# Keyboard Input in iOS

Session 220

Justin Garcia

[redacted]

[redacted]

## $4 Billion

paid to iOS developers

100%

of chart topping Apps have 4+ stars

[redacted]

## Introduction

## Introduction

- Avoid common pitfalls

## Introduction

- Avoid common pitfalls
- New features in iOS 6

## What You Will Learn

## What You Will Learn

- Managing the keyboard

## What You Will Learn

- Managing the keyboard
- Managing static text

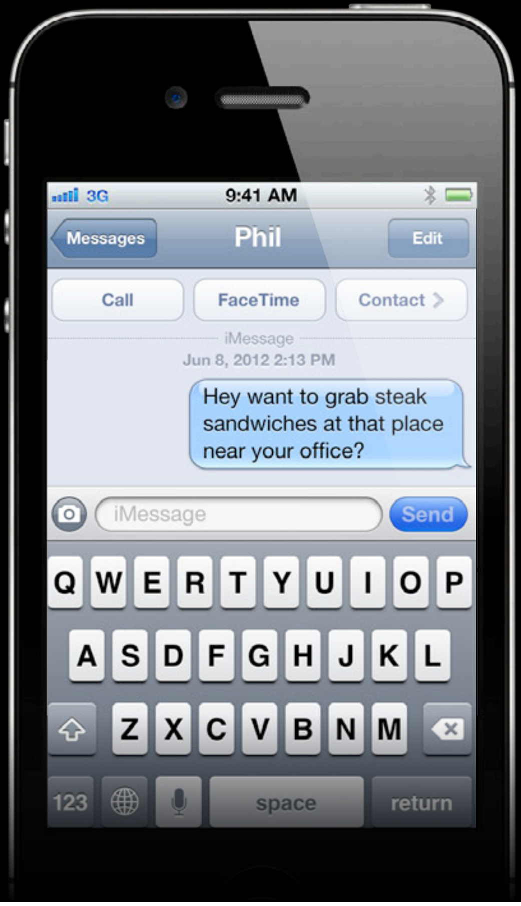
## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

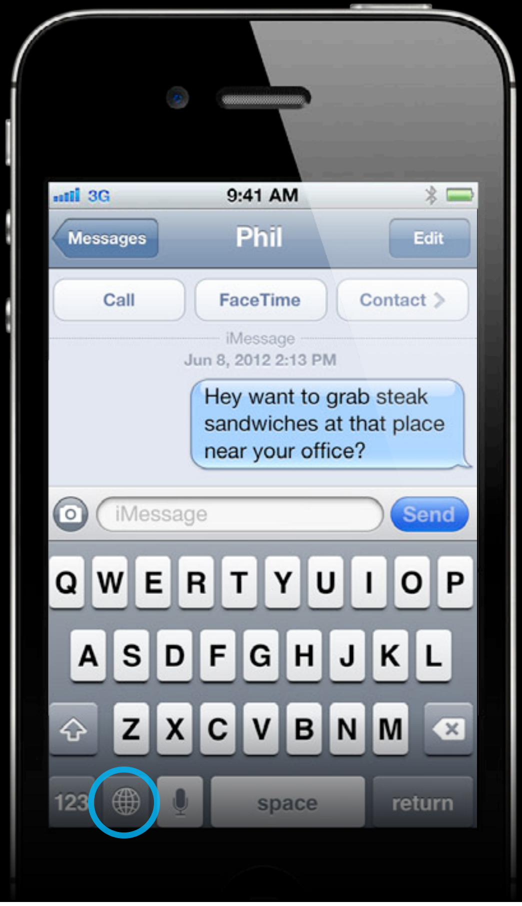
## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

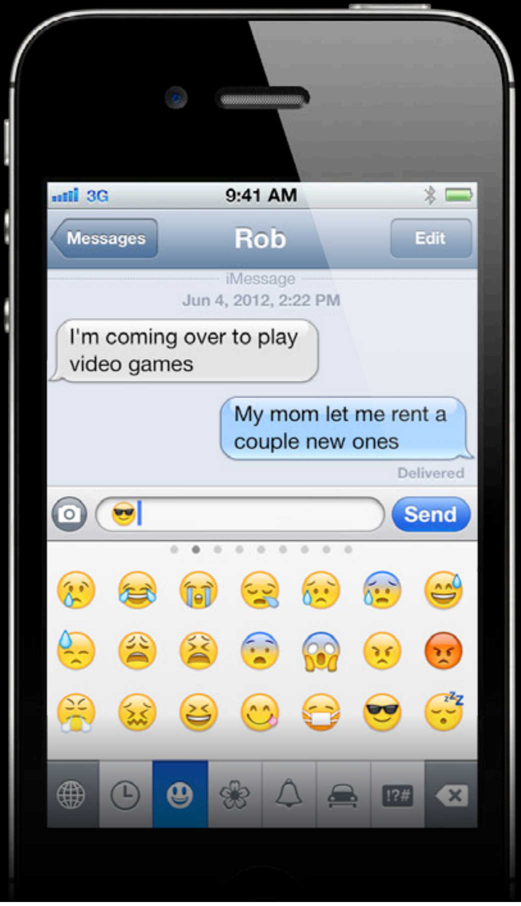
# Managing the Keyboard



## Managing the Keyboard

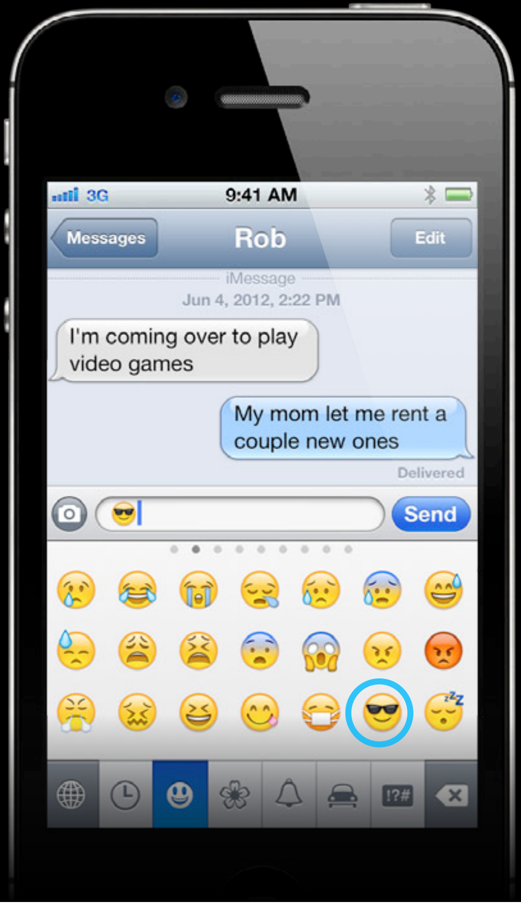


## Managing the Keyboard



iMessage
Jun 4, 2012, 2:22 PM
I'm coming over to play video games
My mom let me rent a couple new ones
Delivered

## Managing the Keyboard



I'm coming over to play video games

My mom let me rent a couple new ones

## Managing the Keyboard

## Managing the Keyboard

- Size and position changes

## Managing the Keyboard

- Size and position changes
- Attaching views

## Managing the Keyboard

- Size and position changes
- Attaching views

## Keyboard Size and Position Changes

## Keyboard Size and Position Changes

- Bring up

## Keyboard Size and Position Changes

- Bring up
- Candidate bar

## Keyboard Size and Position Changes

- Bring up
- Candidate bar
- Split and undocked

## Keyboard Size and Position Changes

- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

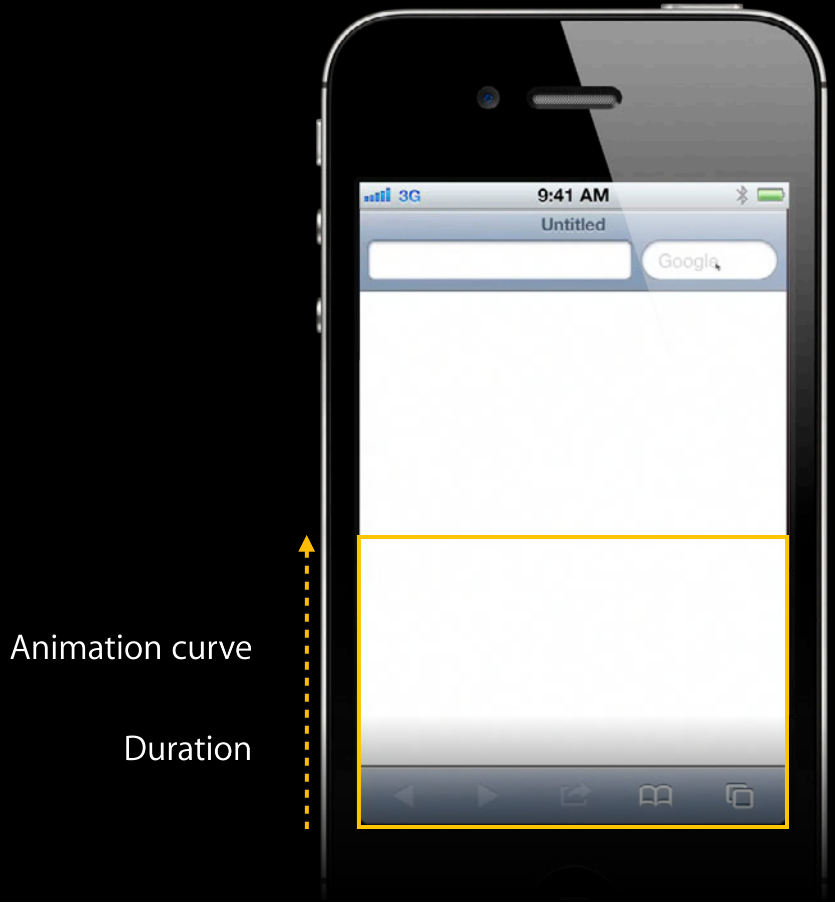
- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

### Responding to keyboard bring up

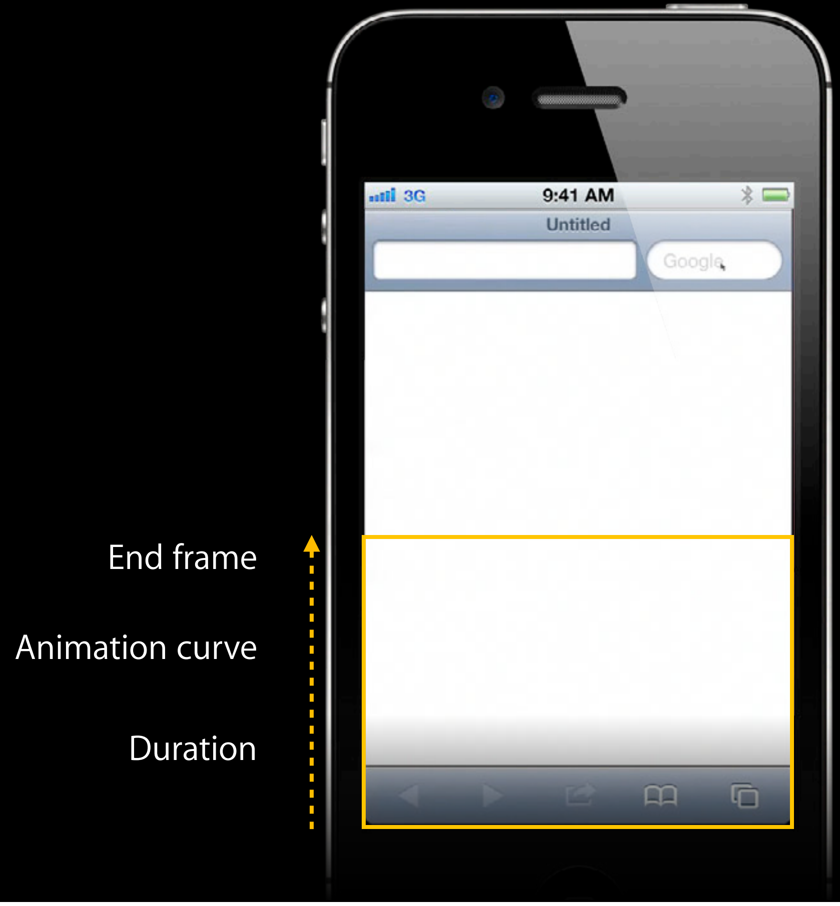
## Keyboard Size and Position Changes

### Responding to keyboard bring up



## Keyboard Size and Position Changes

### Responding to keyboard bring up



## Keyboard Size and Position Changes

### Responding to keyboard bring up

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
  // Note: rects are in screen coordinates.
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
  // Note: rects are in screen coordinates.
  UIAnimationCurve *curve =
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
  // Note: rects are in screen coordinates.
  UIAnimationCurve *curve =
   [[info objectForKey:UIKeyboardAnimationCurveInfoKey] intValue];
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
  // Note: rects are in screen coordinates.
  UIAnimationCurve *curve =
   [[info objectForKey:UIKeyboardAnimationCurveInfoKey] intValue];
  CGFloat duration =
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
  // Note: rects are in screen coordinates.
  UIAnimationCurve *curve =
   [[info objectForKey:UIKeyboardAnimationCurveInfoKey] intValue];
  CGFloat duration =
   [[info objectForKey:UIKeyboardAnimationDurationUserInfoKey] floatValue];
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardWillShowNotification fired:
- (void)keyboardWillShow:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];
  // Note: rects are in screen coordinates.
  UIAnimationCurve *curve =
   [[info objectForKey:UIKeyboardAnimationCurveInfoKey] intValue];
  CGFloat duration =
   [[info objectForKey:UIKeyboardAnimationDurationUserInfoKey] floatValue];


  // Kick off your animation…
}
```

## Keyboard Size and Position Changes

### Responding to keyboard bring up

```
// UIKeyboardDidShowNotification fired:
- (void)keyboardDidShow:(NSNotification *n) {
  // You could scroll to reveal the cursor here, like Notes does.
}
```

## Keyboard Size and Position Changes

### Responding to keyboard dismissal

```
// UIKeyboardWillHideNotification fired:
- (void)keyboardWillHide:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  // …etc. Use the same keys as -keyboardWillShow:.
}
```

## Keyboard Size and Position Changes

### Responding to keyboard dismissal

```
// UIKeyboardDidHideNotification fired:
- (void)keyboardDidHide:(NSNotification *n) {
  // Keyboard is hidden.
}
```

## Keyboard Size and Position Changes

- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

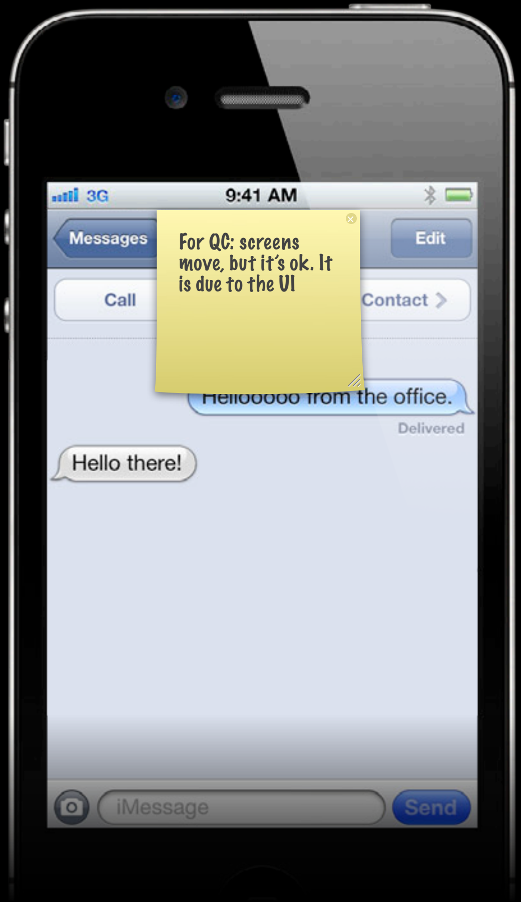
- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

"We experienced very strong iPhone sales growth in all of our segments, led by our Asia Pacific and Japan segments where sales more than doubled year-over-year."

Tim Cook

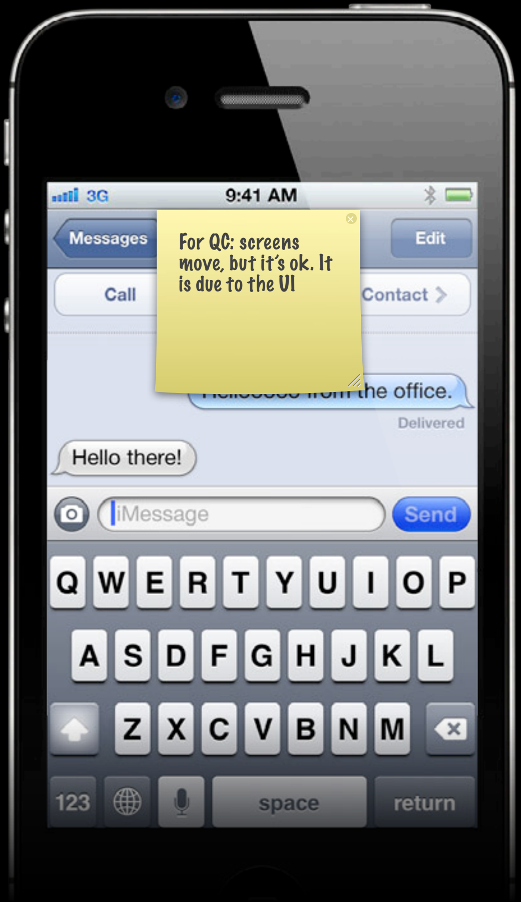
## Keyboard Size and Position Changes

### Candidate bar


3G
9:41 AM
Messages
Edit
Call
Contact
For QC: screens move, but it's ok. It is due to the UI
Helloooooo from the office.
Delivered
Hello there!
iMessage
Send

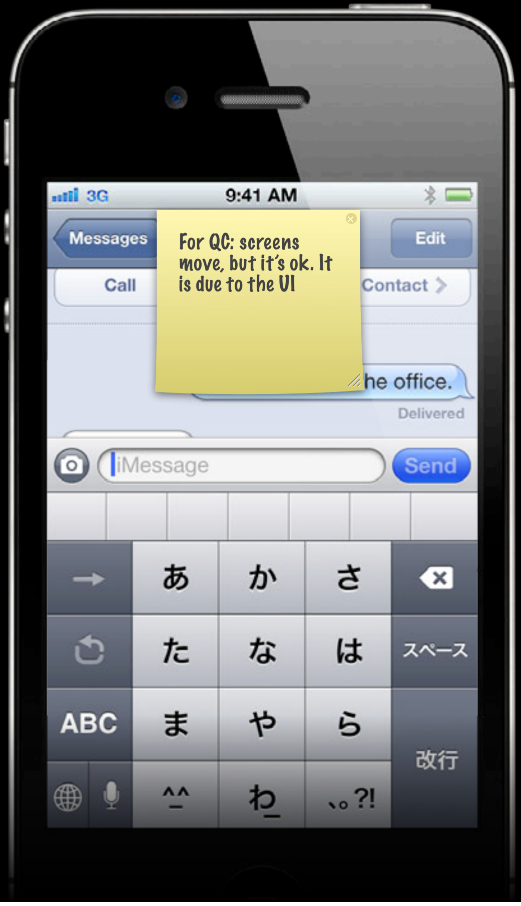
## Keyboard Size and Position Changes

### Candidate bar



## Keyboard Size and Position Changes

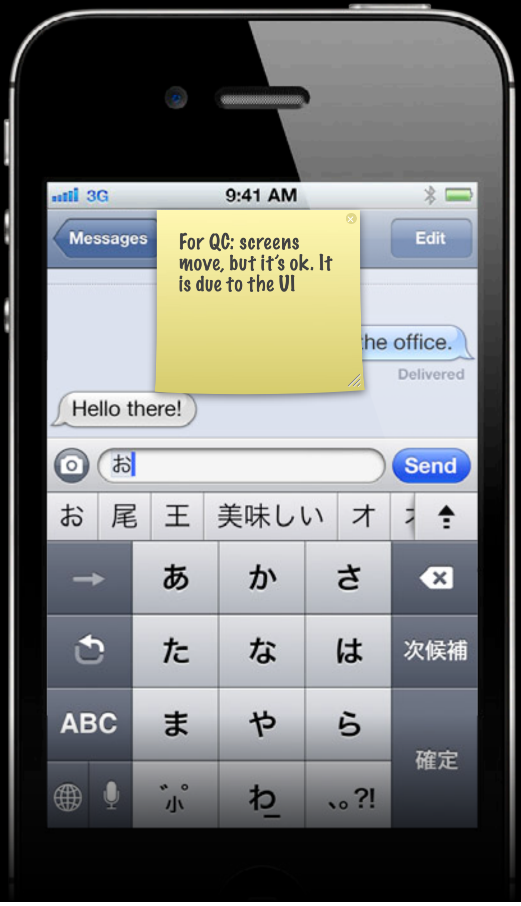
### Candidate bar



For QC: screens move, but it's ok. It is due to the UI

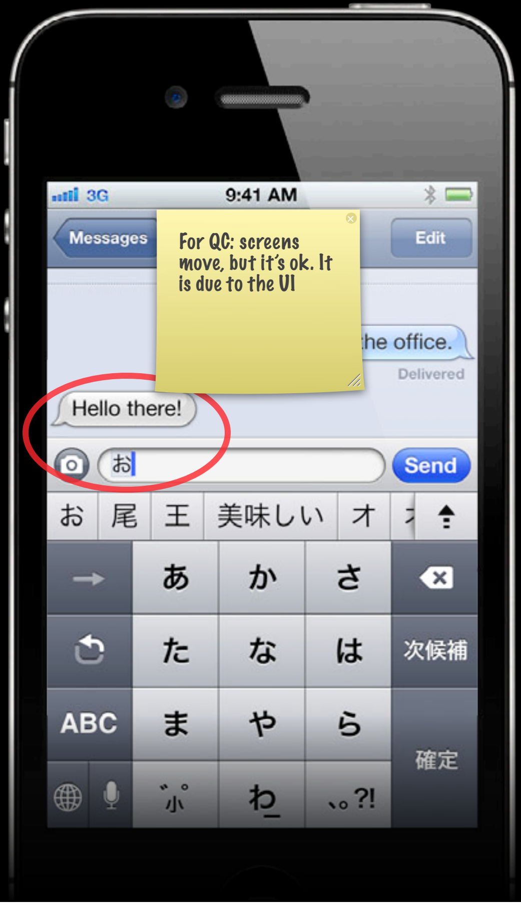
## Keyboard Size and Position Changes

### Candidate bar



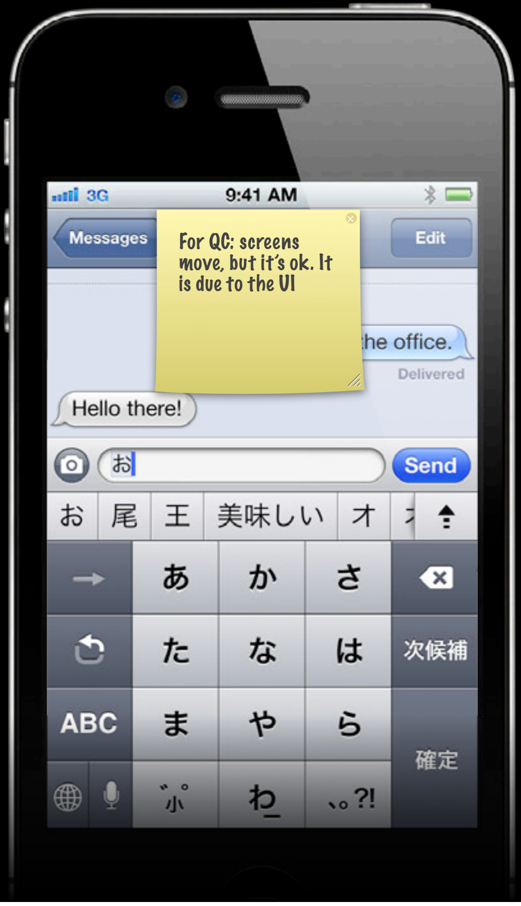
## Keyboard Size and Position Changes

### Candidate bar



## Keyboard Size and Position Changes

### Candidate bar



## Keyboard Size and Position Changes

### Candidate bar

```
// UIKeyboardDidChangeFrameNotification fired:
- (void)keyboardDidChangeFrame:(NSNotification *n) {
  NSDictionary *info = [n userInfo];
  CGRect beginFrame =
   [[info objectForKey:UIKeyboardFrameBeginUserInfoKey] CGRectValue];
  CGRect endFrame =
   [[info objectForKey:UIKeyboardFrameEndUserInfoKey] CGRectValue];

  // ... etc. Adjust your content.
}
```

## Keyboard Size and Position Changes

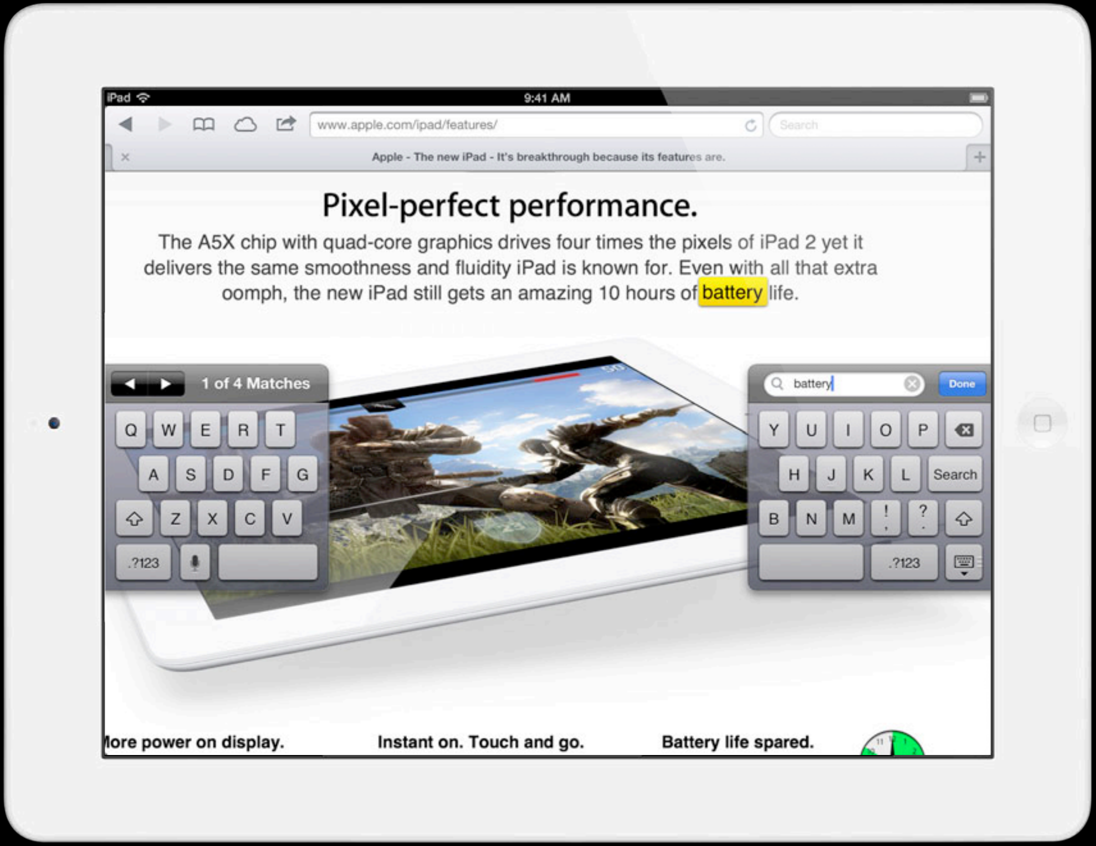
- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

### Split and undocked keyboard


iPad
9:41 AM
www.apple.com/ipad/features/
Search
Apple - The new iPad - It's breakthrough because its features are.
1 of 4 Matches
battery
Done
Search
.?123
More power on display.
Instant on. Touch and go.
Battery life spared.

Pixel-perfect performance.

The A5X chip with quad-core graphics drives four times the pixels of iPad 2 yet it delivers the same smoothness and fluidity iPad is known for. Even with all that extra oomph, the new iPad still gets an amazing 10 hours of battery life.

*Demo*

## Keyboard Size and Position Changes

### Split and undocked keyboard

```
// Fires on UIKeyboardWillHideNotification:
- (void)keyboardWillHide:(NSNotification *n) {
  // Prepare for the keyboard to be undocked, split or hidden.
}
```

## Keyboard Size and Position Changes

### Split and undocked keyboard

```
// Fires on UIKeyboardDidHideNotification:
- (void)keyboardDidHide:(NSNotification *n) {
  // Resize content.
}
```

## Keyboard Size and Position Changes

### Split and undocked keyboard

```
// Fires on UIKeyboardFrameWillChangeNotification:
- (void)keyboardFrameWillChange:(NSNotification *n) {
  // Don’t forget, split keyboards are shorter!
}
```

## Keyboard Size and Position Changes

- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

- Bring up
- Candidate bar
- Split and undocked
- Hardware keyboards

## Keyboard Size and Position Changes

### Hardware keyboards

## Keyboard Size and Position Changes

### Hardware keyboards

80 WPM

## Keyboard Size and Position Changes

### Hardware keyboards

```
// Same as responding to keyboard bring up and dismissal!
```

*Demo*

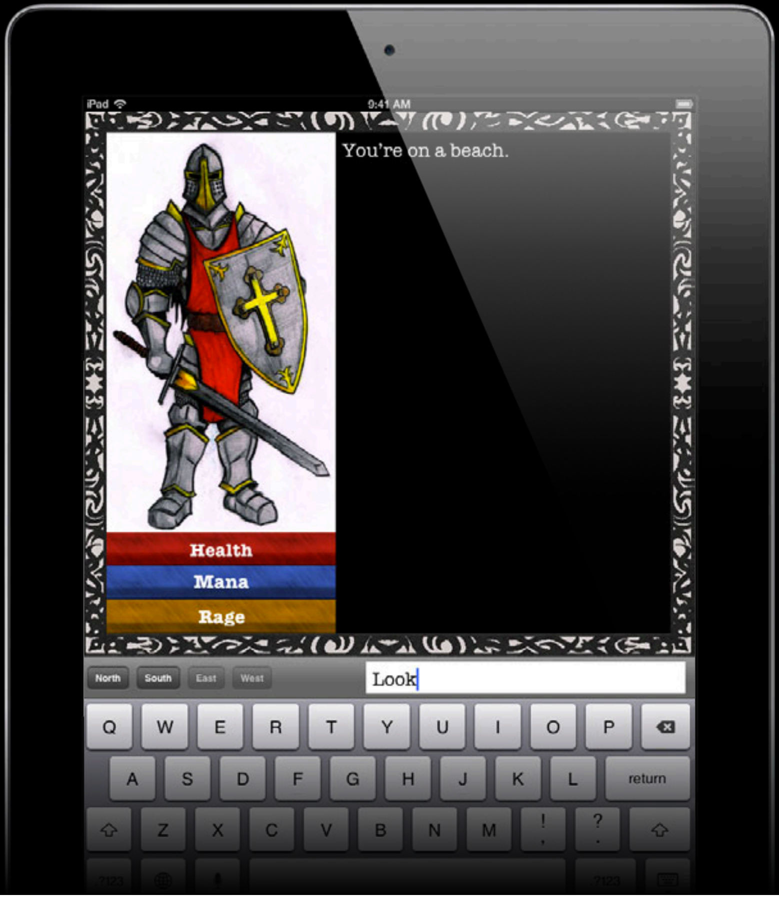
## Managing the Keyboard

- Keyboard size and position changes
- Attaching views

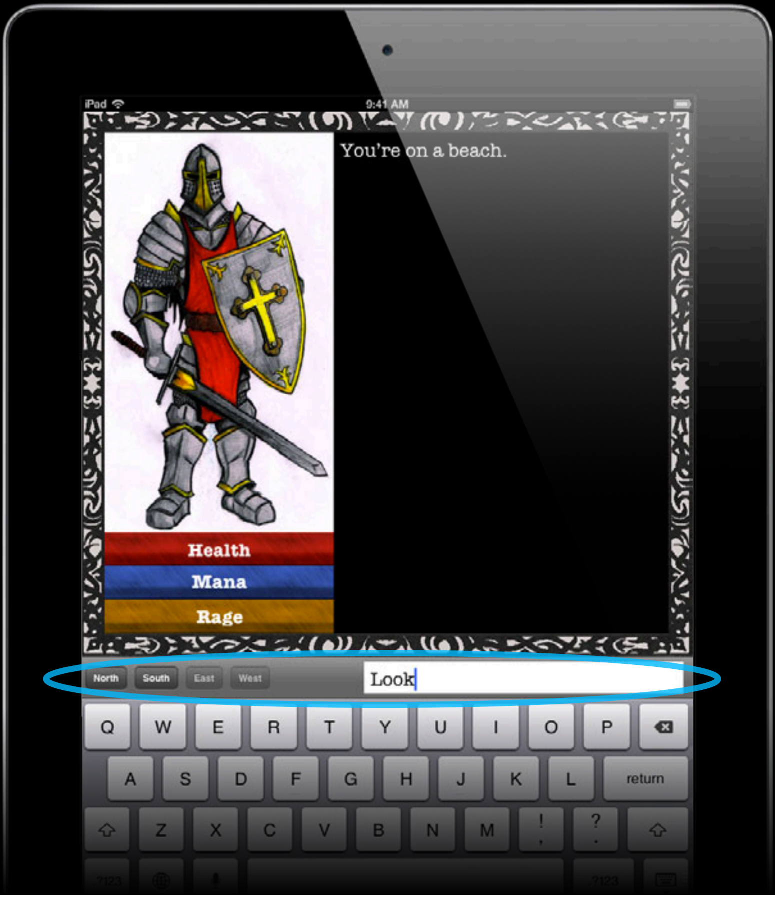
## Managing the Keyboard

- Keyboard size and position changes
- Attaching views

# Attaching Views to the Keyboard



## Attaching Views to the Keyboard



## Keyboard Size and Position Changes

### inputAccessoryView

```
@property (readwrite, retain) UIView *inputAccessoryView;
```

## *Demo*

## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

## Managing Static Text

## Managing Static Text

- Unicode essentials

## Managing Static Text

- Unicode essentials
- System selection in custom text views

## Managing Static Text

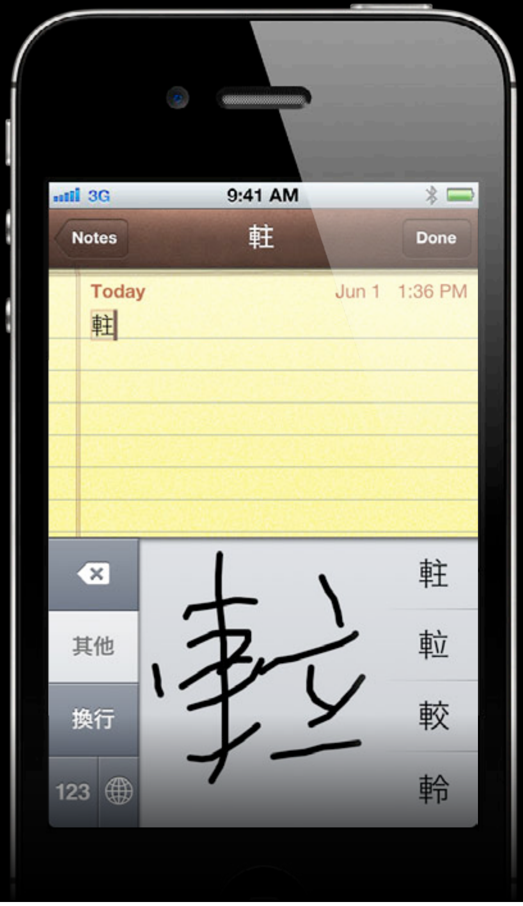
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## Managing Static Text

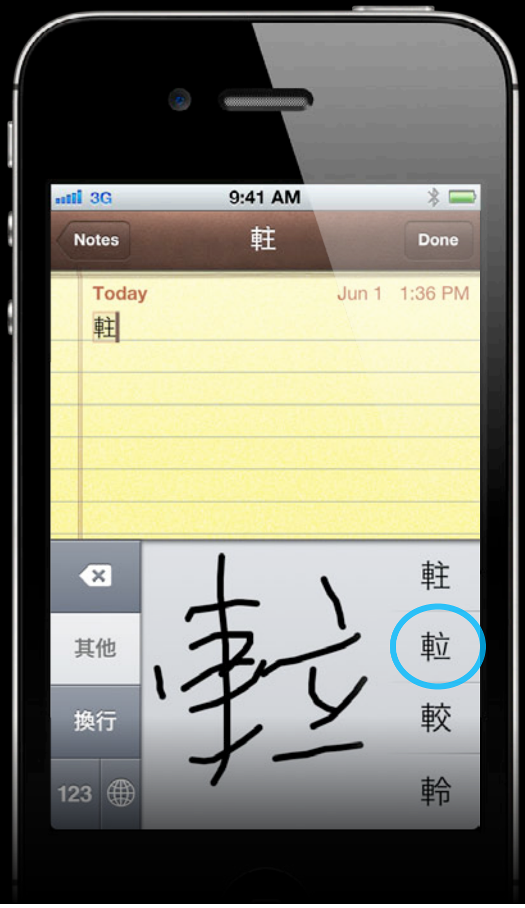
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## Unicode Essentials

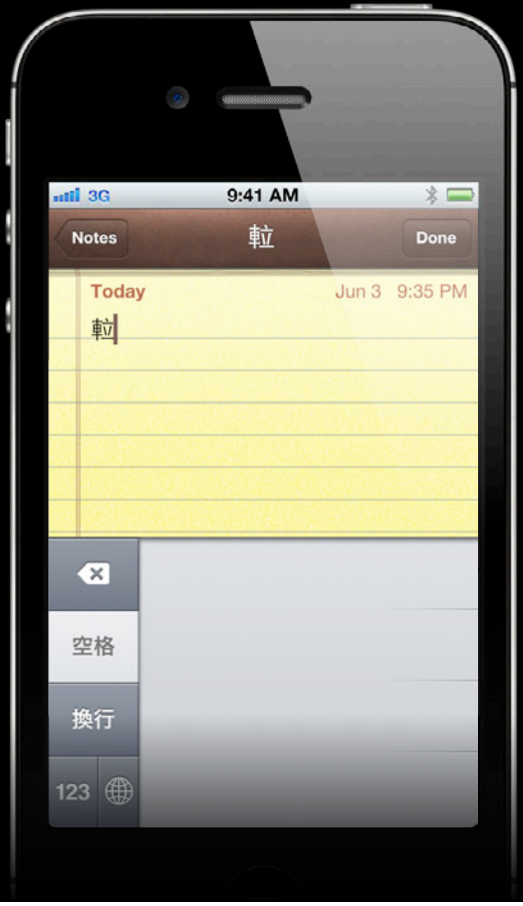
## Unicode Essentials



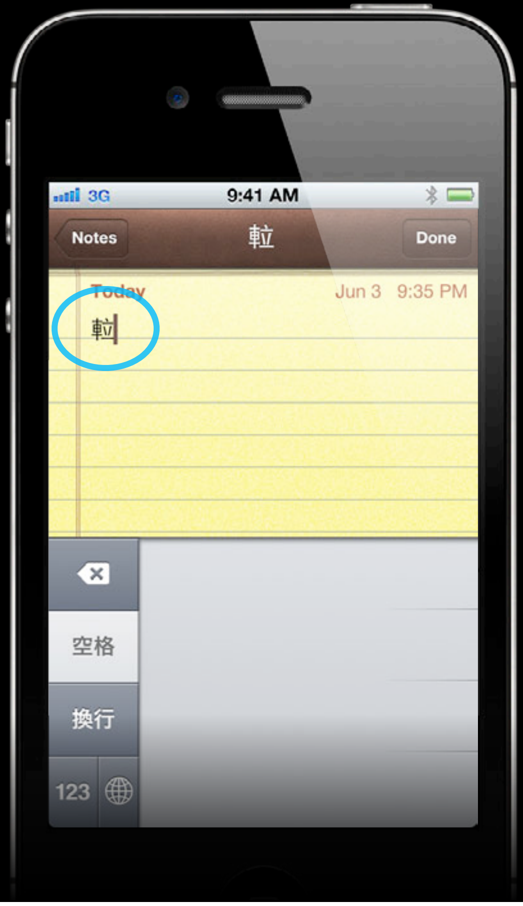
## Unicode Essentials



# Unicode Essentials

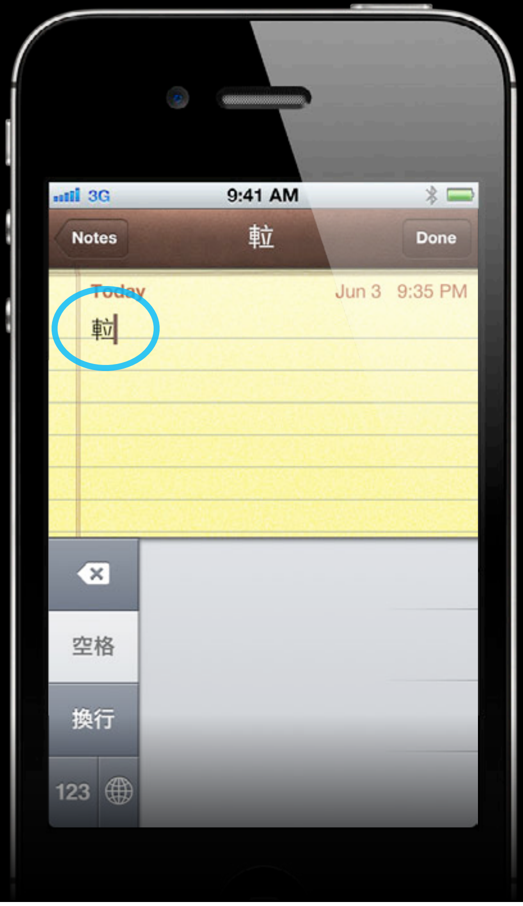


## Unicode Essentials



## Unicode Essentials

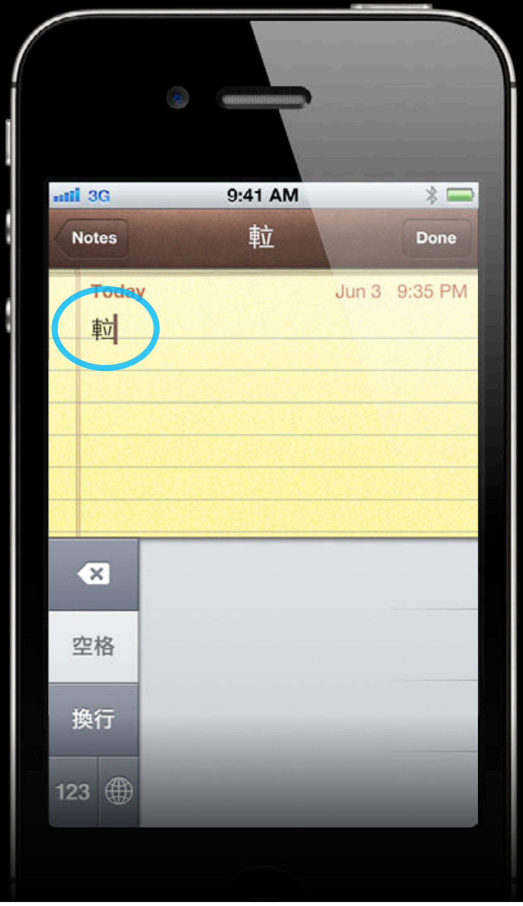
```
[[UITextView text]
characterAtIndex:0]?
```



## Unicode Essentials

```
[[UITextView text]
characterAtIndex:0]?
```

�



## Unicode Essentials

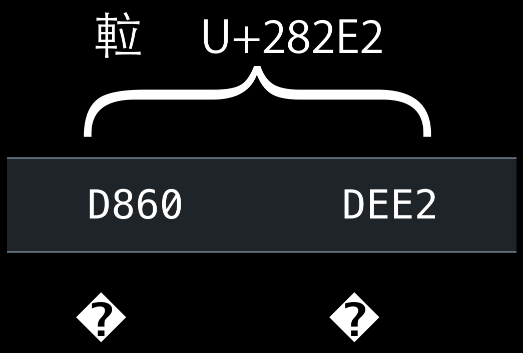
### Composed character sequence

This character occupies four bytes in memory

## Unicode Essentials

### Composed character sequence

This character occupies four bytes in memory



Memory representation

Interpretation as unichars

## Unicode Essentials

### Composed character sequence

## Unicode Essentials

### Composed character sequence

```
- (NSString *)firstCharacter {
```

## Unicode Essentials

### Composed character sequence

```
- (NSString *)firstCharacter {
    NSString *text = [myTextView text];
```

## Unicode Essentials

### Composed character sequence

```
- (NSString *)firstCharacter {
    NSString *text = [myTextView text];
    NSRange range = [text rangeOfComposedCharacterSequenceAtIndex:0];
```

## Unicode Essentials

### Composed character sequence

```
- (NSString *)firstCharacter {
    NSString *text = [myTextView text];
    NSRange range = [text rangeOfComposedCharacterSequenceAtIndex:0];
    NSString *sequence = [text substringWithRange:range];
```

## Unicode Essentials

### Composed character sequence

```
- (NSString *)firstCharacter {
    NSString *text = [myTextView text];
    NSRange range = [text rangeOfComposedCharacterSequenceAtIndex:0];
    NSString *sequence = [text substringWithRange:range];
    return sequence;
```

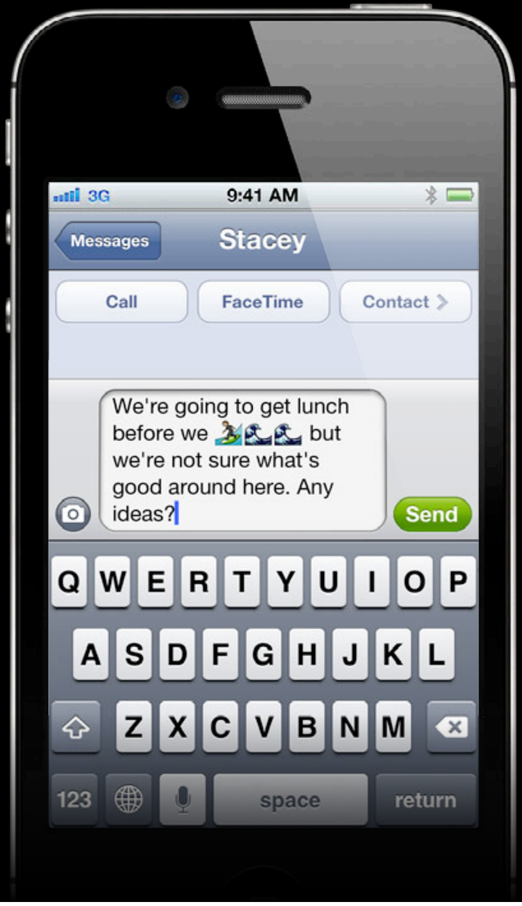
## Unicode Essentials

### Composed character sequence

```
- (NSString *)firstCharacter {
    NSString *text = [myTextView text];
    NSRange range = [text rangeOfComposedCharacterSequenceAtIndex:0];
    NSString *sequence = [text substringWithRange:range];
    return sequence;
}
```

## Unicode Essentials

### Composed character sequence



We're going to get lunch before we 🏄🌊🌊 but we're not sure what's good around here. Any ideas?

## Unicode Essentials

### Composed character sequence

## Unicode Essentials

### Composed character sequence

```
- (void)enumerateCharacterSequences {
```

## Unicode Essentials

### Composed character sequence

```
- (void)enumerateCharacterSequences {
    NSString *text = [myTextView text];
    NSRange fullRange = NSMakeRange(0, text.length);
    [text enumerateSubstringsInRange:fullRange
          options:NSStringEnumerationByComposedCharacterSequences
          usingBlock:
```

## Unicode Essentials

### Composed character sequence

```
- (void)enumerateCharacterSequences {
    NSString *text = [myTextView text];
    NSRange fullRange = NSMakeRange(0, text.length);
    [text enumerateSubstringsInRange:fullRange
          options:NSStringEnumerationByComposedCharacterSequences
          usingBlock:^(NSString *substring,
                     NSRange substringRange,
                     NSRange enclosingRange,
                     BOOL *stop) {
                       // Do something with each character sequence.
    }];
}
```

## Unicode Essentials

## Unicode Essentials

- Character ≠ unichar

## Unicode Essentials

- Character ≠ unichar
- Think composed character sequences

## Unicode Essentials

- Character ≠ unichar
- Think composed character sequences
- See Text and Linguistic Analysis

## Unicode Essentials

- Character ≠ unichar
- Think composed character sequences
- See Text and Linguistic Analysis
- See Internationalization Tips and Tricks

## Managing Static Text

## Managing Static Text

- Unicode essentials

## Managing Static Text

- Unicode essentials
- System selection in custom text views

## Managing Static Text

- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## Managing Static Text

- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

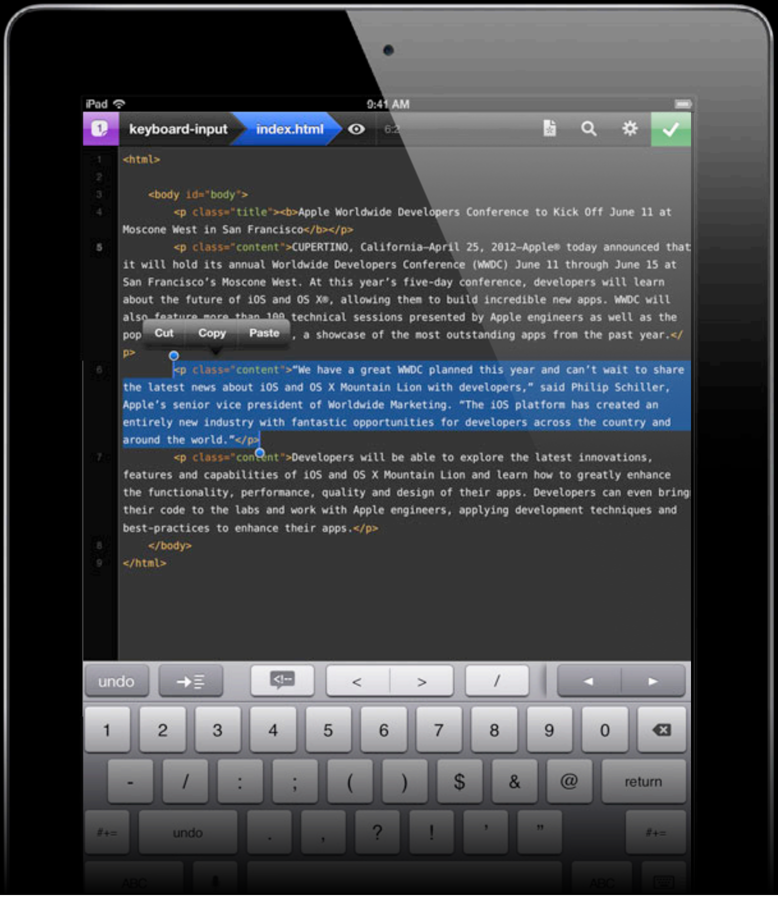
## Managing Static Text

New

- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## System Selection in Custom Views

### Why use a custom text view?



```
<html>

    <body id="body">
        <p class="title"><b>Apple Worldwide Developers Conference to Kick Off June 11 at
Moscone West in San Francisco</b></p>
        <p class="content">CUPERTINO, California—April 25, 2012—Apple® today announced that
it will hold its annual Worldwide Developers Conference (WWDC) June 11 through June 15 at
San Francisco's Moscone West. At this year's five-day conference, developers will learn
about the future of iOS and OS X®, allowing them to build incredible new apps. WWDC will
also feature more than 100 technical sessions presented by Apple engineers as well as the
pop                 , a showcase of the most outstanding apps from the past year.</
p>
        <p class="content">"We have a great WWDC planned this year and can't wait to share
the latest news about iOS and OS X Mountain Lion with developers," said Philip Schiller,
Apple's senior vice president of Worldwide Marketing. "The iOS platform has created an
entirely new industry with fantastic opportunities for developers across the country and
around the world."</p>
        <p class="content">Developers will be able to explore the latest innovations,
features and capabilities of iOS and OS X Mountain Lion and learn how to greatly enhance
the functionality, performance, quality and design of their apps. Developers can even bring
their code to the labs and work with Apple engineers, applying development techniques and
best-practices to enhance their apps.</p>
    </body>
</html>
```

## System Selection in Custom Views

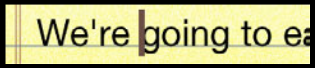
### What you need to implement

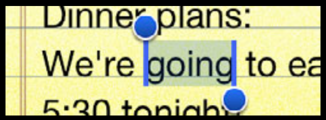
- UITextInput protocol
  - Turns a UIResponder into a editable text view
- UITextInputTokenizer subclass (optional)
  - Specifies how to break characters, words, sentences, paragraphs, etc.

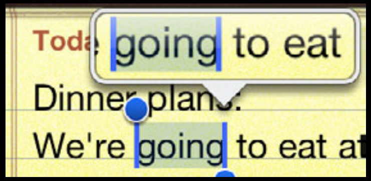
## System Selection in Custom Views

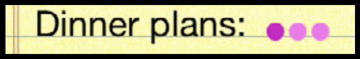
### What you get with system selection

- Insertion caret
- Selection tint and paddles
- Adjustment UI
- Selection and editing gestures
- Selection magnifiers
- Dictation placeholder









## System Selection in Custom Views

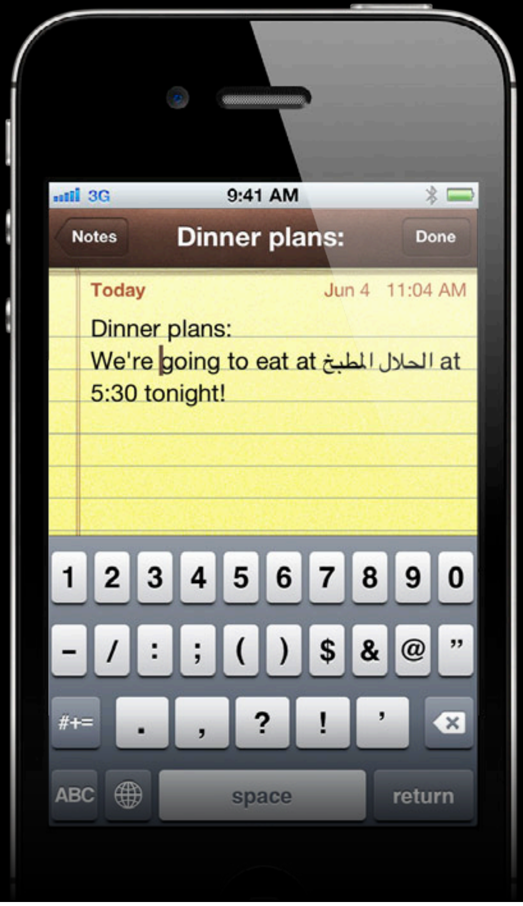
### What else you need to implement

- But subclass UITextView, rather than UIResponder/UIView
  - Implement UITextInput using your own text storage
- Also, -selectionRectsForRange:
  - UITextSelectionRect objects
- Don’t forget UITextInputDelegate methods!

## System Selection in Custom Views

### UITextPosition

- Encapsulated object of a cursor location
- No required methods
  - No assumptions about what it represents
- Doesn't have to be unique

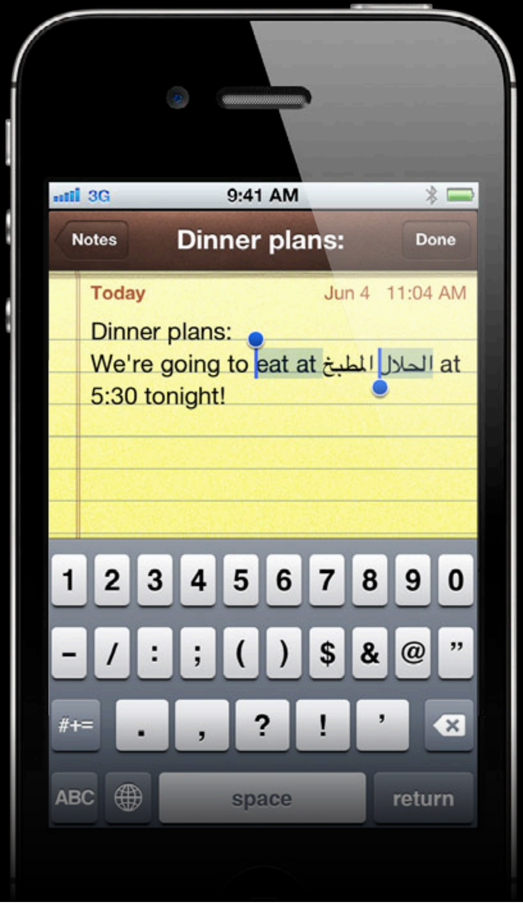


Dinner plans:
We're going to eat at المطبخ الحلال at 5:30 tonight!

# System Selection in Custom Views

## UITextRange

- Two UITextPositions
- May contain multiple text writing directions
- Must be contiguous in the document
- An "empty" range is a caret

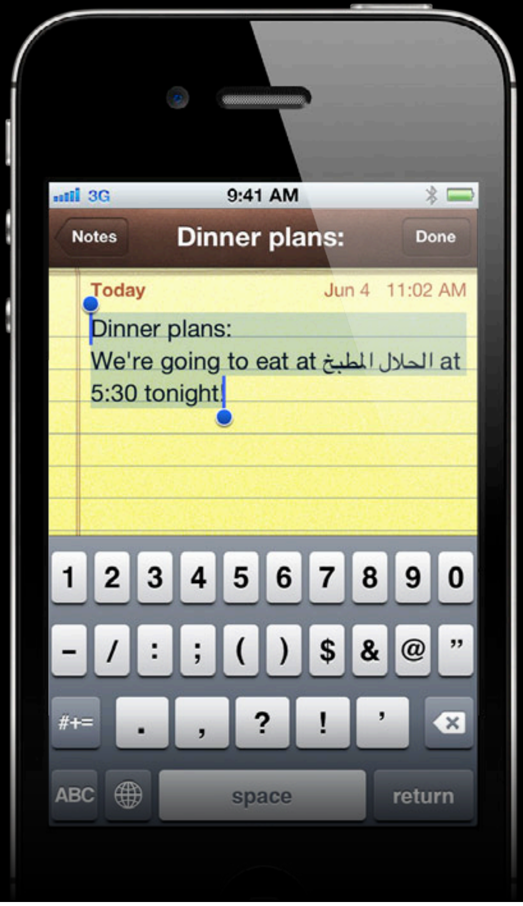


Dinner plans:
We're going to eat at المطبخ الحلال at 5:30 tonight!

## System Selection in Custom Views

### UITextSelectionRect

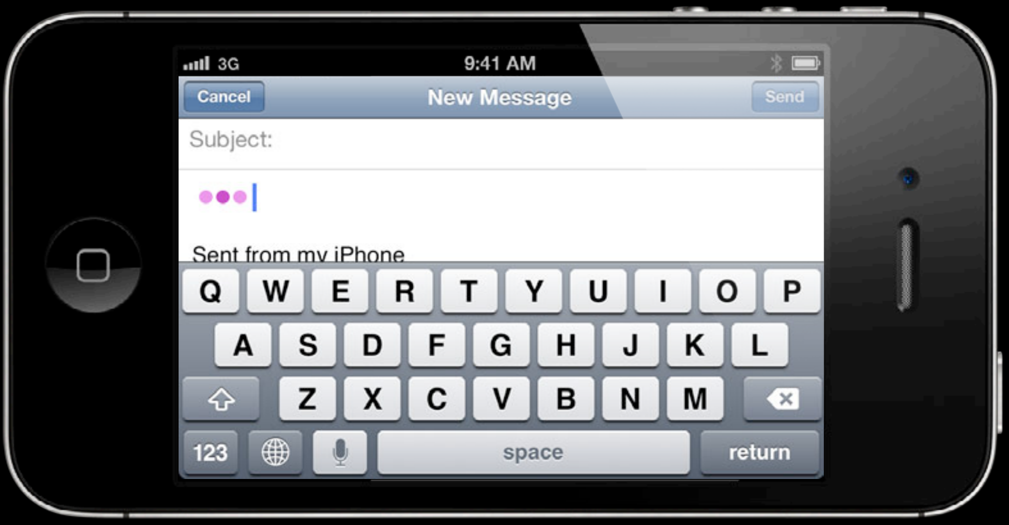
- Reflects the on-screen area of a UITextRange
- containsStart and containsEnd
  - Determine selection paddle display
  - Return YES once and only once for each!
- Beware of line-wrapping
- Beware of overlapping rects
- The fewer the better



Dinner plans:
We're going to eat at الحلال المطبخ at
5:30 tonight!

## System Selection in Custom Views

### Dictation thinking dots



## System Selection in Custom Views

### Get dictation thinking dots in your custom text view

- You specify the size and position
- The system takes care of the rest

```
- (id)insertDictationResultPlaceholder;
- (CGRect)frameForDictationResultPlaceholder:(id)placeholder;
- (void)removeDictationResultPlaceholder:(id)placeholder willInsertResult:
(BOOL)willInsertResult;
```

## *Demo*

Morgan Winer

## Managing Static Text

New

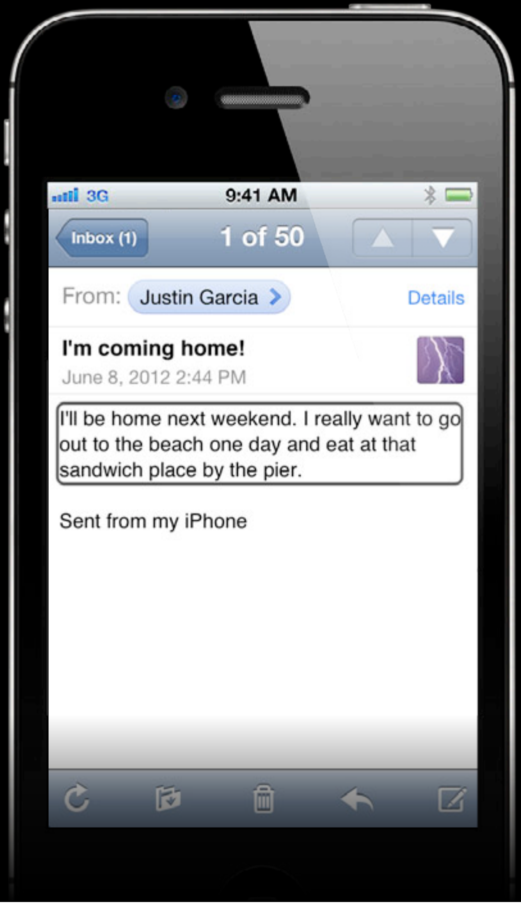
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## Managing Static Text

- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## UITextInput in Standard Text Views

### Paragraph at a point



I'm coming home!

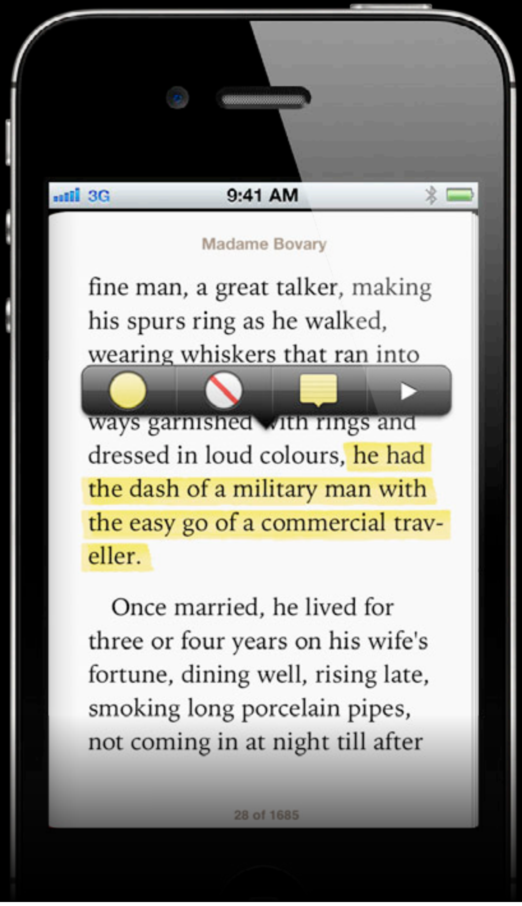
June 8, 2012 2:44 PM

I'll be home next weekend. I really want to go out to the beach one day and eat at that sandwich place by the pier.

Sent from my iPhone

## UITextInput in Standard Text Views

### Word at a point



fine man, a great talker, making his spurs ring as he walked, wearing whiskers that ran into ways garnished with rings and dressed in loud colours, he had the dash of a military man with the easy go of a commercial trav-eller.

Once married, he lived for three or four years on his wife's fortune, dining well, rising late, smoking long porcelain pipes, not coming in at night till after

## UITextInput

### Word at a point

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
    [_textView rangeEnclosingPosition:position
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
    [_textView rangeEnclosingPosition:position
                      withGranularity:UITextGranularityWord
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
    [_textView rangeEnclosingPosition:position
                      withGranularity:UITextGranularityWord
                          inDirection:UITextLayoutDirectionForward];
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
    [_textView rangeEnclosingPosition:position
                      withGranularity:UITextGranularityWord
                          inDirection:UITextLayoutDirectionForward];
  // UITextStorageDirectionForward, not UITextStorageDirectionRight!
```

## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
    [_textView rangeEnclosingPosition:position
                      withGranularity:UITextGranularityWord
                          inDirection:UITextLayoutDirectionForward];
  // UITextStorageDirectionForward, not UITextStorageDirectionRight!
  return range;
```

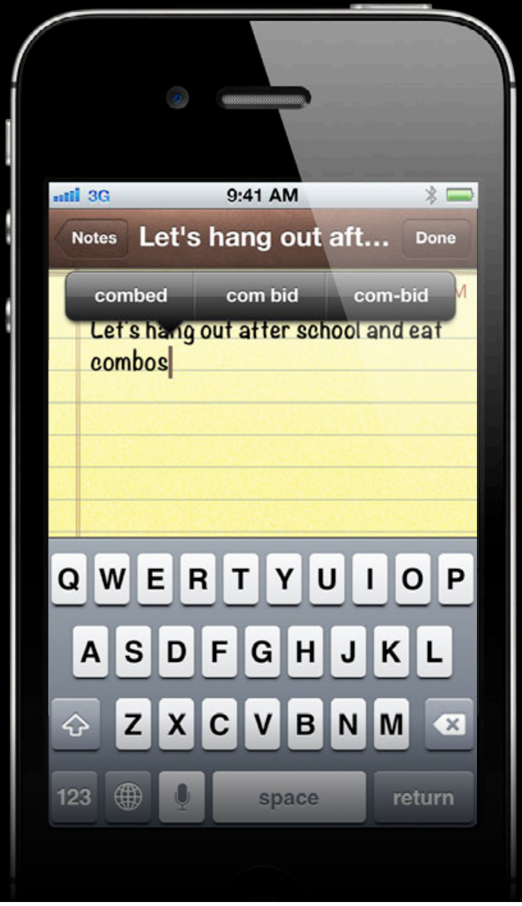
## UITextInput

### Word at a point

```
- (UITextRange *)rangeForWordAtPoint:(CGPoint p) {
  UITextPosition *position = [_textView closestPositionToPoint:p];
  UITextRange *range =
    [_textView rangeEnclosingPosition:position
                      withGranularity:UITextGranularityWord
                          inDirection:UITextLayoutDirectionForward];
  // UITextStorageDirectionForward, not UITextStorageDirectionRight!
  return range;
}
```

## UITextInput

### Iterate back one word



## UITextInput

### Iterate back one word

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
  UITextRange *wordRange =
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
  UITextRange *wordRange =
    [_textView textRangeFromPosition:position
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
  UITextRange *wordRange =
    [_textView textRangeFromPosition:position
                          toPosition:previousWordStart];
```

# UITextInput

## Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
  UITextRange *wordRange =
    [_textView textRangeFromPosition:position
                          toPosition:previousWordStart];
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
  UITextRange *wordRange =
    [_textView textRangeFromPosition:position
                          toPosition:previousWordStart];

  return [_textView textInRange:wordRange];
```

## UITextInput

### Iterate back one word

```
- (NSString *)previousWord {
  UITextRange *currentSelectedRange = [_textView selectedTextRange];
  UITextPosition *position = [currentSelectedRange start];
  UITextPosition *previousWordStart =
    [_textView positionFromPosition:position
                         toBoundary:UITextGranularityWord
                        inDirection:UITextStorageDirectionBackward];
  // UITextStorageDirectionBackward, not UITextStorageDirectionLeft!
  UITextRange *wordRange =
    [_textView textRangeFromPosition:position
                          toPosition:previousWordStart];

  return [_textView textInRange:wordRange];
}
```

## *Demo*

## Managing Static Text

- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

## Handling User Input

## Handling User Input

- UITextInputTraits

## Handling User Input

- UITextInputTraits
- Rich text editing

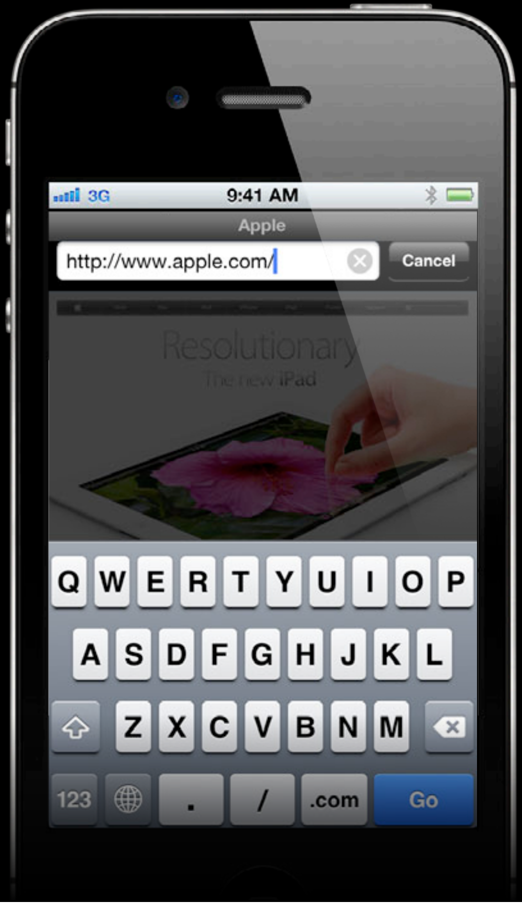
## Handling User Input

- UITextInputTraits
- Rich text editing
- Dictation API

## Handling User Input

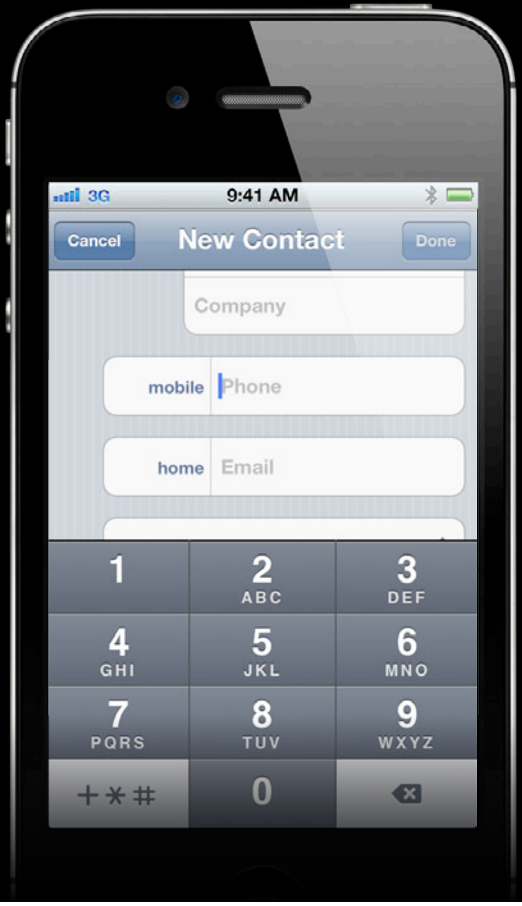
- UITextInputTraits
- Rich text editing
- Dictation API

## UITextInputTraits
### UIKeyboardType



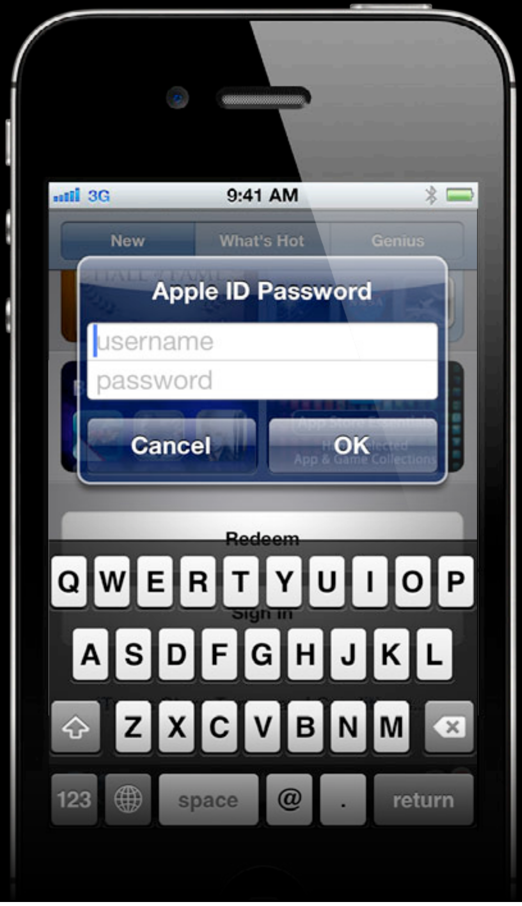
# UITextInputTraits

## UIKeyboardType

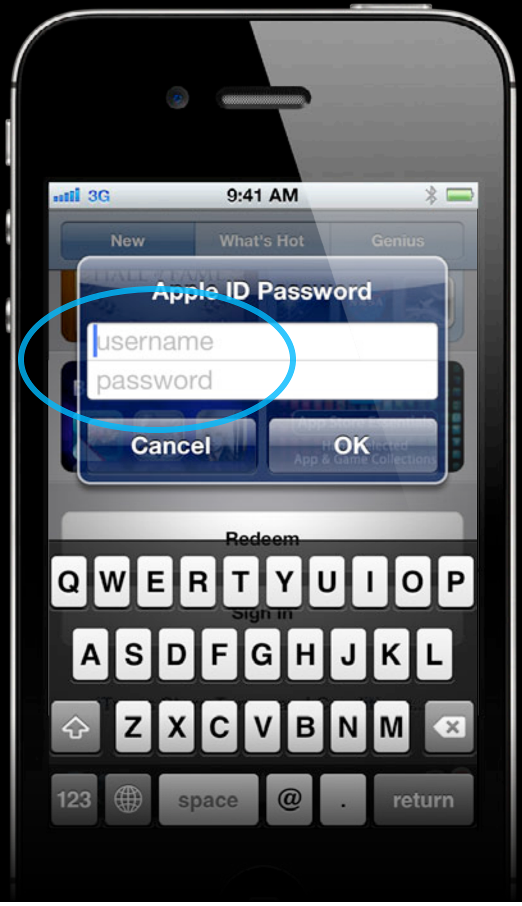


## UITextInputTraits

### UIKeyboardType

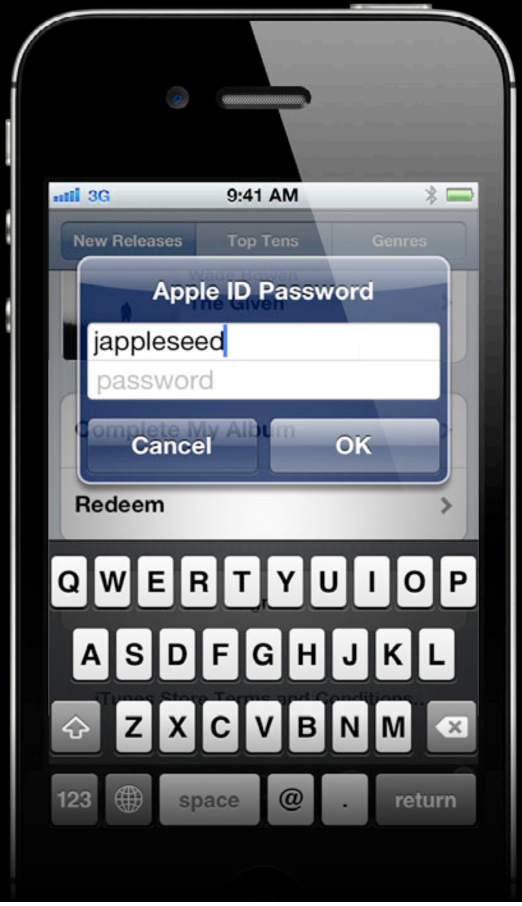


## UITextInputTraits
### UIKeyboardType



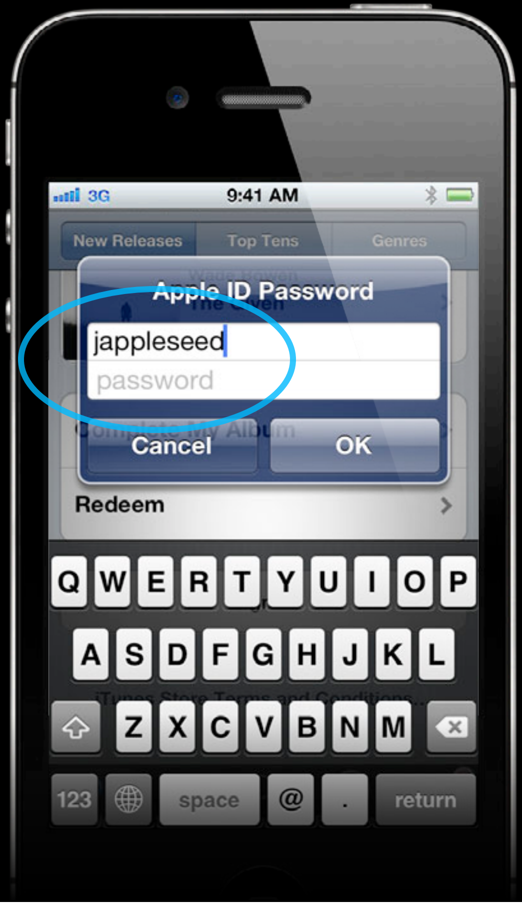
## UITextInputTraits

### UITextAutocorrectionType

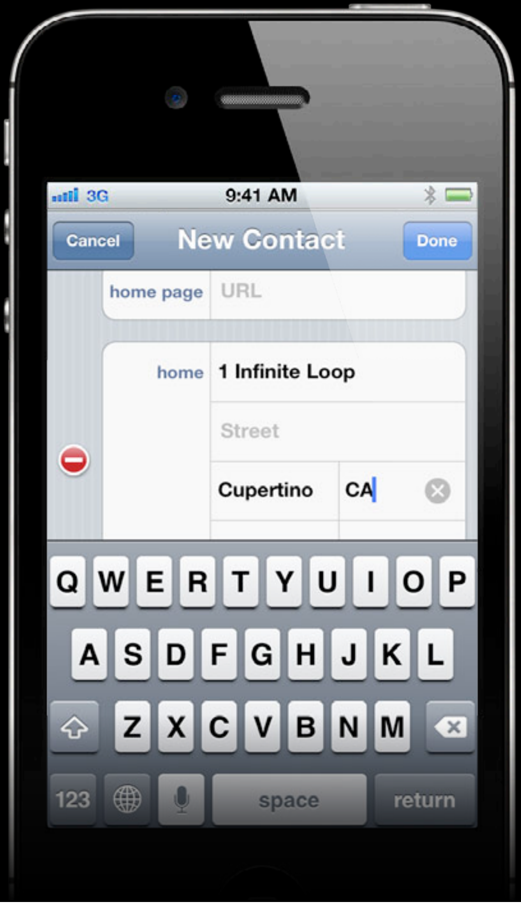


## UITextInputTraits

### UITextAutocorrectionType

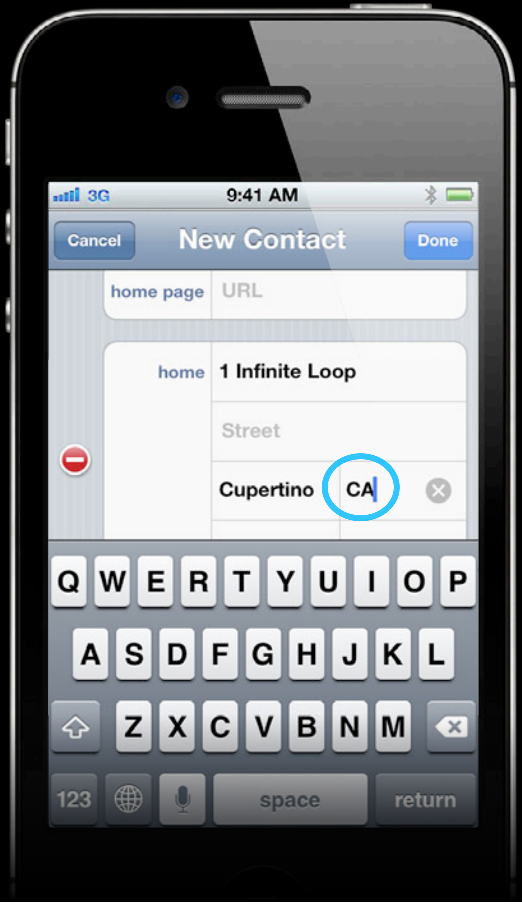


## UITextInputTraits
### UIAutocapitalizationType



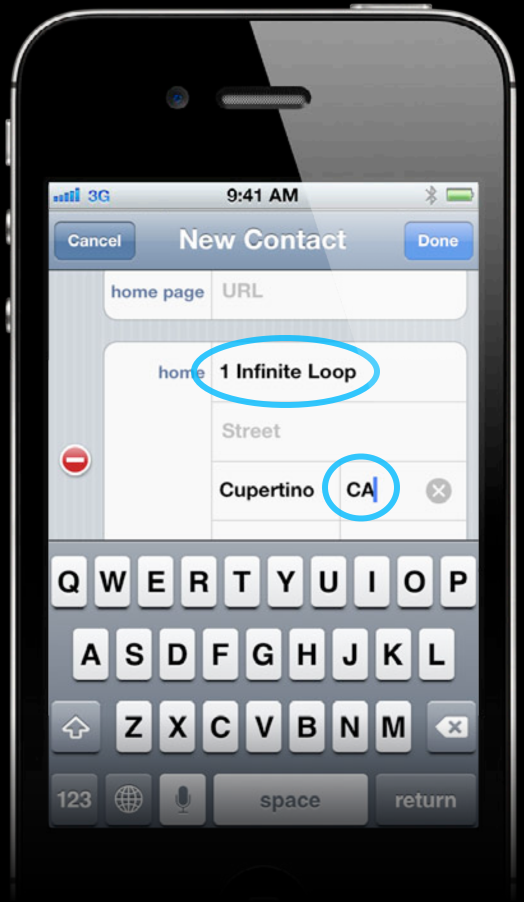
## UITextInputTraits

### UIAutocapitalizationType



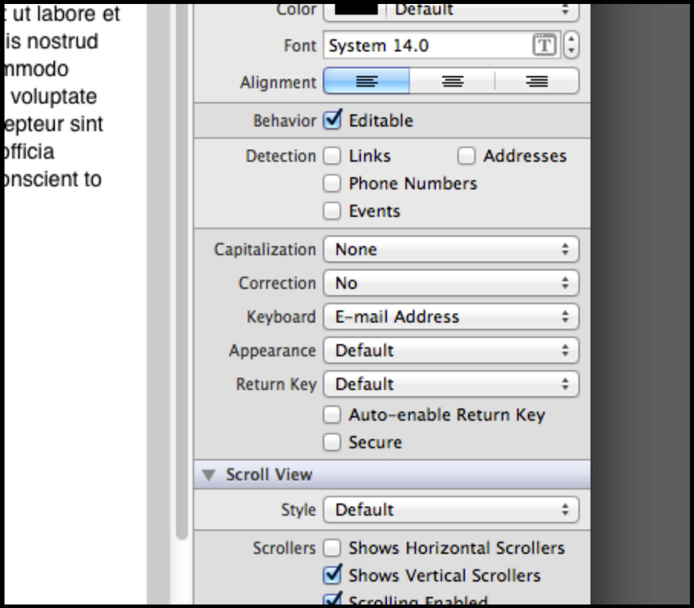
# UITextInputTraits

## UIAutocapitalizationType



## UITextInputTraits

### Choosing traits



t ut labore et
is nostrud
nmodo
voluptate
epteur sint
officia
onscient to

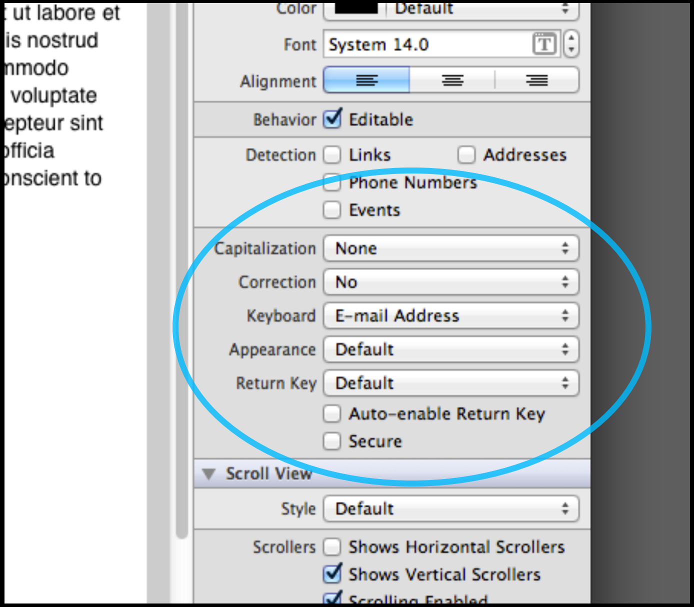
Color Default
Font System 14.0
Alignment
Behavior ☑ Editable
Detection ☐ Links ☐ Addresses
☐ Phone Numbers
☐ Events
Capitalization None
Correction No
Keyboard E-mail Address
Appearance Default
Return Key Default
☐ Auto-enable Return Key
☐ Secure

Scroll View

Style Default
Scrollers ☐ Shows Horizontal Scrollers
☑ Shows Vertical Scrollers
☑ Scrolling Enabled

## UITextInputTraits

### Choosing traits



Color Default

Font System 14.0

Alignment

Behavior ✓ Editable

Detection ☐ Links ☐ Addresses ☐ Phone Numbers ☐ Events

Capitalization None

Correction No

Keyboard E-mail Address

Appearance Default

Return Key Default

☐ Auto-enable Return Key

☐ Secure

Scroll View

Style Default

Scrollers ☐ Shows Horizontal Scrollers ✓ Shows Vertical Scrollers ✓ Scrolling Enabled

## Handling User Input

- UITextInputTraits
- Rich text editing
- Dictation API

## Handling User Input

- UITextInputTraits
- Rich text editing
- Dictation API

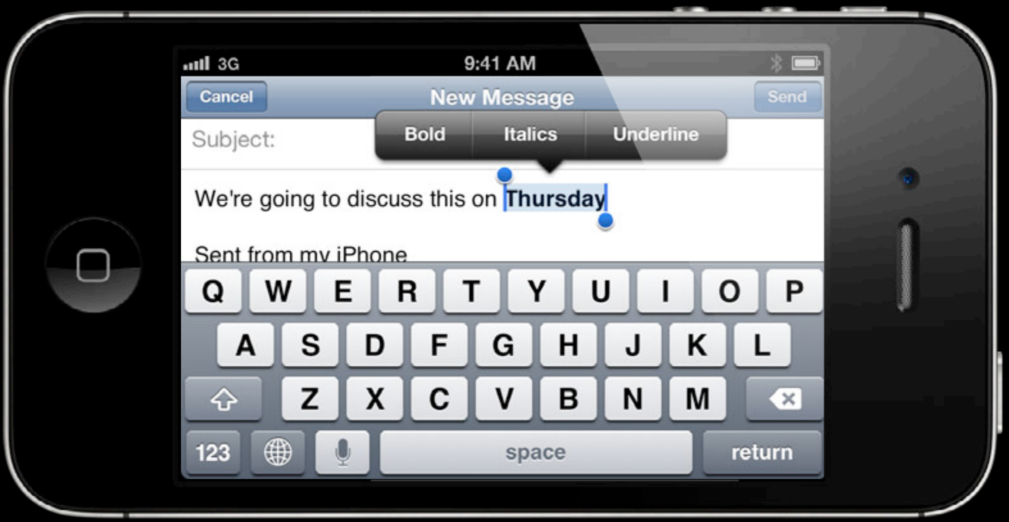
## Handling User Input

New

- UITextInputTraits
- Rich text editing
- Dictation API

## Rich Text Editing

New



We're going to discuss this on Thursday

Sent from my iPhone

## Rich Text Editing

## Rich Text Editing

- BIU controls

## Rich Text Editing

- BIU controls
- NSAttributedString

## Rich Text Editing

- BIU controls
- NSAttributedString
- Typing attributes

## Rich Text Editing

## Rich Text Editing

- BIU controls

## Rich Text Editing

- BIU controls
- NSAttributedString

## Rich Text Editing

- BIU controls
- NSAttributedString
- Typing attributes

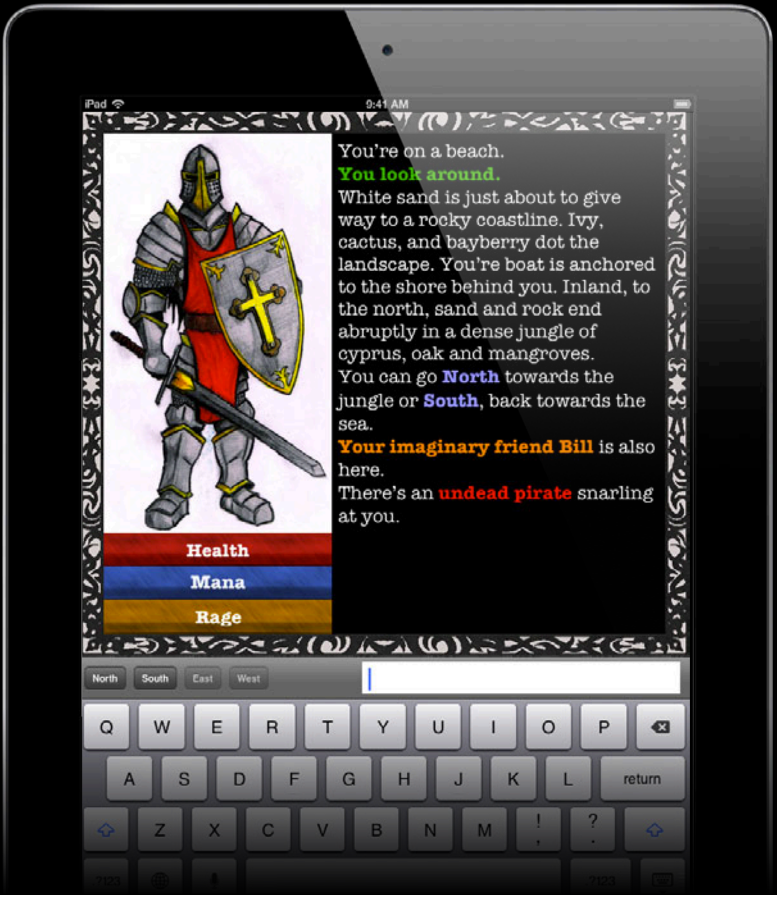
## Rich Text Editing

### Bold, Italic and Underline controls

```
@property(nonatomic) BOOL allowsEditingTextAttributes;
```

## Rich Text Editing

### NSAttributedString



You're on a beach.
You look around.
White sand is just about to give way to a rocky coastline. Ivy, cactus, and bayberry dot the landscape. You're boat is anchored to the shore behind you. Inland, to the north, sand and rock end abruptly in a dense jungle of cyprus, oak and mangroves.
You can go **North** towards the jungle or **South**, back towards the sea.
**Your imaginary friend Bill** is also here.
There's an **undead pirate** snarling at you.

## Rich Text Editing

NSAttributedString

## Rich Text Editing
### NSAttributedString

hello

## Rich Text Editing

NSAttributedString

## Rich Text Editing

### NSAttributedString

- hello

## Rich Text Editing

### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
```

## Rich Text Editing

### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];
```

## Rich Text Editing

### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];
```

## Rich Text Editing

### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@"hello"];

[attributedString addAttribute:NSForegroundColorAttributeName
```

## Rich Text Editing

### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@"hello"];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
```

## Rich Text Editing

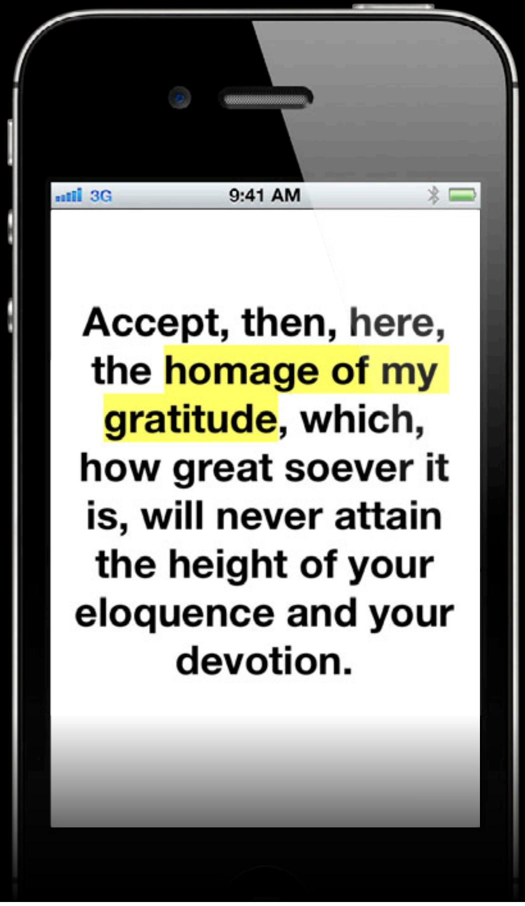
### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
                         range:NSMakeRange(2, 2)];
```

## Rich Text Editing

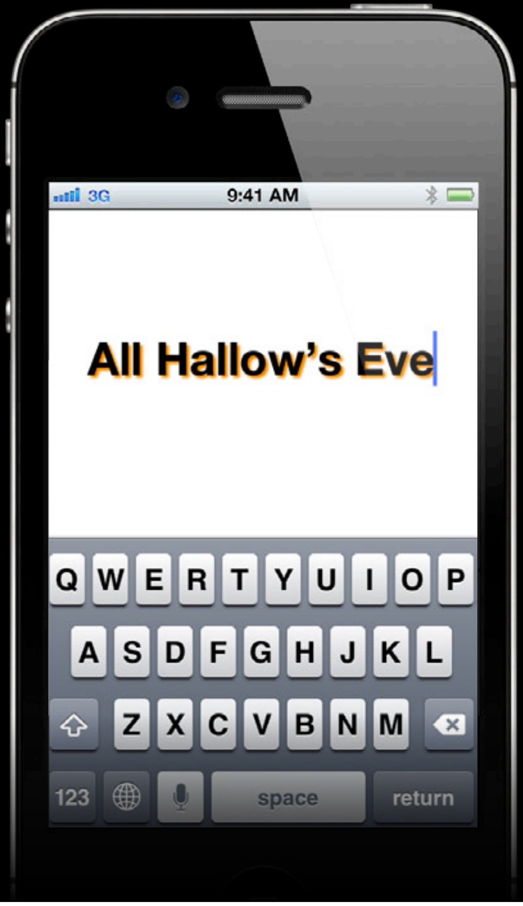
`NSBackgroundColorAttributeName`



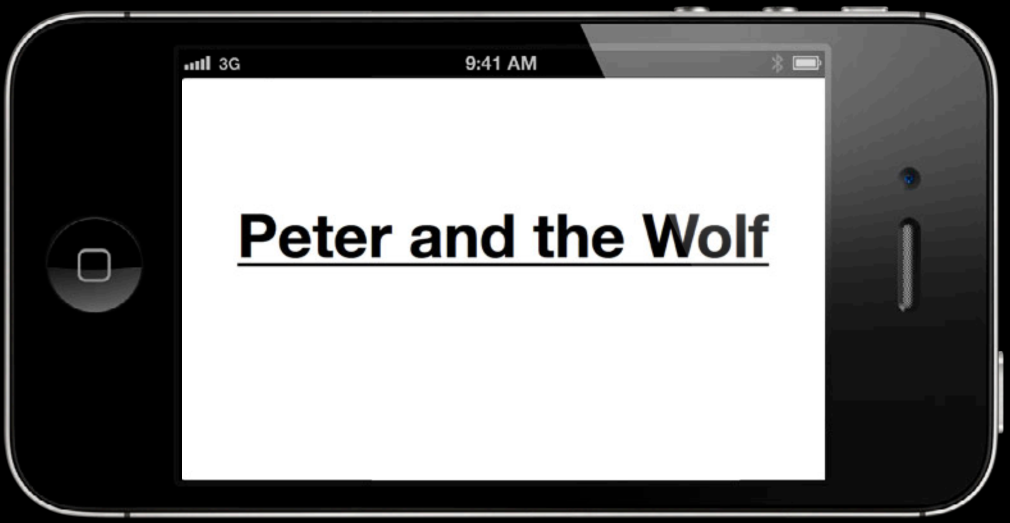
Accept, then, here, the homage of my gratitude, which, how great soever it is, will never attain the height of your eloquence and your devotion.

## Rich Text Editing

`NSShadowAttributeName`



## Rich Text Editing



Peter and the Wolf

NSUnderlineStyleAttributeName

## Rich Text Editing

NSAttributedString

## Rich Text Editing

### NSAttributedString

- hello

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
                         range:NSMakeRange(2, 2)];
```

## Rich Text Editing

### NSAttributedString

- ~~hello~~

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
                         range:NSMakeRange(2, 2)];

 [attributedString addAttribute:NSStrikethroughAttributeName
                          value:
                          range:NSMakeRange(0, 5)];
```

## Rich Text Editing

### NSAttributedString

- ~~hello~~

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
                         range:NSMakeRange(2, 2)];

 [attributedString addAttribute:NSStrikethroughAttributeName
                          value:[NSNumber numberWithBool:YES]
                          range:NSMakeRange(0, 5)];
```

# Rich Text Editing

## NSAttributedString

- ~~hello~~

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
                         range:NSMakeRange(2, 2)];

 [attributedString addAttribute:NSStrikethroughAttributeName
                          value:@YES
                          range:NSMakeRange(0, 5)];
```

## Rich Text Editing

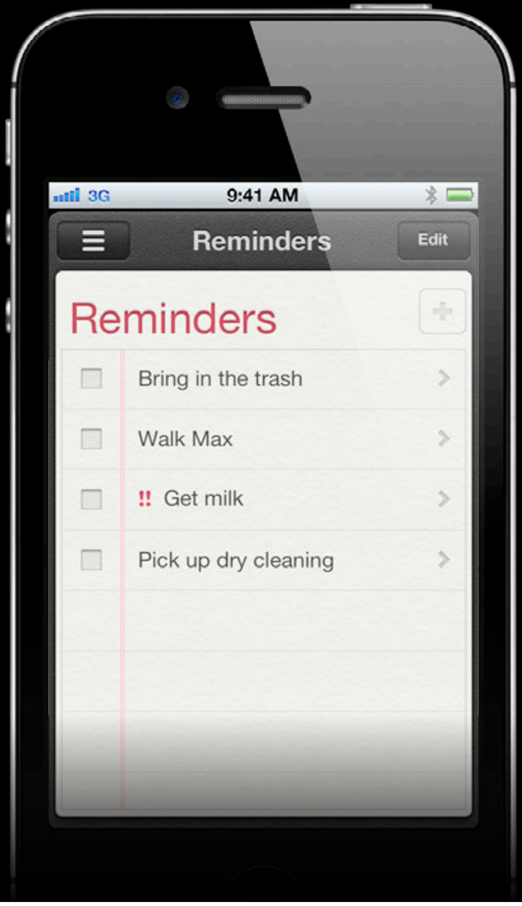
### NSAttributedString

- ~~hello~~

```
NSMutableAttributedString *attributedString =
[[NSMutableAttributedString alloc] initWithString:@”hello”];

[attributedString addAttribute:NSForegroundColorAttributeName
                         value:[UIColor greenColor]
                         range:NSMakeRange(2, 2)];

 [attributedString addAttribute:NSStrikethroughAttributeName
                          value:@YES
                          range:NSMakeRange(0, 5)];

myTextView.attributedText = attributedString;
```

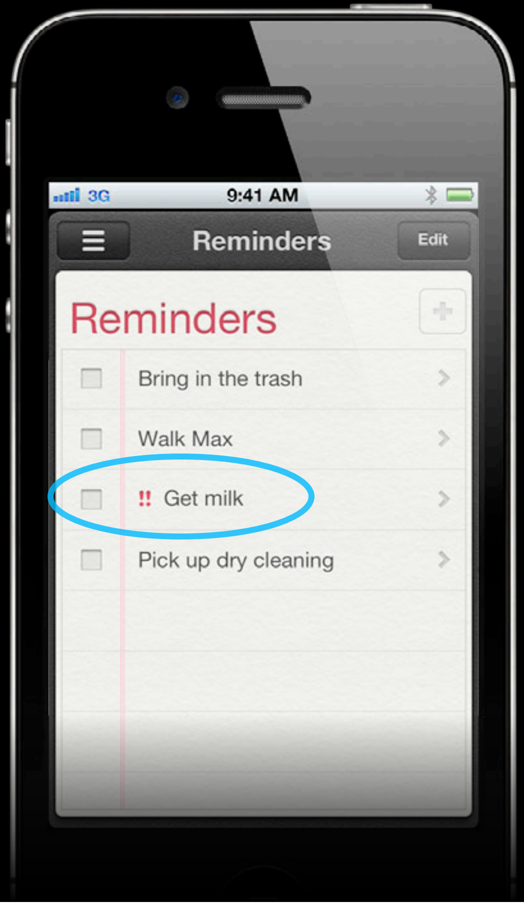
## Rich Text Editing

### NSAttributedString



## Rich Text Editing

### NSAttributedString



Reminders

Bring in the trash

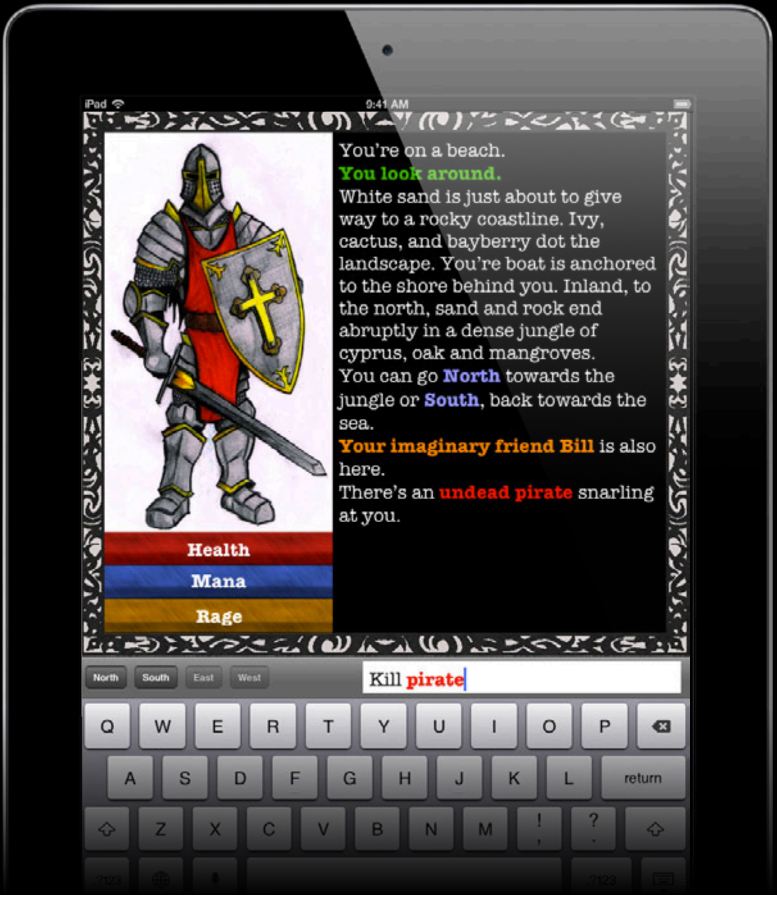
Walk Max

!! Get milk

Pick up dry cleaning

## Rich Text Editing

### Typing attributes

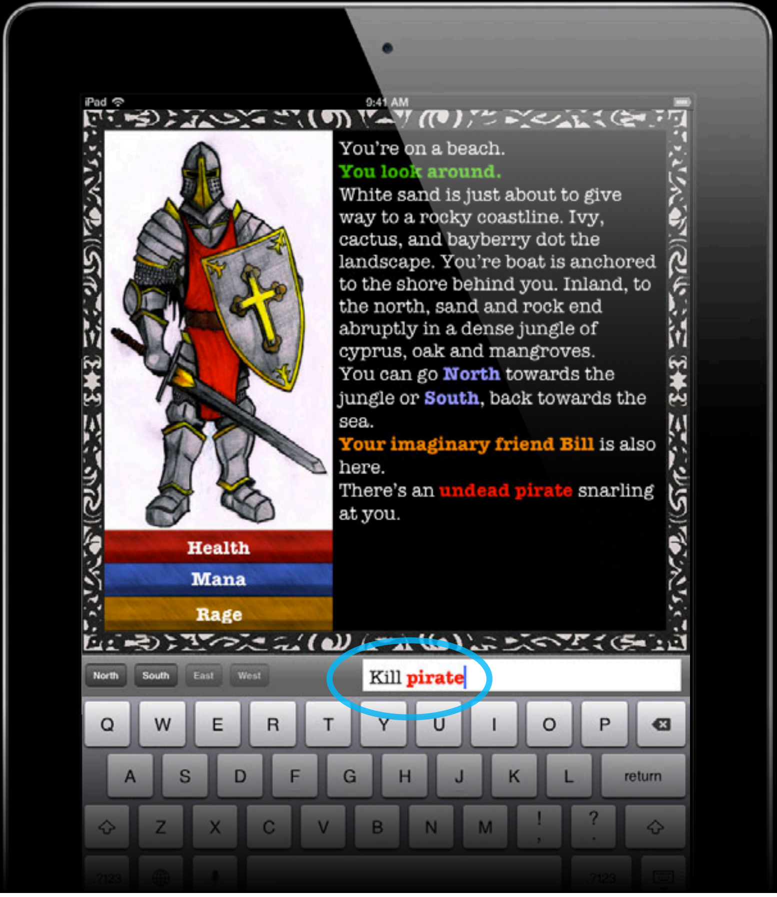


You're on a beach.
**You look around.**
White sand is just about to give way to a rocky coastline. Ivy, cactus, and bayberry dot the landscape. You're boat is anchored to the shore behind you. Inland, to the north, sand and rock end abruptly in a dense jungle of cyprus, oak and mangroves.
You can go **North** towards the jungle or **South**, back towards the sea.
**Your imaginary friend Bill** is also here.
There's an **undead pirate** snarling at you.

Kill **pirate**

# Rich Text Editing

## Typing attributes



You're on a beach.
**You look around.**
White sand is just about to give way to a rocky coastline. Ivy, cactus, and bayberry dot the landscape. You're boat is anchored to the shore behind you. Inland, to the north, sand and rock end abruptly in a dense jungle of cyprus, oak and mangroves.
You can go **North** towards the jungle or **South**, back towards the sea.
**Your imaginary friend Bill** is also here.
There's an **undead pirate** snarling at you.

Kill **pirate**

## Rich Text Editing

### Typing attributes

```
NSMutableDictionary *attributes =
  [myTextView.typingAttributes mutableCopy];

[attributes addObject:[UIColor redColor]
               forKey:NSForegroundColorAttributeName];

myTextView.typingAttributes = attributes;
```

## *Demo*

Justin Garcia

## Handling User Input

New

- UITextInputTraits
- Rich text editing
- Dictation API

## Handling User Input

- UITextInputTraits
- Rich text editing
- Dictation API

“[Dictation] is now one of my favorite features of the iPhone 4S.”

John Gruber

## Dictation API

### Access dictation alternatives

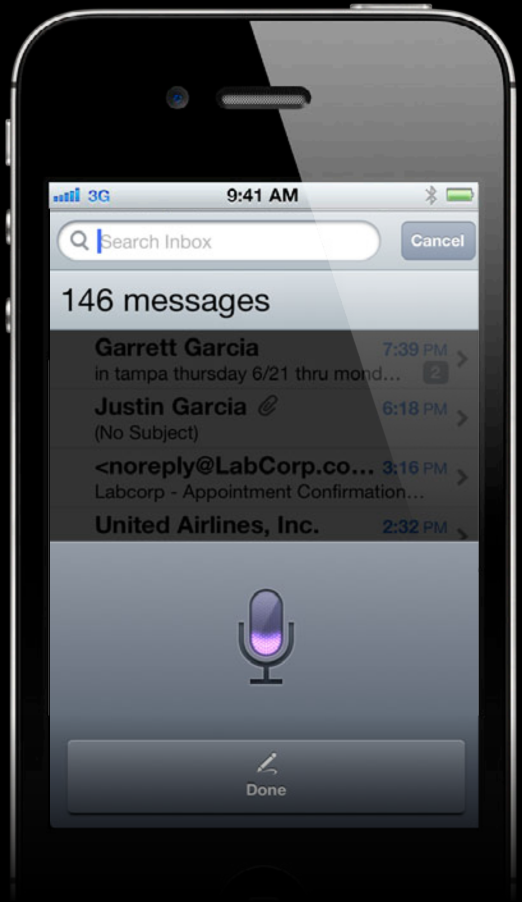
## Dictation API

### Access dictation alternatives

3G
9:41 AM
Search Inbox
Cancel
146 messages
Garrett Garcia 7:39 PM
in tampa thursday 6/21 thru mond...
Justin Garcia 6:18 PM
(No Subject)
<noreply@LabCorp.co... 3:16 PM
Labcorp - Appointment Confirmation...
United Airlines, Inc. 2:32 PM
123
space
Search

## Dictation API

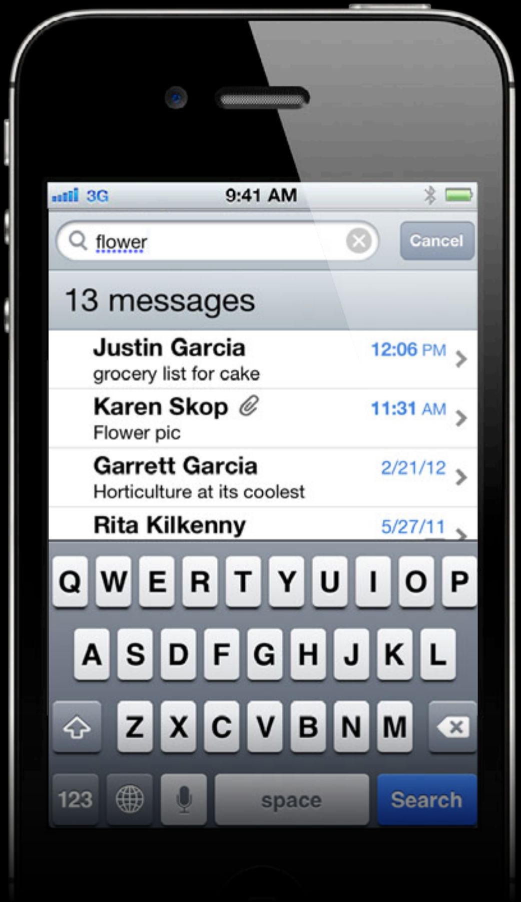
### Access dictation alternatives



| | | |
|---|---|---|
| Garrett Garcia | 7:39 PM | |
| in tampa thursday 6/21 thru mond... | | 2 |
| Justin Garcia | 6:18 PM | |
| (No Subject) | | |
| <noreply@LabCorp.co... | 3:16 PM | |
| Labcorp - Appointment Confirmation... | | |
| United Airlines, Inc. | 2:32 PM | |

# Dictation API

## Access dictation alternatives



13 messages

Justin Garcia 12:06 PM
grocery list for cake

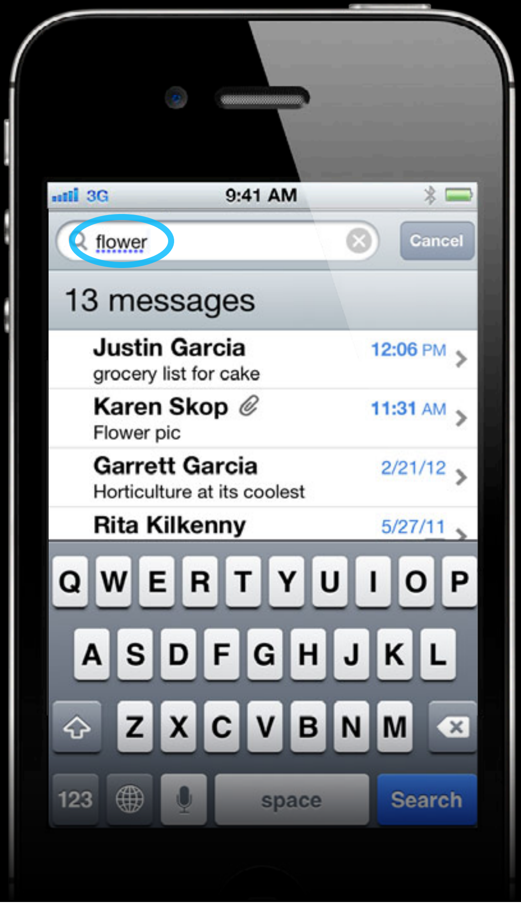
Karen Skop 11:31 AM
Flower pic

Garrett Garcia 2/21/12
Horticulture at its coolest

Rita Kilkenny 5/27/11

## Dictation API

### Access dictation alternatives



Justin Garcia 12:06 PM
grocery list for cake

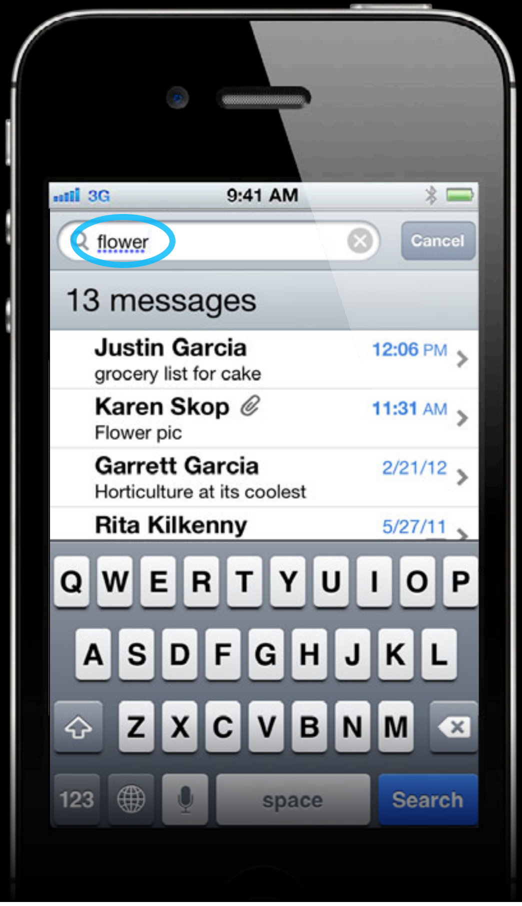
Karen Skop 11:31 AM
Flower pic

Garrett Garcia 2/21/12
Horticulture at its coolest

Rita Kilkenny 5/27/11

## Dictation API

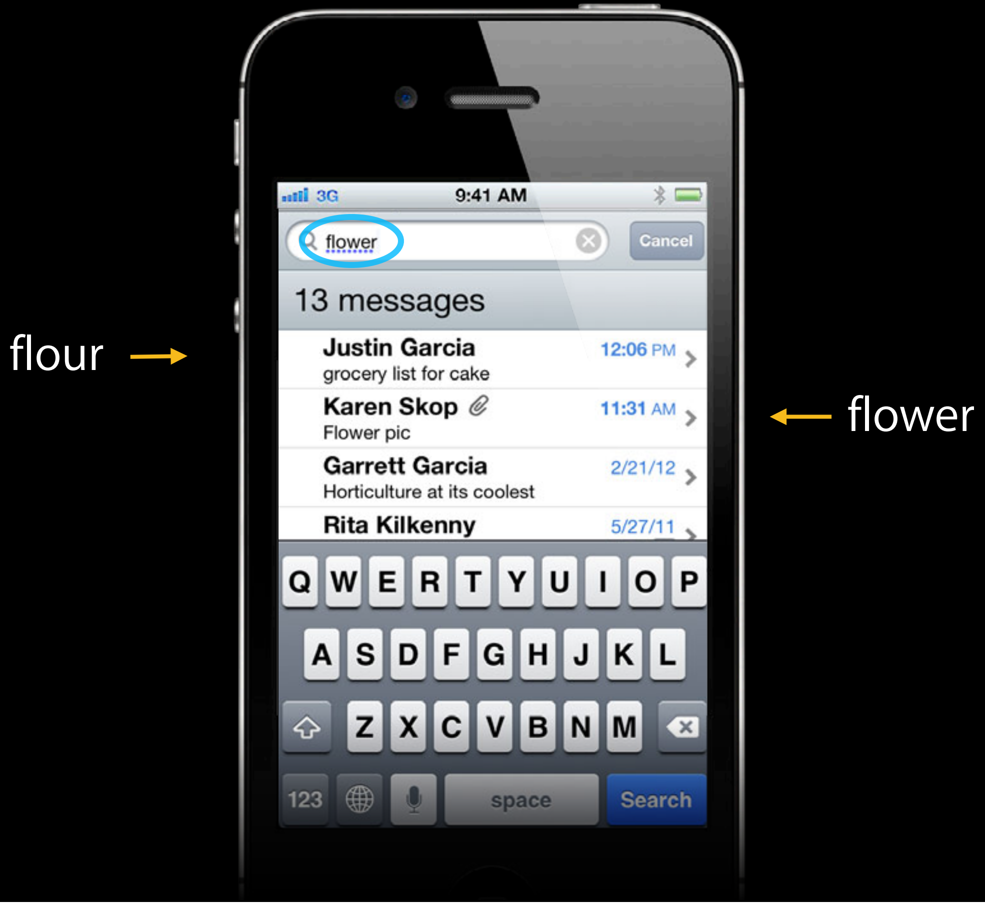
### Access dictation alternatives



← flower

# Dictation API

## Access dictation alternatives



## Dictation API

### Access dictation alternatives

```
// Override in your UITextField or UITextView subclass:
- (void)insertDictationResult:(NSArray *)result {
   for (UIDictationPhrase *p in result) {
      NSArray *alternatives = p.alternatives;
      for (NSString *alternative in alternatives) {
        // Build the set of alternate interpretations.
      }
   }

   // Search messages with the full set of interpretations.

   [super insertDictationResult:result];
}
```

## Handling User Input

- UITextInputTraits
- Rich text editing
- Dictation API

## What You Will Learn

- Managing the keyboard
- Managing static text
- Handling user input

## More Information

**Paul Marcos**
Application Services Evangelist
pmarcos@apple.com

**Documentation**
Text, Web and Editing Programming Guide
http://developer.apple.com/library/ios/#documentation/StringsTextFonts/Conceptual/TextAndWebiPhoneOS/Introduction/Introduction.html#//apple_ref/doc/uid/TP40009542-CH1-SW1

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

## Related Sessions

| | |
|---|---|
| Introduction to Attributed Strings for iOS | Mission<br>Wednesday 3:15PM |
| Advanced Attributed Strings for iOS | Mission<br>Thursday 10:00AM |
| Internationalization Tips and Tricks | Marina<br>Friday 10:15AM |

## Labs

| | |
|---|---|
| Attributed Strings & Text Lab | Essentials Lab A<br>Thursday 11:30AM |
| Internationalization Lab | App Services Lab A<br>Friday 11:30AM |

## Summary

## Summary

- Keyboard size and position changes

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard
- Unicode essentials

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard
- Unicode essentials
- System selection in custom text views

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views
- UITextInputTraits

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views
- UITextInputTraits
- Rich text editing

## Summary

- Keyboard size and position changes
- Attaching views to the keyboard
- Unicode essentials
- System selection in custom text views
- UITextInput in standard text views
- UITextInputTraits
- Rich text editing
- Dictation APIs

WWDC2012

The last 3 slides after the logo are intentionally left blank for all presentations.

The last 3 slides after the logo are intentionally left blank for all presentations.

The last 3 slides after the logo are intentionally left blank for all presentations.