

Creating User Activities

Document-based apps

Add `NSUbiquitousDocumentUserActivityType`
to each `CFBundleDocumentTypes` entry



Key	Type	Value
▼ Information Property List	Dictionary	(15 items)
Localization native development region	String	en
▼ Document types	Array	(1 item)
▼ Item 0 (DocumentType)	Dictionary	(7 items)
▶ CFBundleTypeExtensions	Array	(1 item)
Icon File Name	String	
Document Type Name	String	DocumentType
▶ Document OS Types	Array	(1 item)
Role	String	Editor
Cocoa NSDocument Class	String	Document
NSUbiquitousDocumentUserActivityType	String	com.company.editing-mydoc

Creating User Activities

Document-based apps

Creating User Activities

Document-based apps

We set `userActivity` automatically when the document is in iCloud

Creating User Activities

Document-based apps

We set `userActivity` automatically when the document is in iCloud

- On OS X, you can KVO

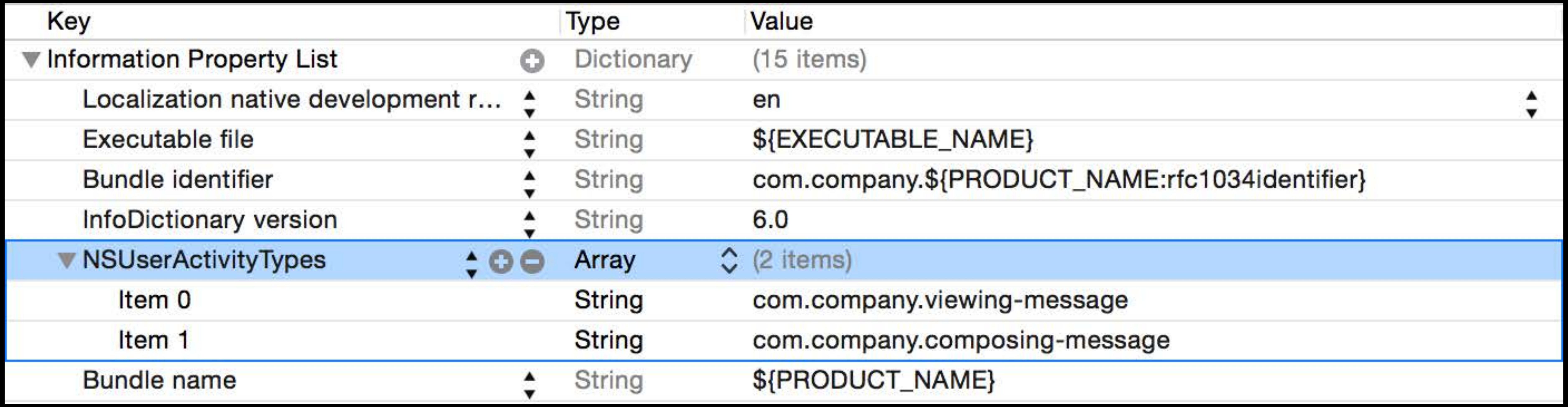
Creating User Activities

Other apps

Creating User Activities

Other apps

`NSUserActivityTypes` in Info.plist



Key	Type	Value
▼ Information Property List	Dictionary	(15 items)
Localization native development r...	String	en
Executable file	String	${EXECUTABLE_NAME}
Bundle identifier	String	com.company.${PRODUCT_NAME:rfc1034identifier}
InfoDictionary version	String	6.0
▼ NSUserActivityTypes	Array	(2 items)
Item 0	String	com.company.viewing-message
Item 1	String	com.company.composing-message
Bundle name	String	${PRODUCT_NAME}

Creating User Activities

Documents and Responders

Creating User Activities

Documents and Responders

We manage it for you

Creating User Activities

Documents and Responders

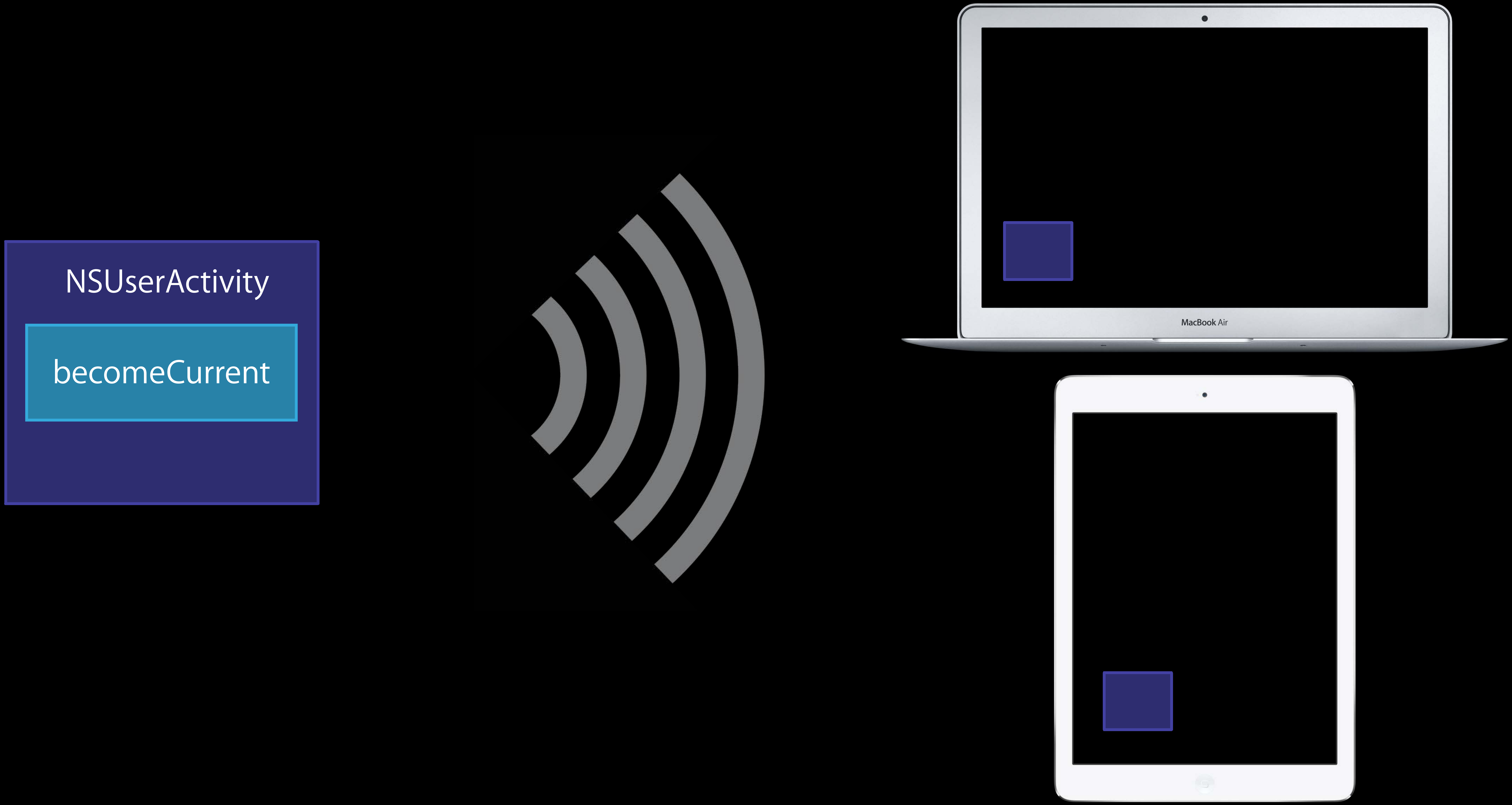
We manage it for you

- We call `becomeCurrent`

Creating User Activities

NSUserActivity
becomeCurrent

Creating User Activities



Creating User Activities

becomeCurrent on iOS

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

- UIKit walks the view controller hierarchy

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

- UIKit walks the view controller hierarchy
    - Including presented view controllers

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

- UIKit walks the view controller hierarchy
  - Including presented view controllers
  - The view controller's view must be in the view hierarchy

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

- UIKit walks the view controller hierarchy
  - Including presented view controllers
  - The view controller’s view must be in the view hierarchy

When userActivity is set:

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

- UIKit walks the view controller hierarchy
  - Including presented view controllers
  - The view controller's view must be in the view hierarchy

When userActivity is set:

- If the view controller is in a transition, we wait until after it's done

Creating User Activities

becomeCurrent on iOS

When the app is launched, comes into the foreground, or tabs are switched:

- UIKit walks the view controller hierarchy
    - Including presented view controllers
    - The view controller's view must be in the view hierarchy

When userActivity is set:

- If the view controller is in a transition, we wait until after it's done
- If the view controller's view is in the window hierarchy

Creating User Activities

becomeCurrent on iOS

Creating User Activities

becomeCurrent on iOS

UIDocument will not `becomeCurrent` automatically.

Creating User Activities

becomeCurrent on iOS

UIDocument will not `becomeCurrent` automatically.

Share the userActivity:

```
[document openWithCompletionHandler:^(BOOL success) {
   viewController.userActivity = document.userActivity;
   …
}];
```

Creating User Activities

becomeCurrent on OS X

Creating User Activities

becomeCurrent on OS X

AppKit looks for a `userActivity`:

Creating User Activities

becomeCurrent on OS X

AppKit looks for a `userActivity`:

- Main window’s responder chain

Creating User Activities

becomeCurrent on OS X

AppKit looks for a `userActivity`:

- Main window’s responder chain
- Main window controller’s document

Creating User Activities

becomeCurrent on OS X

AppKit looks for a `userActivity`:

- Main window's responder chain
- Main window controller's document

We'll reevaluate when appropriate

Creating User Activities

Documents and Responders

Creating User Activities

Documents and Responders

We manage it for you

- We call `becomeCurrent`

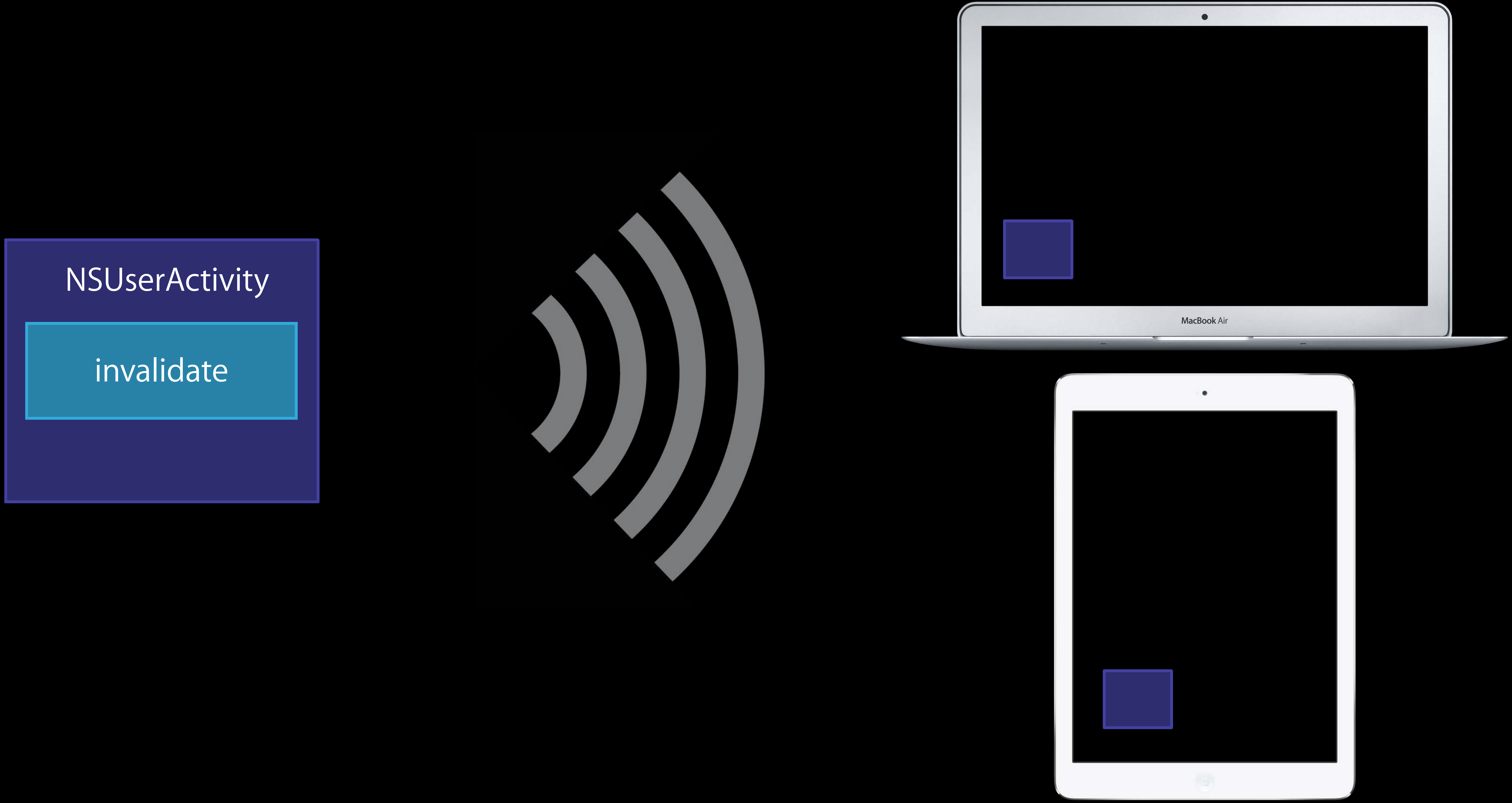
Creating User Activities

Documents and Responders

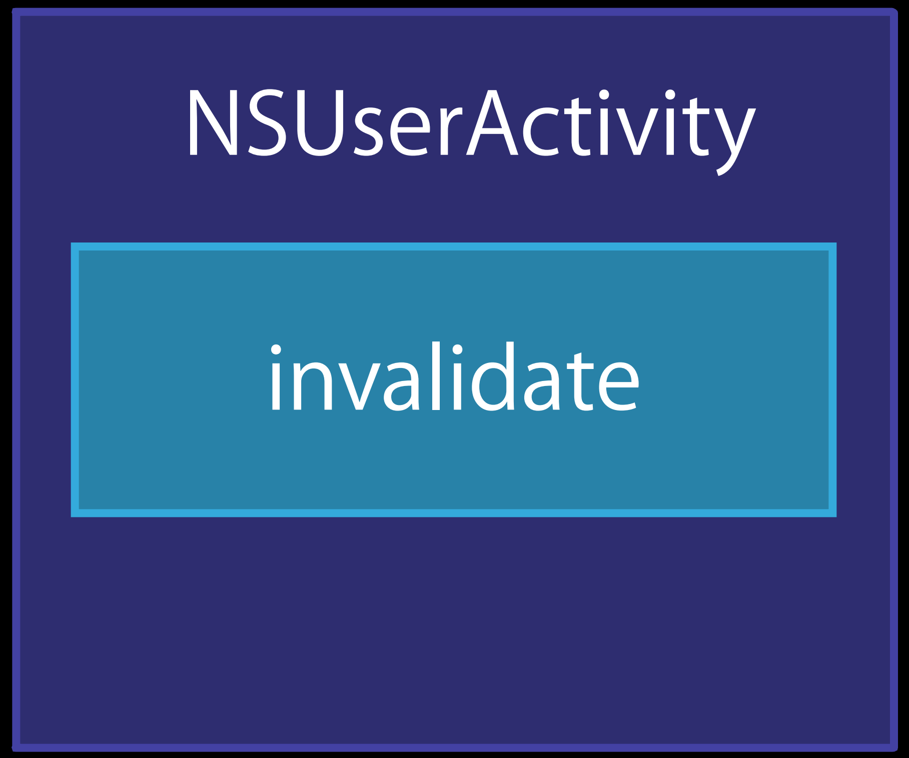
We manage it for you

- We call `becomeCurrent`
- We call `invalidate`

Creating User Activities



Creating User Activities



Updating User Activities

Updating User Activities

Documents and Responders

Updating User Activities

Documents and Responders

NSUserActivity has a `userInfo` dictionary

Updating User Activities

Documents and Responders

NSUserActivity has a `userInfo` dictionary

Override:

```
- (void)updateUserActivityState:(NSUserActivity *)userActivity
```

Updating User Activities

Documents and Responders

NSUserActivity has a `userInfo` dictionary

Override:

```
- (void)updateUserActivityState:(NSUserActivity *)userActivity
```

The userInfo is emptied each time

Updating User Activities

Documents and Responders

Updating User Activities

Documents and Responders

Something like this:

```
- (void)updateUserActivityState:(NSUserActivity *)userActivity {
    [super updateUserActivityState:userActivity];

    [userActivity addUserInfoEntriesFromDictionary:@{
       @“messageID”: self.messageID,
    }];
}
```

Updating User Activities

Documents and Responders

Something like this:

```
- (void)updateUserActivityState:(NSUserActivity *)userActivity {
   [super updateUserActivityState:userActivity];

   [userActivity addUserInfoEntriesFromDictionary:@{
      @“messageID”: self.messageID,
   }];
}
```

When your info is stale:

userActivity.needsSave = YES;

Updating User Activities

What to include

Updating User Activities

What to include

Can store NSArray, NSData, NSDate, NSDictionary, NSNull, NSNumber, NSSet, NSString, NSUUID, or NSURL

Updating User Activities

What to include

Can store NSArray, NSData, NSDate, NSDictionary, NSNull, NSNumber, NSSet, NSString, NSUUID, or NSURL

File URLs in iCloud or from a document provider are OK

Updating User Activities

What to include

Updating User Activities

What to include

Keep the minimal amount of information in the userInfo

Updating User Activities

What to include

Keep the minimal amount of information in the userInfo

- Just the state

Updating User Activities

What to include

Keep the minimal amount of information in the userInfo

- Just the state
- Avoid platform specifics

Updating User Activities

What to include

Keep the minimal amount of information in the userInfo

- Just the state
- Avoid platform specifics
- NS/UIDocument will add its `fileURL` with `NSUserActivityDocumentURLKey`

Updating User Activities

What to include

Updating User Activities

What to include

Think about versioning

Updating User Activities

What to include

Think about versioning

Maybe something like:

```
- (void)application:(NS/UIApplication *)application
        didUpdateUserActivity:(NSUserActivity *)userActivity {
   [userActivity addUserInfoEntriesFromDictionary:@{
      @“handoffVersion”: @“2.0”,
   }];
}
```

Continuing User Activities

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

We start fetching it from the other device:

```
- (BOOL)application:(NS/UIApplication *)application
        willContinueUserActivityWithType:(NSString *)activityType;
```

Continuing User Activity

App Delegate

We start fetching it from the other device:

```
- (BOOL)application:(NS/UIApplication *)application
        willContinueUserActivityWithType:(NSString *)activityType;
```

Use this to show the user what's being continued

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

```
- (BOOL)application:(NS/UIApplication *)application
       willContinueUserActivityWithType:(NSString *)activityType {
   if ([activityType isEqual:@“com.company.viewing-message”]) {

      id vc = [[MessageViewController alloc] init];
      vc.showLoadingIndicator = YES;
      [self showMessageViewController:vc];

      return YES;
   }

   return NO;
}
```

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

We got the activity:

```
- (BOOL)application:(NS/UIApplication *)application
  continueUserActivity:(NSUserActivity *)userActivity
  restorationHandler:
    (void(^)(NSArray *restorableObjects))restorationHandler;
```

Continuing User Activity

App Delegate

We got the activity:

```
- (BOOL)application:(NS/UIApplication *)application
  continueUserActivity:(NSUserActivity *)userActivity
  restorationHandler:
    (void(^)(NSArray *restorableObjects))restorationHandler;
```

Reconstruct the user's activity

Continuing User Activity

App Delegate

We got the activity:

```
- (BOOL)application:(NS/UIApplication *)application
  continueUserActivity:(NSUserActivity *)userActivity
  restorationHandler:
    (void(^)(NSArray *restorableObjects))restorationHandler;
```

Reconstruct the user’s activity

Call the restorationHandler, passing it an array of documents or responders that present the user activity

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

Here’s an example:

```
- (BOOL)application:continueUserActivity:restorationHandler: {
   NSString *activityType = activity.activityType;
   if ([activityType isEqual:@“com.company.viewing-message”]) {
      id vc = [[MessageViewController alloc] init];
      …
      restorationHandler(@[vc]);

      return YES;
   }
   return NO;
}
```

Continuing User Activity

App Delegate

Here's an example:

```
- (BOOL)application:continueUserActivity:restorationHandler: {
   NSString *activityType = activity.activityType;
   if ([activityType isEqual:@“com.company.viewing-message”]) {
      id vc = [[MessageViewController alloc] init];
      …
      restorationHandler(@[vc]);

      return YES;
   }
   return NO;
}
```

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

```
@implementation MessageViewController
…
- (void)restoreUserActivityState:(NSUserActivity *)activity {
   [super restoreUserActivityState:activity];

   [self setMessageID:activity.userInfo[@“messageID”]];
   …
   id cvc = [[ConversationViewController alloc] init];
   …
   [cvc restoreUserActivityState:activity];
}
…
@end
```

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

```
@implementation ConversationViewController
…
- (void)restoreUserActivityState:(NSUserActivity *)activity {
   [super restoreUserActivityState:activity];

   NSString *version = activity.userInfo[@“handoffVersion”]
   BOOL isOldVersion = [self isOldVersion:version];

   NSString *recipientKey = isOldVersion ? @“to” : @“rcptID”;
   self.recipient = activity.userInfo[recipientKey];
   [self updateRecipientImage];
}
…
```

Continuing User Activity

App Delegate

Continuing User Activity

App Delegate

If there was an error:

```
- (void)application:(NS/UIApplication *)application
    didFailToContinueUserActivityWithType:(NSString *)activityType
    error:(NSError *)error;
```

Continuing User Activity

App Delegate

If there was an error:

```
- (void)application:(NS/UIApplication *)application
    didFailToContinueUserActivityWithType:(NSString *)activityType
    error:(NSError *)error;
```

Can be `NSUserCancelledError`!

Continuing User Activity

Document-based app

Continuing User Activity

Document-based app

On iOS, you continue the user activity:

```
- (BOOL)application:continueUserActivity:restorationHandler: {
…
   NSURL *url = activity.userInfo[NSUserActivityDocumentURLKey];
   MyDocument *doc = [[MyDocument alloc] initWithFileURL:url];

   restorationHandler(@[doc]);

   return YES;
…
```

Continuing User Activity

Document-based app

Continuing User Activity

Document-based app

On OS X, AppKit can use NSDocumentController

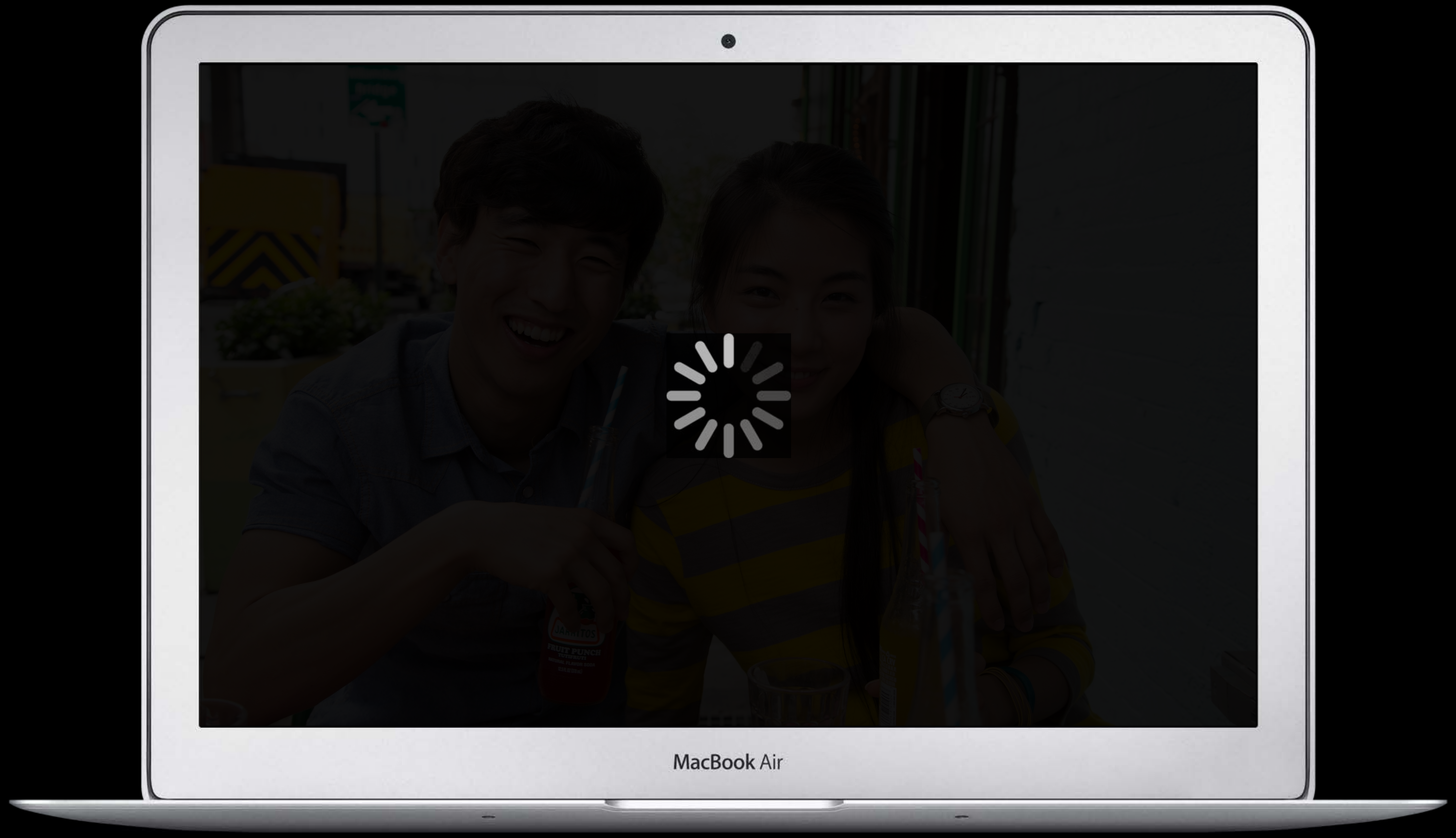
`restoreUserActivityState:`

Continuing User Activity

Continuing User Activity

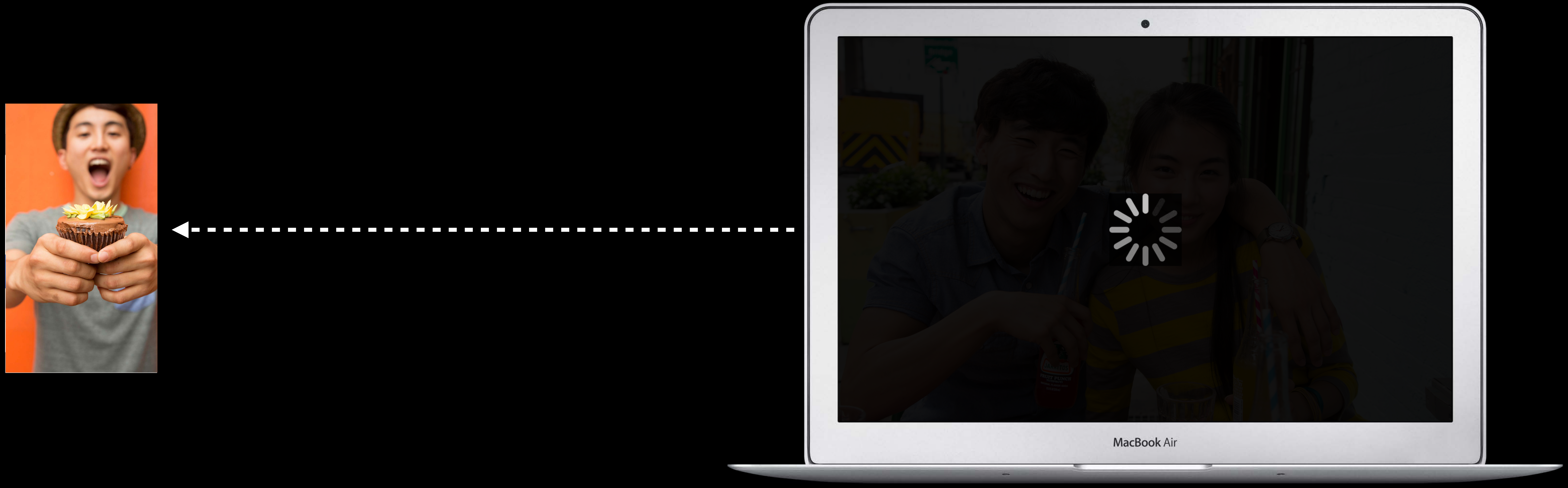
application:willContinueUserActivityWithType:

Continuing User Activity



application:willContinueUserActivityWithType:

Continuing User Activity

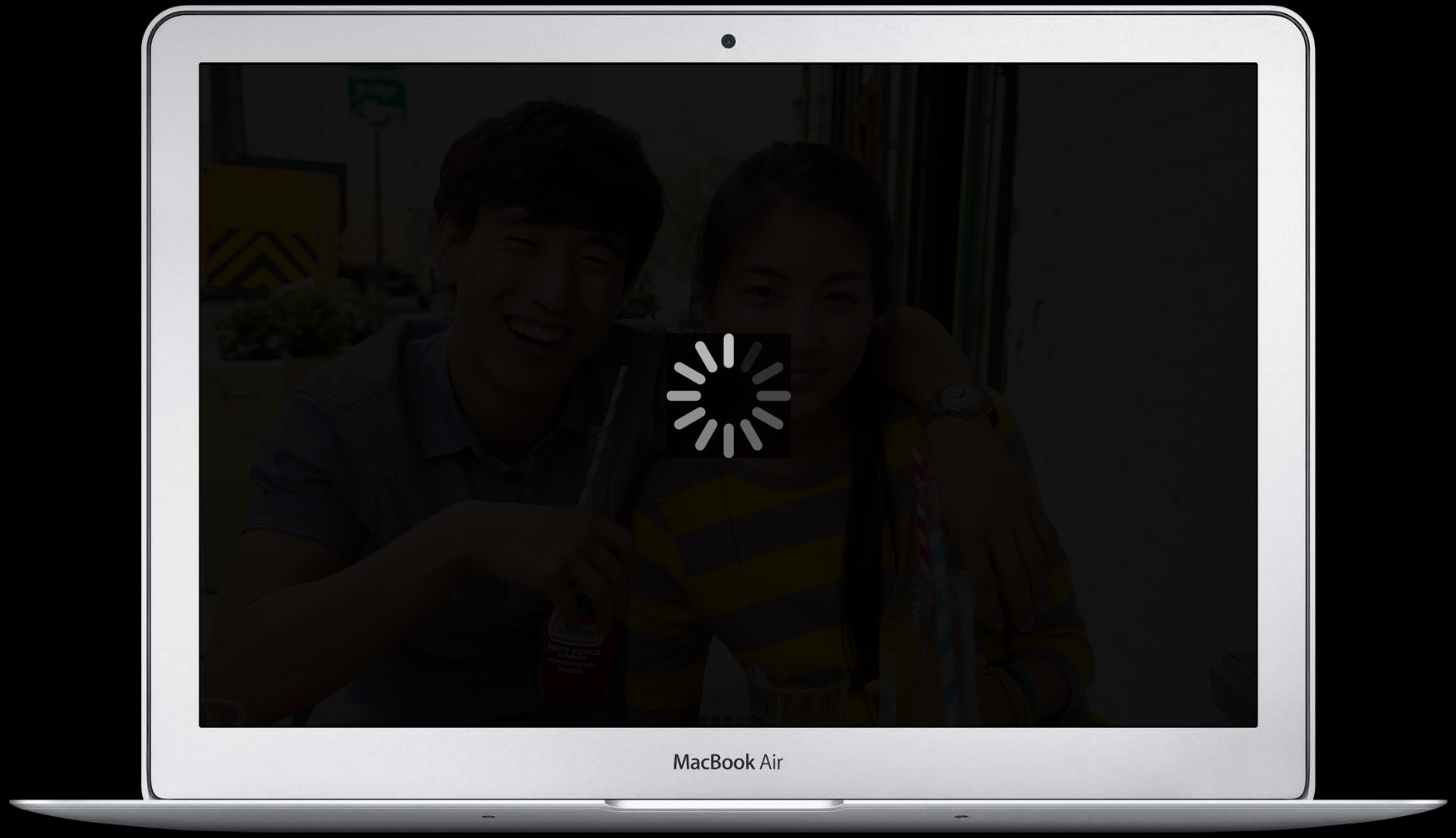


application:willContinueUserActivityWithType:

Continuing User Activity

MacBook Air

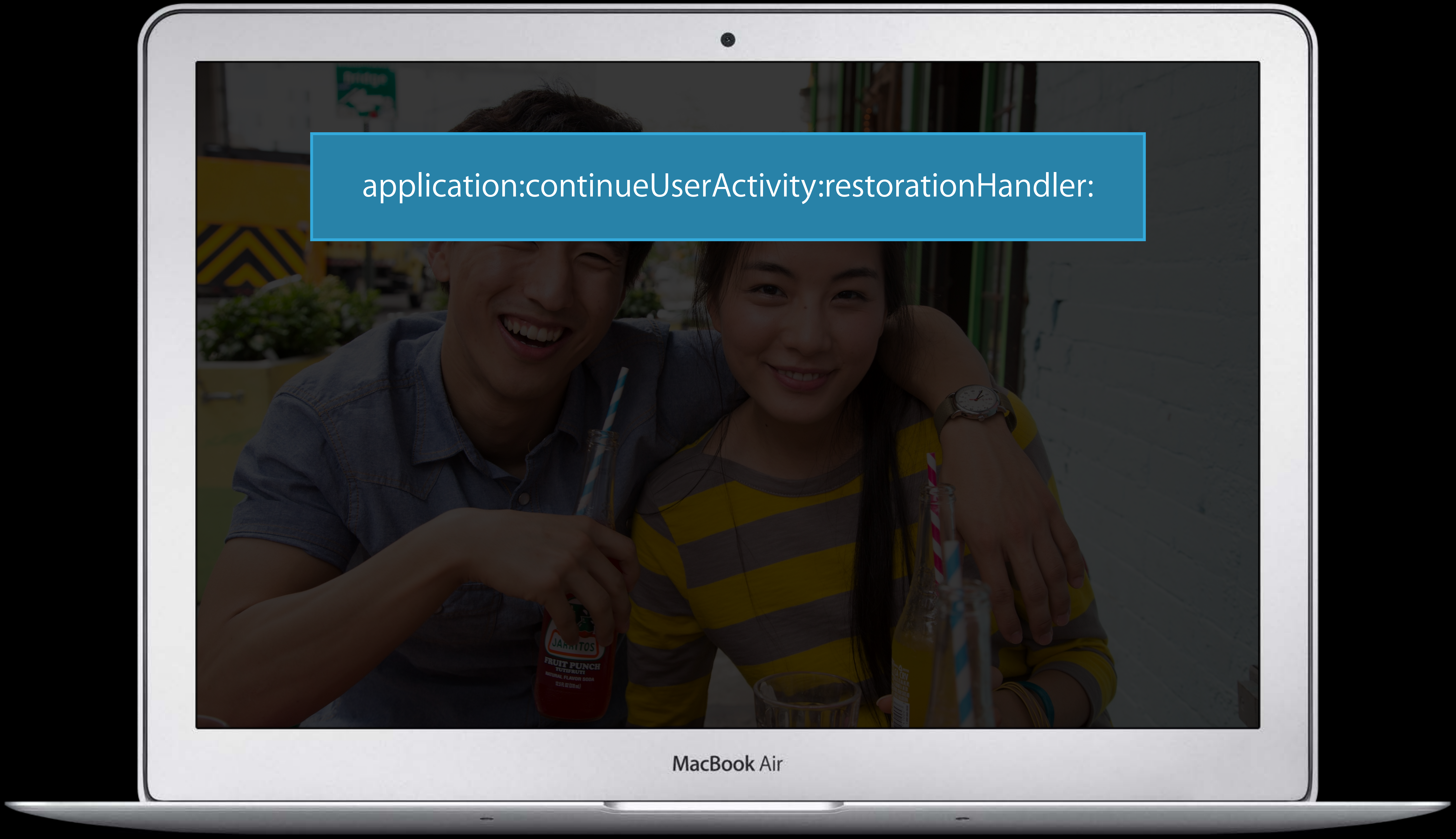
Continuing User Activity



`application:continueUserActivity:restorationHandler:`

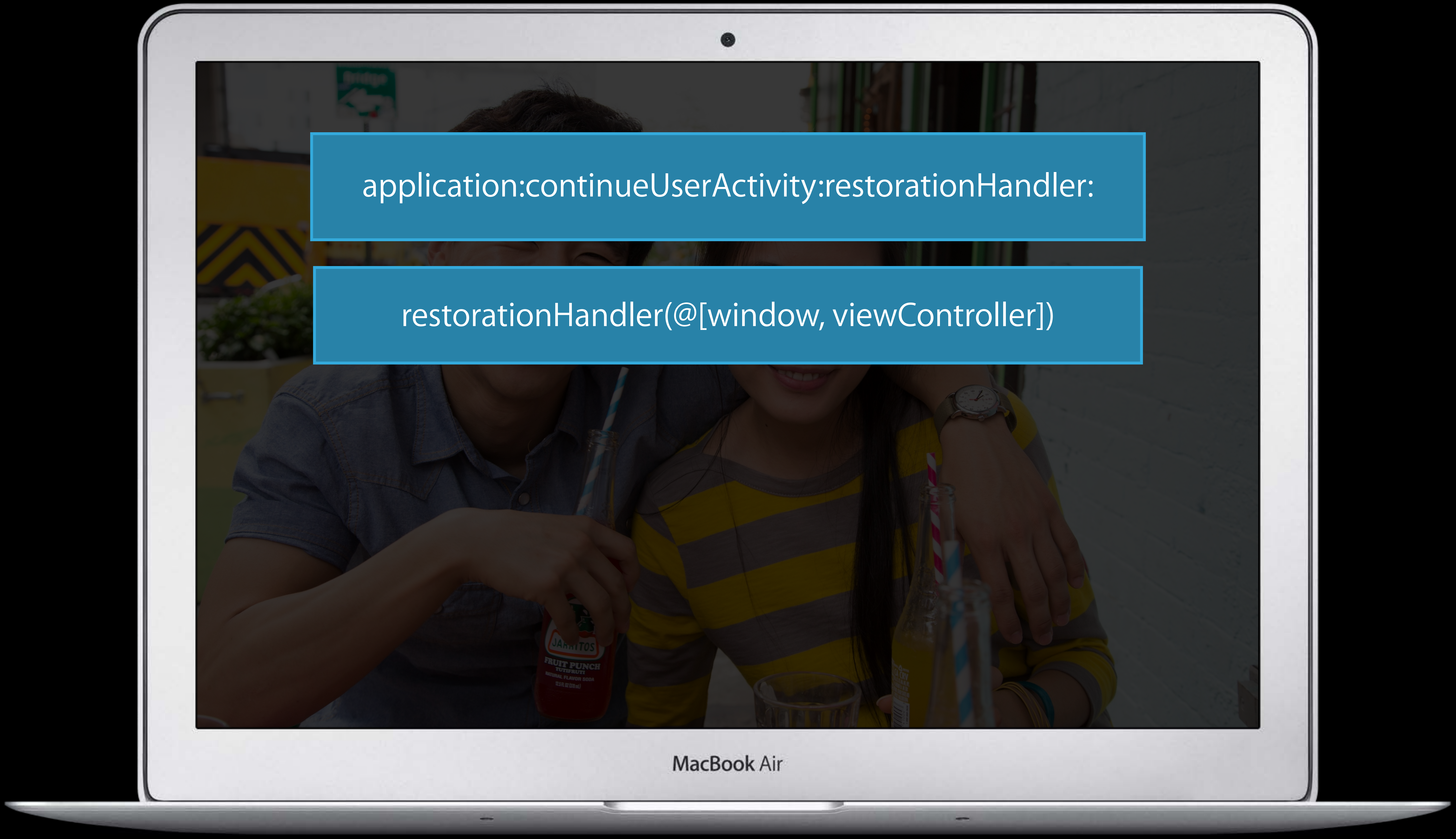
Continuing User Activity

Continuing User Activity



application:continueUserActivity:restorationHandler:

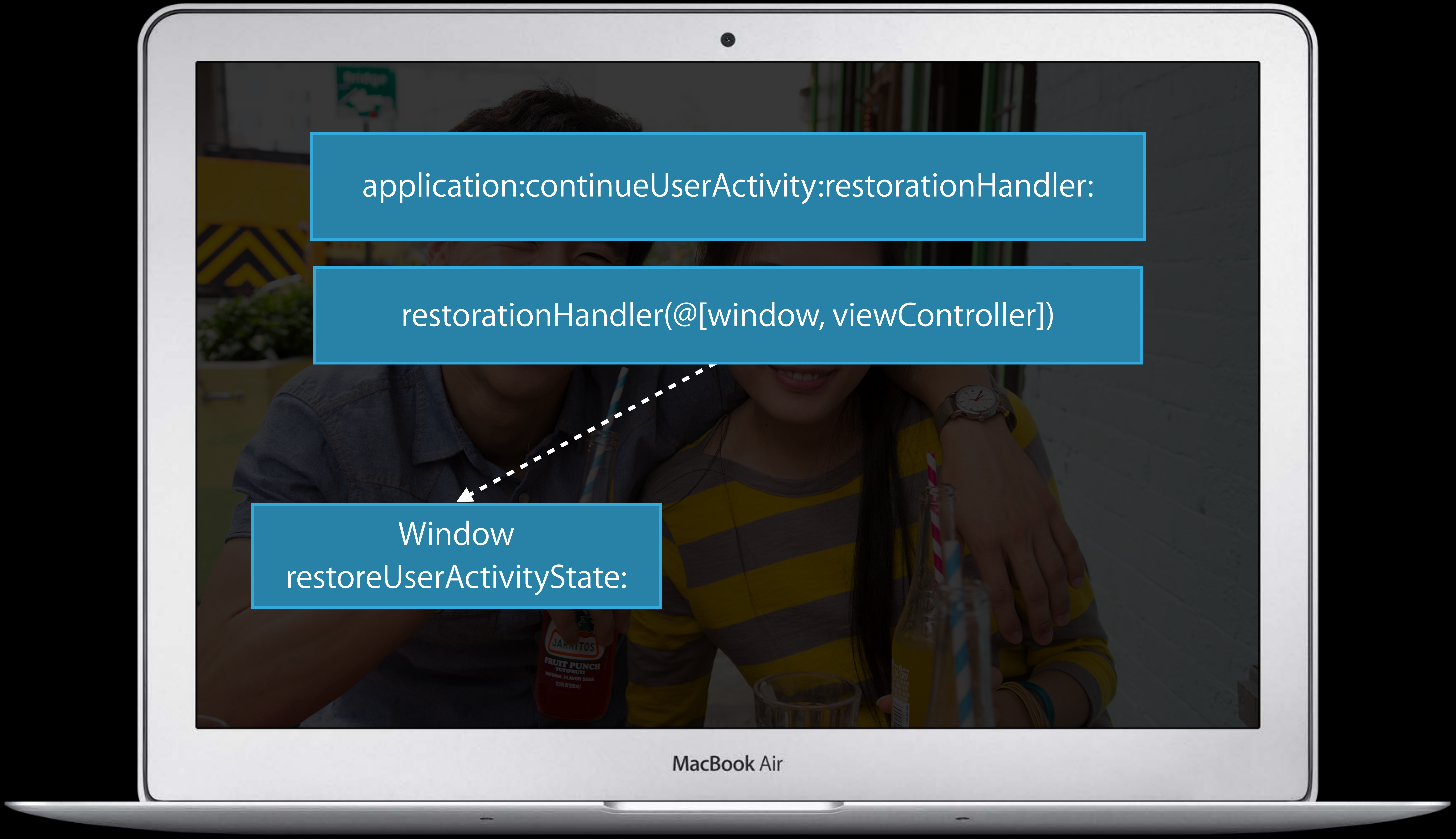
Continuing User Activity



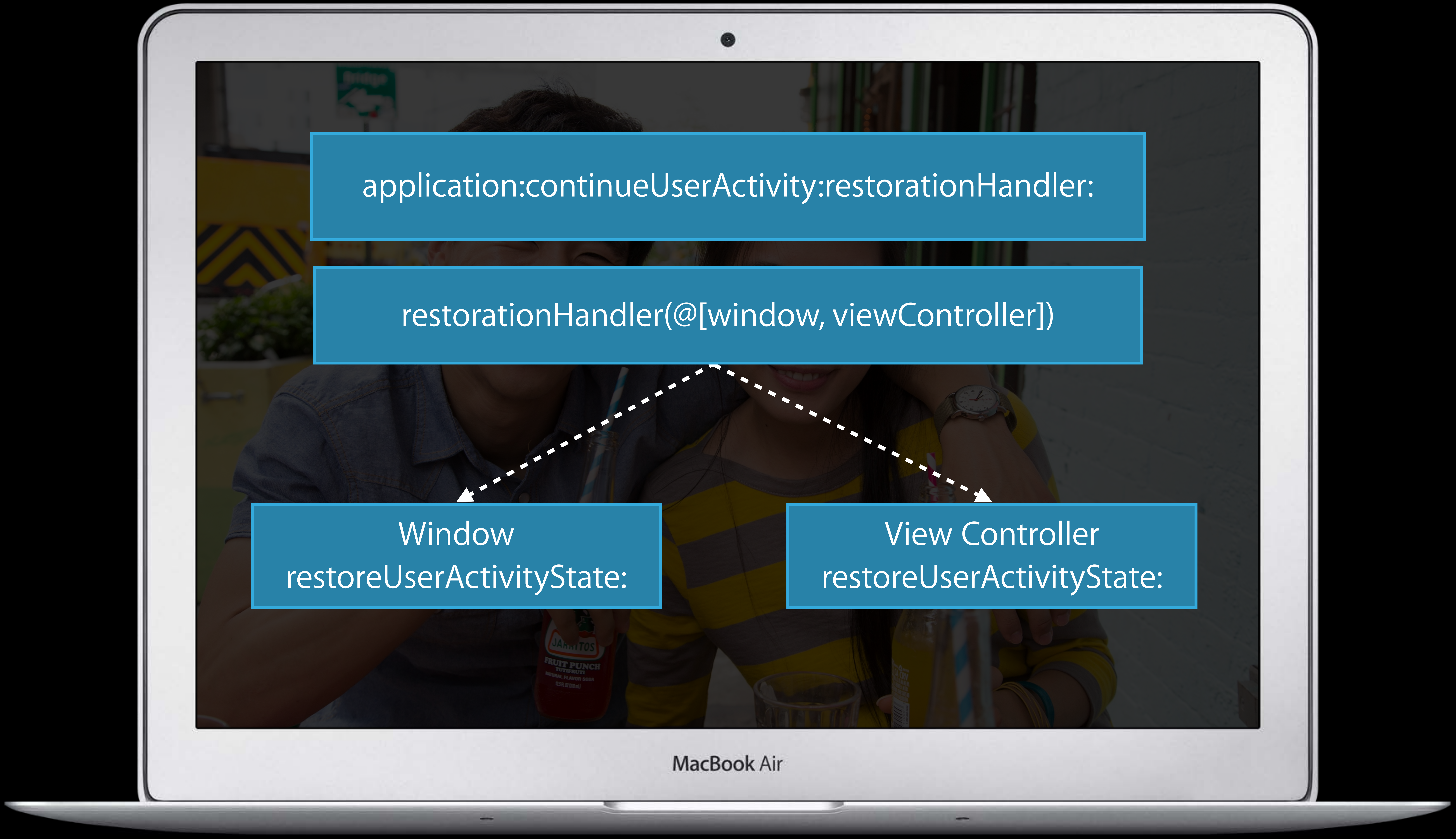
```
application:continueUserActivity:restorationHandler:
```

```
restorationHandler(@[window, viewController])
```

Continuing User Activity


application:continueUserActivity:restorationHandler:
restorationHandler(@[window, viewController])
Window
restoreUserActivityState:
MacBook Air

Continuing User Activity


application:continueUserActivity:restorationHandler:
restorationHandler(@[window, viewController])
Window
restoreUserActivityState:
View Controller
restoreUserActivityState:
MacBook Air

Continuing User Activity

Continuing User Activity

NSUserActivity In-depth

Keith Stattenfield
CoreFrameworks Engineer

Non AppKit/UIKit uses

NSUserActivity

Non AppKit/UIKit uses

NSUserActivity

Your application creates an activity with an activity type string

```
[[NSUserActivity alloc] initWithActivityType:@“com.company.edit.foo”];
```

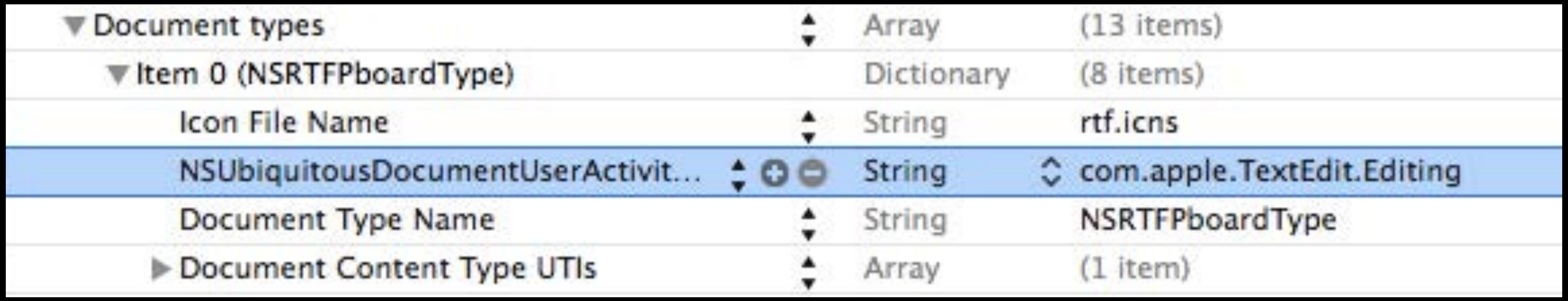
Activity Type Strings

NSUserActivity

Activity Type Strings

NSUserActivity

- Applications which want to receive activities claim them in their Info.plist
  - Either in NSUserActivityTypes or in CFBundleDocumentTypes



```
▼Document types                      Array        (13 items)
  ▼Item 0 (NSRTFPboardType)          Dictionary   (8 items)
      Icon File Name                 String       rtf.icns
      NSUbiquitousDocumentUserActivit…  String    com.apple.TextEdit.Editing
      Document Type Name             String       NSRTFPboardType
    ▶ Document Content Type UTIs     Array        (1 item)
```

Activity Type Strings

NSUserActivity

Activity Type Strings

NSUserActivity

All applications from the same developer can exchange activities

Activity Type Strings

NSUserActivity

All applications from the same developer can exchange activities

Applications don’t have to claim the same activity types they create

Applications don’t have to claim any activity types, but can still create them

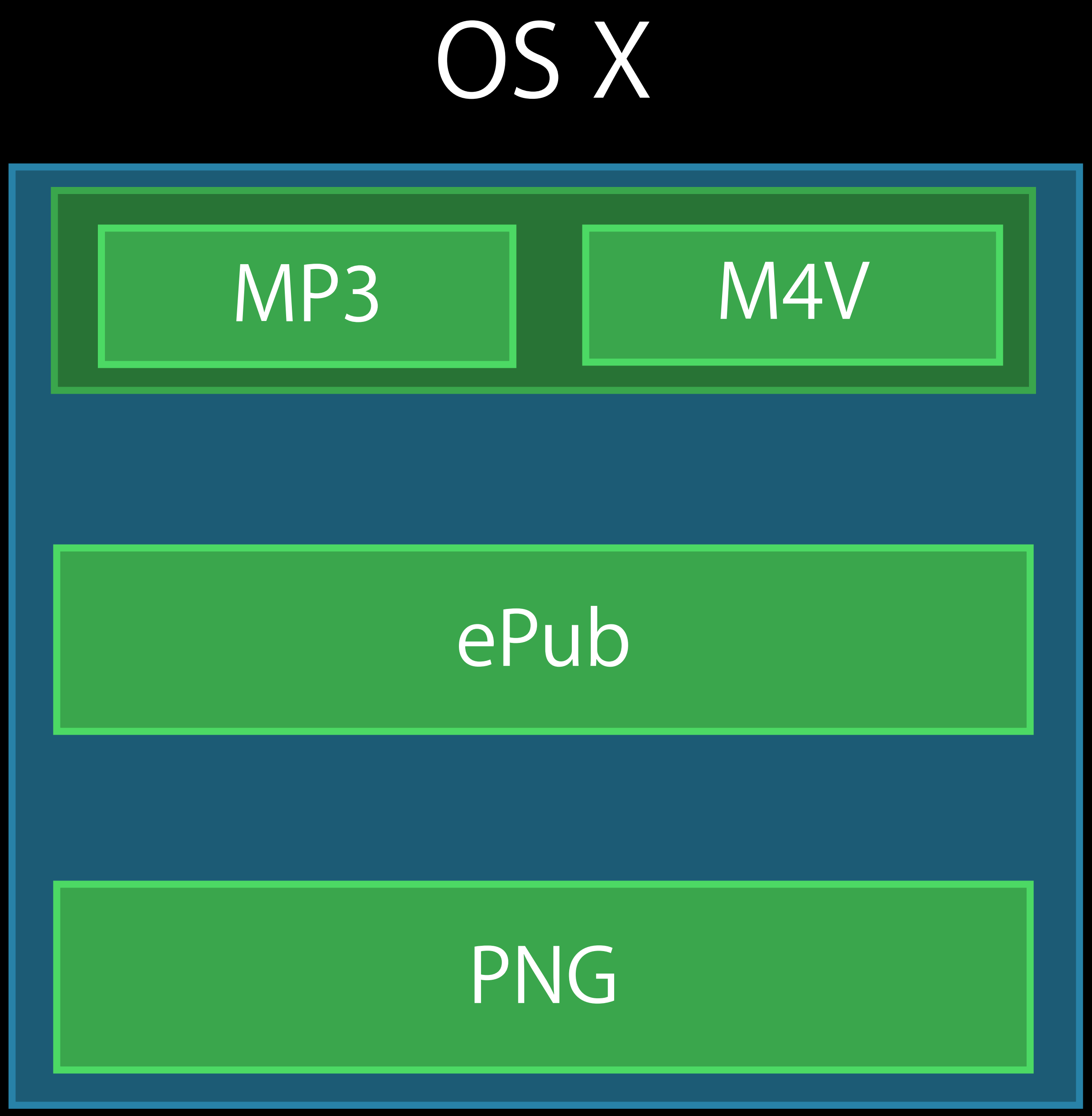
Activity Type Strings

NSUserActivity

All applications from the same developer can exchange activities

Applications don't have to claim the same activity types they create

Applications don't have to claim any activity types, but can still create them



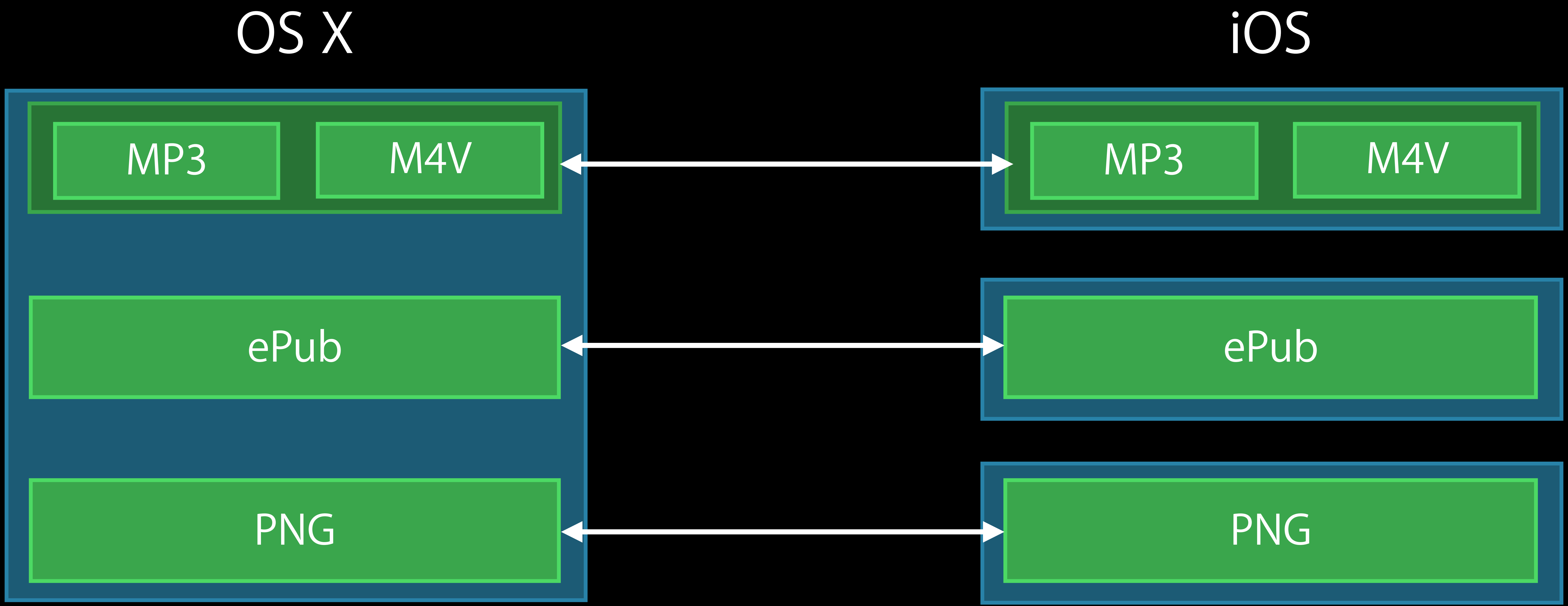
Activity Type Strings

NSUserActivity

All applications from the same developer can exchange activities

Applications don't have to claim the same activity types they create

Applications don't have to claim any activity types, but can still create them



Advanced NSUserActivity

Setting the activity information

Advanced NSUserActivity

Setting the activity information

```
activity.title = @" … "
activity.userInfo = @{ … }
    [activity addUserInfoEntriesFromDictionary: @{ … }]
[activity becomeCurrent]
[activity invalidate]
```

Advanced NSUserActivity

NSUserActivityDelegate

Advanced NSUserActivity

NSUserActivityDelegate

```
activity.delegate = self;
…
activity.needsSave = YES;
```

Then, when the system needs information from your activity

```
- (void)userActivityWillSave:(NSUserActivity *)userActivity
```

Advanced NSUserActivity

NSUserActivityDelegate

Advanced NSUserActivity

NSUserActivityDelegate

When continued from another device:

```
- (void)userActivityWasContinued:(NSUserActivity *)userActivity
```

Called when this activity was successfully continued on another device

Most applications won’t need this at all

Website Handoff

Native application to web browser

Network

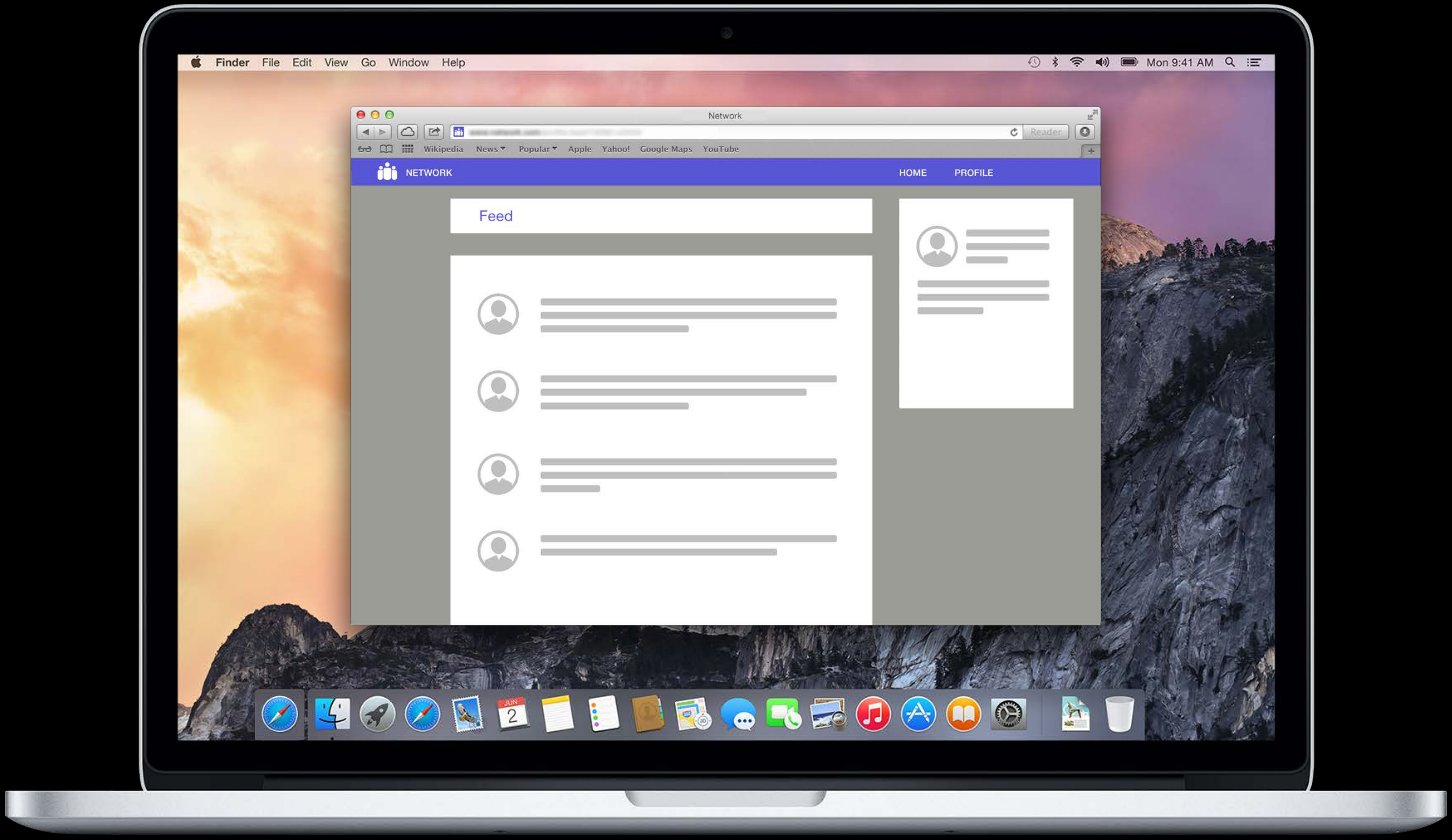
Website Handoff

Native application to web browser

Network

Website Handoff

Native application to web browser


Finder File Edit View Go Window Help
Mon 9:41 AM
Network
Wikipedia News Popular Apple Yahoo! Google Maps YouTube
NETWORK
HOME
PROFILE
Feed

Network

Website Handoff

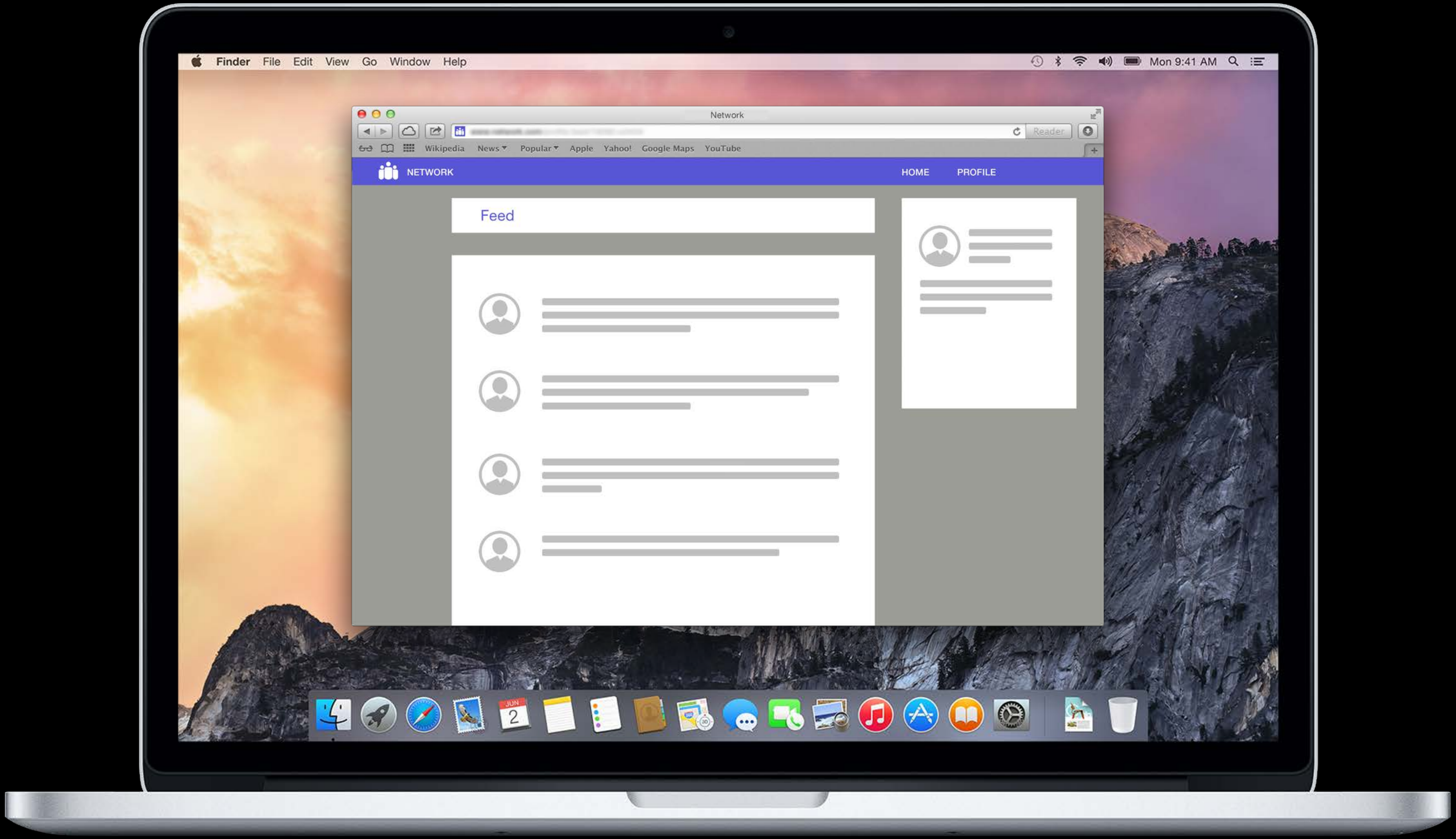
Native application to web browser

```
NSUserActivity* activity = [[NSUserActivity alloc]
initWithActivityType:…];


activity.userInfo = @{ … }
activity.webpageURL = [NSURL URLWithString: …];
```

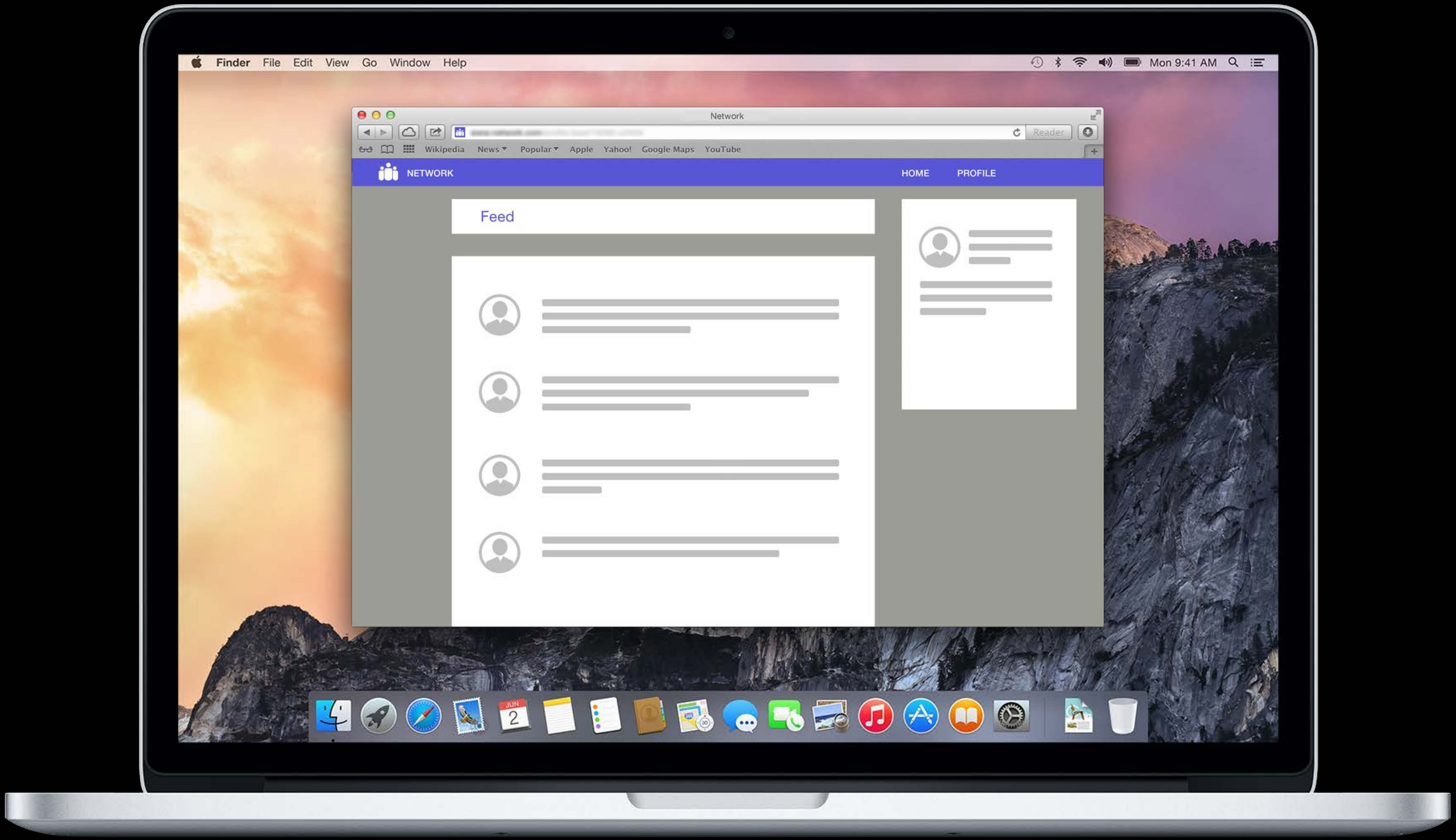
Website Handoff

Web browser to native application


Finder File Edit View Go Window Help
Mon 9:41 AM
Network
Wikipedia News Popular Apple Yahoo! Google Maps YouTube
NETWORK
HOME
PROFILE
Feed

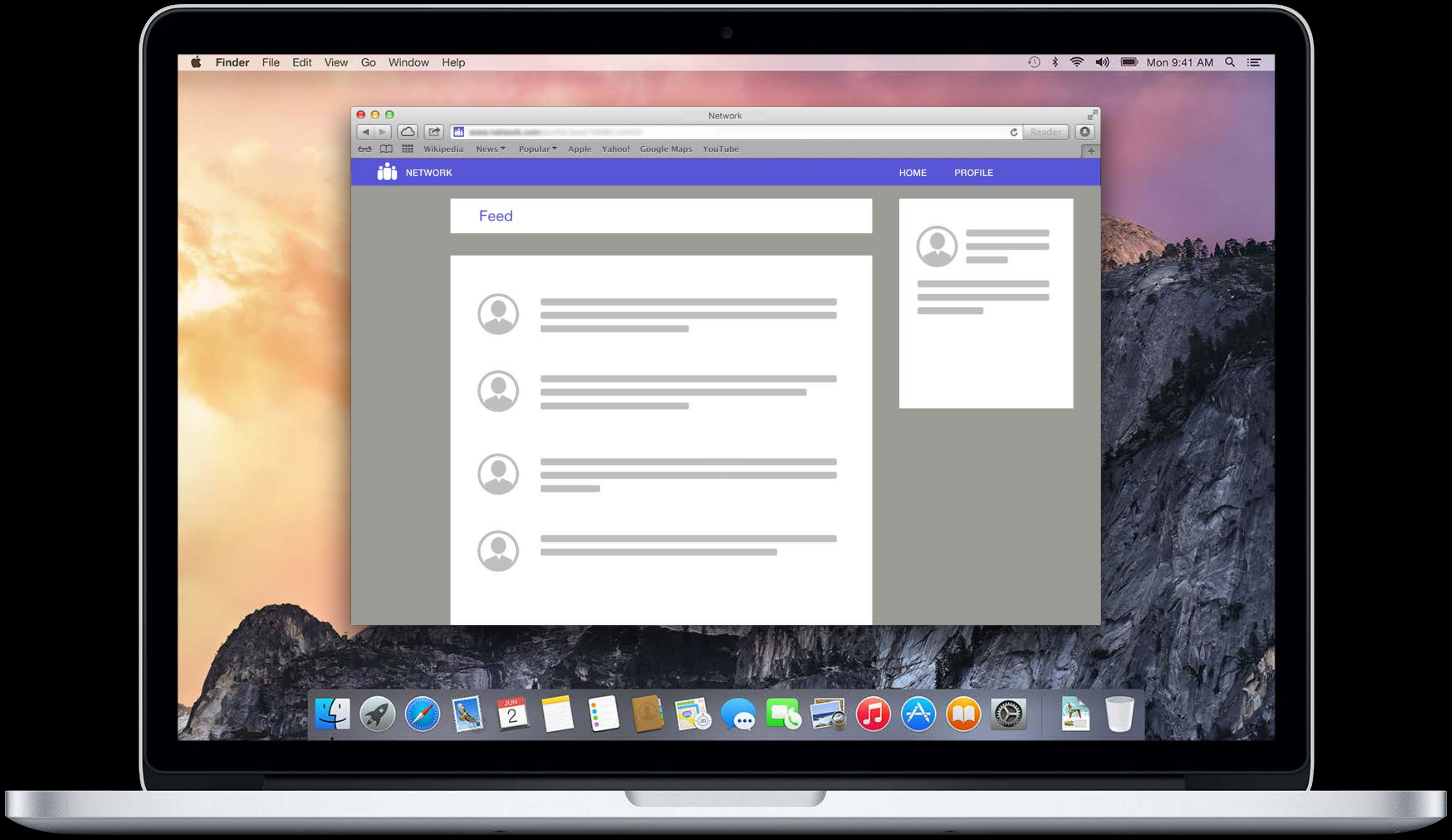
Website Handoff

Web browser to native application


Finder File Edit View Go Window Help
Mon 9:41 AM
Network
Wikipedia News Popular Apple Yahoo! Google Maps YouTube
NETWORK
HOME
PROFILE
Feed

Website Handoff

Web browser to native application


Finder File Edit View Go Window Help
Mon 9:41 AM
Network
Wikipedia News Popular Apple Yahoo! Google Maps YouTube
NETWORK
HOME
PROFILE
Feed

Website Handoff

Web browser to native application

Website Handoff

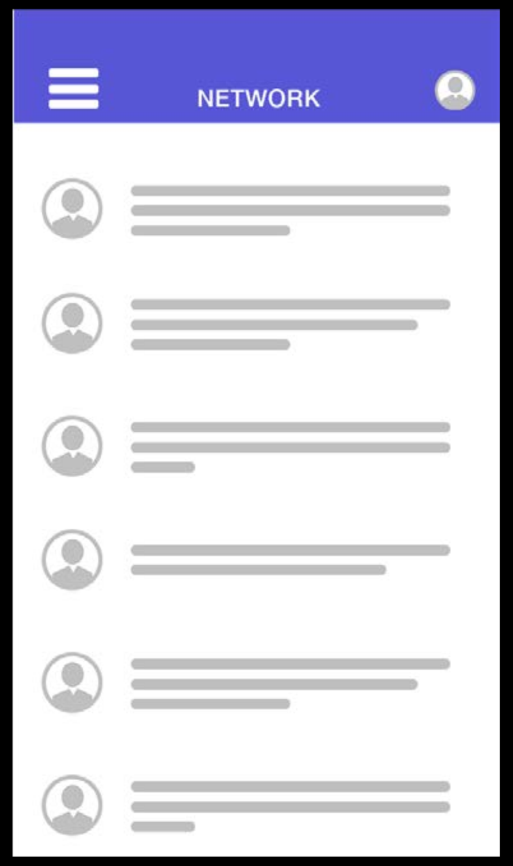
Web browser to native application

```
application:continueUserActivity:(NSUserActivity*)userActivity
restorationHandler:(void(^)(NSArray
*restorableObjects))restorationHandler {
   if ([userActivity.activityType
         isEqual:NSUserActivityTypeContinuingFromWebBrowser]) {
      /* resume an activity based on the webpageURL */
      …
   } else if ([userActivity isEqual:@“com.company.type12”]) {
      …
   }
```

Continuation Streams

Need more than a one-way, one time exchange of data from creator to receiver

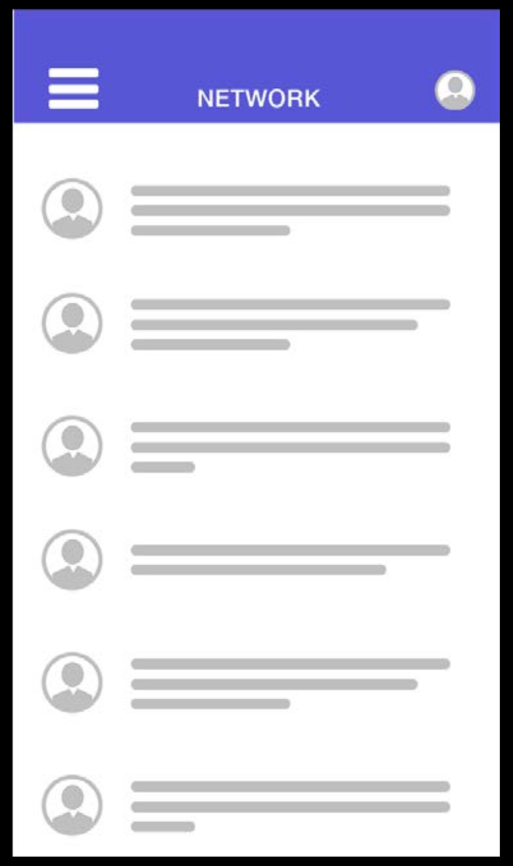
Establishes a bidirectional stream for some kind of interactive purposes



Continuation Streams

Need more than a one-way, one time exchange of data from creator to receiver

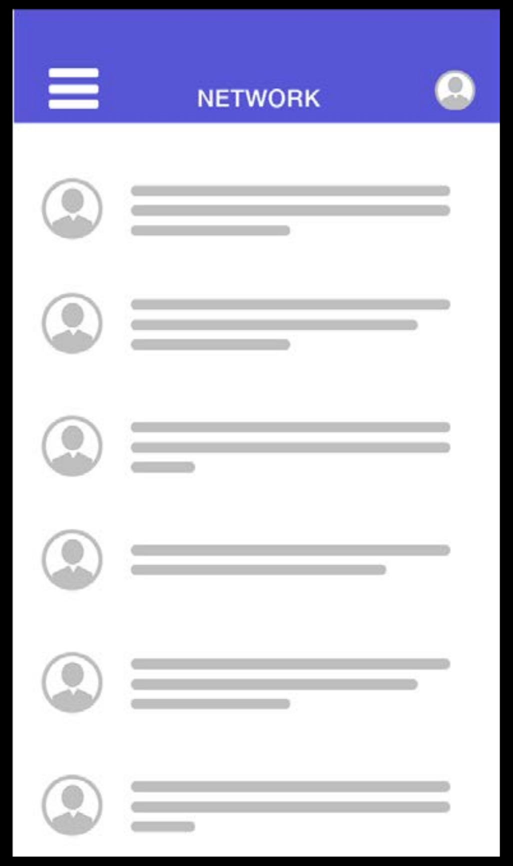
Establishes a bidirectional stream for some kind of interactive purposes



Continuation Streams

Need more than a one-way, one time exchange of data from creator to receiver

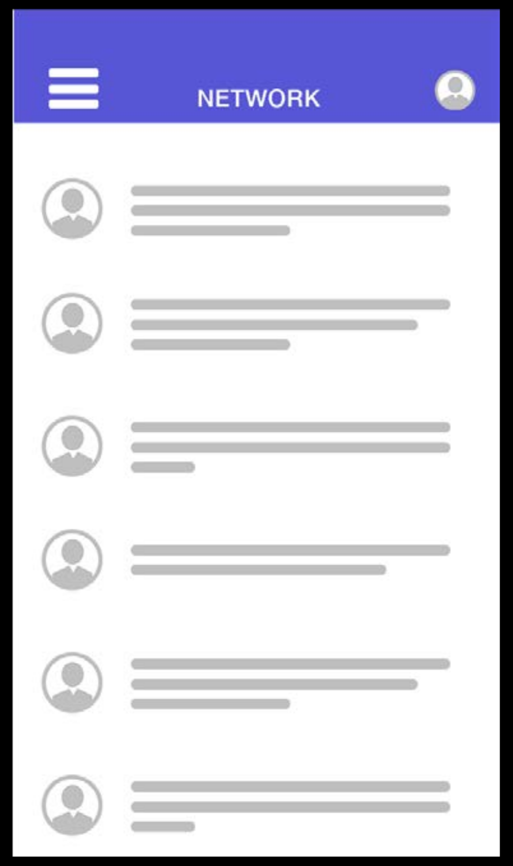
Establishes a bidirectional stream for some kind of interactive purposes



Continuation Streams

Need more than a one-way, one time exchange of data from creator to receiver

Establishes a bidirectional stream for some kind of interactive purposes



Continuation Streams

NSUserActivity

```
NSUserActivity* activity = [[NSUserActivity alloc] initWithActivityType:
@“com.company.interact” ];
activity.userInfo = @{ … }
activity.delegate = self;
activity.supportsContinuationStreams = YES;
[activity becomeCurrent ];
```

Continuation Streams

NSUserActivity, on the receiving device

```
- application:(NS/UIApplication*) continueUserActivity:
(NSUserActivity*)activity restorationHandler:…
{
   if (activity.supportsContinuationStreams ) {
      [activity getContinuationStreamsWithCompletionHandler:
         ^(NSInputStream* inputstream,NSOutputStream*
           outputstream, NSError* error) {
           if (!error) {
              /* You can send and receive over these streams! */
           }
```

ConnectBack

NSUserActivity, back on the initiating device

Lastly, this delegate method is called with the streams

```
-(void) userActivity:(NSUserActivity *)userActivity
didReceiveInputStream:(NSInputStream *)inputStream outputStream:
(NSOutputStream *)outputStream {

   …

}
```

So, you've learned

AppKit/UIKit support

NS/UIDocument support

Continuation streams

Website interoperability

More Information

Jake Behrens
Frameworks Evangelist
behrens@apple.com

Documentation

Handoff Programming Guide

http://apple.com

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

Related Sessions

●	Cloud Documents	Marina	Thursday 11:30AM
●	Your App, Your Website, and Safari	Nob Hill	Tuesday 4:30PM

Labs

Handoff Lab	Frameworks Lab B	Thursday 9:00AM
Cocoa Touch Lab	Frameworks Lab A	Thursday 2:00PM
Cocoa Lab	Frameworks Lab B	Thursday 4:30PM

WWDC14