

What's New in iMessage Apps

Session 234

Eugene Bistolas, Messages Engineer

Jay Chae, Messages Engineer

Stephen Lottermoser, Messages Engineer

What's New in Messages

Direct Send

Live Message Layouts

Best Practices

What's New in Messages

Direct Send

Live Message Layouts

Best Practices

What's New in Messages

Direct Send

Live Message Layouts

Best Practices

What's New in Messages

Direct Send

Live Message Layouts

Best Practices

What's New in Messages

Direct Send

Live Message Layouts

Best Practices

iMessage App Fundamentals

iMessage App Fundamentals

iMessage Apps and Stickers Part 2

WWDC 2016

What's New in Messages



9:41 AM

100%



Stephen



iMessage





9:41 AM

100%



Stephen



iMessage



-

\$1

+

Show Keypad

Request

Pay





9:41 AM

100%



Stephen



iMessage



WORDS with friends™

Create Game



Search




JibJab
JibJab Media Inc.



JIBJAB

Search party food drive

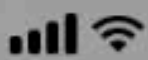


CELEBRATION

Snap a s
to get this part

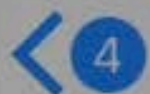


Snap a S
Fill the oval space with



9:41 AM

100%



Add your face to GIFs,
eCards and Music Videos.

Sign in with Facebook

Sign up with Email

Already have an account? [Sign In](#)



NEW

Direct Send



WORDS with friends
 Tap to Play
 Only on iOS 10

Play With Me!
 Click to play or enable game in iMessage

Add comment or Send

By continuing, you agree to Zynga's Terms of Service and Privacy Policy


```
// iMessage App Insert Draft API
```

```
open func insert(_ message: MSMessage, completionHandler: ((Error?) -> Void)? = nil)
```

```
open func insert(_ sticker: MSSticker, completionHandler: ((Error?) -> Void)? = nil)
```

```
open func insertText(_ text: String, completionHandler: ((Error?) -> Void)? = nil)
```

```
open func insertAttachment(_ URL: URL, withAlternateFilename filename: String?,  
completionHandler: ((Error?) -> Void)? = nil)
```

```
// iMessage App Direct Send API
```

```
open func send(_ message: MSMessage, completionHandler: ((Error?) -> Void)? = nil)
```

```
open func send(_ sticker: MSSticker, completionHandler: ((Error?) -> Void)? = nil)
```

```
open func sendText(_ text: String, completionHandler: ((Error?) -> Void)? = nil)
```

```
open func sendAttachment(_ URL: URL, withAlternateFilename filename: String?,  
completionHandler: ((Error?) -> Void)? = nil)
```

Choosing a Send API

Insert draft API

Insert is recommended for most iMessage apps

Best for rich message composition

- Append comments
- Send with effects, full screen moments

Allow user confirmation for sent content

Choosing a Send API

Direct send API

Provides quick fire and forget user experience

Great for flows where inserting message, then sending, adds extra step

Maintain trust in your app

- Clearly indicate to user what content will be sent
- Clearly denote UI elements that trigger a send

iMessage App Direct Send

API requirements

Messages enforces requirements for API use

- App must be visible
- App can send one message per user interaction
- No further messages can be sent until next interaction
- New error codes

iMessage App Direct Send

API requirements

Messages enforces requirements for API use

- App must be visible
- App can send one message per user interaction
- No further messages can be sent until next interaction
- New error codes

```
// MSMessage Error Codes

public enum MSMessageErrorCode: Int {
    // New iOS 11 Error Codes
    @available(iOS 11.0, *)
    case sendWithoutRecentInteraction

    @available(iOS 11.0, *)
    case sendWhileNotVisible
}
```

Direct Send

New fast and easy send experience

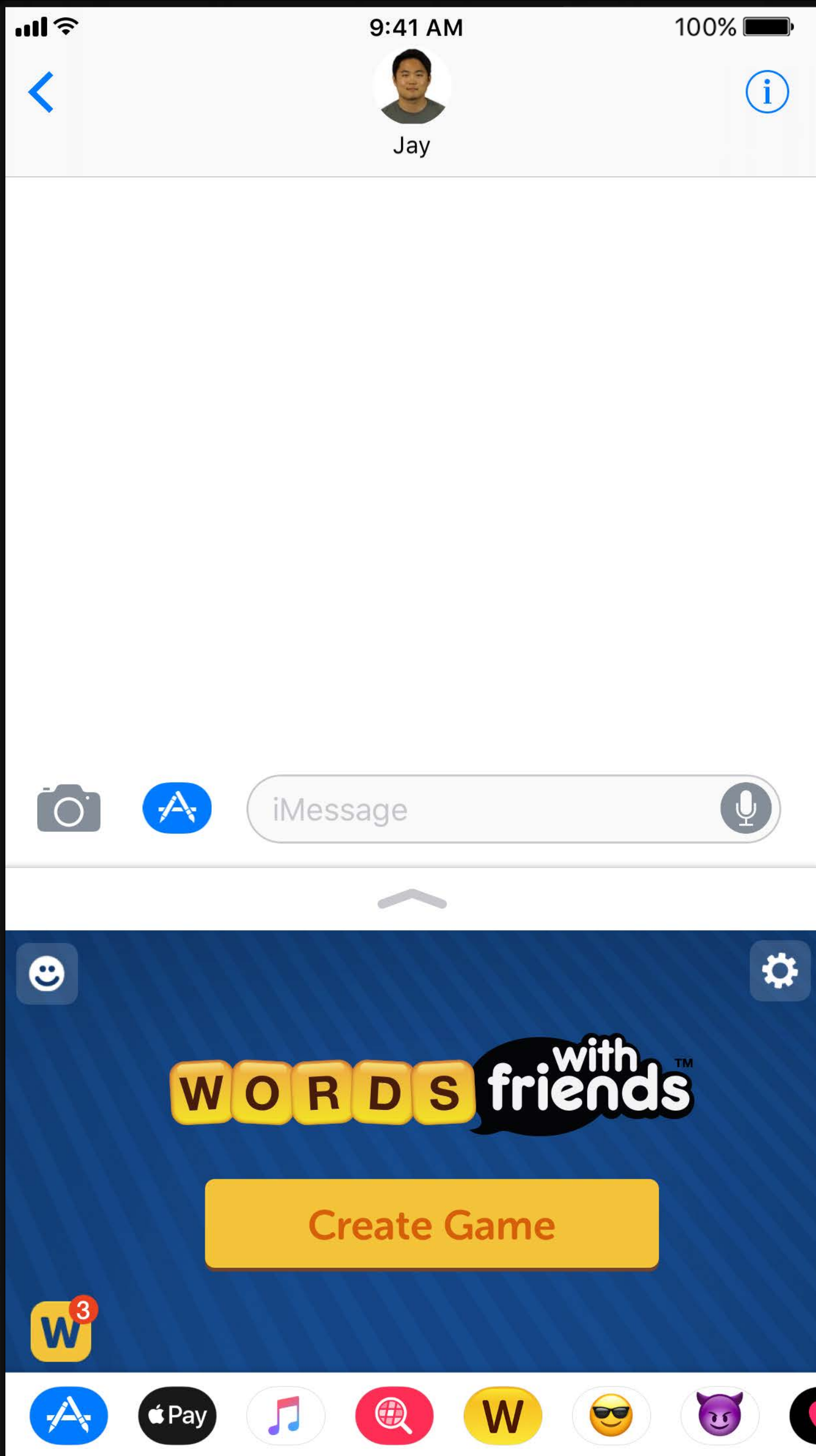
For some apps, direct send provides excellent alternative to staging drafts

Consider which API is best for your app

NEW

Live Message Layouts

Stephen Lottermoser, Messages Engineer





9:41 AM

100%



Jay



iMessage



Your App






9:41 AM

100%



Jay



 Template Image

Template message
A simple template message

Read 4:47 PM

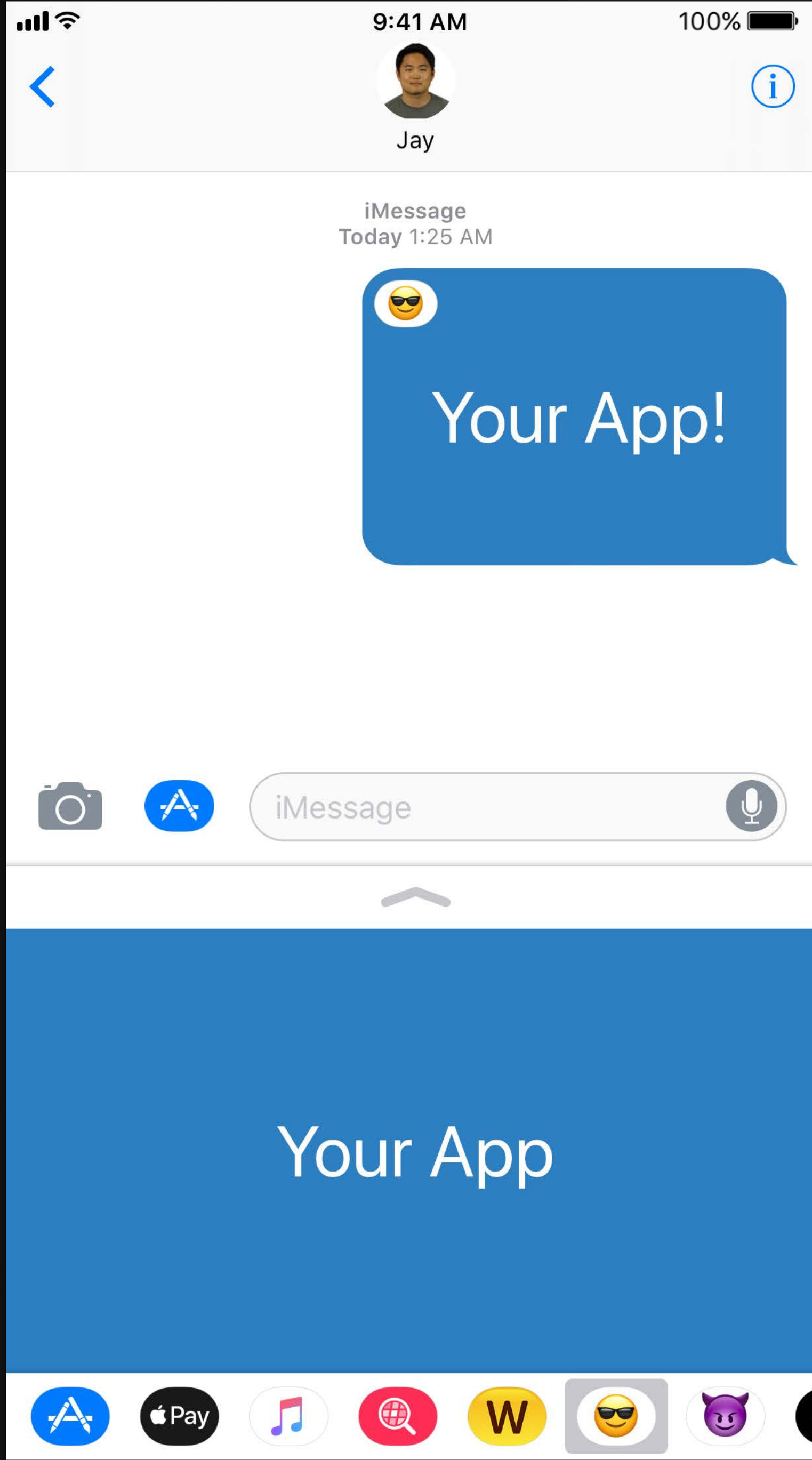


iMessage



Your App

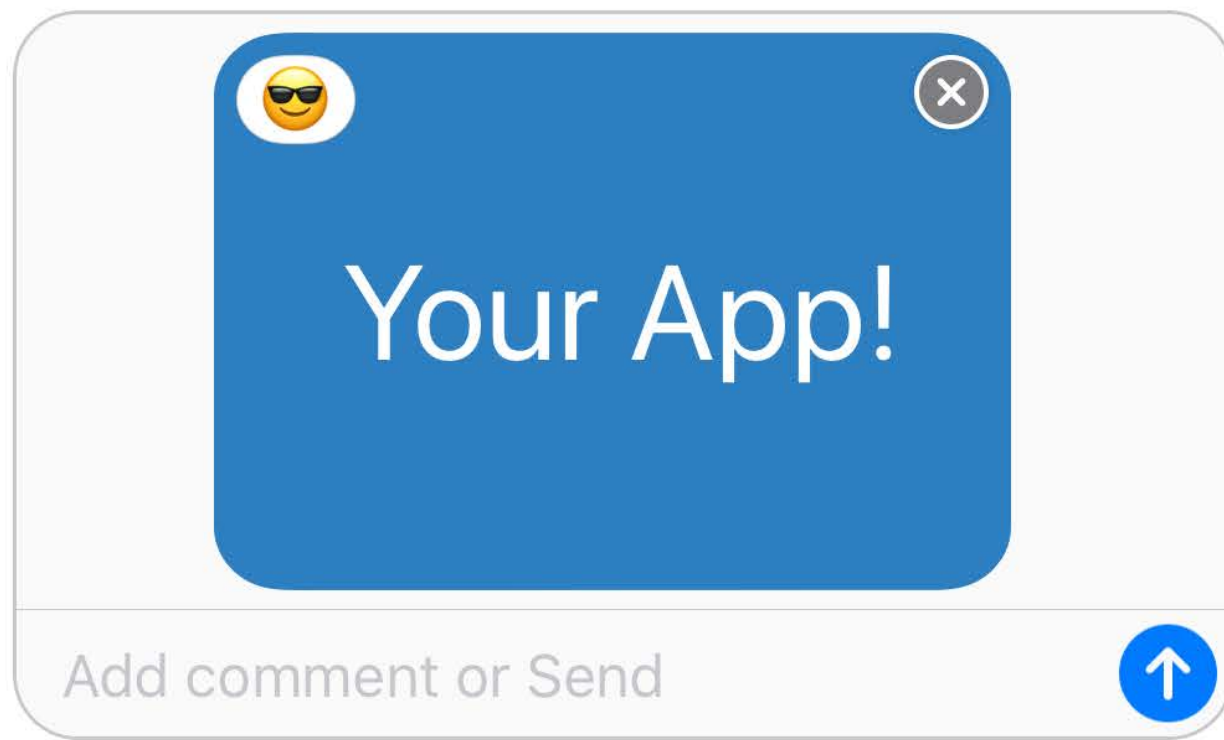




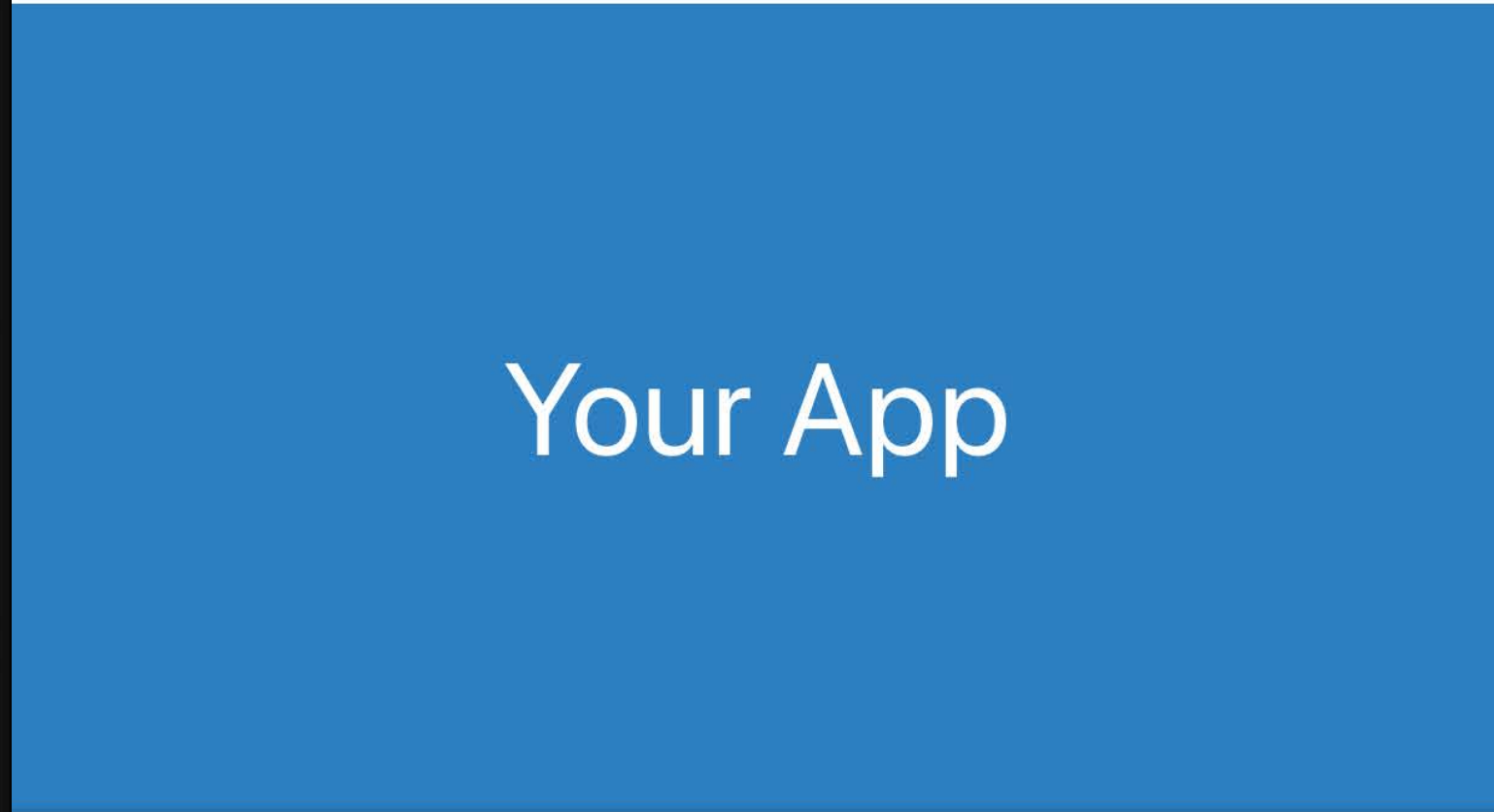
NEW



Jay



Add comment or Send



Your App





9:41 AM


100%





Jay



iMessage
Today 1:25 AM

 Your App!

 Your App!

 Your App!

Read 1:26 AM



iMessage




NEW





Jay



iMessage
Today 1:25 AM

 Your App!

 Your App!

 Someone Else's App

Read 1:26 AM



iMessage



NEW



9:41 AM


100%





Jay



iMessage
Today 1:25 AM

 Your App!

 Your App!

 Your App!

Read 1:26 AM



iMessage



NEW

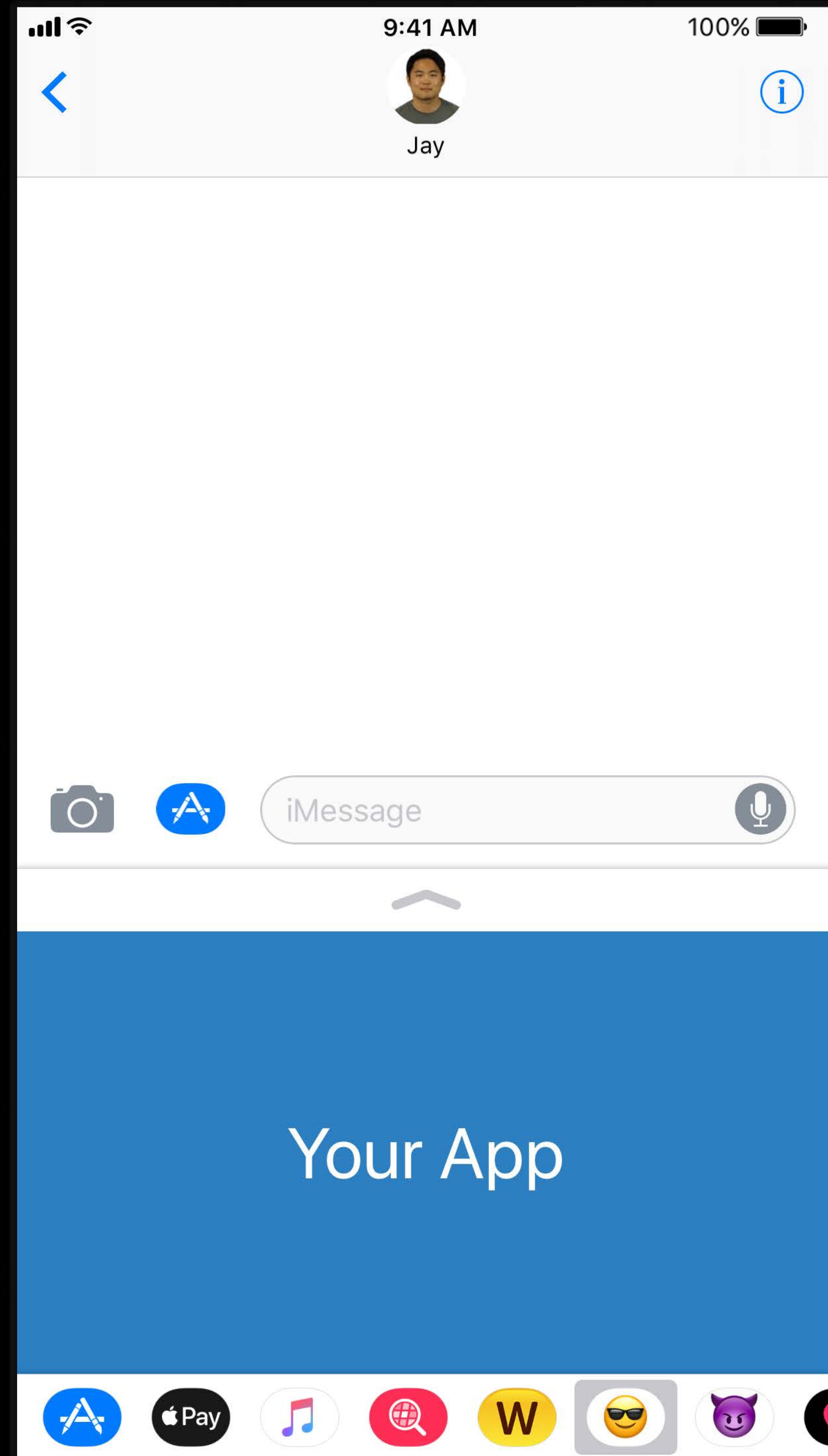
Demo

Live Message Layouts

How do they work?

iMessage app

MSMessagesAppViewController
.compact



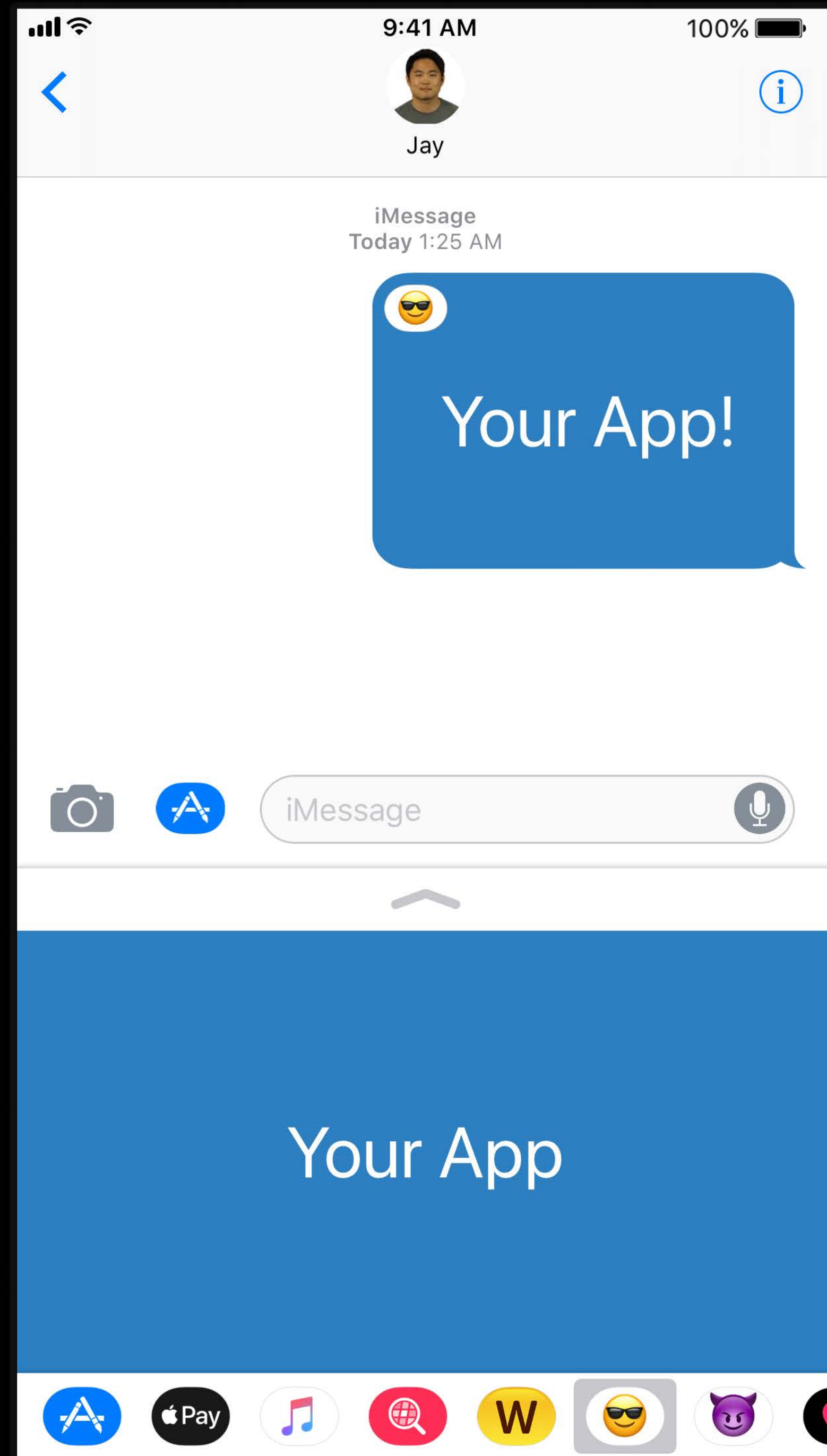
Live Message Layouts

How do they work?

iMessage app

MSMessagesAppViewController
.compact

MSMessagesAppViewController
.transcript



Live Message Layouts

How do they work?

iMessage app

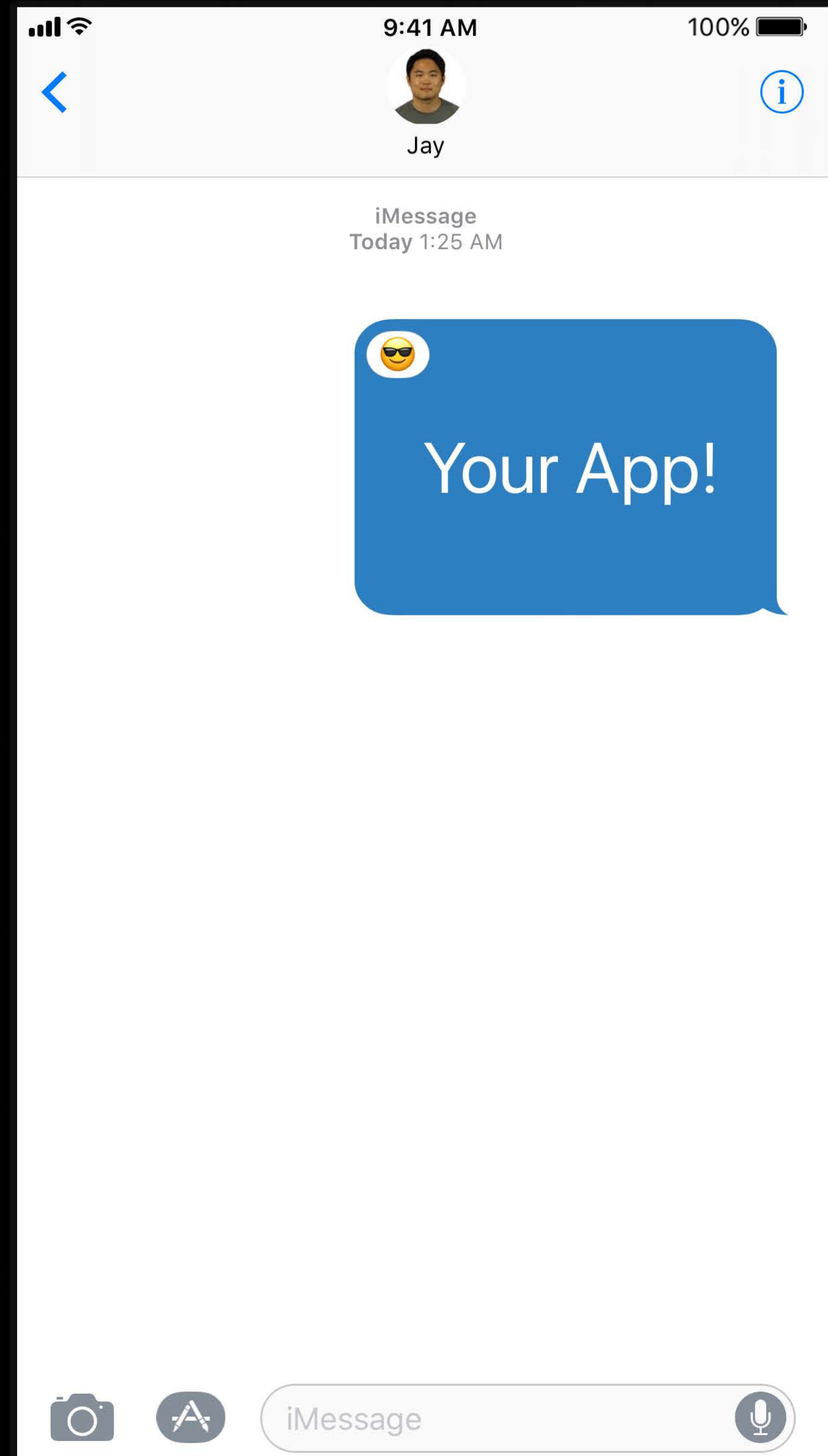


Live Message Layouts

How do they work?

iMessage app

```
MSMessagesAppViewController  
.transcript
```



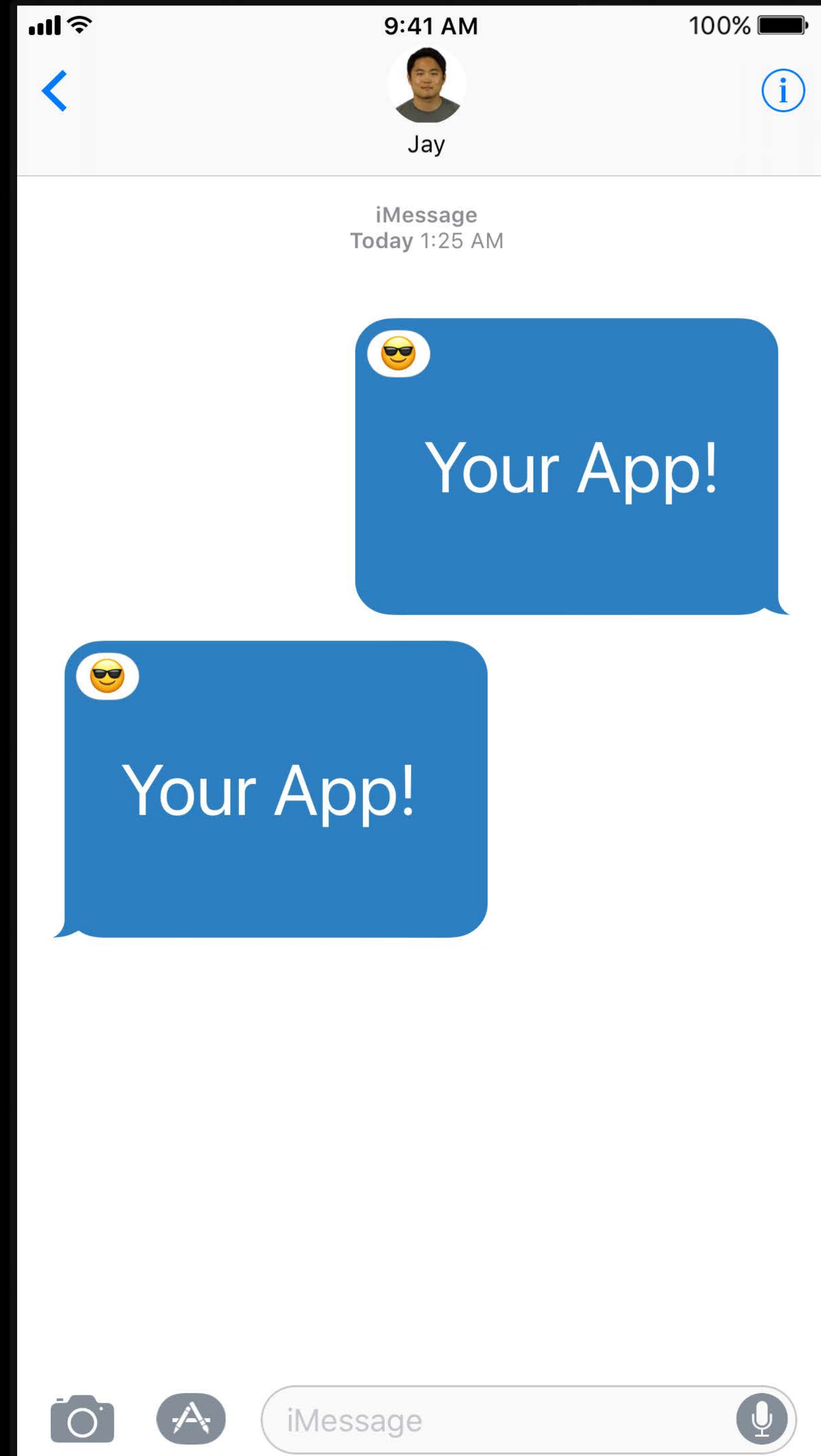
Live Message Layouts

How do they work?

iMessage app

```
MSMessagesAppViewController  
.transcript
```

```
MSMessagesAppViewController  
.transcript
```



Live Message Layouts

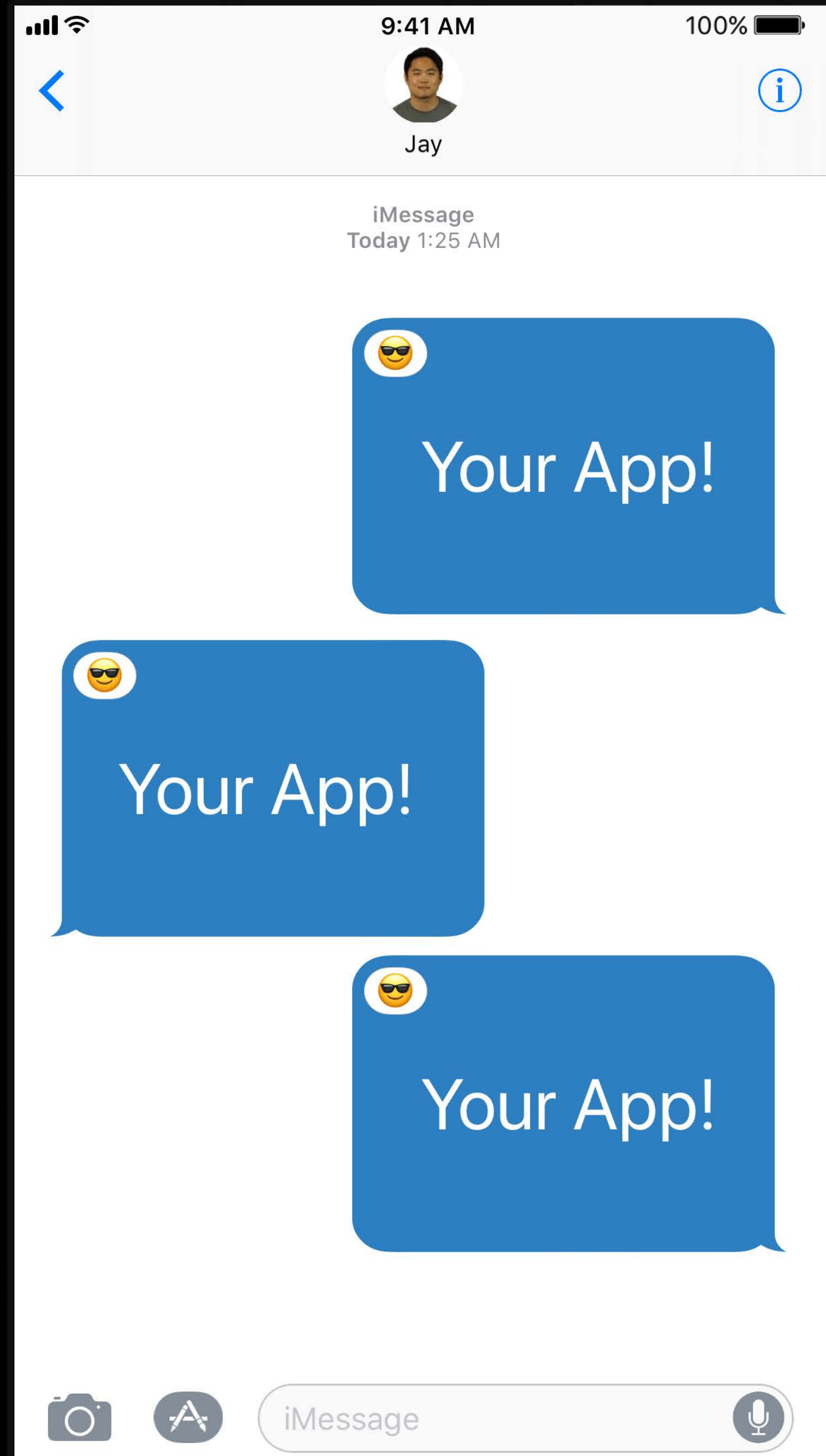
How do they work?

iMessage app

MSMessagesAppViewController
.transcript

MSMessagesAppViewController
.transcript

MSMessagesAppViewController
.transcript



Live Message Layouts

Sending and receiving messages

Jay Chae, Messages Engineer

MSMessageLayout

MSMessageTemplateLayout

MSMessageLayout

MSMessageTemplateLayout

MSMessageLiveLayout


```
// MSMessageLiveLayout

@available(iOS 11.0, *)
open class MSMessageLiveLayout : MSMessageLayout {

    public init(alternateLayout: MSMessageTemplateLayout)

    open var alternateLayout: MSMessageTemplateLayout { get }
}
```

MSMessageLiveLayout

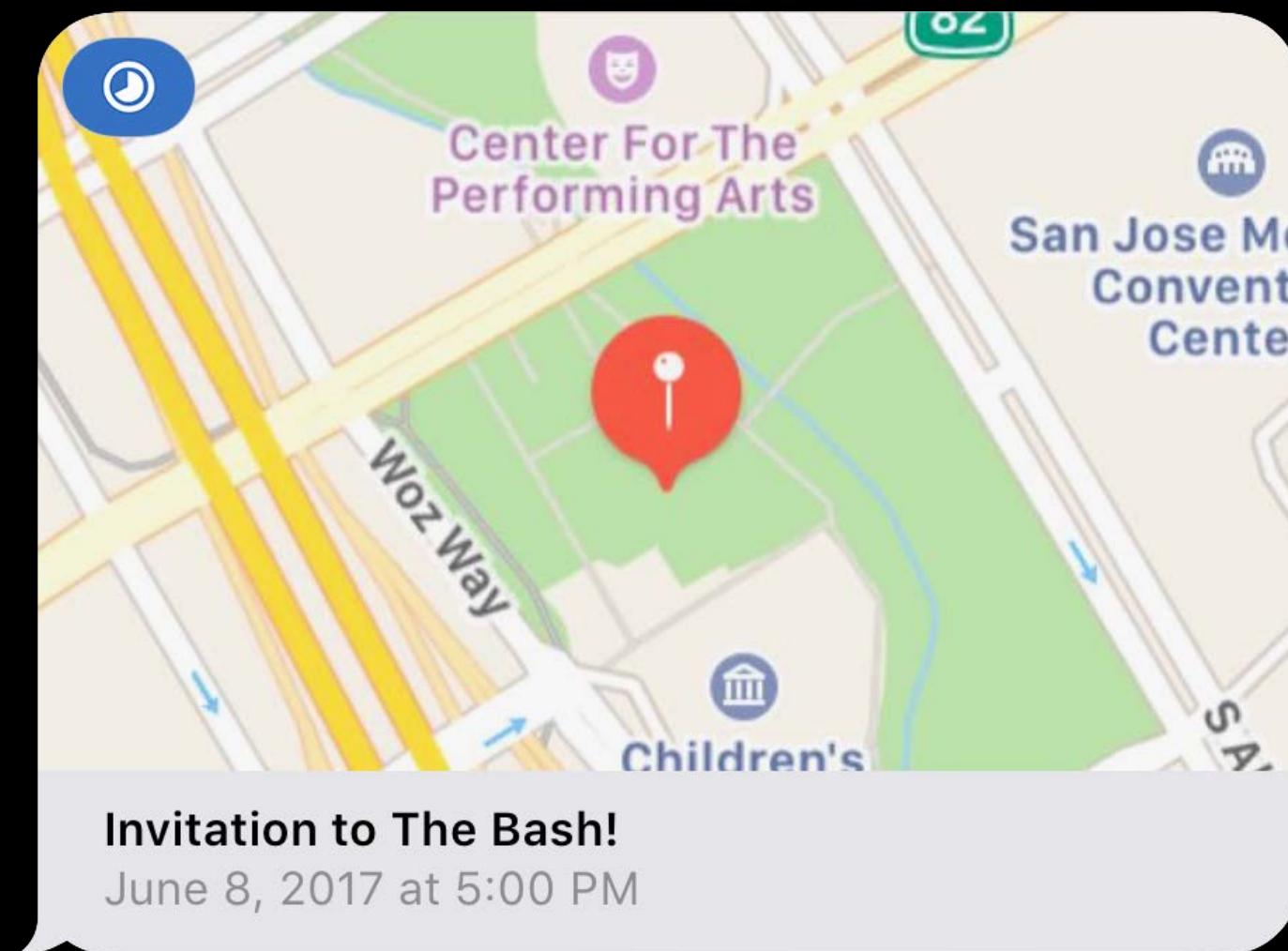
Alternate layout

Used on:

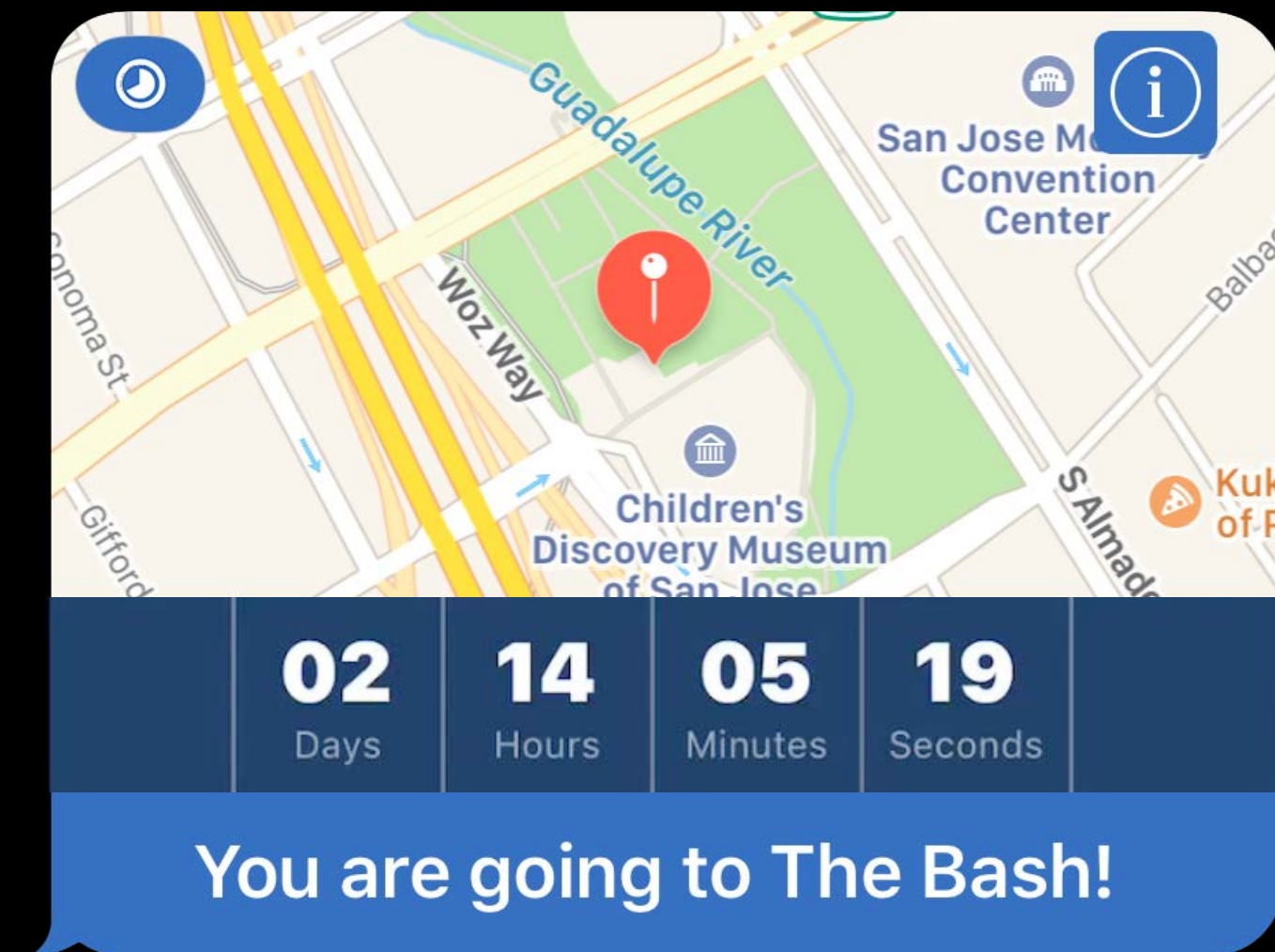
- Devices without your iMessage app installed
- Devices running iOS 10
- Devices running watchOS 3 or macOS Sierra and later

MSMessageLiveLayout

Alternate layout



iMessage app not installed



iMessage app installed

```
// Composing a Message, Sample code
```

```
let alternateLayout = MSMessageTemplateLayout()  
alternateLayout.caption = summaryText  
alternateLayout.subcaption = readableDateFormatter.string(from: event.date)  
alternateLayout.image = UIImage(named: event.fallbackImageName)
```

```
let layout = MSMessageLiveLayout(alternateLayout: alternateLayout)  
let message = MSMessage()  
message.layout = layout
```

```
conversation.send(message, completionHandler: nil)
```



```
// Composing a Message, Sample code
```

```
let alternateLayout = MSMessageTemplateLayout()  
alternateLayout.caption = summaryText  
alternateLayout.subcaption = readableDateFormatter.string(from: event.date)  
alternateLayout.image = UIImage(named: event.fallbackImageName)
```

```
let layout = MSMessageLiveLayout(alternateLayout: alternateLayout)  
let message = MSMessage()  
message.layout = layout
```

```
conversation.send(message, completionHandler: nil)
```

```
// Composing a Message, Sample code
```

```
let alternateLayout = MSMessageTemplateLayout()  
alternateLayout.caption = summaryText  
alternateLayout.subcaption = readableDateFormatter.string(from: event.date)  
alternateLayout.image = UIImage(named: event.fallbackImageName)
```

```
let layout = MSMessageLiveLayout(alternateLayout: alternateLayout)  
let message = MSMessage()  
message.layout = layout
```

```
conversation.send(message, completionHandler: nil)
```

```
// Composing a Message, Sample code
```

```
let alternateLayout = MSMessageTemplateLayout()  
alternateLayout.caption = summaryText  
alternateLayout.subcaption = readableDateFormatter.string(from: event.date)  
alternateLayout.image = UIImage(named: event.fallbackImageName)
```

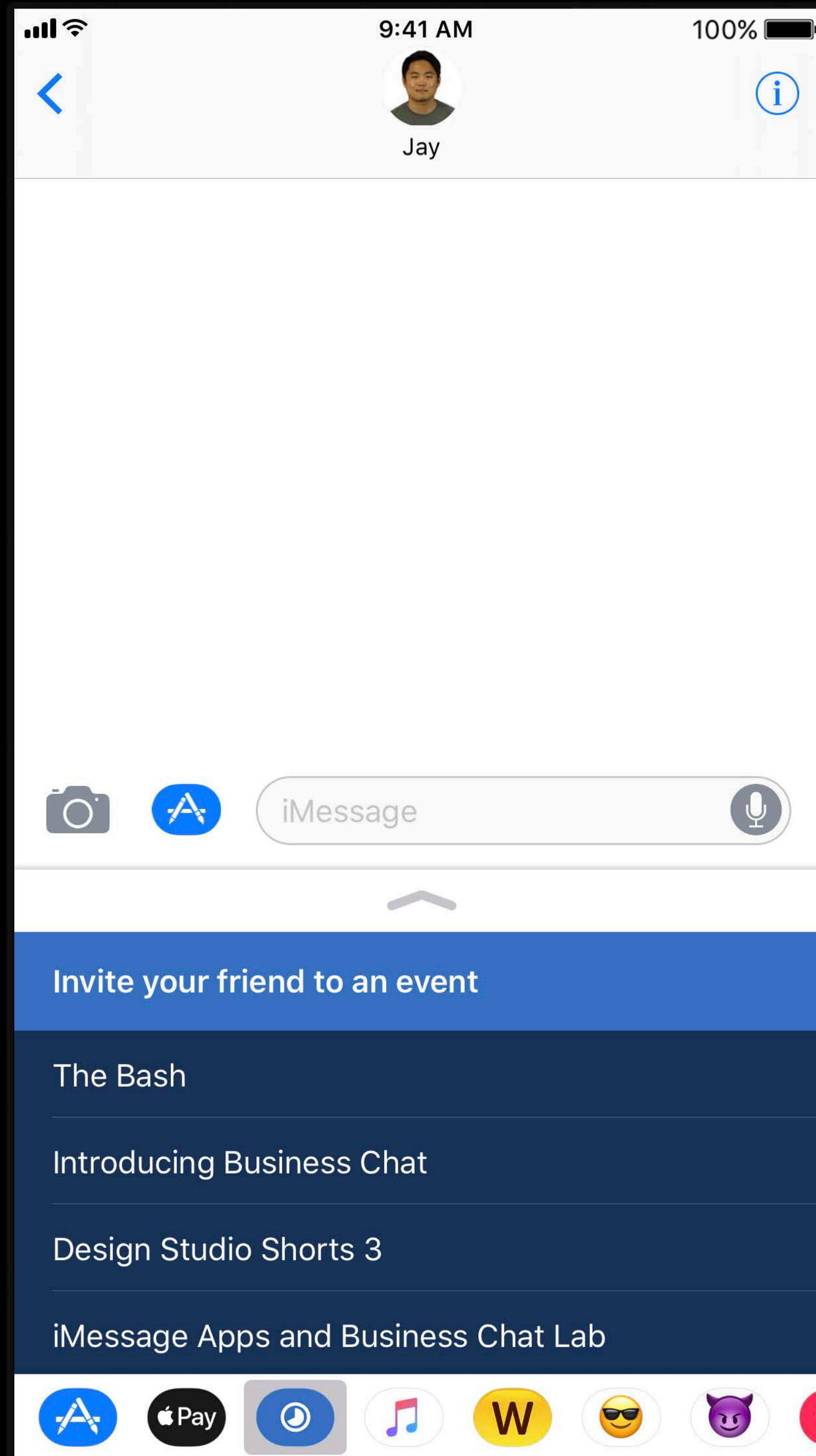
```
let layout = MSMessageLiveLayout(alternateLayout: alternateLayout)  
let message = MSMessage()  
message.layout = layout
```

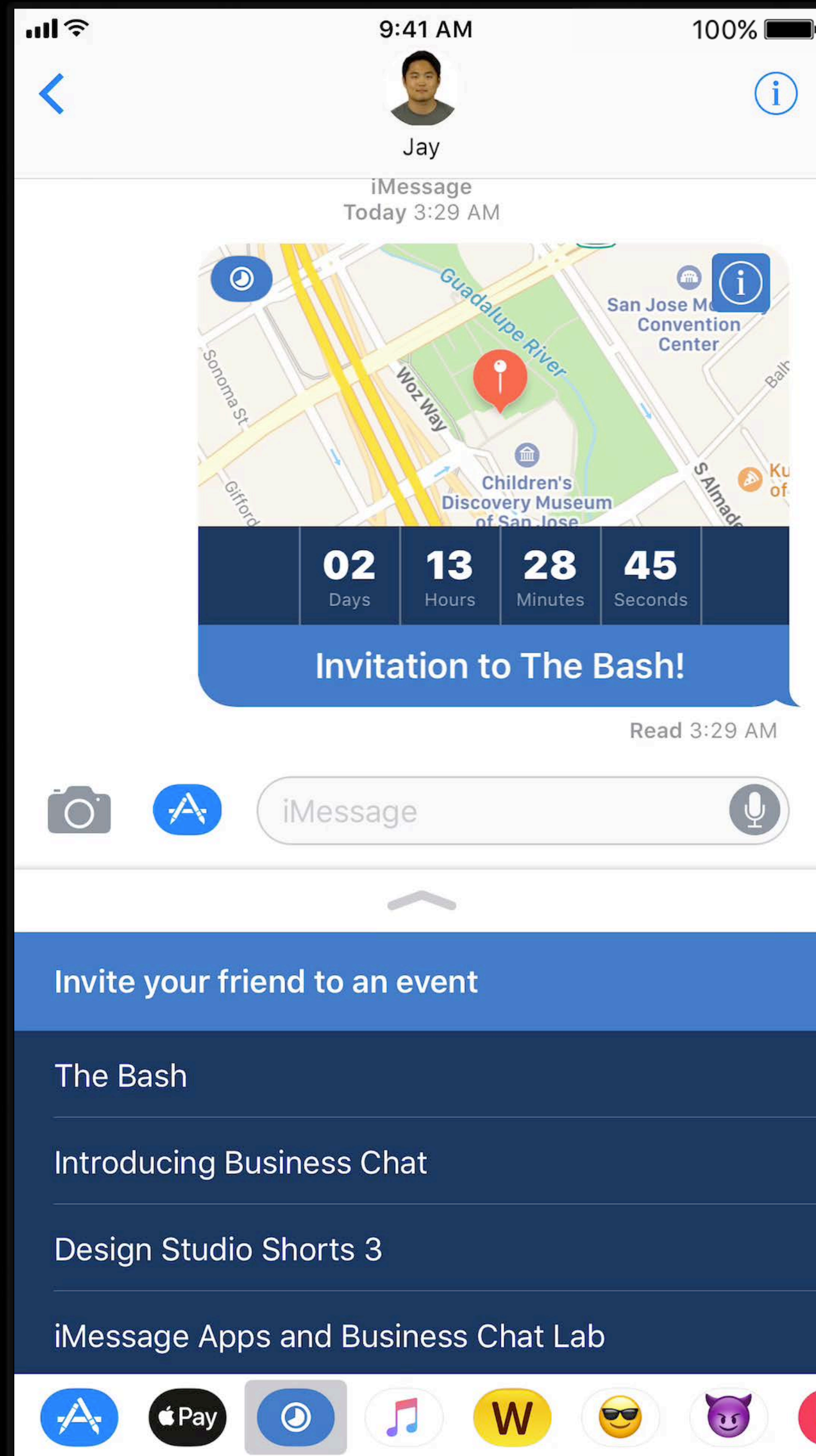
```
conversation.send(message, completionHandler: nil)
```

MSMessagesAppViewController

MSMessagesAppViewController

MSMessagesAppViewController





MSMessagesAppViewController

MSMessagesAppViewController

```
// Configuring a MSMessagesAppViewController subclass for display
```

```
public enum MSMessagesAppPresentationStyle : UInt {  
    case compact  
    case expanded  
    @available(iOS 11.0, *)  
    case transcript  
}
```



```
// Configuring a MSMessagesAppViewController subclass for display
```

```
public enum MSMessagesAppPresentationStyle : UInt {  
    case compact  
    case expanded  
    @available(iOS 11.0, *)  
    case transcript  
}
```

MSMessagesAppViewController

Configuring for display

When `willBecomeActive(with:)` is called, you have enough information to configure your view controller

`presentationStyle`

`activeConversation?.selectedMessage`

MSMessagesAppViewController

Configuring for display

When `willBecomeActive(with:)` is called, you have enough information to configure your view controller

`presentationStyle`

`activeConversation?.selectedMessage`



```
graph TD; A[MSMessagesAppViewController] --- B[Transcript Child View Controller (selectedMessage)];
```

MSMessagesAppViewController
.transcript

Transcript Child View Controller
(selectedMessage)

MSMessagesAppViewController

Configuring for display

When `willBecomeActive(with:)` is called, you have enough information to configure your view controller

`presentationStyle`

`activeConversation?.selectedMessage`

MSMessagesAppViewController
.transcript

Transcript Child View Controller
(selectedMessage)

MSMessagesAppViewController
.compact

Compact Child View Controller
(selectedMessage)

```
// Configuring a MSMessagesAppViewController subclass for display
```

```
override func willBecomeActive(with conversation: MSConversation) {  
    super.willBecomeActive(with: conversation)  
    let message = activeConversation?.selectedMessage  
  
    switch presentationStyle {  
    case .compact:  
        presentSummaryViewController(for: message)  
    case .expanded:  
        presentDetailViewController(for: message)  
  
    }  
}
```

```
// Configuring a MSMessagesAppViewController subclass for display

override func willBecomeActive(with conversation: MSConversation) {
    super.willBecomeActive(with: conversation)
    let message = activeConversation?.selectedMessage

    switch presentationStyle {
    case .compact:
        presentSummaryViewController(for: message)
    case .expanded:
        presentDetailViewController(for: message)
    case .transcript:
        presentTranscriptViewController(for: message)
    }
}
}
```

```
// Configuring a MSMessagesAppViewController subclass for display
```

```
override func willBecomeActive(with conversation: MSConversation) {  
    super.willBecomeActive(with: conversation)  
    let message = activeConversation?.selectedMessage  
  
    switch presentationStyle {  
    case .compact:  
        presentSummaryViewController(for: message)  
    case .expanded:  
        presentDetailViewController(for: message)  
    case .transcript:  
        presentTranscriptViewController(for: message)  
    }  
}
```

```
// Configuring a MSMessagesAppViewController subclass for display
```

```
public enum MSMessagesAppPresentationStyle : UInt {  
    case compact  
    case expanded  
    @available(iOS 11.0, *)  
    case transcript  
}
```



```
// Configuring a MSMessagesAppViewController subclass for display
```

```
public enum MSMessagesAppPresentationStyle : UInt {  
    case compact  
    case expanded  
    @available(iOS 11.0, *)  
    case transcript  
}
```

```
@available(iOS 11.0, *)  
public protocol MSMessagesAppTranscriptPresentation {  
    public func contentSizeThatFits(_ size: CGSize) -> CGSize  
}
```

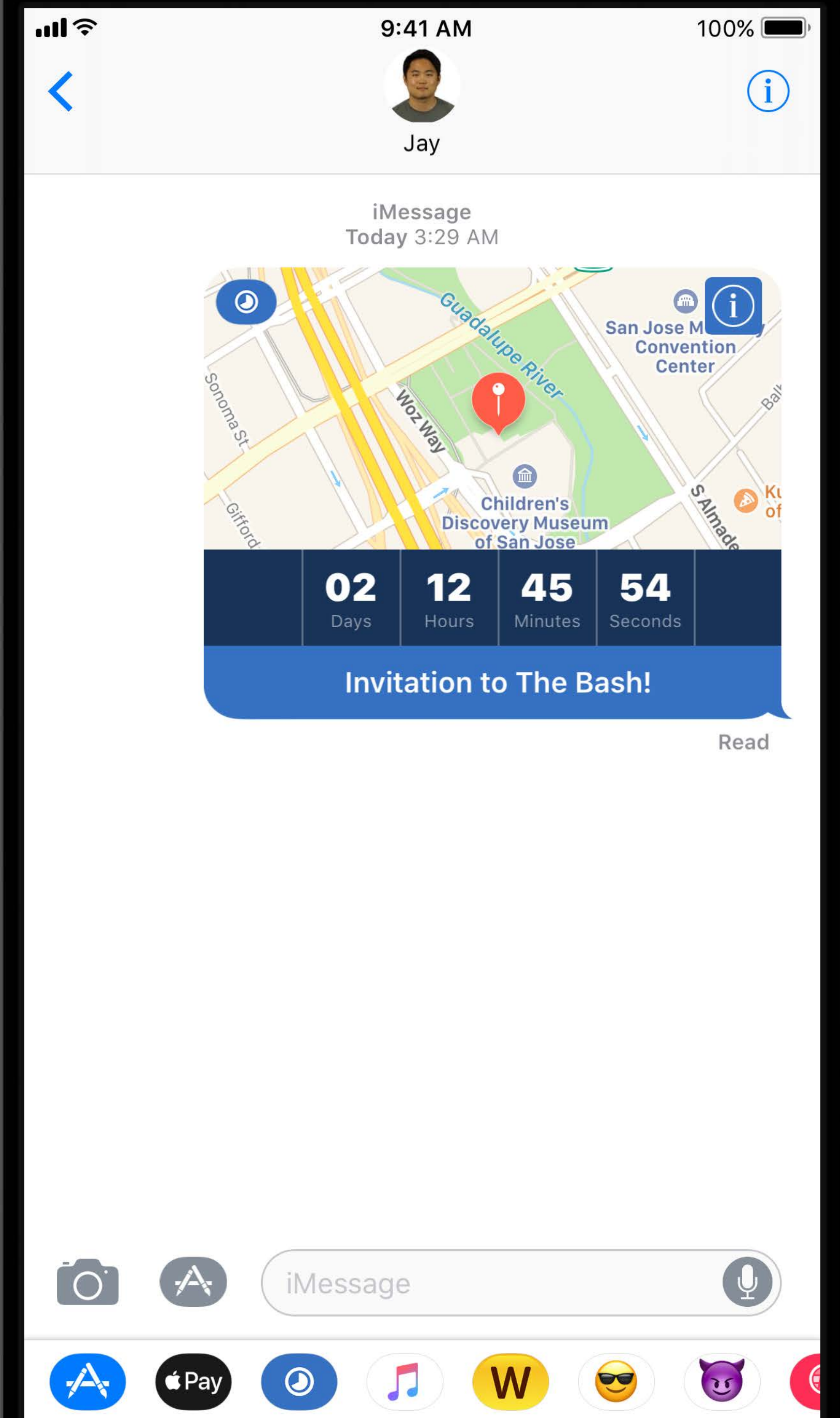
Displaying a Message

Sizing your content

```
override func contentSizeThatFits(_ size: CGSize) -> CGSize {
    let contentHeight: CGFloat = 217.0

    let titleFont = EventCountdownTranscriptView.titleFont
    let titleHeight = titleFont.lineHeight

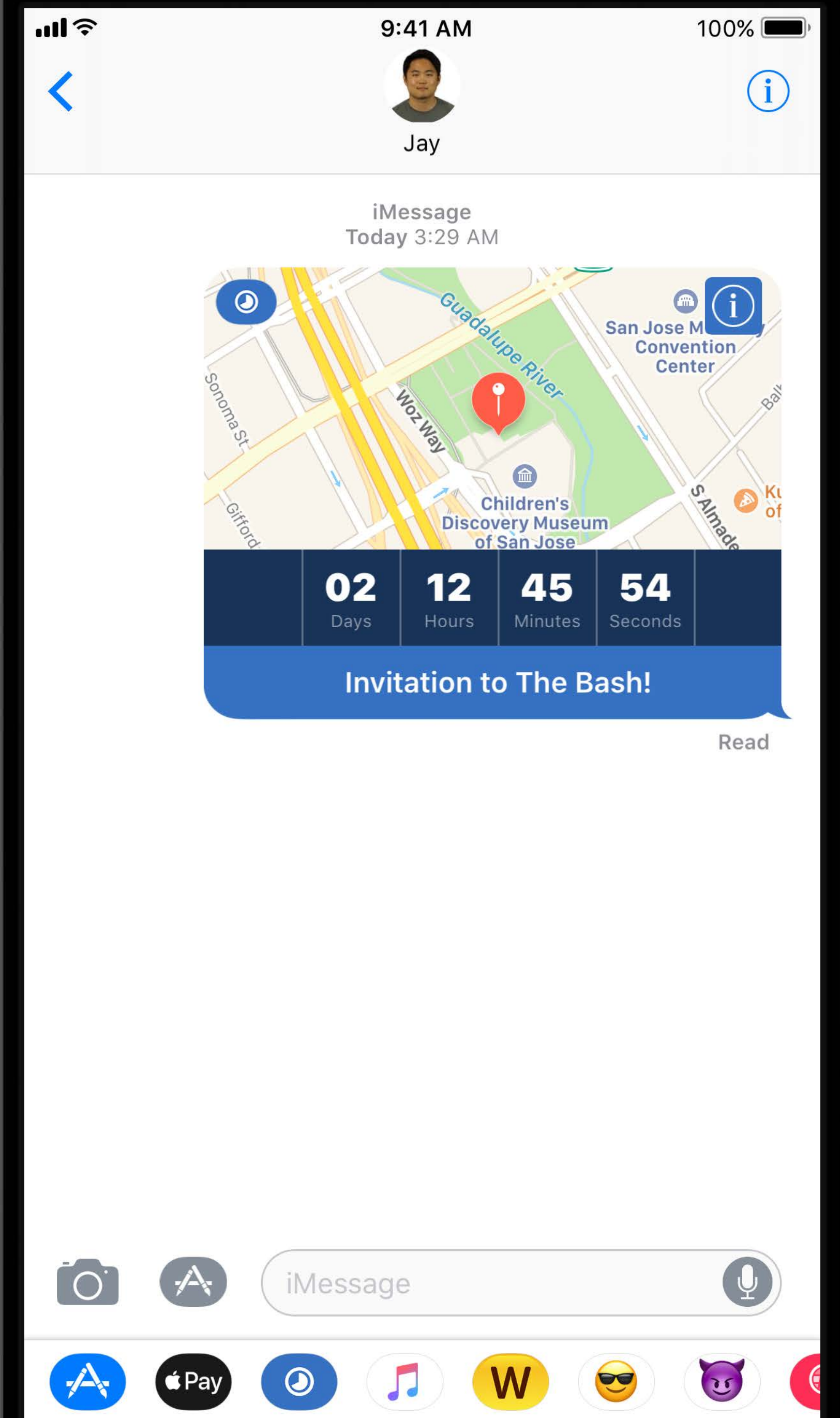
    let totalHeight = contentHeight + titleHeight
    return CGSize(width: size.width, height: totalHeight)
}
```



Displaying a Message

Sizing your content

```
override func contentSizeThatFits(_ size: CGSize) -> CGSize {  
    let contentHeight: CGFloat = 217.0  
  
    let titleFont = EventCountdownTranscriptView.titleFont  
    let titleHeight = titleFont.lineHeight  
  
    let totalHeight = contentHeight + titleHeight  
    return CGSize(width: size.width, height: totalHeight)  
}
```



Displaying a Message

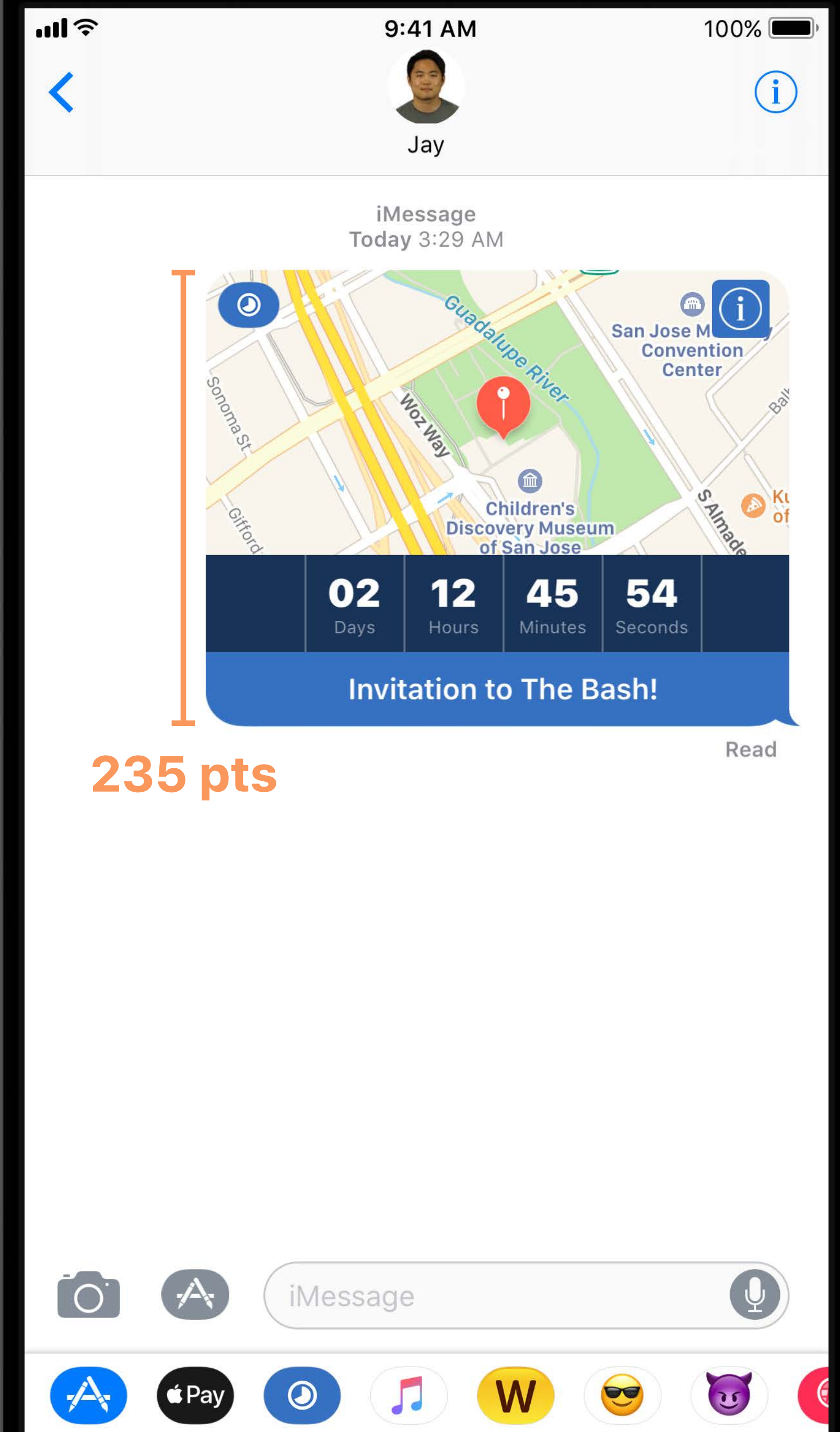
Sizing your content

```
override func contentSizeThatFits(_ size: CGSize) -> CGSize {  
    let contentHeight: CGFloat = 217.0
```

```
    let titleFont = EventCountdownTranscriptView.titleFont  
    let titleHeight = titleFont.lineHeight
```

```
    let totalHeight = contentHeight + titleHeight  
    return CGSize(width: size.width, height: totalHeight)
```

```
}
```

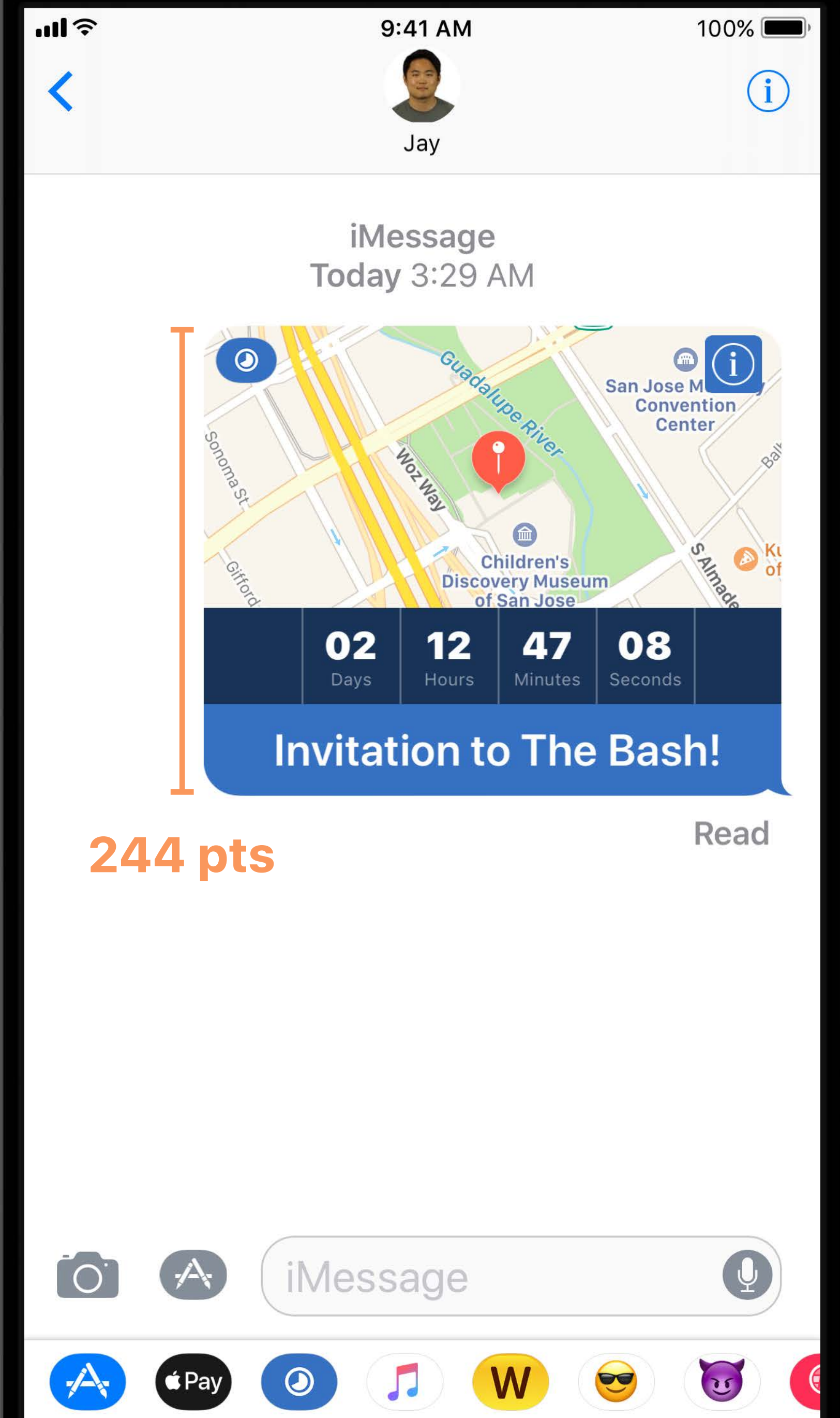


235 pts

Displaying a Message

Sizing your content

```
override func contentSizeThatFits(_ size: CGSize) -> CGSize {  
    let contentHeight: CGFloat = 217.0  
  
    let titleFont = EventCountdownTranscriptView.titleFont  
    let titleHeight = titleFont.lineHeight  
  
    let totalHeight = contentHeight + titleHeight  
    return CGSize(width: size.width, height: totalHeight)  
}
```



244 pts



Code Demo

Live Message Layouts

Interaction and more

Stephen Lottermoser, Messages Engineer

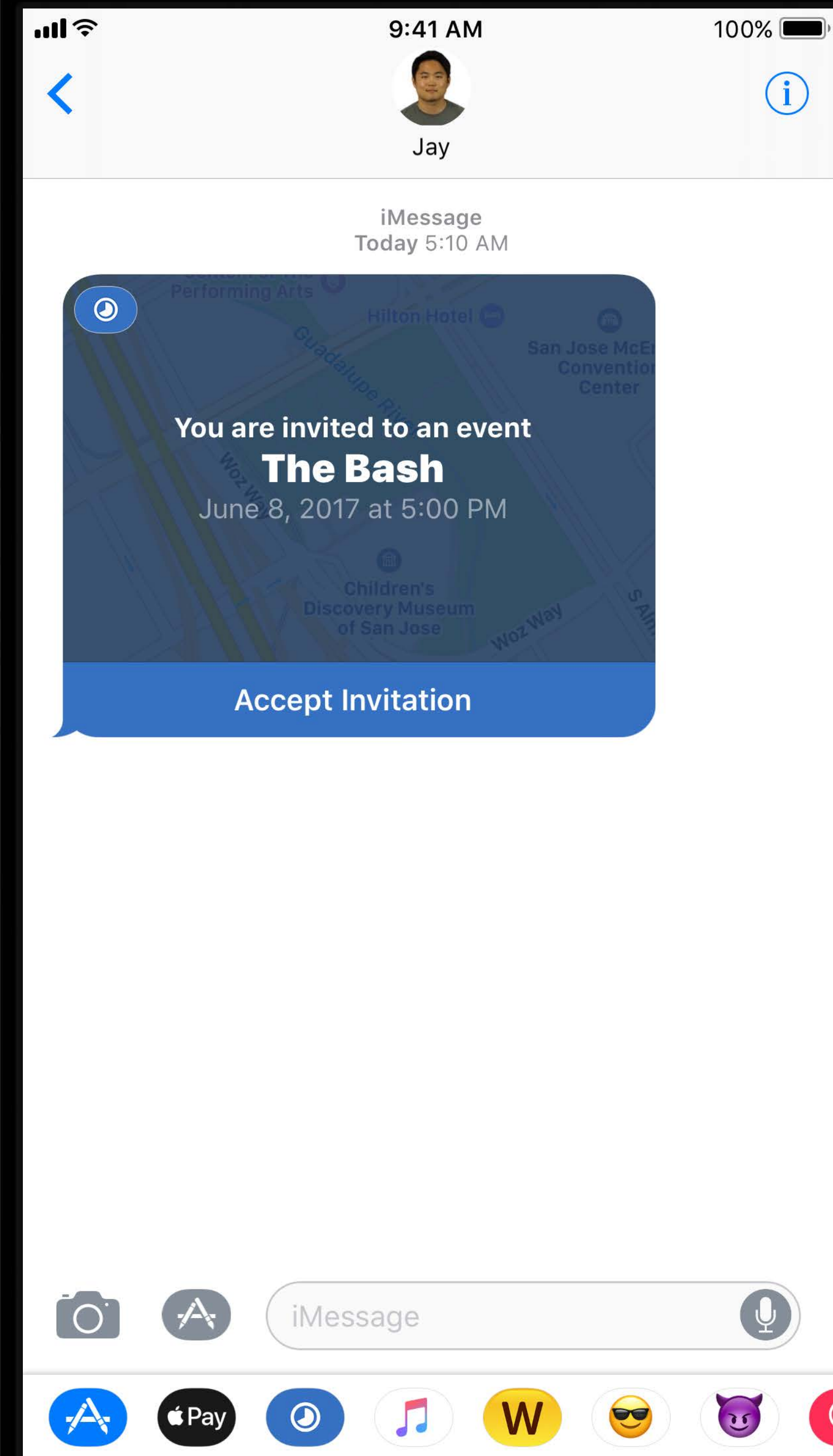
Demo

Adding interaction

Interaction

Keep views lightweight

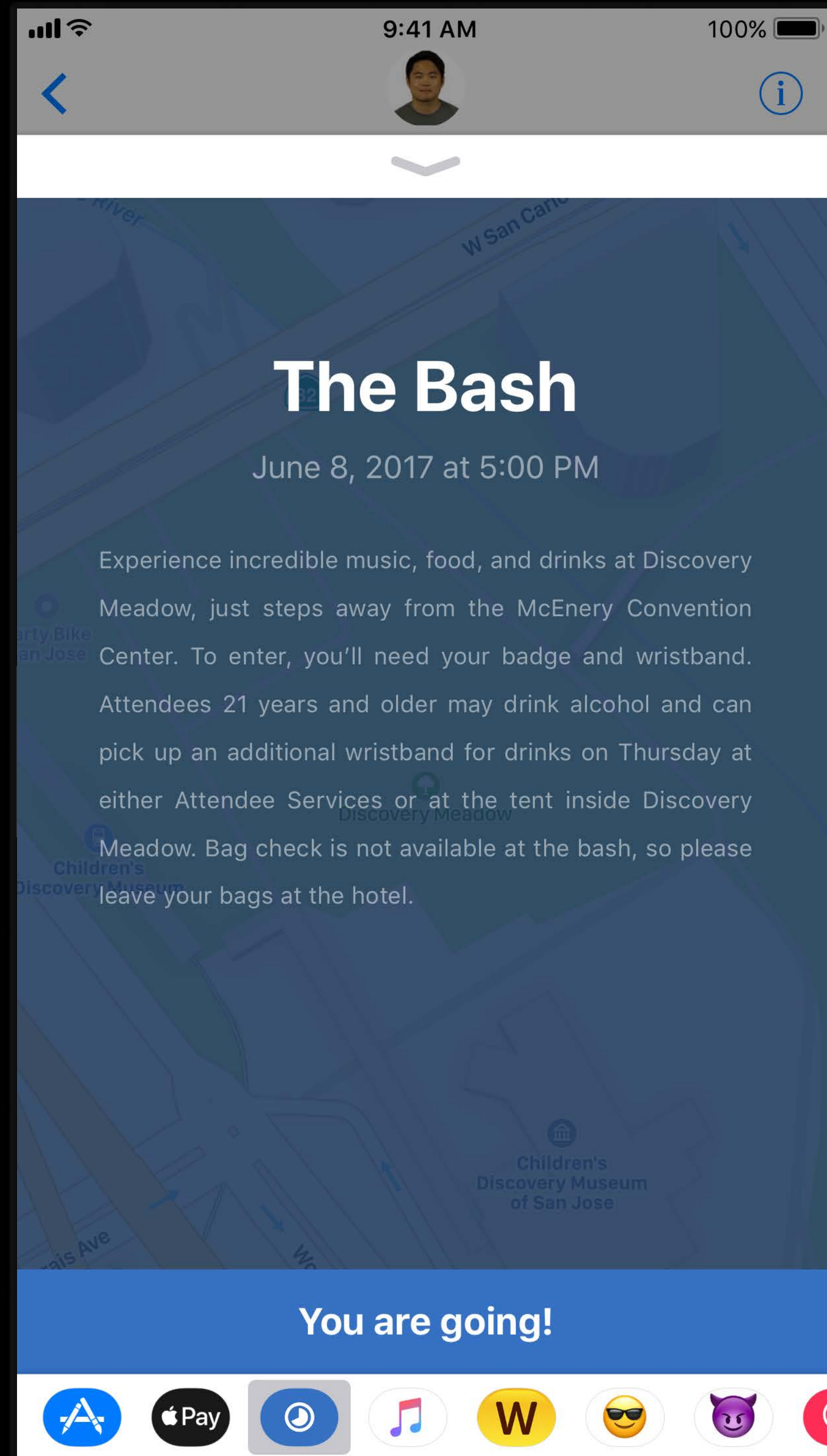
- Simple buttons or tap gestures
- Limit use of scroll views
- No keyboard input



Interaction

Use other presentation styles for complex interactions

```
requestPresentationStyle(.expanded)
```



Lifecycle

Lifecycle

Extension callbacks

In iOS 10, these methods referred to the **extension's** lifecycle

```
class MSMessagesAppViewController {  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didResignActive(with conversation: MSConversation)  
  
}
```

Lifecycle

Extension callbacks

In iOS 10, these methods referred to the **extension's** lifecycle

```
class MSMessagesAppViewController {  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didResignActive(with conversation: MSConversation)  
  
}
```

Lifecycle

Extension callbacks

In iOS 11, these methods referred to the [view controller's](#) lifecycle

```
class MSMessagesAppViewController {  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didResignActive(with conversation: MSConversation)  
  
}
```

Lifecycle

Extension callbacks

Observe extension notifications for extension lifecycle events

```
public static let NSExtensionHostWillEnterForeground: NSNotification.Name
```

```
public static let NSExtensionHostDidEnterBackground: NSNotification.Name
```

```
public static let NSExtensionHostWillResignActive: NSNotification.Name
```

```
public static let NSExtensionHostDidBecomeActive: NSNotification.Name
```


View Controller Lifecycle

Live message layout

When presenting a transcript presentation style view controller, methods will be called in this order

```
class MSMessagesAppViewController {
```

```
}
```

View Controller Lifecycle

Live message layout

When presenting a transcript presentation style view controller, methods will be called in this order

```
class MSMessagesAppViewController {  
    public init(nibName nibNameOrNil: String?, bundle nibBundleOrNil: Bundle?)  
    open func viewDidLoad()  
  
}
```

View Controller Lifecycle

Live message layout

When presenting a transcript presentation style view controller, methods will be called in this order

```
class MSMessagesAppViewController {  
  
    public init(nibName nibNameOrNil: String?, bundle nibBundleOrNil: Bundle?)  
    open func viewDidLoad()  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didBecomeActive(with conversation: MSConversation)  
  
}
```

View Controller Lifecycle

Live message layout

When presenting a transcript presentation style view controller, methods will be called in this order

```
class MSMessagesAppViewController {  
  
    public init(nibName nibNameOrNil: String?, bundle nibBundleOrNil: Bundle?)  
    open func viewDidLoad()  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didBecomeActive(with conversation: MSConversation)  
  
    open func viewWillAppear(_ animated: Bool)  
    open func viewDidAppear(_ animated: Bool)  
  
}
```

View Controller Lifecycle

Live message layout

When presenting a transcript presentation style view controller, methods will be called in this order

```
class MSMessagesAppViewController {  
  
    public init(nibName nibNameOrNil: String?, bundle nibBundleOrNil: Bundle?)  
    open func viewDidLoad()  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didBecomeActive(with conversation: MSConversation)  
  
    open func viewWillAppear(_ animated: Bool)  
    open func viewDidAppear(_ animated: Bool)  
  
    open func contentSizeThatFits(_ size: CGSize) -> CGSize  
  
}
```

View Controller Lifecycle

Live message layout

When `willBecomeActive` is called, you have enough information to configure your view controller

```
class MSMessagesAppViewController {  
    public init(nibName nibNameOrNil: String?, bundle nibBundleOrNil: Bundle?)  
    open func viewDidLoad()  
  
    open func willBecomeActive(with conversation: MSConversation)  
    open func didBecomeActive(with conversation: MSConversation)  
  
    open func viewWillAppear(_ animated: Bool)  
    open func viewDidAppear(_ animated: Bool)  
  
    open func contentSizeThatFits(_ size: CGSize) -> CGSize  
  
}
```

View Controller Lifecycle

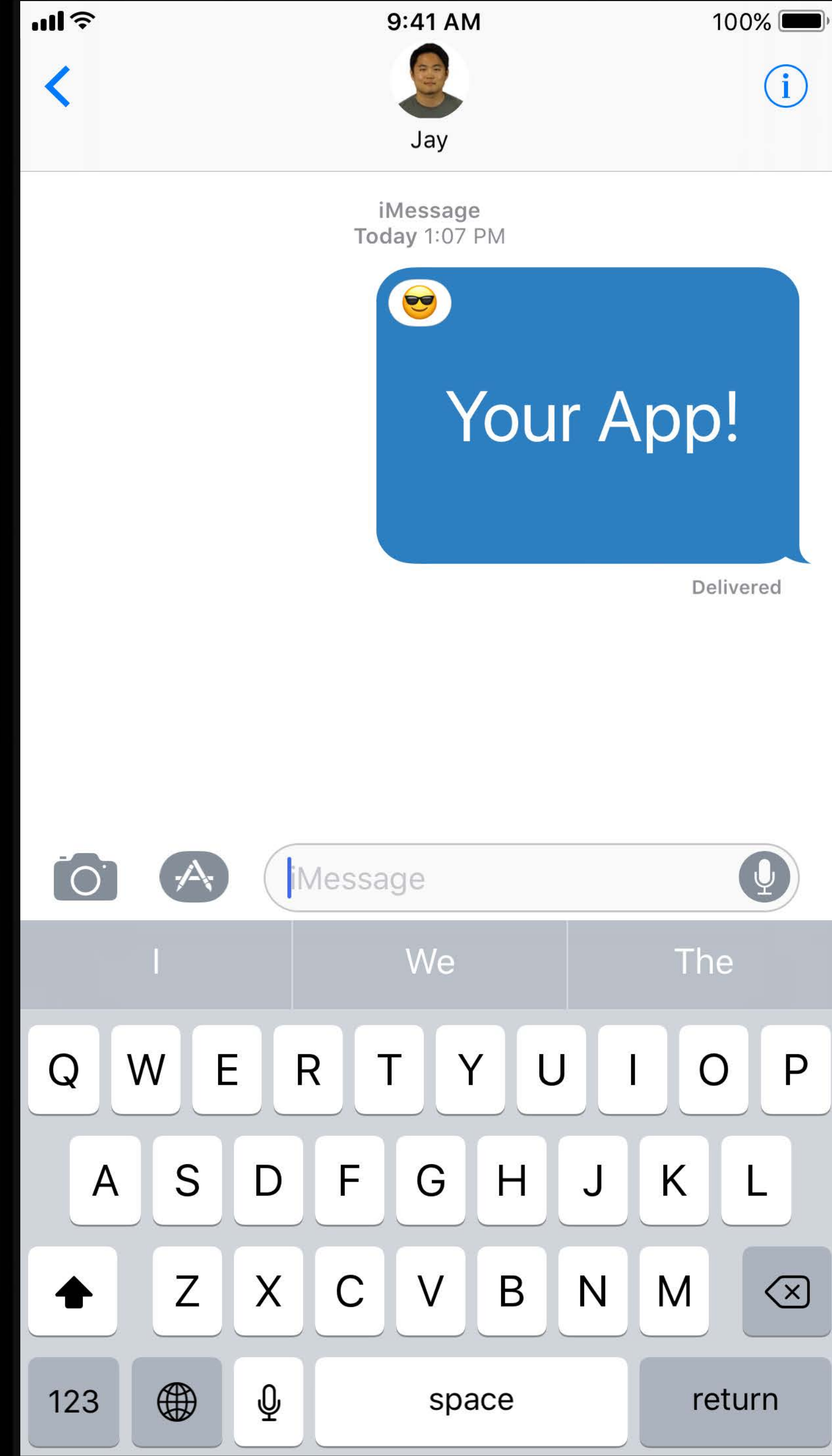
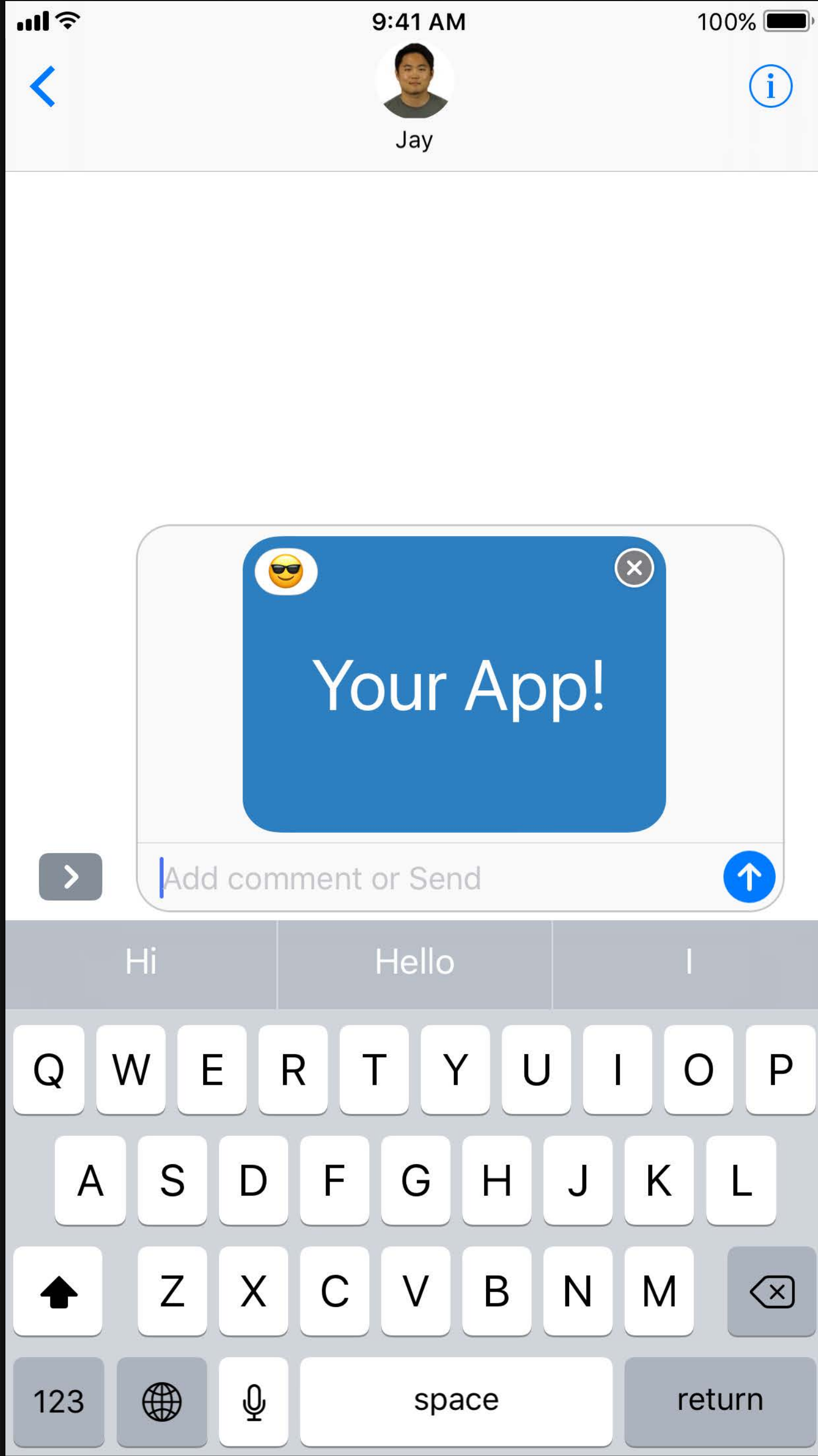
Live message layout

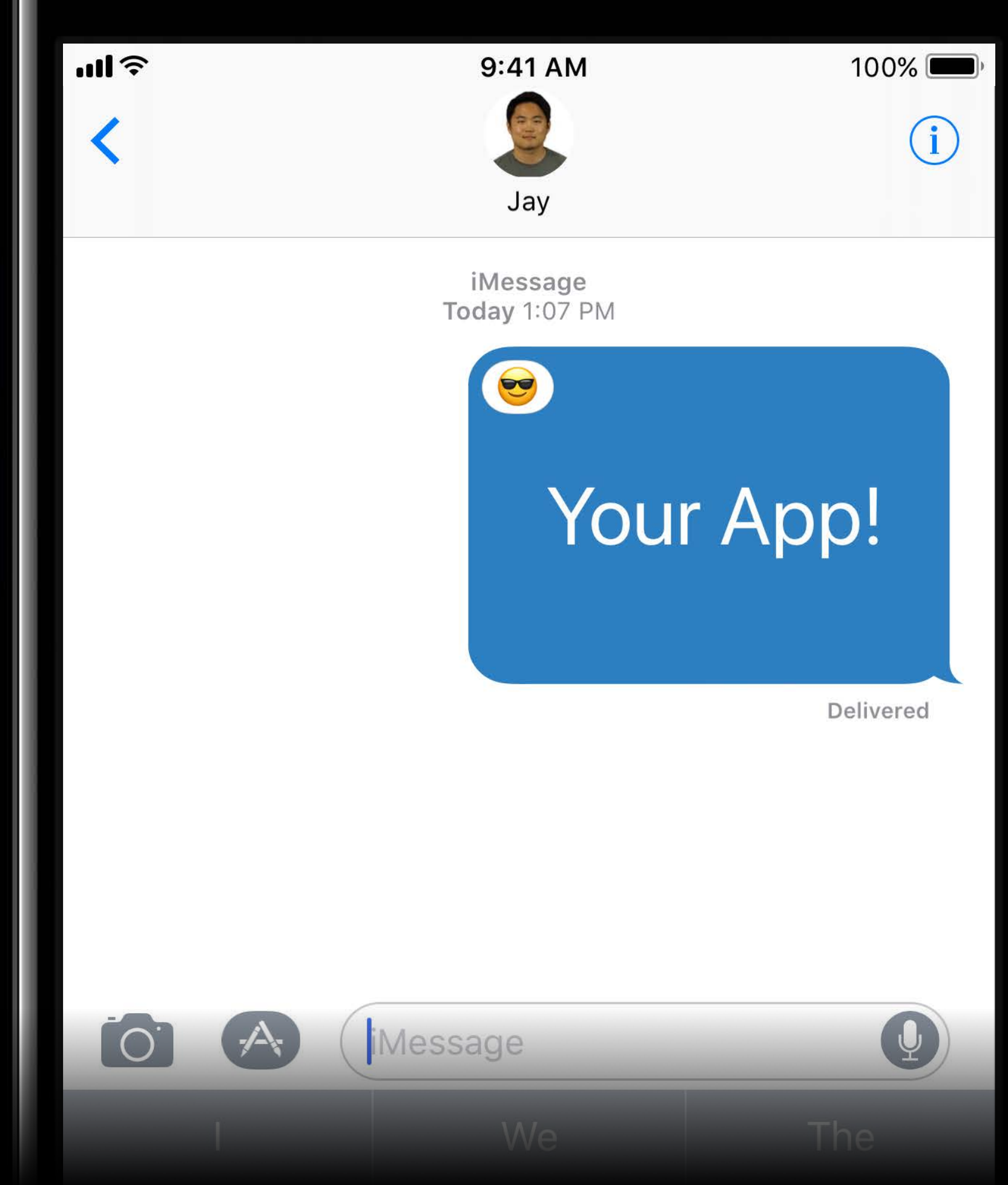
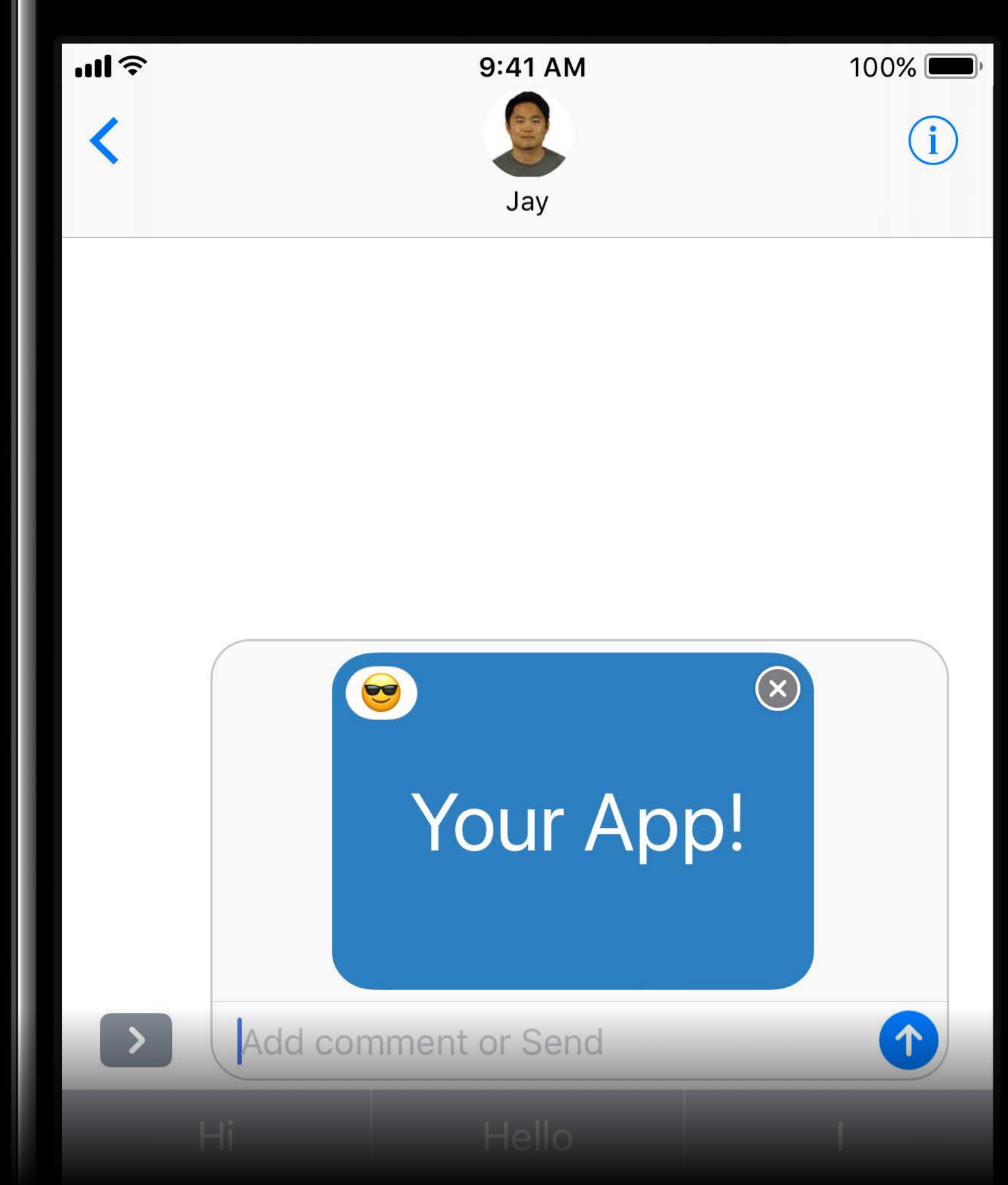
`contentSizeThatFits()` is called last

It may be called multiple times in response to events in Messages

```
open func contentSizeThatFits(_ size: CGSize) -> CGSize
```

Pending Messages





```
open class MSMessage: NSObject, NSCopying, NSSecureCoding {  
  
    @available(iOS 11.0, *)  
    open var isPending: Bool { get }  
  
}
```

Live Message Layouts

Interactive iMessage apps in the transcript!

Same `MSMessagesAppViewController` base class

New layout: `MSMessageLiveLayout`

New presentation style: `.transcript`

Best Practices

Eugene Bistolas, Messages Engineer

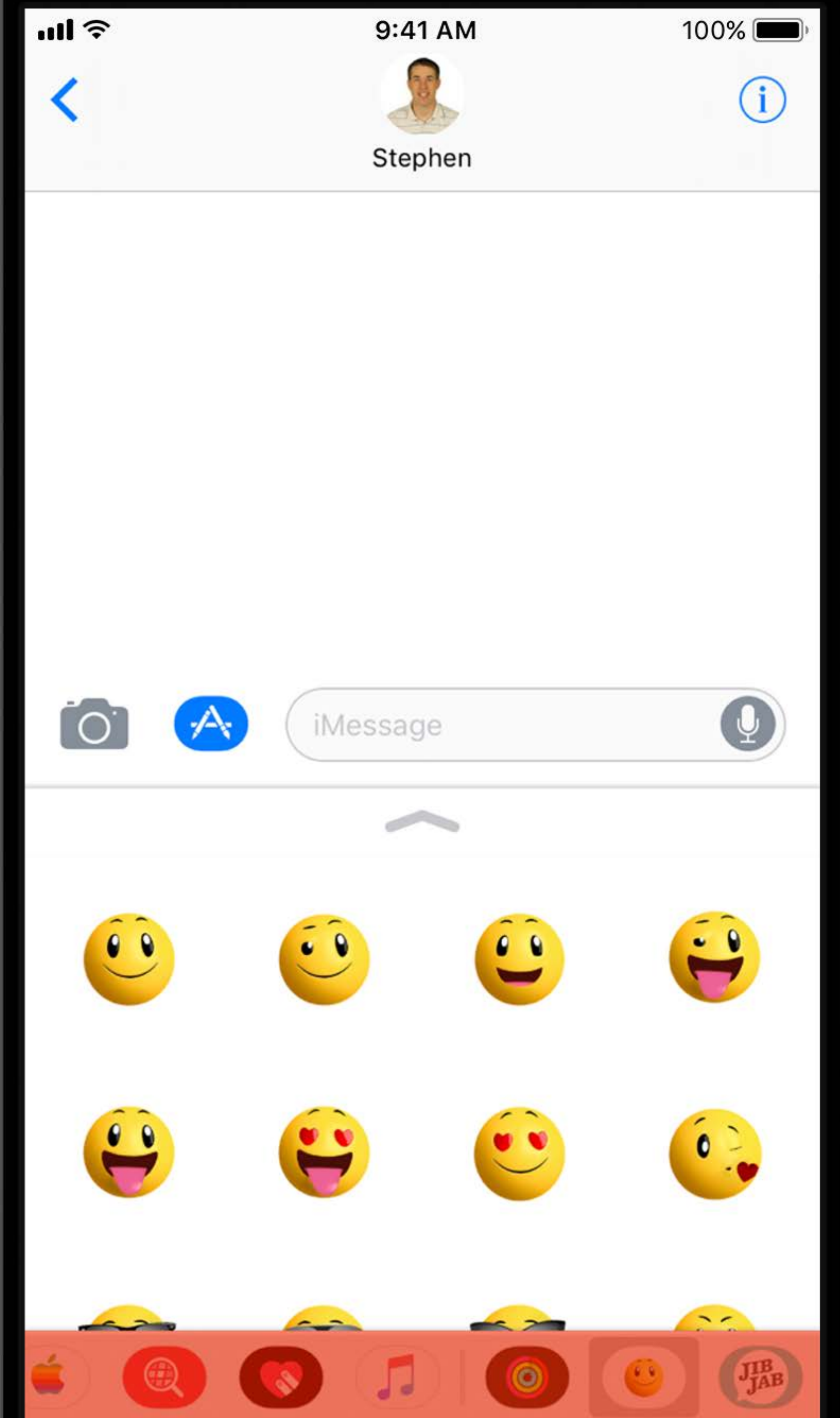
iMessage App Area Insets

Updated insets for Messages app area

Insets can be determined via

- [UIView safeAreaInsets]
- UIViewController top/bottomLayoutGuide
- Autolayout

Legacy apps get old insets



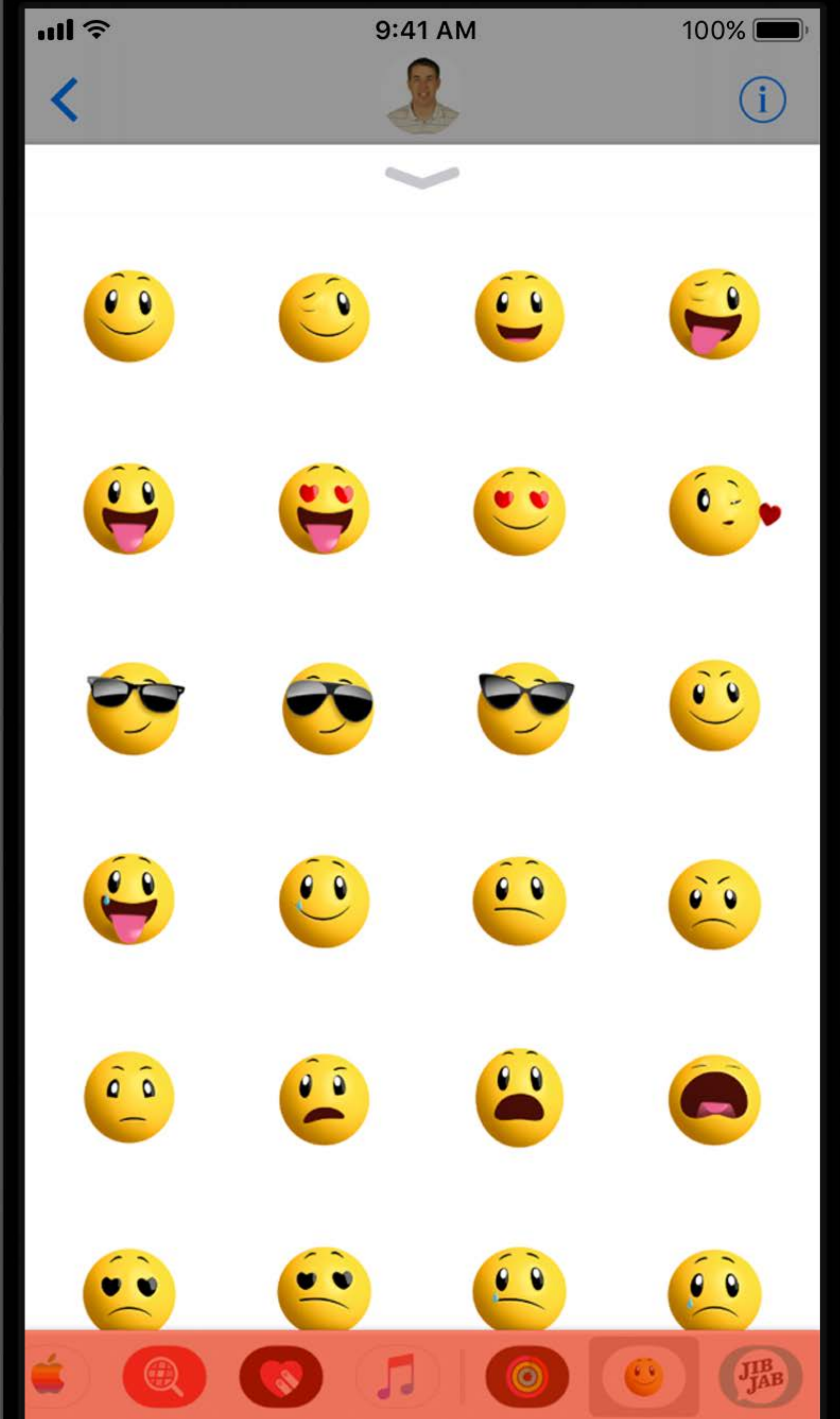
iMessage App Area Insets

Updated insets for Messages app area

Insets can be determined via

- [UIView safeAreaInsets]
- UIViewController top/bottomLayoutGuide
- Autolayout

Legacy apps get old insets



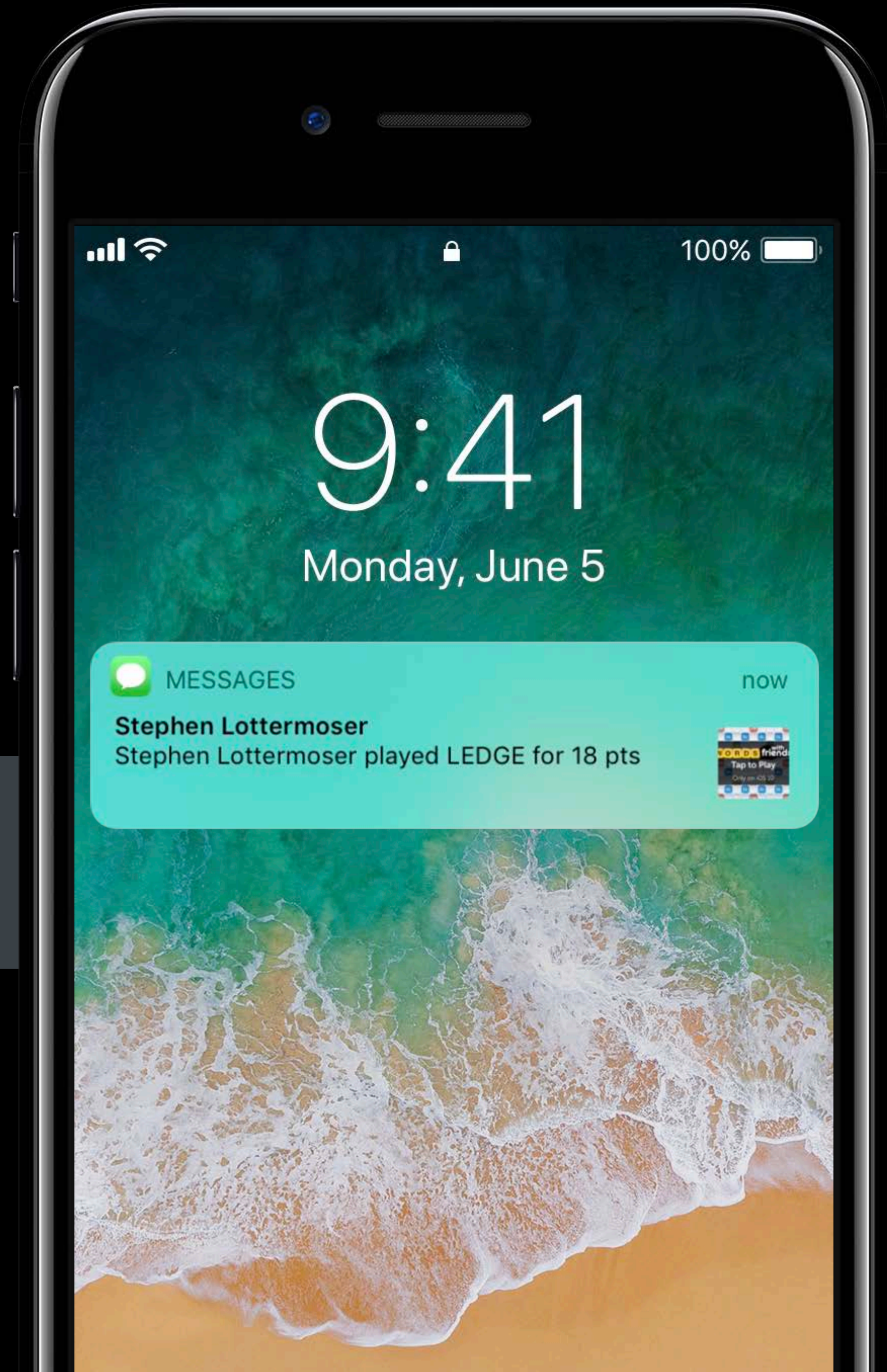
Message Summary Text

Succinct message summary

Provide summary for each message

Visible in notifications and conversation list

```
// MSMessage  
open var summaryText: String?
```



Stickers

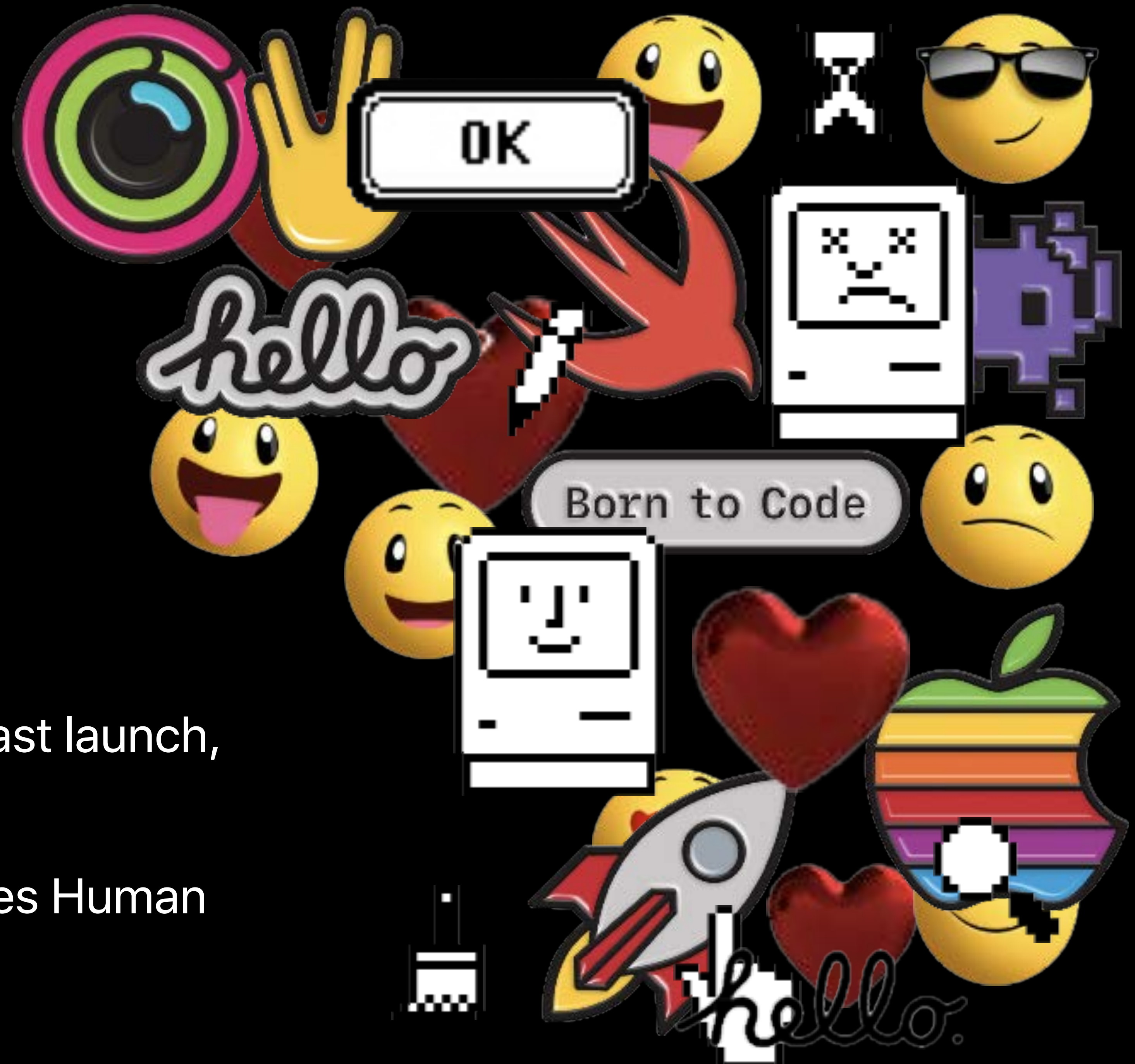
Preferred format: PNG / APNG

Optimizations

- Sticker pixel size
- Frame rate
- File size

Adhering to guidelines ensures fast launch, quick sends

For more info, reference Messages Human Interface Guidelines



MFMessageComposeViewController

MSMessage API

Match user experience between iMessage app and parent app

Template and Live layout support

Received messages can open in iMessage app or parent app

```
// MFMessageComposeViewController

/// This property sets the initial interactive message.
@available(iOS 10.0, *)
@NSCopying open var message: MSMessage?
```

Best Practices

Use layout margins

Set `summaryText` on all `MSMessages`

Consider sticker size, frame rate, for faster stickers

Send app balloons from `MFMessageComposeViewController`

Summary

What's New in Messages

Direct Send

Live Message Layouts

Best Practices

More Information

<https://developer.apple.com/wwdc17/234>

Related Sessions

[Introducing Business Chat](#)

Hall 3

Friday 10:00AM

[Design Studio Shorts 3](#)

Executive Ballroom

Friday 11:00AM

Labs

iMessage Apps and Business Chat Lab

Technology Lab B

Fri 11:00AM–2:00PM

