

The Keys to a Better Text Input Experience

Crafting a better typing experience in your app

Session 242

Kasia Wawer, iOS Keyboards

Shuchen Li, iOS Keyboards

James Magahern, iOS Keyboards

Create a Better Input Experience

Create a Better Input Experience

Integrate the keyboard into your layout

Create a Better Input Experience

Integrate the keyboard into your layout

Create dynamic input accessory views

Create a Better Input Experience

Integrate the keyboard into your layout

Create dynamic input accessory views

Make your app multilingual

Create a Better Input Experience

Integrate the keyboard into your layout

Create dynamic input accessory views

Make your app multilingual

Use traits for smarter QuickType

Create a Better Input Experience

Integrate the keyboard into your layout

Create dynamic input accessory views

Make your app multilingual

Use traits for smarter QuickType

Hardware keyboard support

Create a Better Input Experience

Integrate the keyboard into your layout

Create dynamic input accessory views

Make your app multilingual

Use traits for smarter QuickType

Hardware keyboard support

Custom input views

Create a Better Input Experience

Integrate the keyboard into your layout

Create dynamic input accessory views

Make your app multilingual

Use traits for smarter QuickType

Hardware keyboard support

Custom input views

Keyboard extension tips and best practices

Integrating the Keyboard into Your App

Adaptivity and input accessory views

Kasia Wawer, iOS Keyboards



9:41 AM

100%

CatChat



Simba

I left so much food this morning; how did you eat it all?



Faithful

Let me know when you wake up



Penny

You're not lost, you're probably in the closet again



Nala

Keep an eye on your brother, please



Accounting for changing keyboard heights

Working with non-scrolling layouts

Working with scrolling layouts

Adding an input accessory view

Accounting for the Keyboard

Accounting for the Keyboard

Heights vary by language and settings

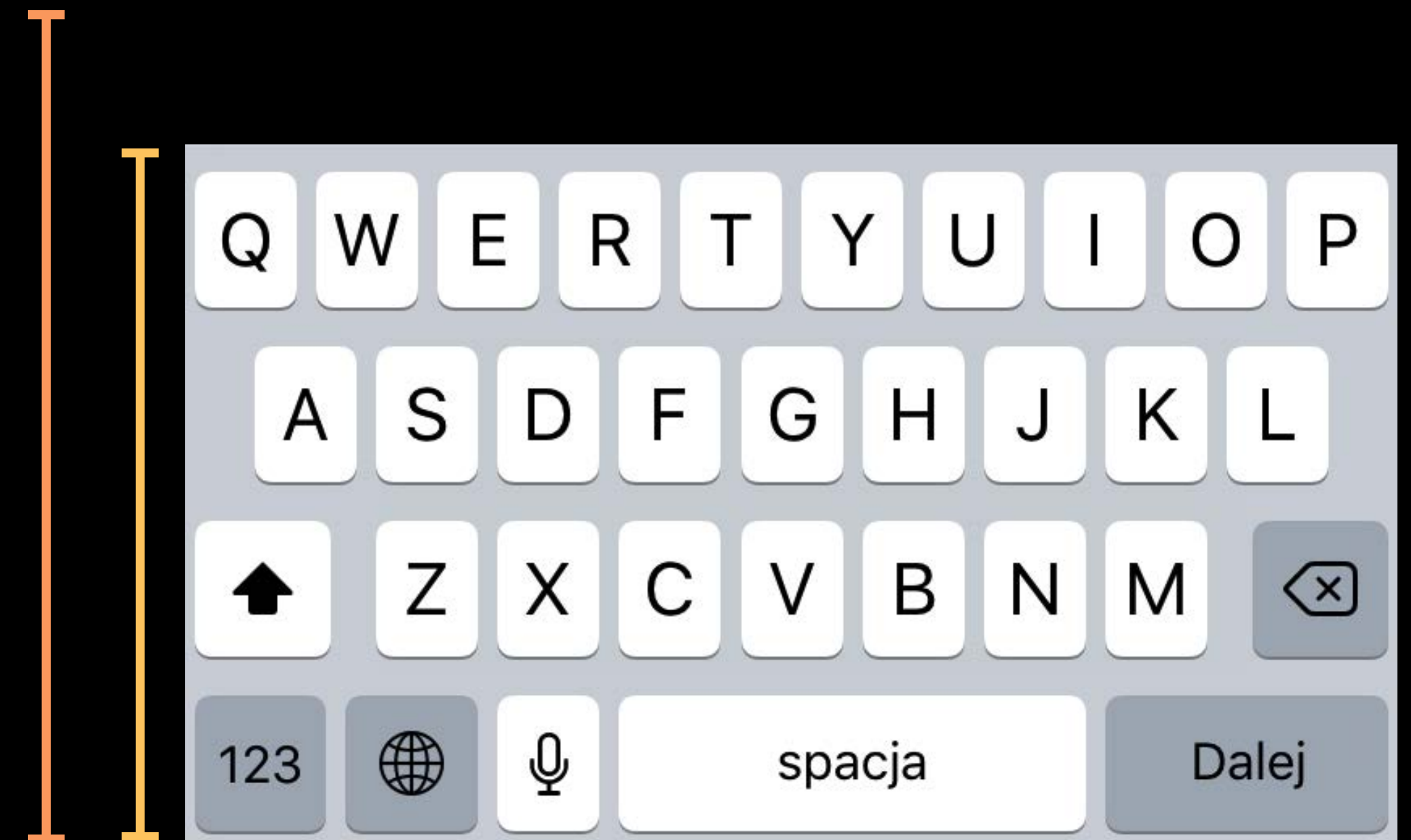
Accounting for the Keyboard

Heights vary by language and settings



Accounting for the Keyboard

Heights vary by language and settings



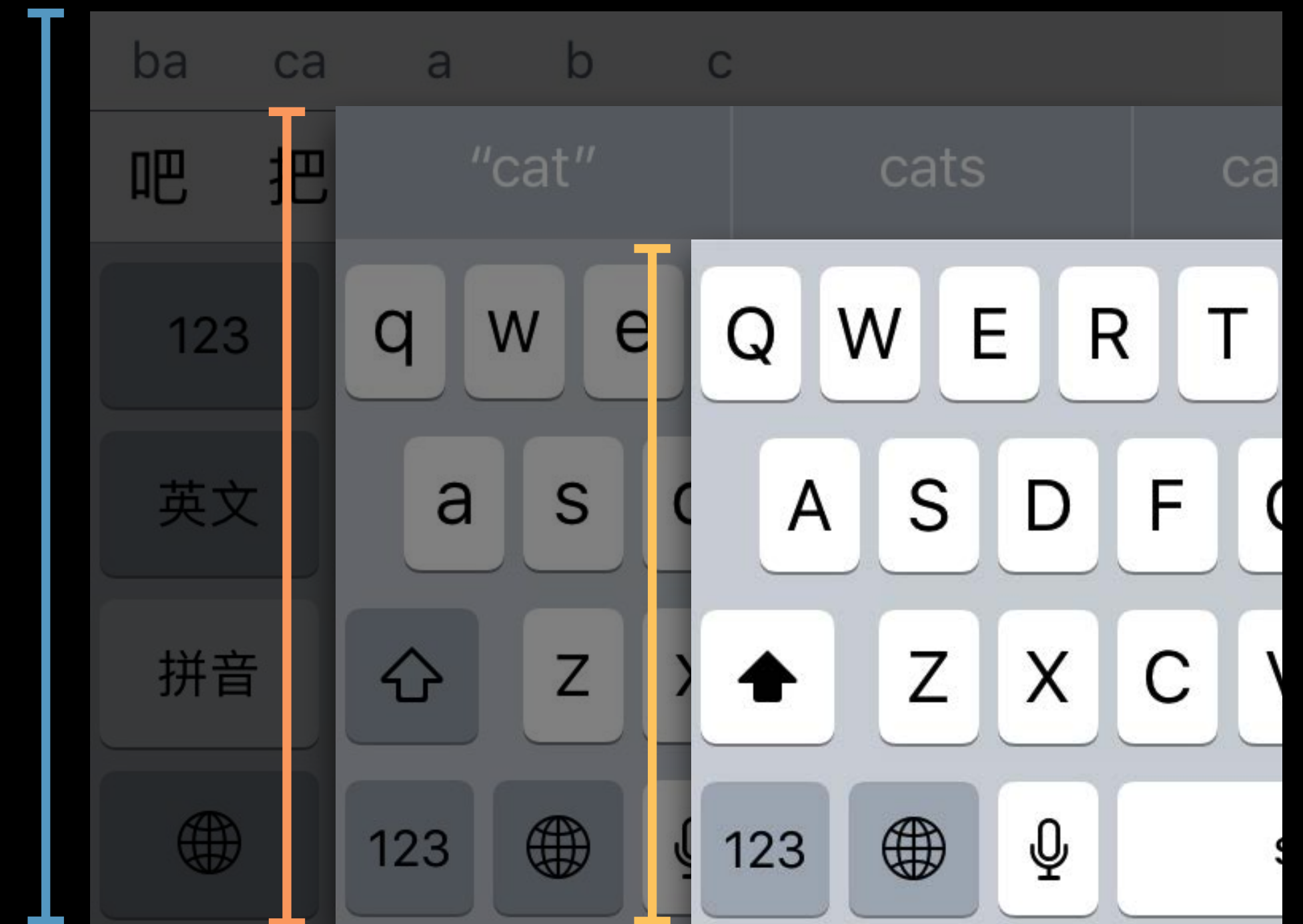
Accounting for the Keyboard

Heights vary by language and settings



Accounting for the Keyboard

Heights vary by language and settings

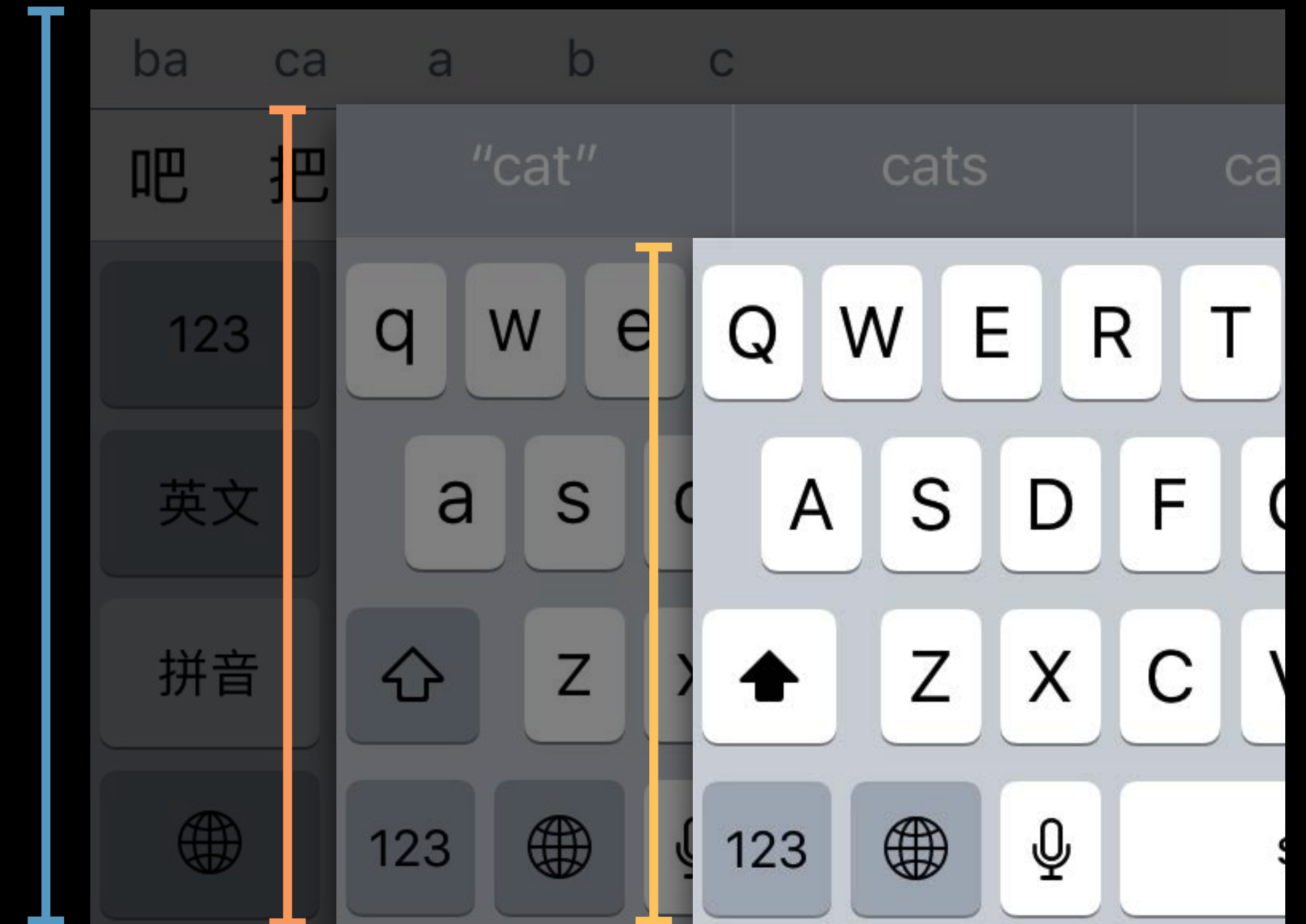


Accounting for the Keyboard

Heights vary by language and settings

Register for notifications

```
UIKeyboardDidShow  
UIKeyboardDidHide  
UIKeyboardDidChangeFrame
```



Undocked and Split Keyboards

Undocked and Split Keyboards

Dismissing and undocking both send Hide notifications

Undocked and Split Keyboards

Dismissing and undocking both send Hide notifications

Frame change notifications will continue when undocked

Undocked and Split Keyboards

Dismissing and undocking both send Hide notifications

Frame change notifications will continue when undocked

Track the most recent Hide or Show event

CatChat



Simba
 

Faithful
Let me know when you wake up

Penny
You're not lost, you're probably in the closet again

Nala
Keep an eye on your brother, please

Food?

I'll make sure to pick up extra on the way home

Please don't try to open the fridge again.

More food?

I left so much food this morning; how did you eat it all already??

Food!

I'll be there as soon as I can

... food?

Sorry, you'll just have to be patient!



I promise you'll get fed as soon as I get home, ok?



Send

CatChat



Simba
 

Faithful
 Let me know when you wake up

Penny
 You're not lost, you're probably in the closet again

Nala
 Keep an eye on your brother, please

Food?

I'll make sure to pick up extra on the way home

Please don't try to open the fridge again.

More food?

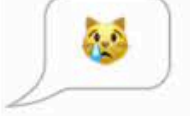
I left so much food this morning; how did you eat it all already??

Food!

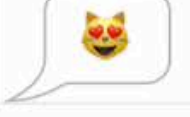
I'll be there as soon as I can

... food?

Sorry, you'll just have to be patient!



I promise you'll get fed as soon as I get home, ok?



Send

Being in the Right Space

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {
    if !keyboardIsHidden {
        guard let userInfo = notification.userInfo,
              let frame = userInfo[UIKeyboardFrameEndUserInfoKey] as? CGRect else { return }
        let convertedFrame = view.convert(frame, from: UIScreen.main.coordinateSpace)
        let intersectedKeyboardHeight = view.frame.intersection(convertedFrame).height
    }
}
```

Being in the Right Space

Keyboard is always in the screen coordinate space

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        guard let userInfo = notification.userInfo,  
              let frame = userInfo[UIKeyboardFrameEndUserInfoKey] as? CGRect else { return }  
        let convertedFrame = view.convert(frame, from: UIScreen.main.coordinateSpace)  
        let intersectedKeyboardHeight = view.frame.intersection(convertedFrame).height  
    }  
}
```

Being in the Right Space

Keyboard is always in the screen coordinate space

Convert the frame from screen coordinates

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {
    if !keyboardIsHidden {
        guard let userInfo = notification.userInfo,
              let frame = userInfo[UIKeyboardFrameEndUserInfoKey] as? CGRect else { return }
        let convertedFrame = view.convert(frame, from: UIScreen.main.coordinateSpace)
        let intersectedKeyboardHeight = view.frame.intersection(convertedFrame).height
    }
}
```

Being in the Right Space

Keyboard is always in the screen coordinate space

Convert the frame from screen coordinates

Use the intersection of the converted keyboard frame and your view

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {
    if !keyboardIsHidden {
        guard let userInfo = notification.userInfo,
              let frame = userInfo[UIKeyboardFrameEndUserInfoKey] as? CGRect else { return }
        let convertedFrame = view.convert(frame, from: UIScreen.main.coordinateSpace)
        let intersectedKeyboardHeight = view.frame.intersection(convertedFrame).height
    }
}
```

Accounting for changing keyboard heights

Working with non-scrolling layouts

Working with scrolling layouts

Adding an input accessory view



9:41 AM

100%

Add new pet

[Save](#)

Hi

I

The

Q

W

E

R

T

Y

U

I

O

P

A

S

D

F

G

H

J

K

L



Z

X

C

V

B

N

M



123



space

return

Adaptive Keyboard-Inclusive Layouts

Create a custom `UILayoutGuide` and height constraint

```
func setUpViews() {  
    // ... view set up ...  
    let keyboardGuide = UILayoutGuide()  
    view.addLayoutGuide(keyboardGuide)  
    heightConstraint = keyboardGuide.heightAnchor.constraint(equalToConstant: kDefaultHeight)  
    heightConstraint.isActive = true  
    // ... view set up ...  
}
```


Adaptive Keyboard-Inclusive Layouts

Create a custom `UILayoutGuide` and height constraint

Tie lowest view to top of your layout guide

```
func setUpViews() {  
    // ...  
    keyboardGuide.bottomAnchor.constraint(equalTo: view.safeAreaLayoutGuide.bottomAnchor)  
        .isActive = true  
    bottomSpacer.bottomAnchor.constraint(equalTo: keyboardGuide.topAnchor).isActive = true  
    // ...  
}
```

Adaptive Keyboard-Inclusive Layouts

Create a custom `UILayoutGuide` and height constraint

Tie lowest view to top of your layout guide

```
func setUpViews() {  
    // ...  
    keyboardGuide.bottomAnchor.constraint(equalTo: view.safeAreaLayoutGuide.bottomAnchor)  
        .isActive = true  
    bottomSpacer.bottomAnchor.constraint(equalTo: keyboardGuide.topAnchor).isActive = true  
    // ...  
}
```

Adaptive Keyboard-Inclusive Layouts

Create a custom `UILayoutGuide` and height constraint

Tie lowest view to top of your layout guide

Use the converted frame to set the height constraint of layout guide

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {
    if !keyboardIsHidden {
        // ... Convert frame ...
        UIView.animate(withDuration: 0.2) {
            heightConstraint.constant = intersectedKeyboardHeight
            view.layoutIfNeeded()
        }
    }
}
```

Adaptive Keyboard-Inclusive Layouts

Create a custom `UILayoutGuide` and height constraint

Tie lowest view to top of your layout guide

Use the converted frame to set the height constraint of layout guide

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        // ... Convert frame ...  
        UIView.animate(withDuration: 0.2) {  
            heightConstraint.constant = intersectedKeyboardHeight  
            view.layoutIfNeeded()  
        }  
    }  
}
```

Accounting for changing keyboard heights

Working with non-scrolling layouts

Working with scrolling layouts

Adding an input accessory view



9:41 AM

100%

< CatChat

Simba

Just checking in!

How's it going today?

Food?

I'll pick up some more on the way home

Please don't try to open the fridge again.

More food now?

I left so much food this morning; how did
you eat it all?

Food!

I'll be there as soon as I can!

... food?

Send

Scrolling Views and Keyboards

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        // ... Convert frame ...  
        scrollView.contentInset.bottom = intersectedKeyboardHeight  
        // ... Handle content scrolling ...  
    }  
}
```

Scrolling Views and Keyboards

Make sure the keyboard is visible

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        // ... Convert frame ...  
        scrollView.contentInset.bottom = intersectedKeyboardHeight  
        // ... Handle content scrolling ...  
    }  
}
```


Scrolling Views and Keyboards

Make sure the keyboard is visible

Convert the frame and get the height

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        // ... Convert frame ...  
        scrollView.contentInset.bottom = intersectedKeyboardHeight  
        // ... Handle content scrolling ...  
    }  
}
```

Scrolling Views and Keyboards

Make sure the keyboard is visible

Convert the frame and get the height

Set content insets appropriately

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        // ... Convert frame ...  
        scrollView.contentInset.bottom = intersectedKeyboardHeight  
        // ... Handle content scrolling ...  
    }  
}
```

Scrolling Views and Keyboards

Make sure the keyboard is visible

Convert the frame and get the height

Set content insets appropriately

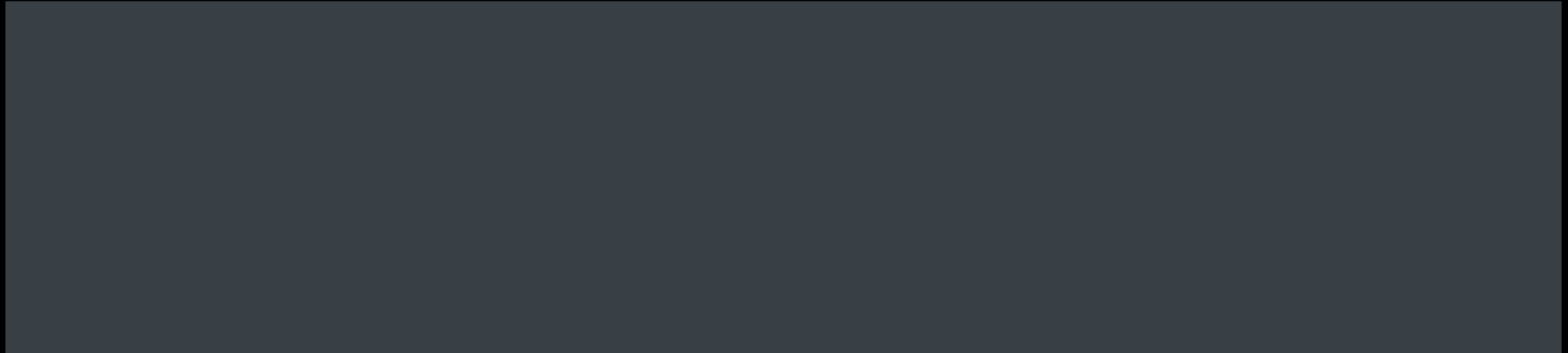
Handle scrolling the content if needed

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    if !keyboardIsHidden {  
        // ... Convert frame ...  
        scrollView.contentInset.bottom = intersectedKeyboardHeight  
        // ... Handle content scrolling ...  
    }  
}
```

UITableViewController

Sets insets for you

Handle scrolling the content if needed



UITableViewController

Sets insets for you

Handle scrolling the content if needed

```
@objc func keyboardFrameChanged(_ notification: Notification) -> Void {  
    let bottomRow = IndexPath(row: items.count - 1, section: 0)  
    tableView.scrollToRow(at: bottomRow, at: UITableViewScrollPosition.bottom, animated: true)  
}
```

Accounting for changing keyboard heights

Working with non-scrolling layouts

Working with scrolling layouts

Adding an input accessory view



9:41 AM

100%

< CatChat

Simba

I left so much food this morning; how did you eat it all?

Food!

I'll be there as soon as I can!

... food?

I can't right now; you'll just have to be more patient!

Send

hi

I

the

Q

W

E

R

T

Y

U

I

O

P

A

S

D

F

G

H

J

K

L



Z

X

C

V

B

N

M



123



space

return

Accessorizing the Keyboard

Accessorizing the Keyboard

Host view controller returns `true` for `canBecomeFirstResponder`

Accessorizing the Keyboard

Host view controller returns `true` for `canBecomeFirstResponder`

Use custom view or view controller

- For view, override `inputAccessoryView`
- For view controller, override `inputAccessoryViewController`

Accessorizing the Keyboard

Host view controller returns `true` for `canBecomeFirstResponder`

Use custom view or view controller

- For view, override `inputAccessoryView`
- For view controller, override `inputAccessoryViewController`

Changing height sends frame change notifications

Making Dynamic Input Accessory Views



Making Dynamic Input Accessory Views

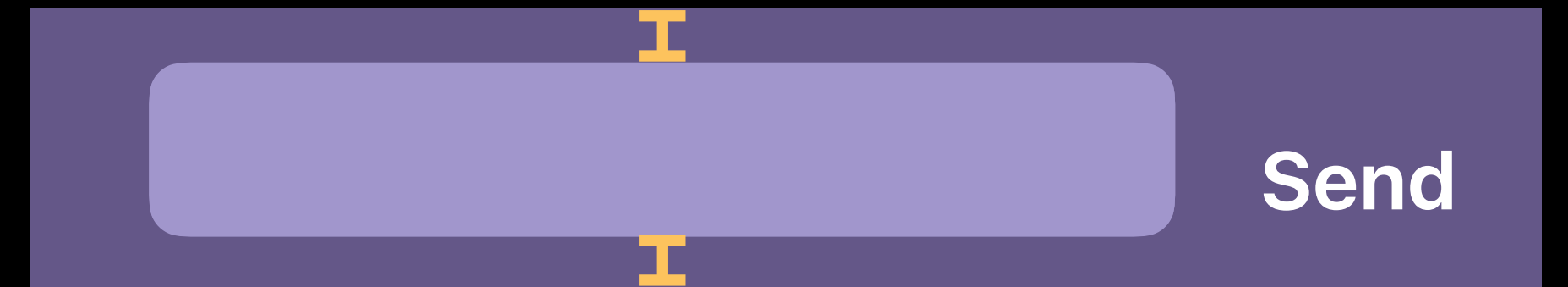
Use expanding view to define height



Making Dynamic Input Accessory Views

Use expanding view to define height

Pin to top and bottom of content view

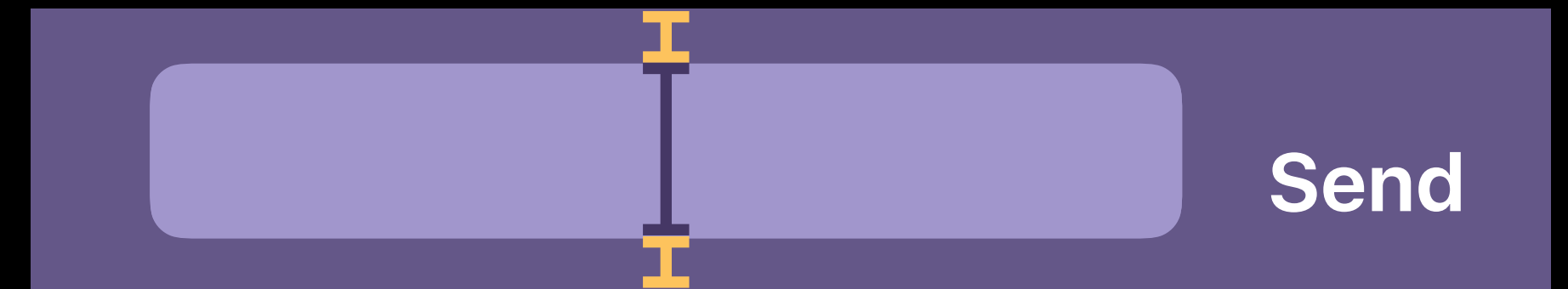


Making Dynamic Input Accessory Views

Use expanding view to define height

Pin to top and bottom of content view

Use or define `intrinsicContentSize`



```
expandingTextView.textContainer.heightTracksTextView = true
```

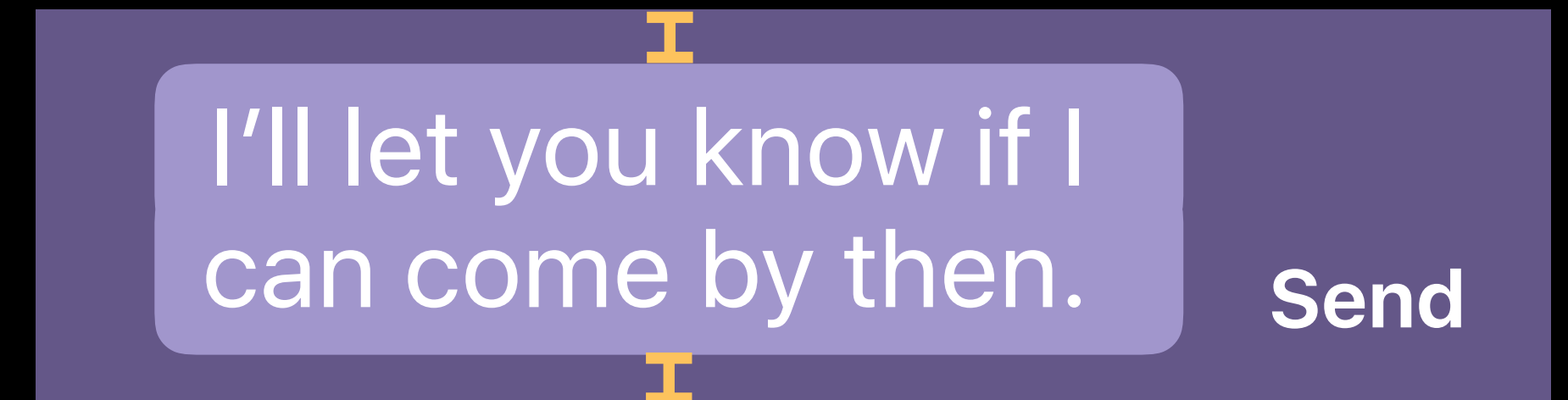
```
expandingTextView.isScrollEnabled = false
```

Making Dynamic Input Accessory Views

Use expanding view to define height

Pin to top and bottom of content view

Use or define `intrinsicContentSize`



```
expandingTextView.textContainer.heightTracksTextView = true
```

```
expandingTextView.isScrollEnabled = false
```



```
// Using intrinsicContentSize to determine height

override var intrinsicContentSize: CGSize {
    var newSize = self.bounds.size
    newSize.height = kMinimumHeight
    if expandingTextView.bounds.size.height > 0.0 {
        newSize.height = expandingTextView.bounds.size.height + kVerticalPadding
    }
    if newSize.height > kMaximumHeight {
        newSize.height = kMaximumHeight
    }
    return newSize
}
```

```
// Using intrinsicContentSize to determine height

override var intrinsicContentSize: CGSize {
    var newSize = self.bounds.size
    newSize.height = kMinimumHeight
    if expandingTextView.bounds.size.height > 0.0 {
        newSize.height = expandingTextView.bounds.size.height + kVerticalPadding
    }
    if newSize.height > kMaximumHeight {
        newSize.height = kMaximumHeight
    }
    return newSize
}
```

```
// Using intrinsicContentSize to determine height

override var intrinsicContentSize: CGSize {
    var newSize = self.bounds.size
    newSize.height = kMinimumHeight
    if expandingTextView.bounds.size.height > 0.0 {
        newSize.height = expandingTextView.bounds.size.height + kVerticalPadding
    }
    if newSize.height > kMaximumHeight {
        newSize.height = kMaximumHeight
    }
    return newSize
}
```

```
// Using intrinsicContentSize to determine height

override var intrinsicContentSize: CGSize {
    var newSize = self.bounds.size
    newSize.height = kMinimumHeight
    if expandingTextView.bounds.size.height > 0.0 {
        newSize.height = expandingTextView.bounds.size.height + kVerticalPadding
    }
    if newSize.height > kMaximumHeight {
        newSize.height = kMaximumHeight
    }
    return newSize
}
```

```
// Using intrinsicContentSize to determine height

override var intrinsicContentSize: CGSize {
    var newSize = self.bounds.size
    newSize.height = kMinimumHeight
    if expandingTextView.bounds.size.height > 0.0 {
        newSize.height = expandingTextView.bounds.size.height + kVerticalPadding
    }
    if newSize.height > kMaximumHeight {
        newSize.height = kMaximumHeight
    }
    return newSize
}
```

```
// Using intrinsicContentSize to determine height

override var intrinsicContentSize: CGSize {
    var newSize = self.bounds.size
    newSize.height = kMinimumHeight
    if expandingTextView.bounds.size.height > 0.0 {
        newSize.height = expandingTextView.bounds.size.height + kVerticalPadding
    }
    if newSize.height > kMaximumHeight {
        newSize.height = kMaximumHeight
    }
    return newSize
}
```



9:41 AM

100%

< CatChat

Simba

More food now?

I left so much food this morning; how did you eat it all?

Food!

I'll be there as soon as I can!

... food?

I'm busy right now, |

Send

and

aid

no

q w e r t y u i o p

a s d f g h j k l

⏏ z x c v b n m ⏪

123 space return

Making Your App Feel Magical

Using context to enrich text input experience

Shuchen Li, iOS Keyboards

Multilingual

Being aware of context

The new “smarts”

Marked text

Hardware keyboard



9:41 AM

100%



John



I'm all set!

Great! Do you want to grab a coffee before we go?

Sure. Do you want to meet in front of the building?

OK. I'll see you in 5 mins.

Need to return the projector to admin before they close

Take your time.

Delivered



iMessage



I

The

You

Q W E R T Y U I O P

A S D F G H J K L

↑ Z X C V B N M ↵

123 space return



9:41 AM

100%



Vivian



周末聚会还有什么需要帮忙的吗?

都准备得差不多了 😊

你的蛋糕需要盘子和蜡烛吗?

我有蜡烛, 你准备盘子吧 😊

我还准备了 🍷

小朋友们肯定会非常开心的 😊



Delivered



iMessage



我 你 在 这 一 不 是 有



q w e r t y u i o p

a s d f g h j k l

⏪ z x c v b n m ⏩

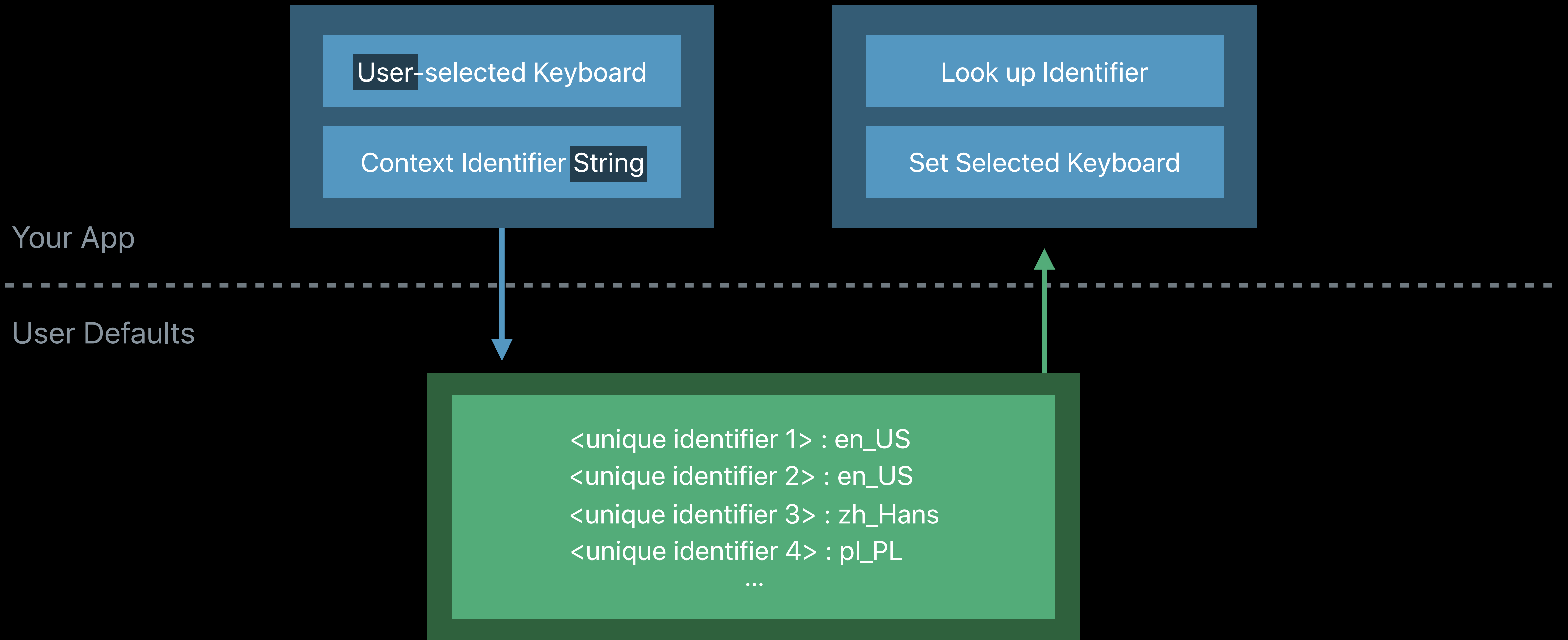
123



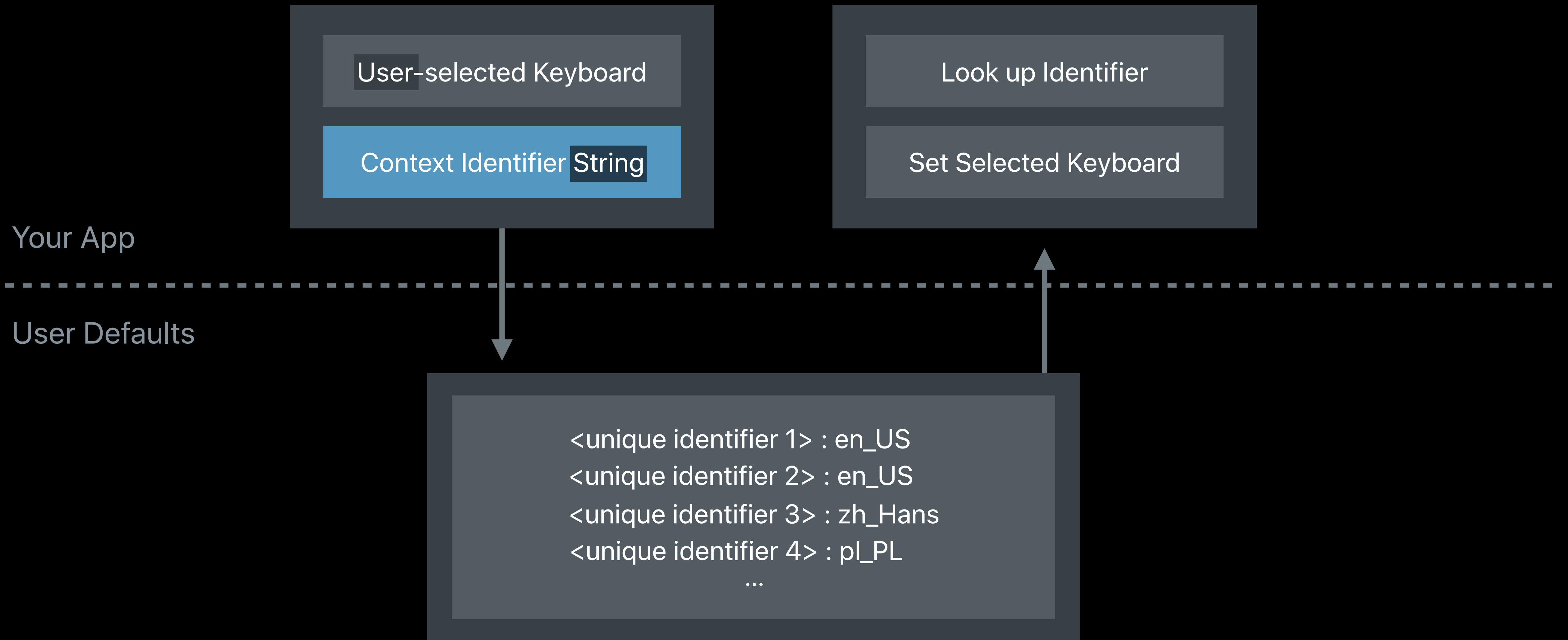
空格

换行

Remembering User Selected Keyboard



Remembering User Selected Keyboard





9:41 AM

100%

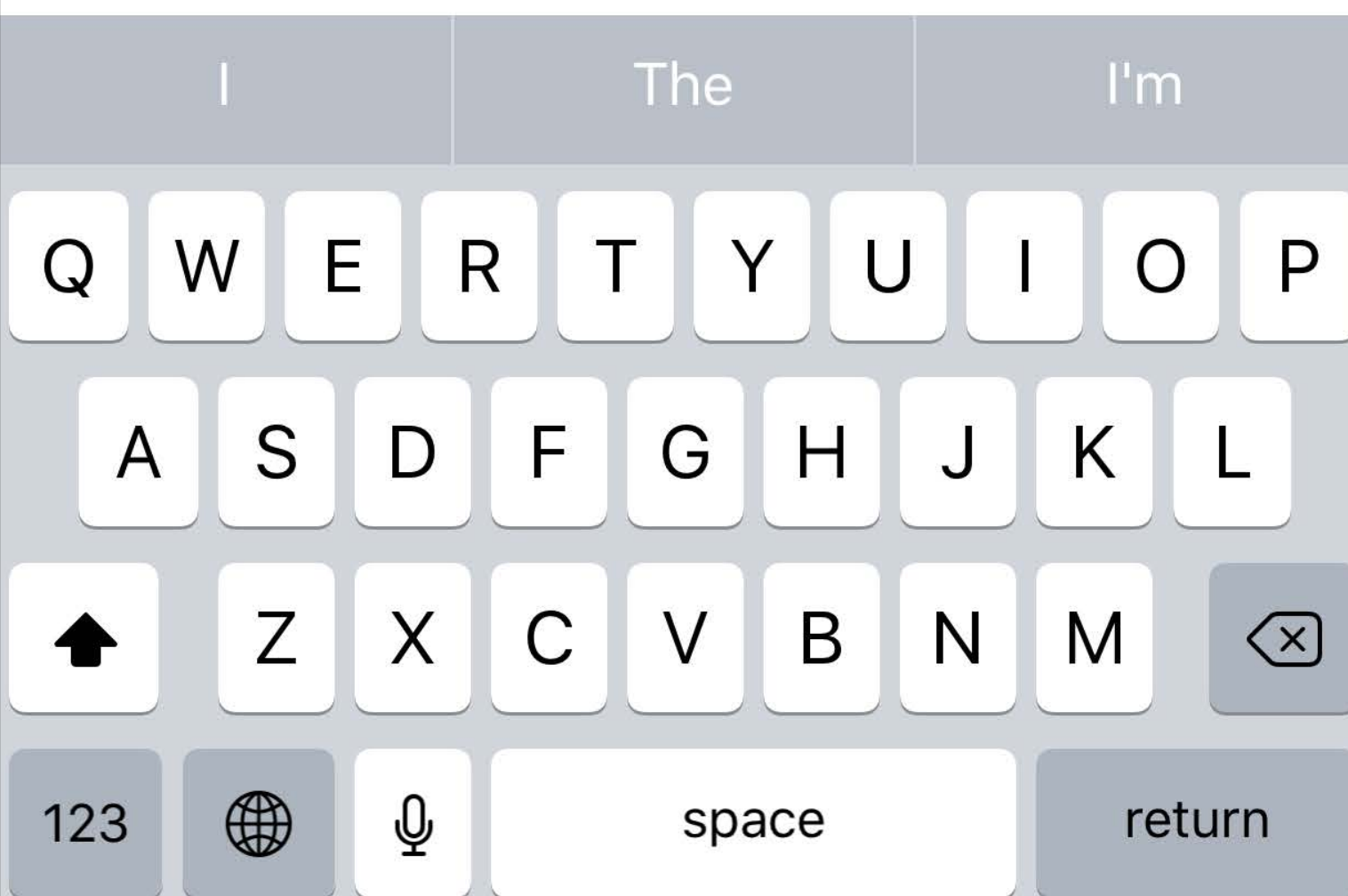
CatChat

Vivian Li

What kind of food does Misiu like?

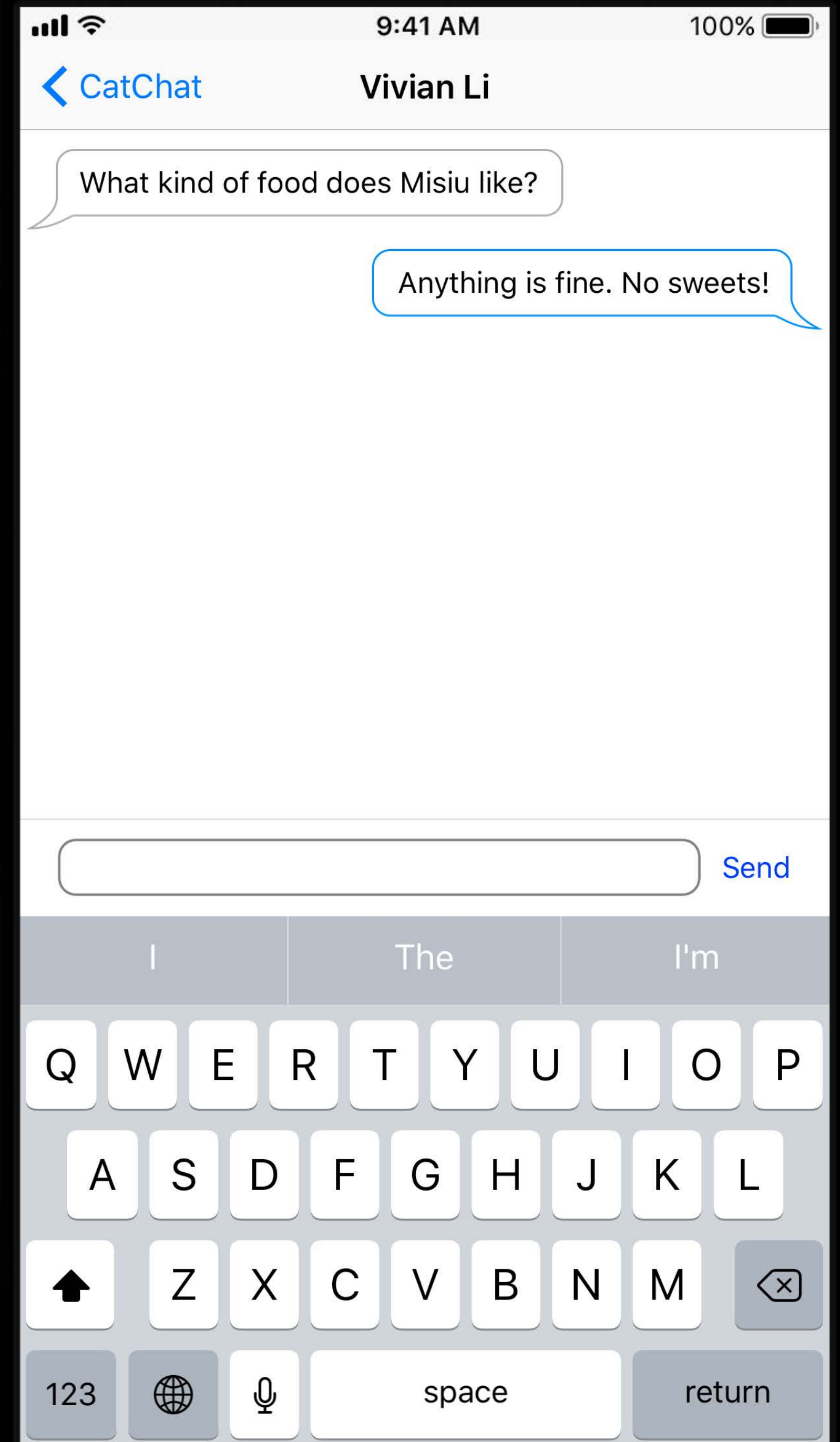
Anything is fine. No sweets!

Send



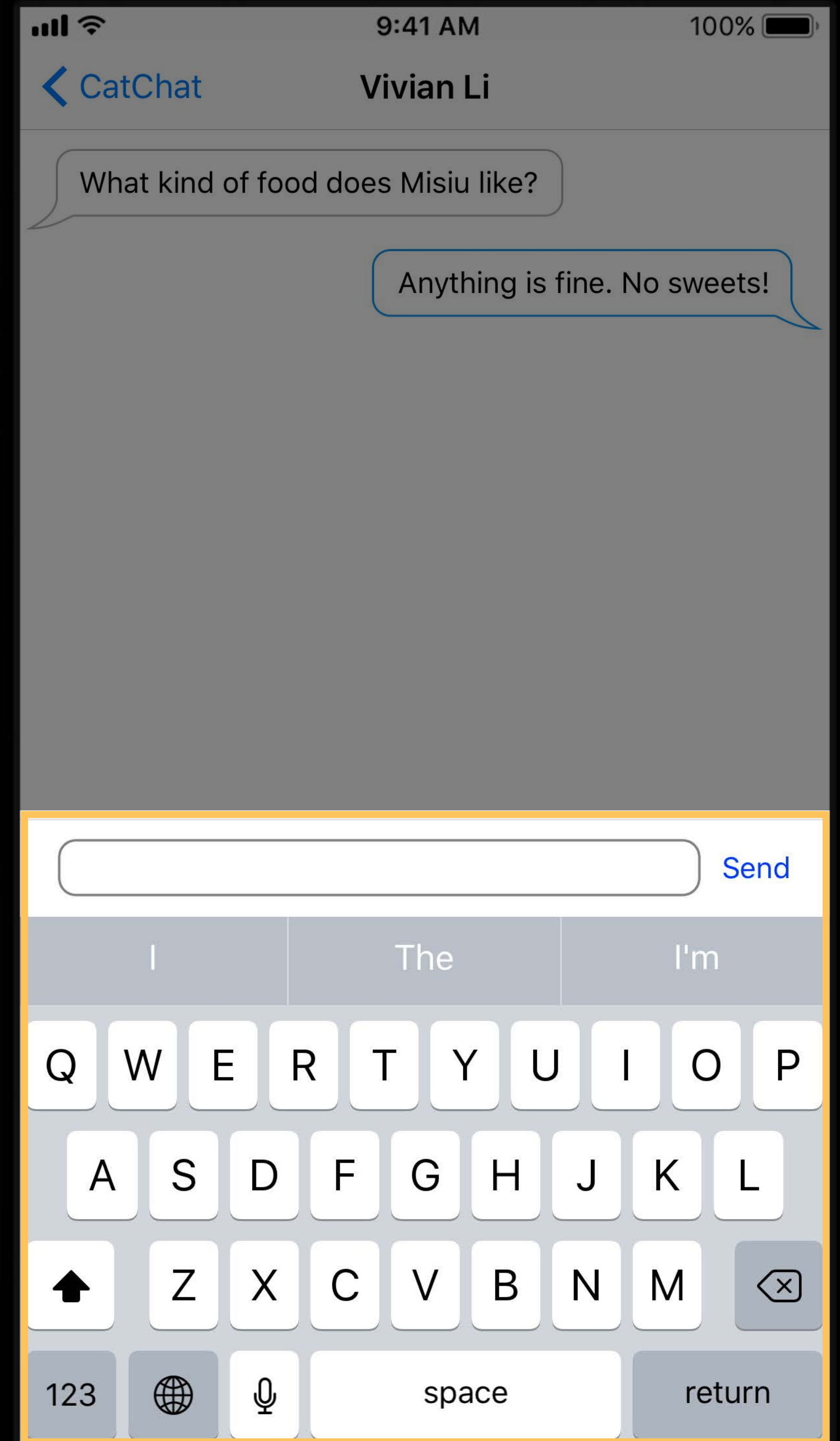
ChatInputAccessoryView

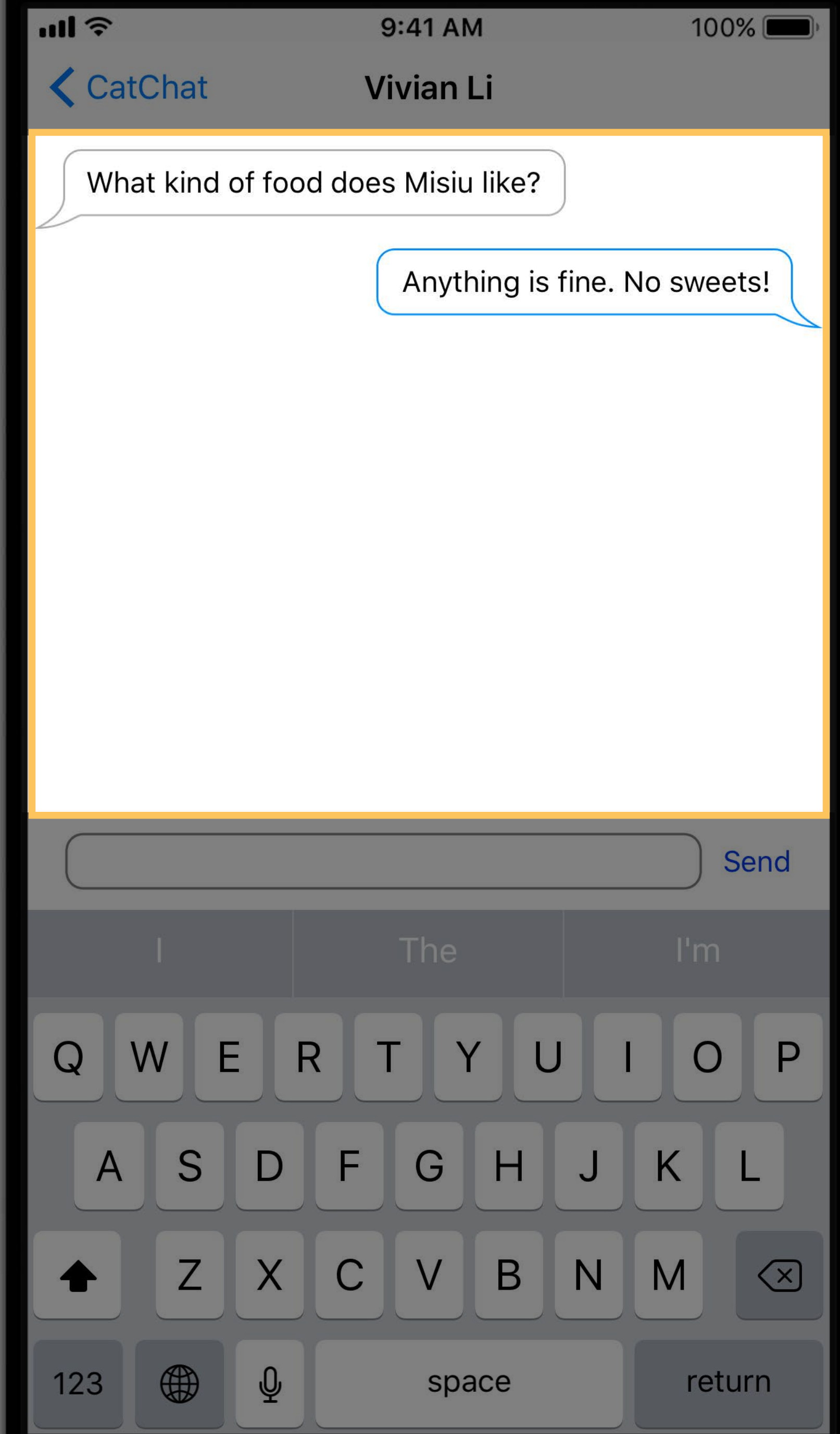
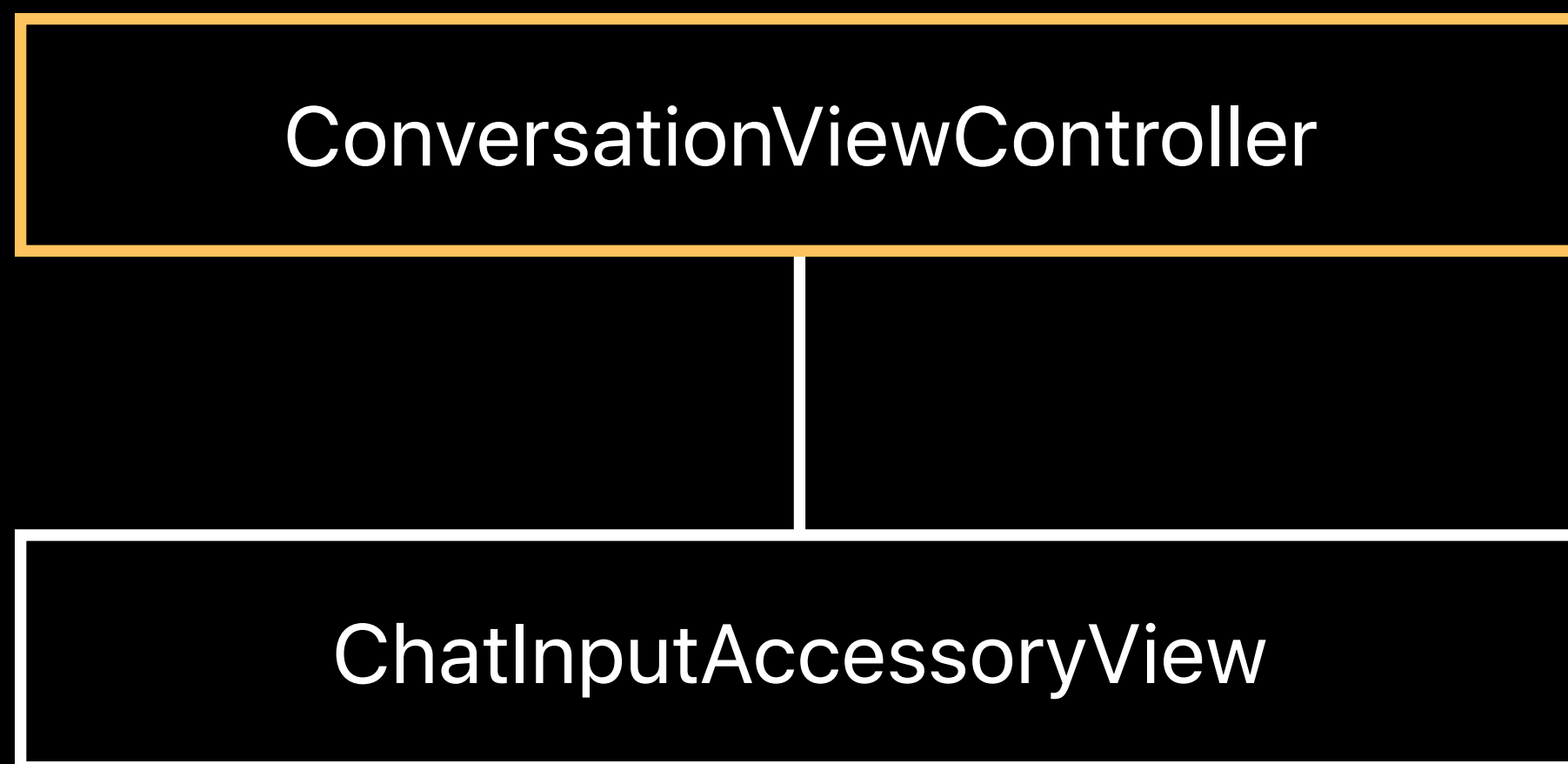
becomesFirstResponder

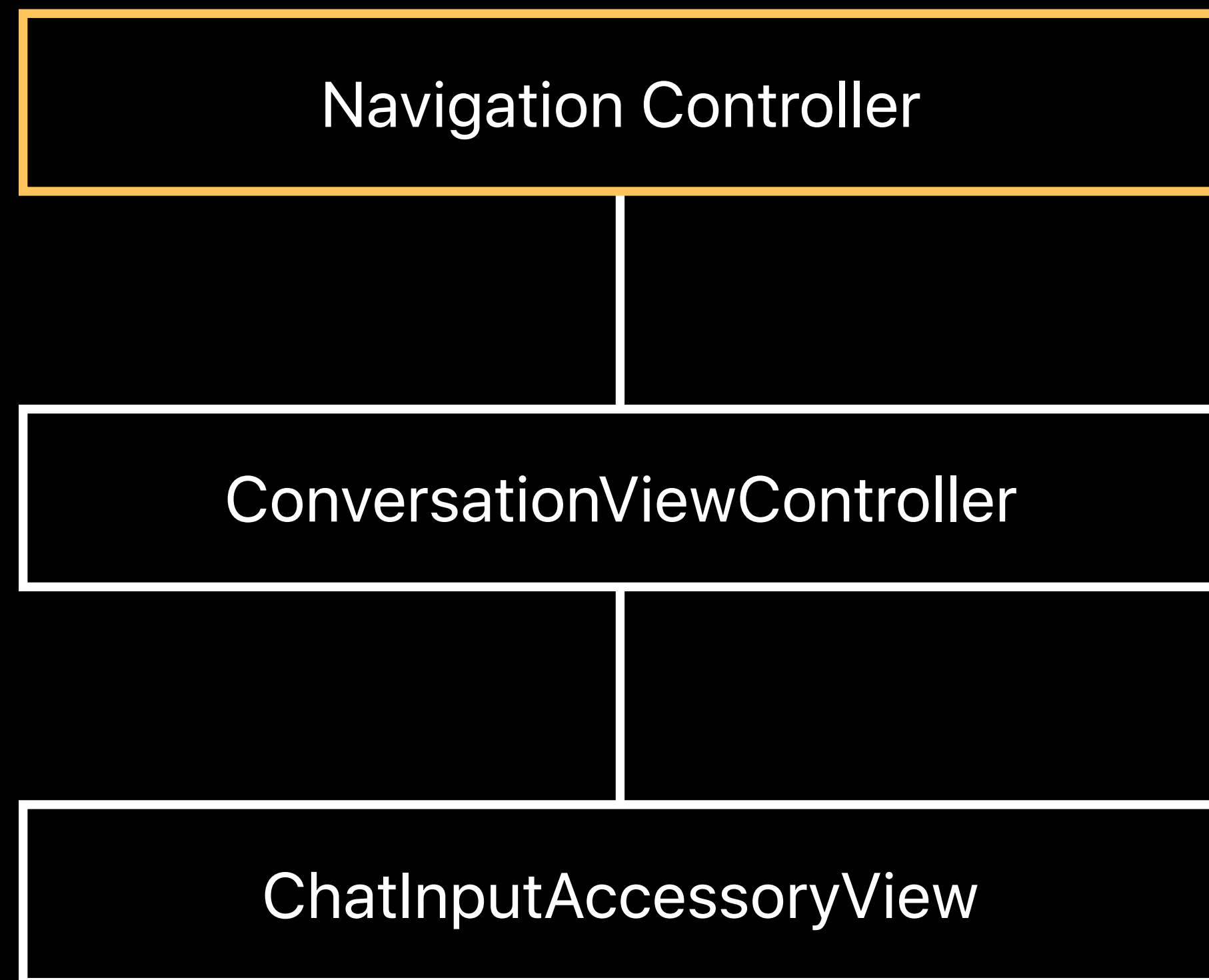


ChatInputAccessoryView

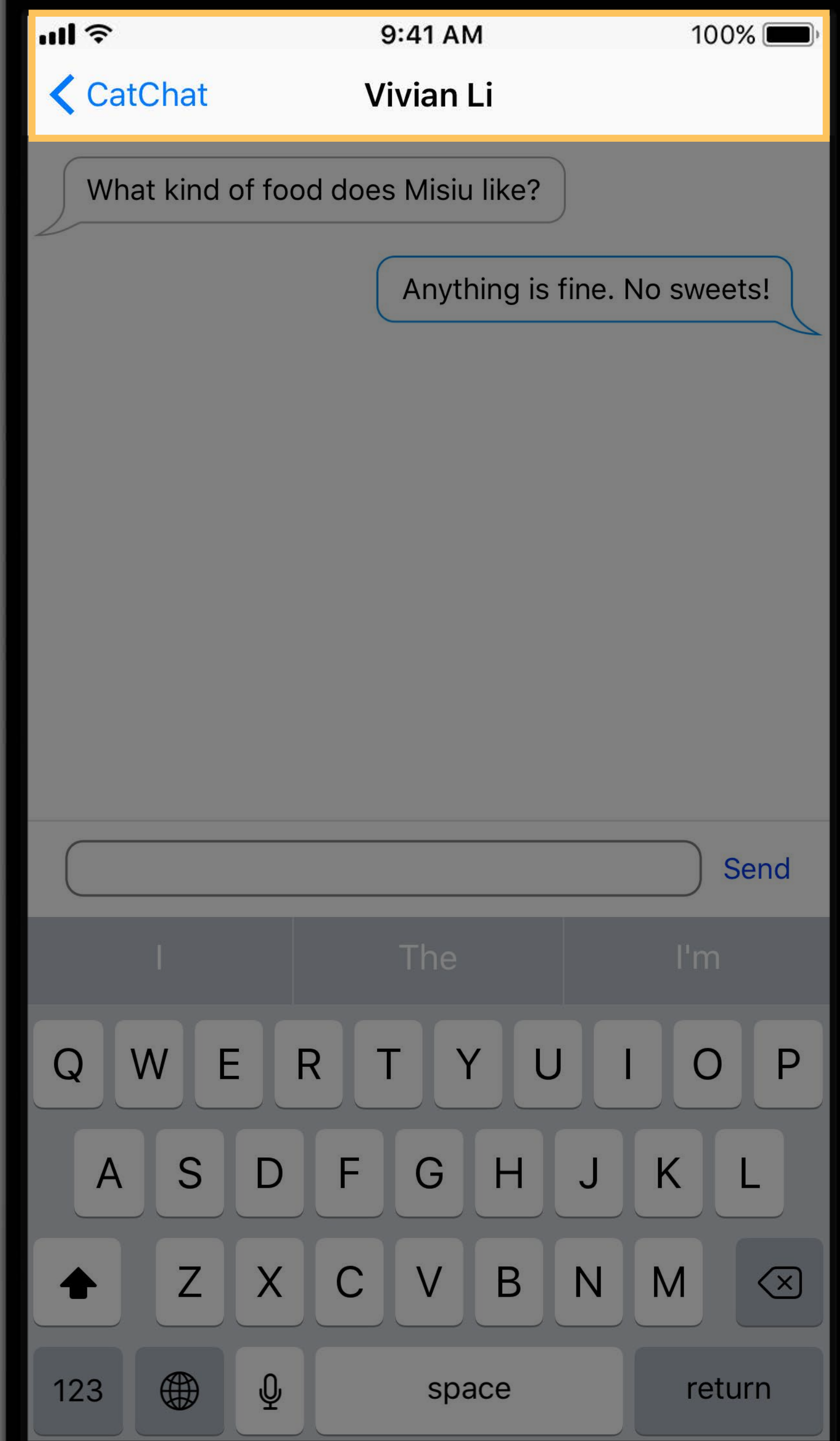
becomesFirstResponder

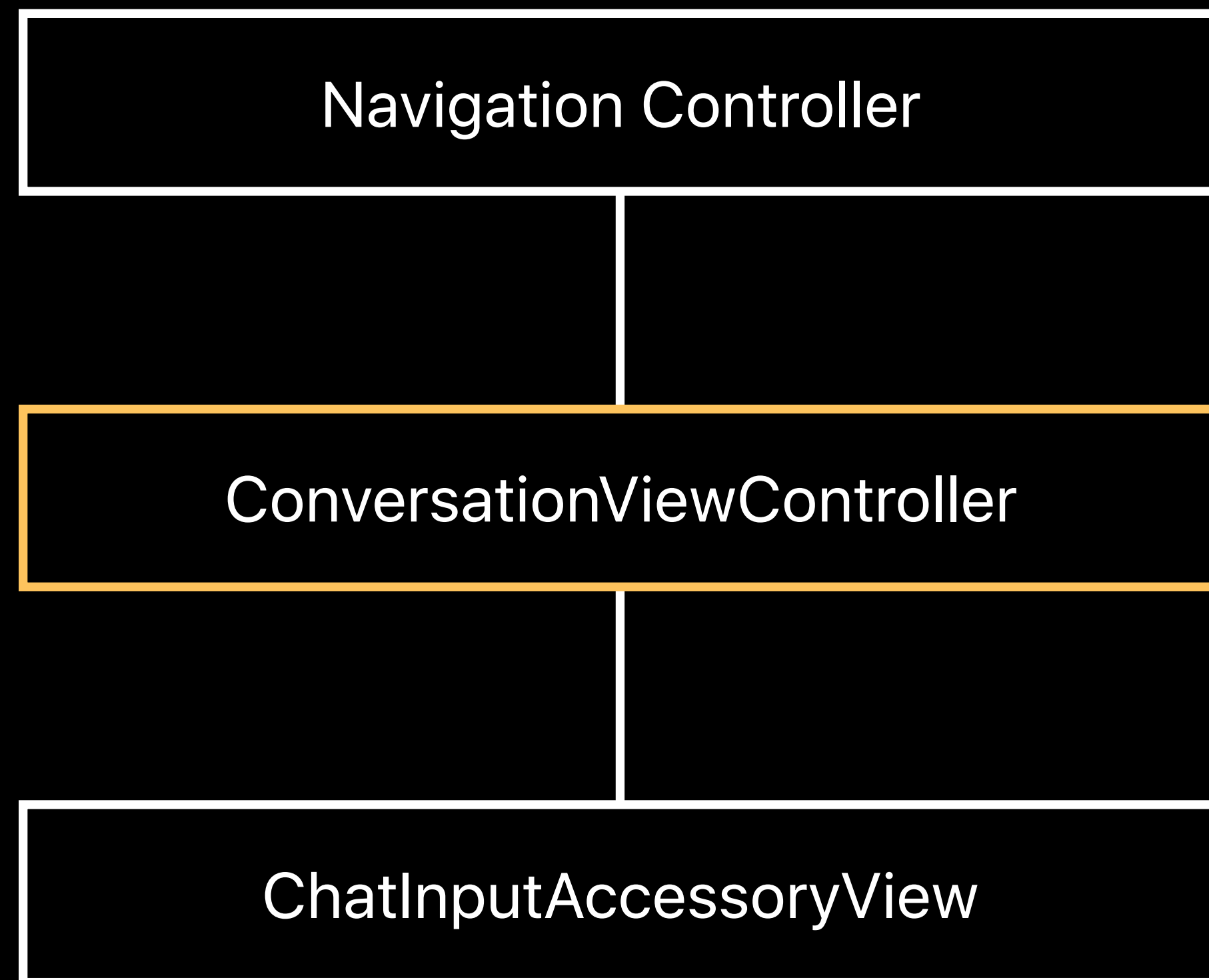




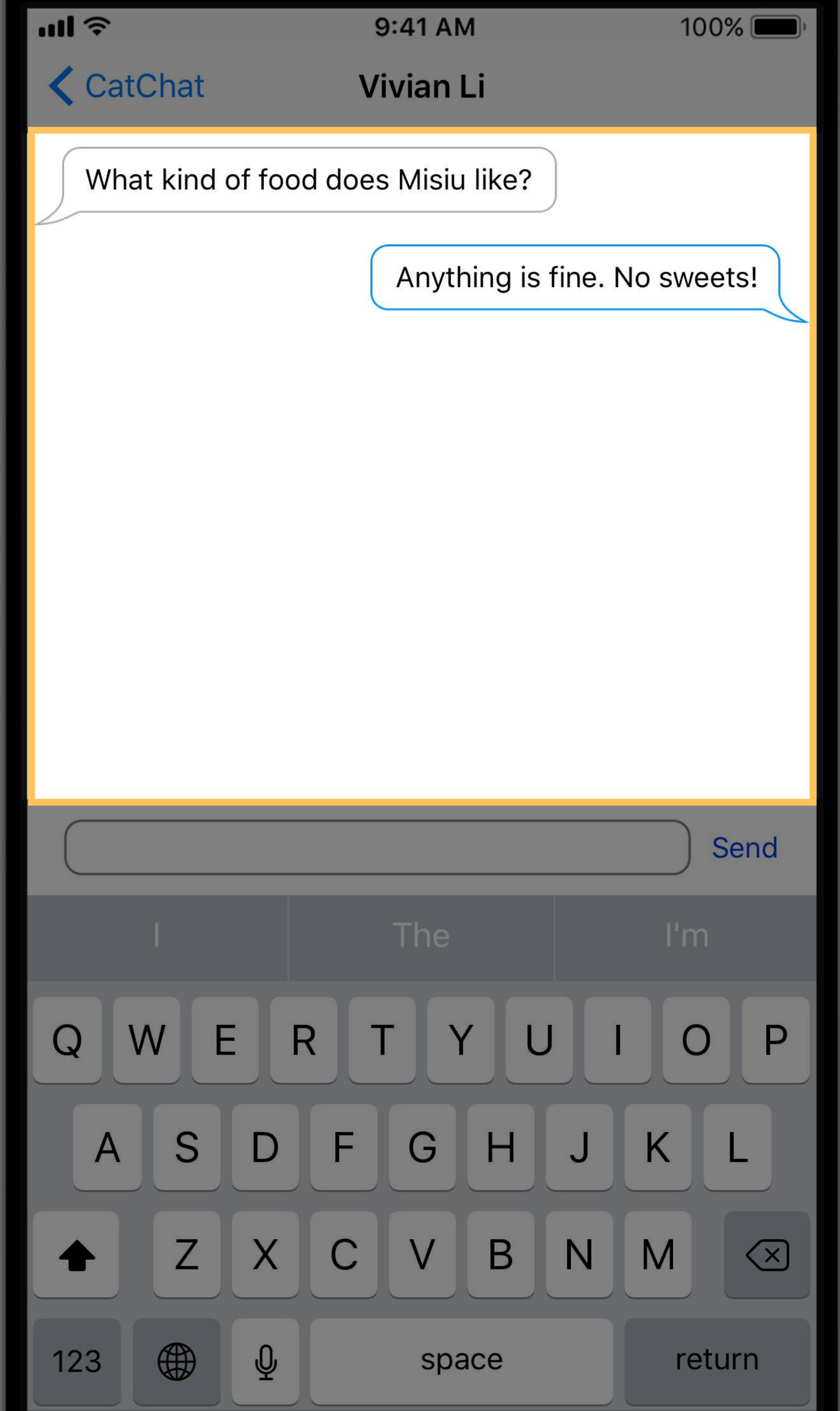


becomesFirstResponder





becomesFirstResponder



Text Input Context Identifier

```
class ConversationViewController: UITableViewController, UITextViewDelegate {  
  
    // ... other code ...  
  
    override var textInputContextIdentifier: String? {  
        // Returning some unique identifier here allows the keyboard to remember  
        // which language the user was typing in when they were last communicating  
        // with this person.  
  
        // It can be anything, as long as it's unique to each  
        // recipient (here we're just returning the name)  
        return self.conversation?.otherParticipant  
    }  
  
    // ... other code ...  
  
}
```

Text Input Context Identifier

```
class ConversationViewController: UITableViewController, UITextViewDelegate {  
  
    // ... other code ...  
  
    override var textInputContextIdentifier: String? {  
        // Returning some unique identifier here allows the keyboard to remember  
        // which language the user was typing in when they were last communicating  
        // with this person.  
  
        // It can be anything, as long as it's unique to each  
        // recipient (here we're just returning the name)  
        return self.conversation?.otherParticipant  
    }  
  
    // ... other code ...  
  
}
```

Demo

Give your app a memory

Kasia Wawer, iOS Keyboards

UIResponder to Remember Keyboard

textInputContextIdentifier

UITextInputMode

Multilingual

Being aware of context

The new “smarts”

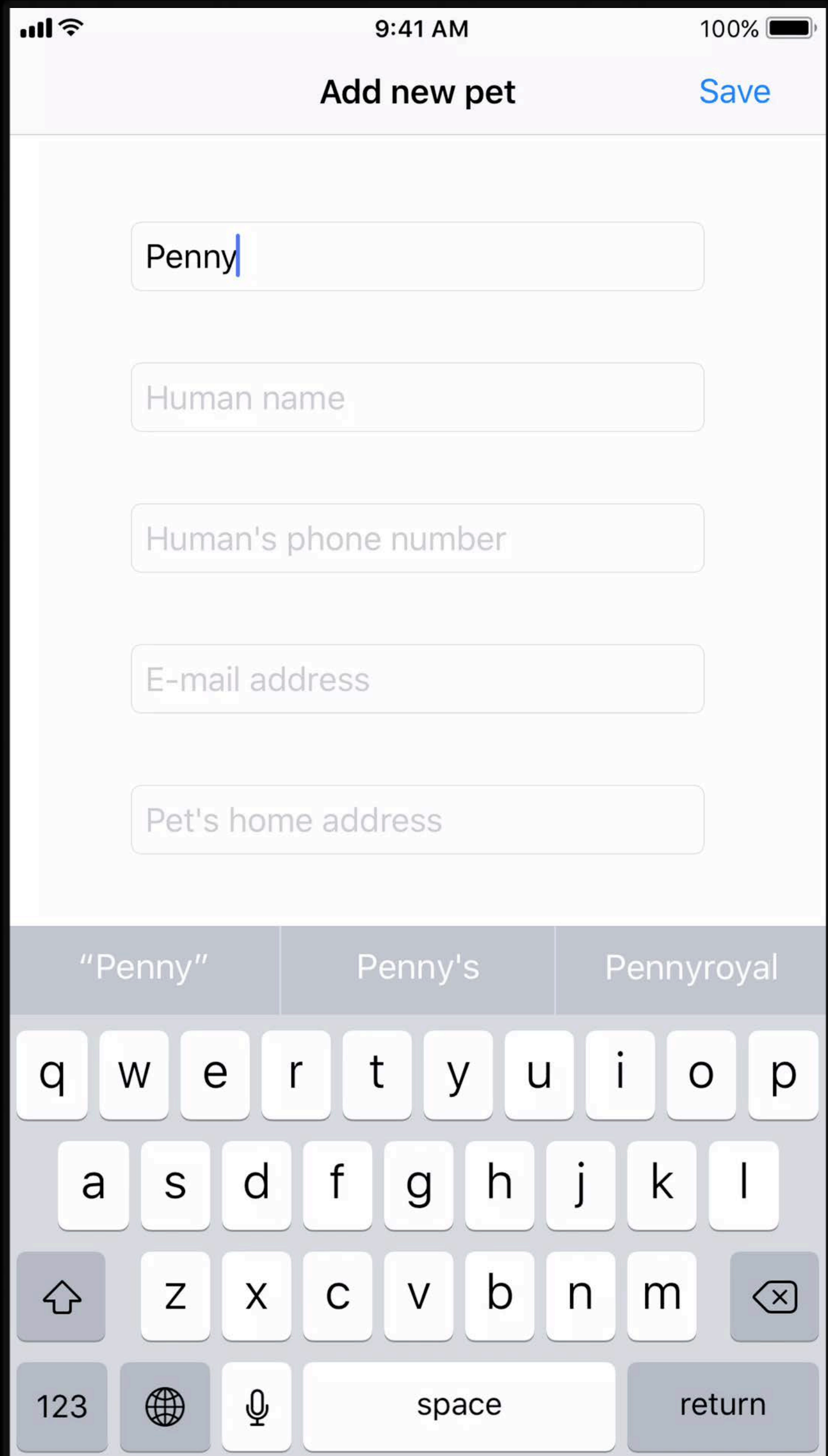
Marked text

Hardware keyboard

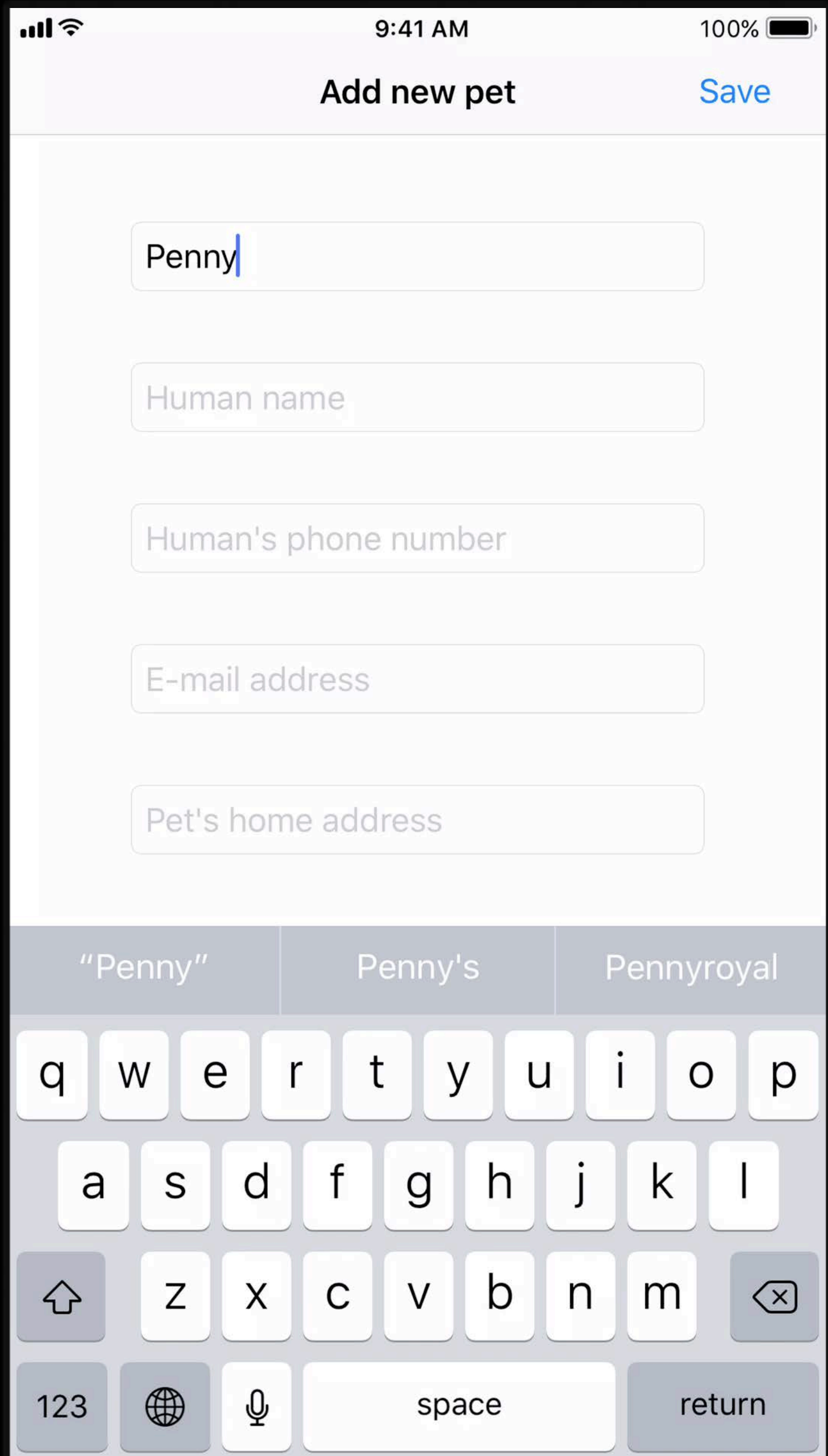
Being Aware of Context

UIKeyboardType

UITextContentTypes



QuickType personal information



QuickType personal information



9:41 AM

100%

Add new pet

Save

Penny

John Appleseed

(505) 555-0155

j.appleseed@icloud.com

Pet's home address

to

and

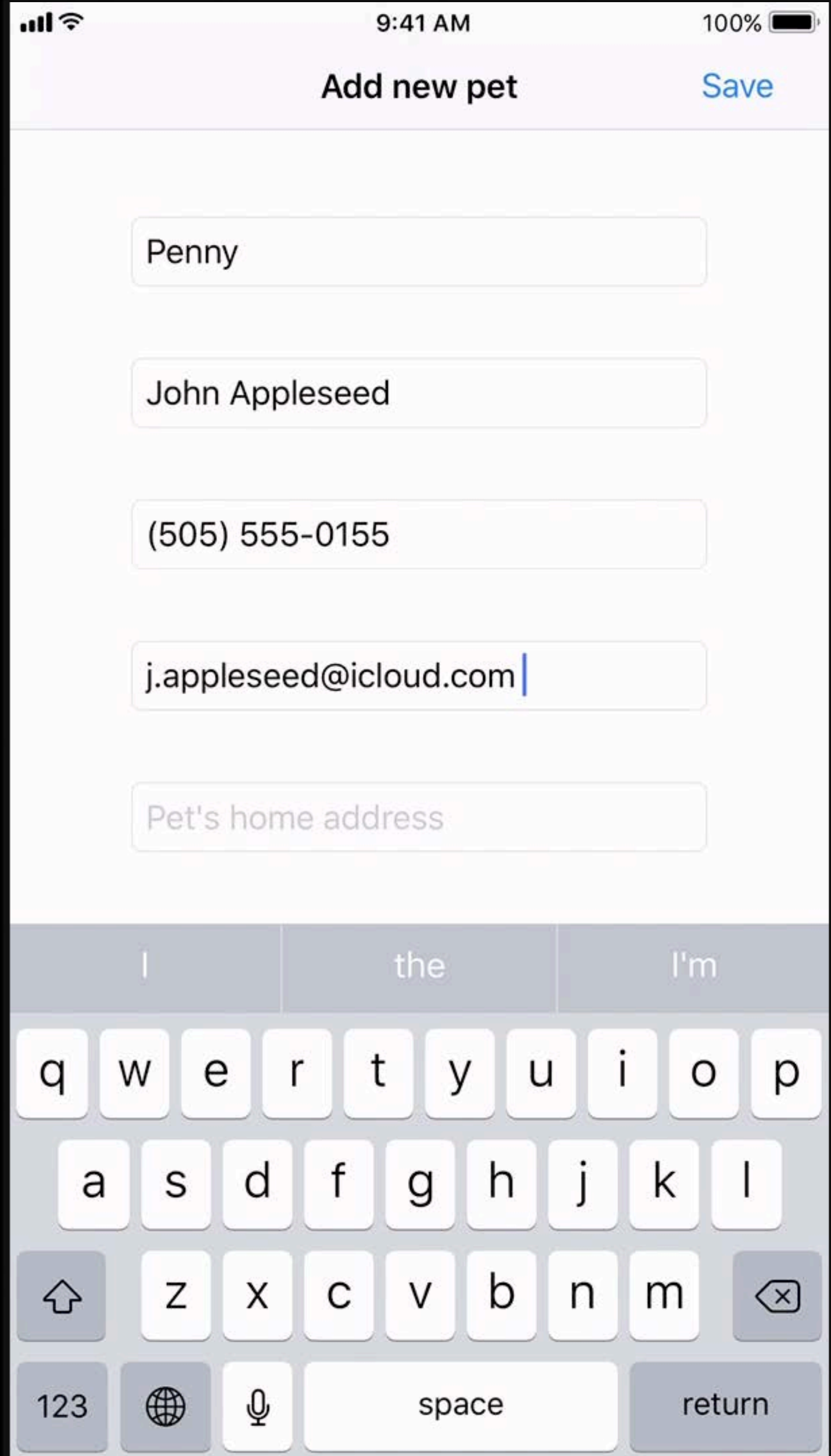
a

q w e r t y u i o p

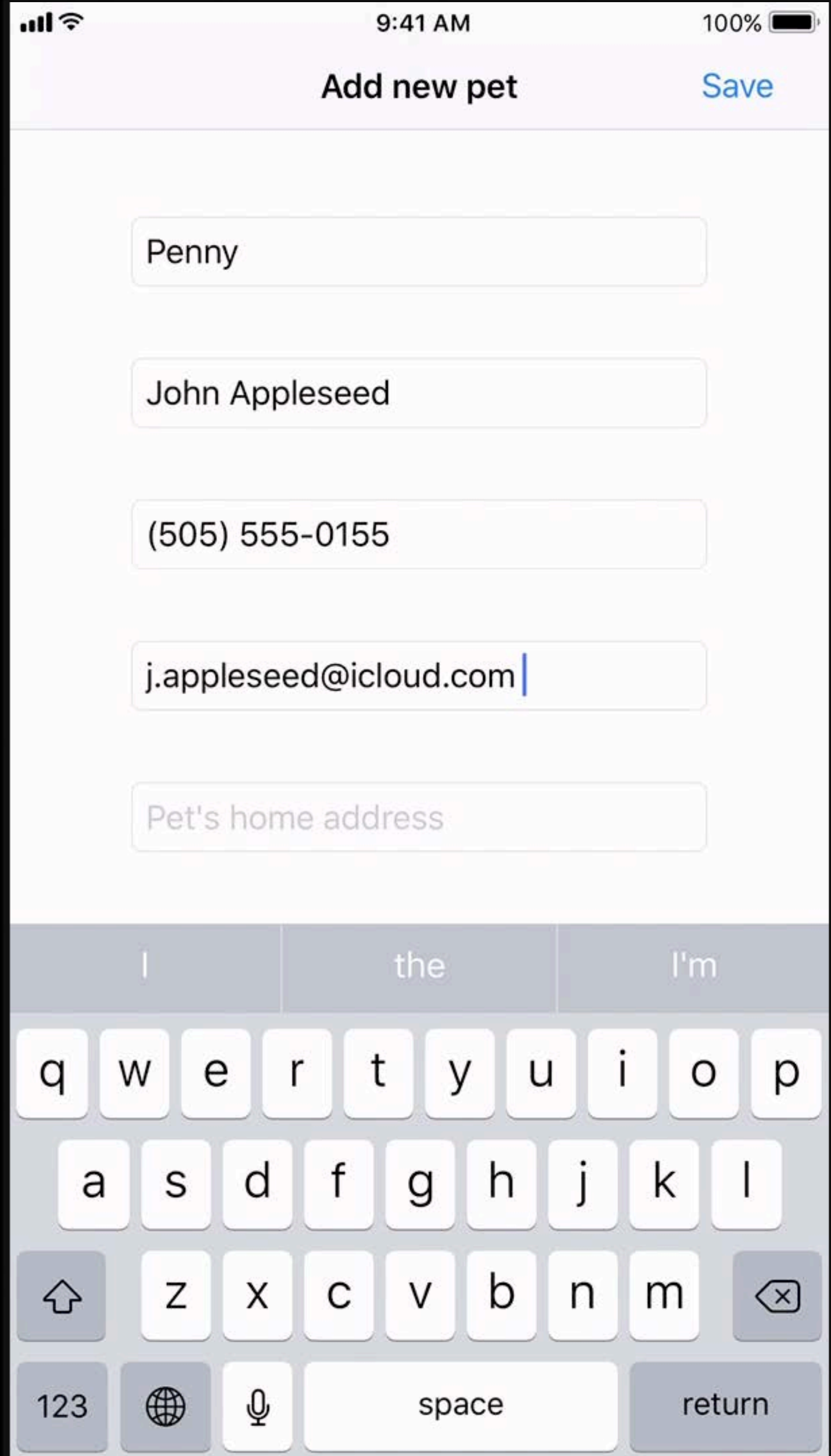
a s d f g h j k l

↑ z x c v b n m ↵

123 space return



QuickType address



QuickType address



9:41 AM

100%

Add new pet

Save

Penny

John Appleseed

(505) 555-0155

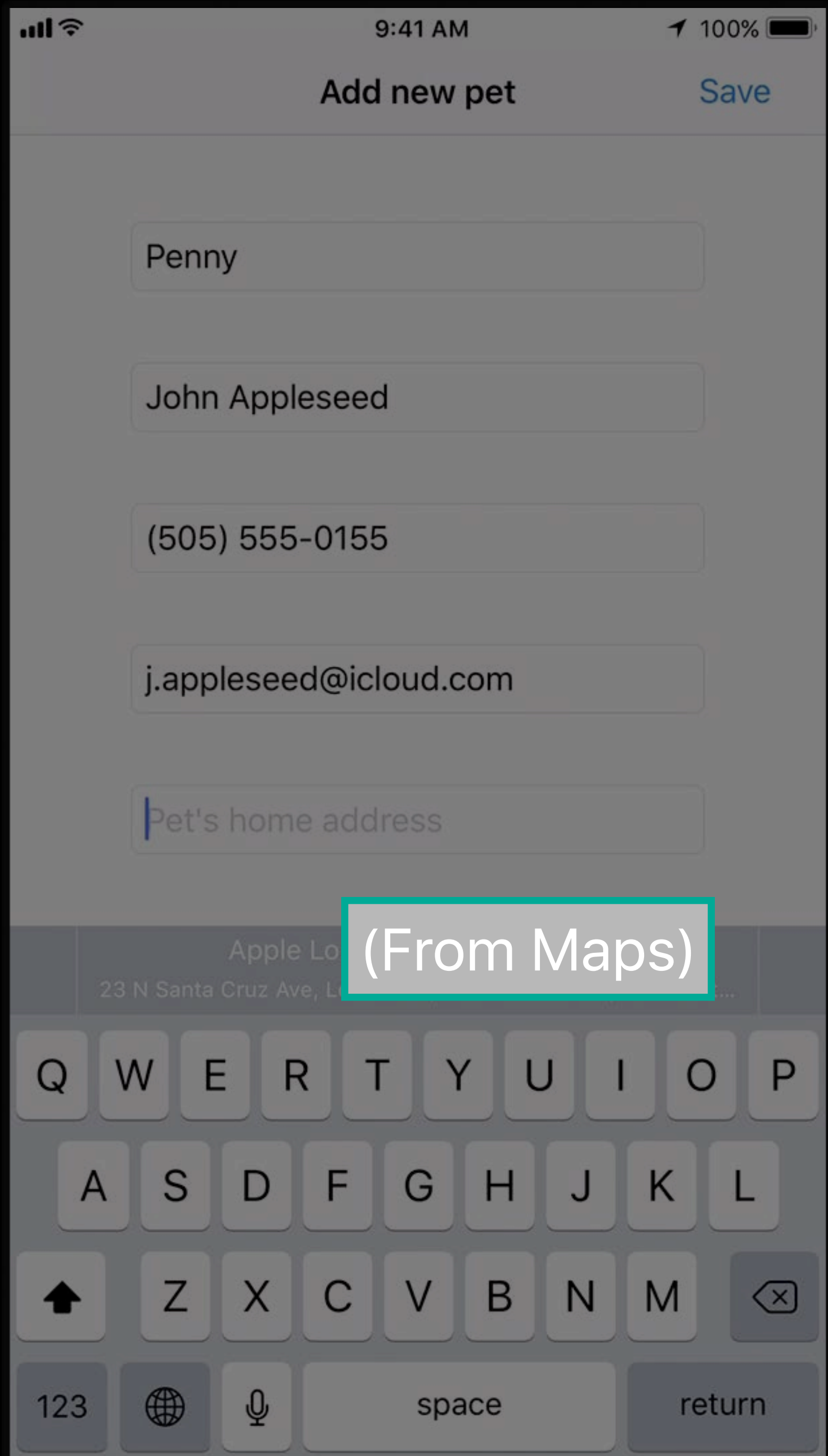
j.appleseed@icloud.com

Pet's home address

Apple Los Gatos (From Maps)

23 N Santa Cruz Ave, Los Gatos, CA 95030-5916, United St...





QuickType sources

UITextContentType

Provides contextual predictions

Displays on QuickType bar



9:41 AM

100%

shiny.

Username

Password

Log In

password for shiny.example.com
rmondello





9:41 AM

100%

shiny.

Username

Password

Log In



Content Types for Password AutoFill

NEW

`UITextContentTypeUsername`

`UITextContentTypePassword`

Log In

Multilingual

Being aware of context

The new "smarts"

Marked text

Hardware keyboard

Smart Quote and Smart Dash

NEW

Smart Quote and Smart Dash

NEW

SF Hello	"a"
Helvetica Neue	"a"
Lucida Grande	"a"
Avenir	"a"
Myriad Set	"a"

Smart Quote and Smart Dash

NEW

SF Hello	"a"	"a"
Helvetica Neue	"a"	“a”
Lucida Grande	"a"	“a”
Avenir	"a"	"a"
Myriad Set	"a"	"a"

Smart Quote and Smart Dash

NEW

SF Hello	"a"	“a”
Helvetica Neue	"a"	“a”
Lucida Grande	"a"	“a”
Avenir	"a"	“a”
Myriad Set	"a"	“a”

Hyphen: 1-dash

- → -

Smart Quote and Smart Dash

NEW

SF Hello	"a"	“a”
Helvetica Neue	"a"	“a”
Lucida Grande	"a"	“a”
Avenir	"a"	“a”
Myriad Set	"a"	“a”

Hyphen: 1-dash

- → -

En dash: 2-dash

- - → -

Smart Quote and Smart Dash

NEW

SF Hello	"a"	“a”
Helvetica Neue	"a"	“a”
Lucida Grande	"a"	“a”
Avenir	"a"	“a”
Myriad Set	"a"	“a”

Hyphen: 1-dash

- → -

En dash: 2-dash

- - → -

Em dash: 3-dash

- - - → —

Smart Insertion and Deletion

Vivian will bring Misiu to the party.

Smart Insertion and Deletion

Vivian will bring Misiu|to the party.

Smart Insertion and Deletion

Vivian will bring Misiu | to the party.

Smart Insertion and Deletion

Vivian will bring Misiu|and Jiwang|to the party.

Smart Insertion and Deletion

Vivian will bring Misiu and Jiwang to the party.

Smart Insertion and Deletion

Vivian will bring Misiu|and Jiwang|to the party.

Smart Insertion and Deletion

Vivian will bring Misiu|to the party.

Smart Insertion and Deletion

Vivian will bring Misiu to the party.

UITextInputTraits

UITextInputTraits

```
.default  
.yes  
.no
```

UITextInputTraits

```
.default  
.yes  
.no
```

Understand your text entry

Multilingual

Being aware of context

The new “smarts”

Marked text

Hardware keyboard



Marked text search

Marked Text

Internationalization Best Practices

WWDC 2016

Giving text widget a memory

Being aware of context

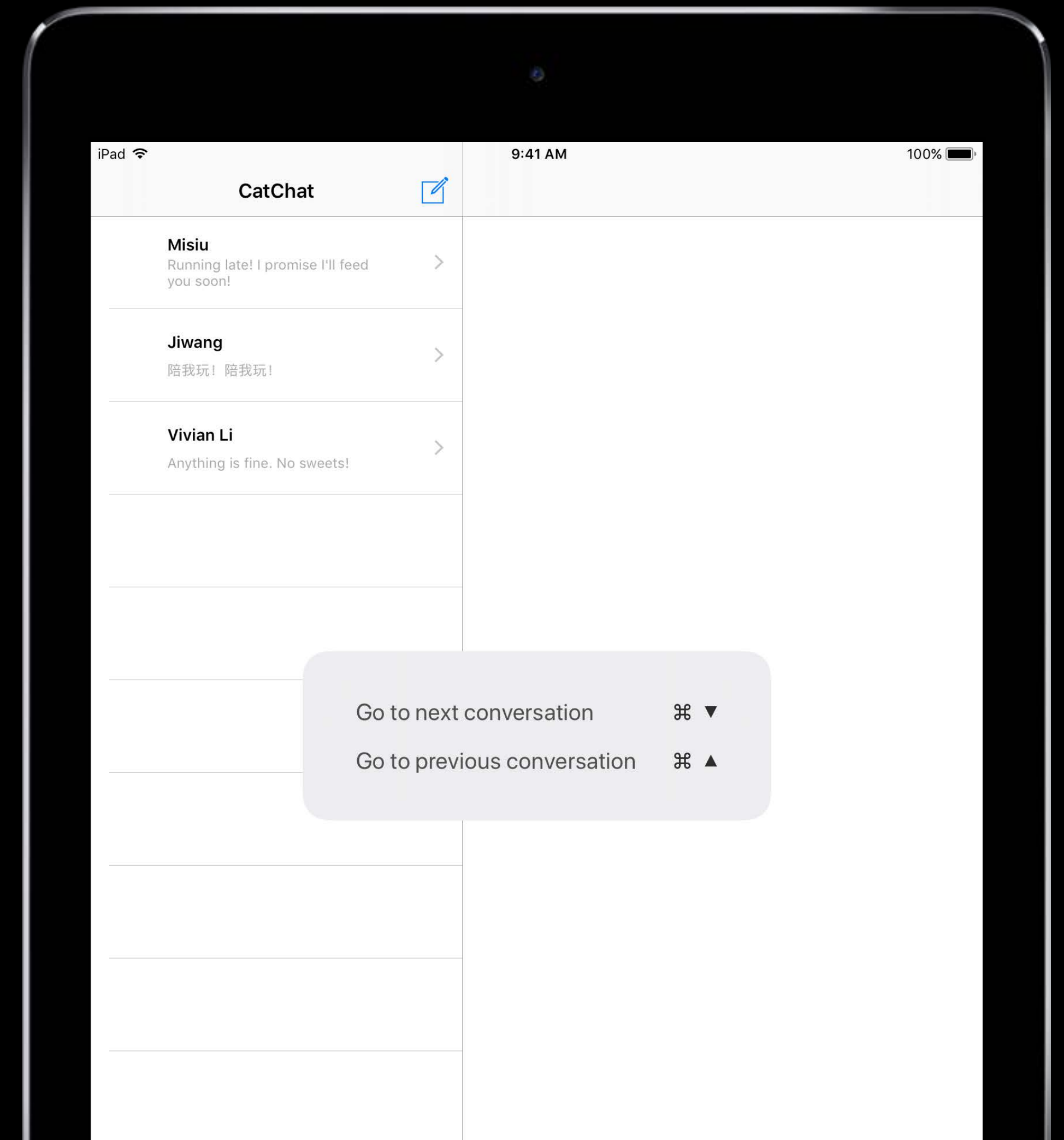
The new "smarts"

Marked text

Hardware keyboard

Working with Hardware Keyboard

Key commands for hardware keyboards



```
// UIKeyCommand

class ConversationViewController: UITableViewController, UITextViewDelegate {

    // ... some code ...

    override var keyCommands: [UIKeyCommand]? {
        return [
            // Command + Down arrow goes to the next conversation
            UIKeyCommand(input: UIKeyInputDownArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_NEXT_CONVERSATION", comment: "")),

            // Command + Up arrow goes to the previous conversation
            UIKeyCommand(input: UIKeyInputUpArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_PREV_CONVERSATION", comment: ""))
        ]
    }

    //... some code ...

}
```

```
// UIKeyCommand

class ConversationViewController: UITableViewController, UITextViewDelegate {

    // ... some code ...

    override var keyCommands: [UIKeyCommand]? {
        return [
            // Command + Down arrow goes to the next conversation
            UIKeyCommand(input: UIKeyInputDownArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_NEXT_CONVERSATION", comment: "")),

            // Command + Up arrow goes to the previous conversation
            UIKeyCommand(input: UIKeyInputUpArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_PREV_CONVERSATION", comment: ""))
        ]
    }

    //... some code ...

}
```

```
// UIKeyCommand

class ConversationViewController: UITableViewController, UITextViewDelegate {

    // ... some code ...

    override var keyCommands: [UIKeyCommand]? {
        return [
            // Command + Down arrow goes to the next conversation
            UIKeyCommand(input: UIKeyInputDownArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_NEXT_CONVERSATION", comment: "")),

            // Command + Up arrow goes to the previous conversation
            UIKeyCommand(input: UIKeyInputUpArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_PREV_CONVERSATION", comment: ""))
        ]
    }

    //... some code ...

}
```

```
// UIKeyCommand

class ConversationViewController: UITableViewController, UITextViewDelegate {

    // ... some code ...

    override var keyCommands: [UIKeyCommand]? {
        return [
            // Command + Down arrow goes to the next conversation
            UIKeyCommand(input: UIKeyInputDownArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_NEXT_CONVERSATION", comment: "")),

            // Command + Up arrow goes to the previous conversation
            UIKeyCommand(input: UIKeyInputUpArrow,
                modifierFlags: .command,
                action: #selector(switchToConversationKeyCommandInvoked(sender:)),
                discoverabilityTitle: NSLocalizedString("GO_TO_PREV_CONVERSATION", comment: ""))
        ]
    }

    //... some code ...

}
```

CatChat



Misiu

Running late! I promise I'll feed you soon!



Jiwang

陪我玩! 陪我玩!



Vivian Li

Anything is fine. No sweets!



Go to next conversation



Go to previous conversation



Send



|

The

iPad



Multilingual

Being aware of context

The new "smarts"

Marked text

Hardware keyboard

- ✓ Multilingual
- ✓ Being aware of context
- ✓ The new "smarts"
- ✓ Marked text
- ✓ Hardware keyboard

Creating Custom Input Views

Accessible input for cats, dogs, and more

James Magahern, iOS Keyboards



9:41 AM

100%



$$\sum_{i=1}^n 573x + 12i + 4i^2$$

1	2	3	4	5	6	7	8	9	0	.
=	+	-	×	÷	Σ					
()	mc	m+	m-	mr					
2 nd	x ²	x ³	x ^y	e ^x	10 ^x					
1/x	√x	∛x	∛x	ln	log ₁₀					
x!	sin	cos	tan	e	EE					
Rad	sinh	cosh	tanh	π	Rand					

learned up to now. (You might need your **debugging** skills, too!)

```
moveForward()
```

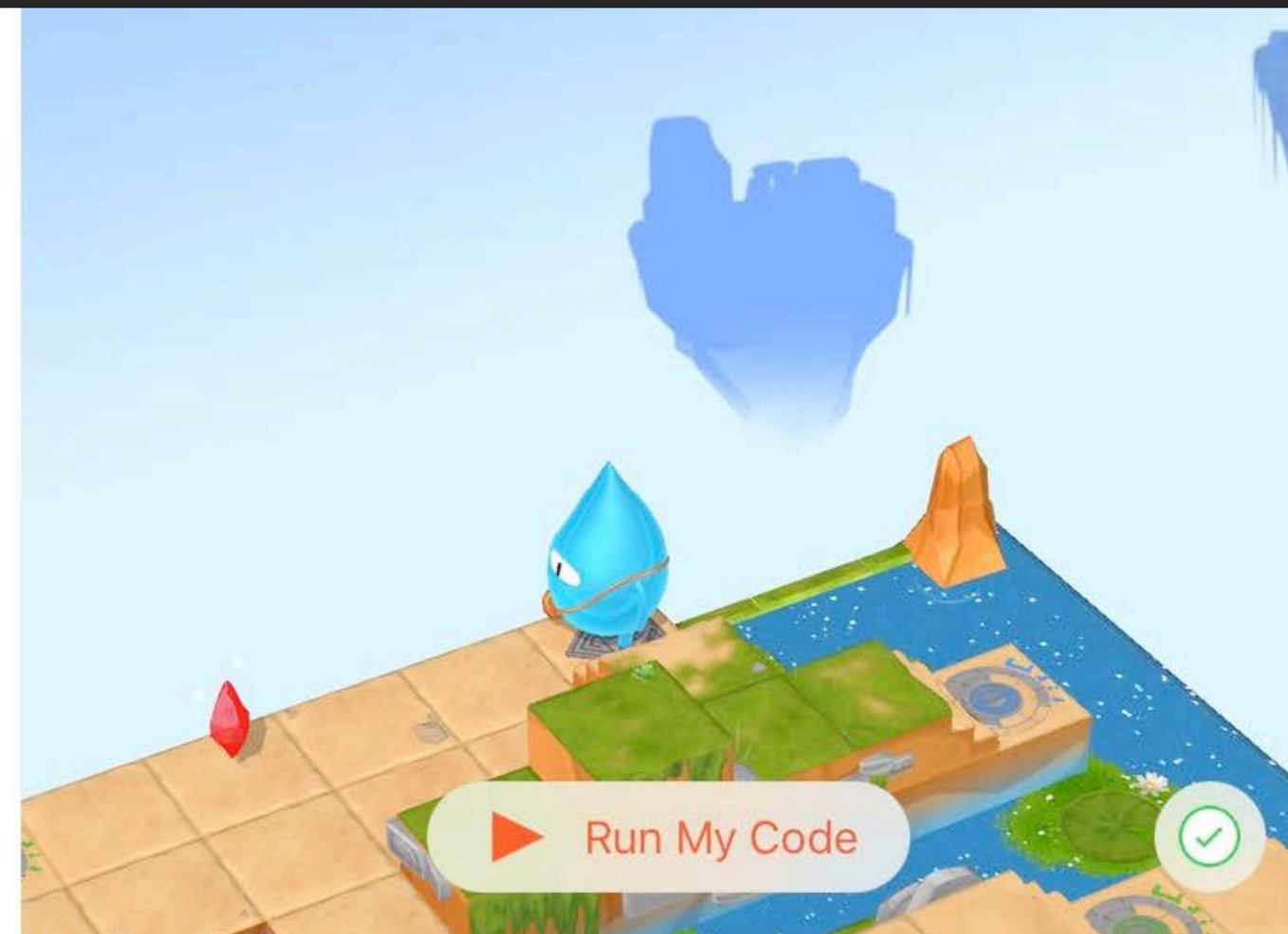
```
moveForward()
```

```
collectGem()
```

```
moveForward()
```

```
moveForward()
```

```
moveForward()
```

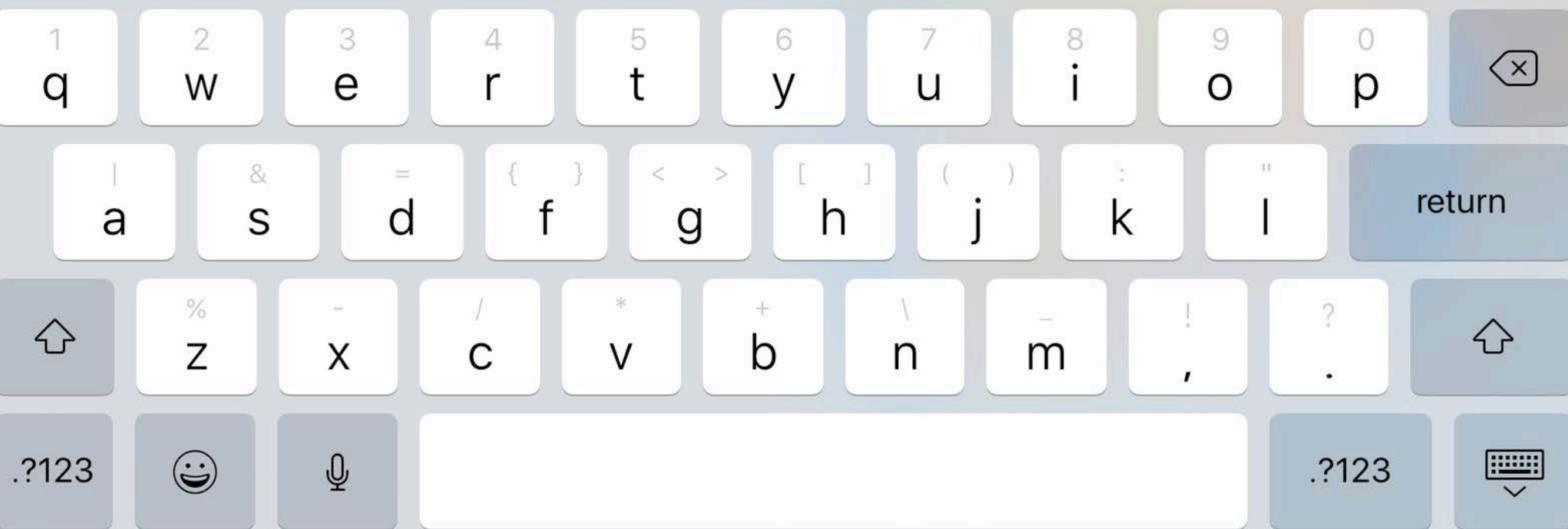


collectGem()

moveForward()

toggleSwitch()

turnLeft()



	B	C
--	---	---

1	Song	Score
2	Monkey Business	999870
3	Blew My Mind	999980
4	Come To Life	999990
5		...
6		
7		

✕ | ✓

fx + - × ÷ () "text" < ⚡

Q W E R T Y U I O P

A S D F G H J K L

↑ Z X C V B N M ↵

123 🌐 🗣️ space return

Custom Accessible Input Views for Pets

Custom Accessible Input Views for Pets

Our pets have no way to talk back

Custom Accessible Input Views for Pets

Our pets have no way to talk back

Cats and dogs have trouble using QWERTY

Custom Accessible Input Views for Pets

Our pets have no way to talk back

Cats and dogs have trouble using QWERTY

Limited vocabulary



9:41 AM

100%


< CatChat

Poochie

Food!

Okay I'll be home in a bit. Please don't eat the furniture!

Send



Food



Outside

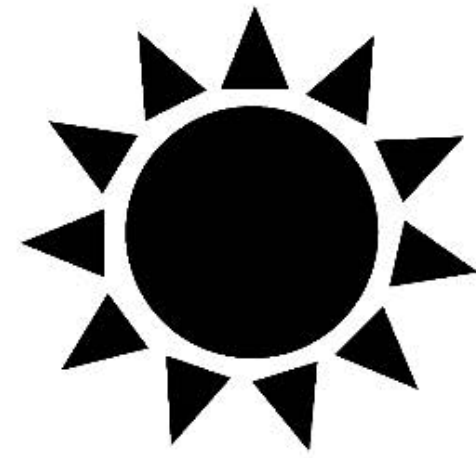


Stop!

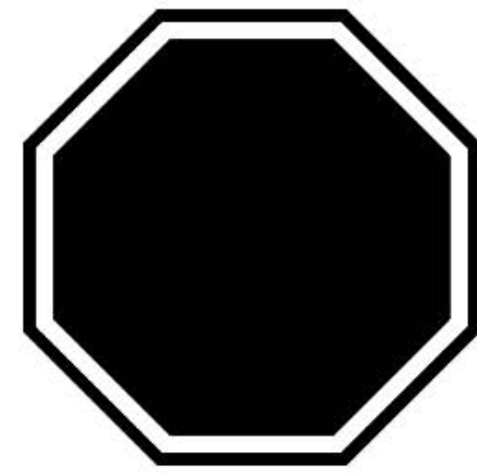




Food



Outside

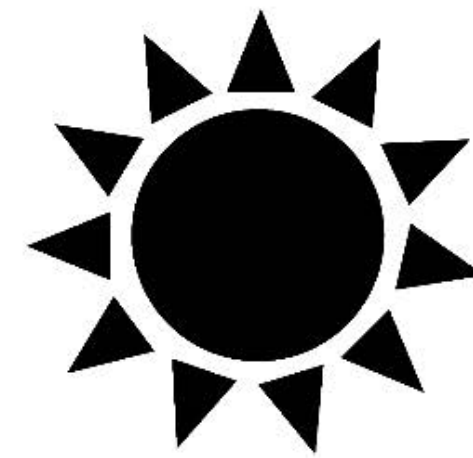


Stop!

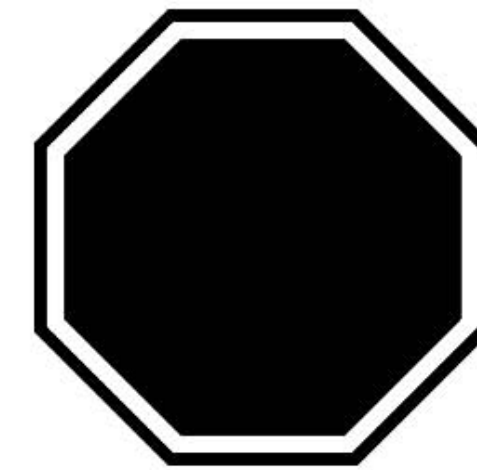




Food



Outside



Stop!



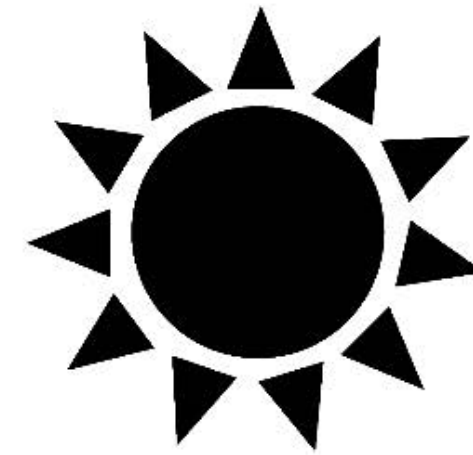
UIInputViewController

UIInputView

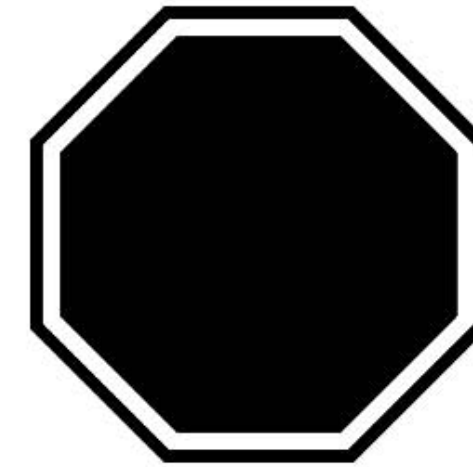
Send



Food



Outside



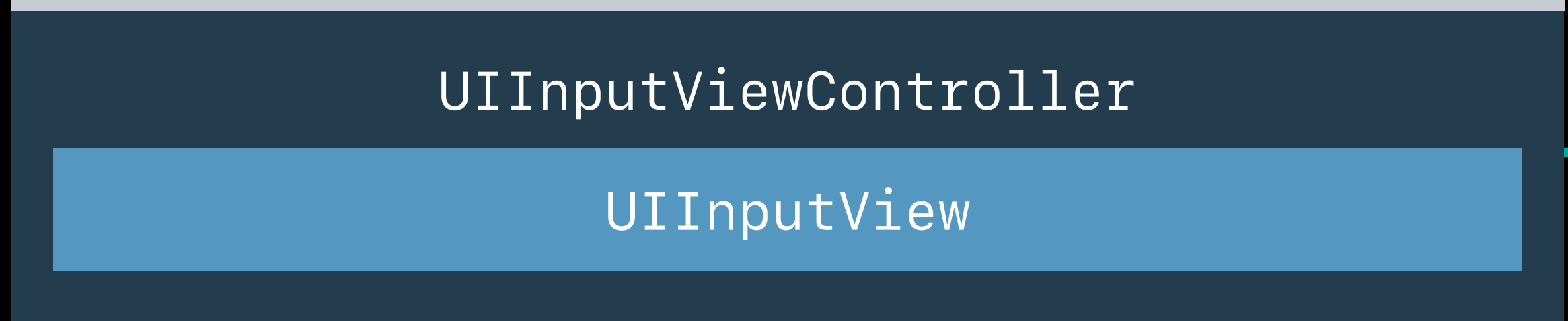
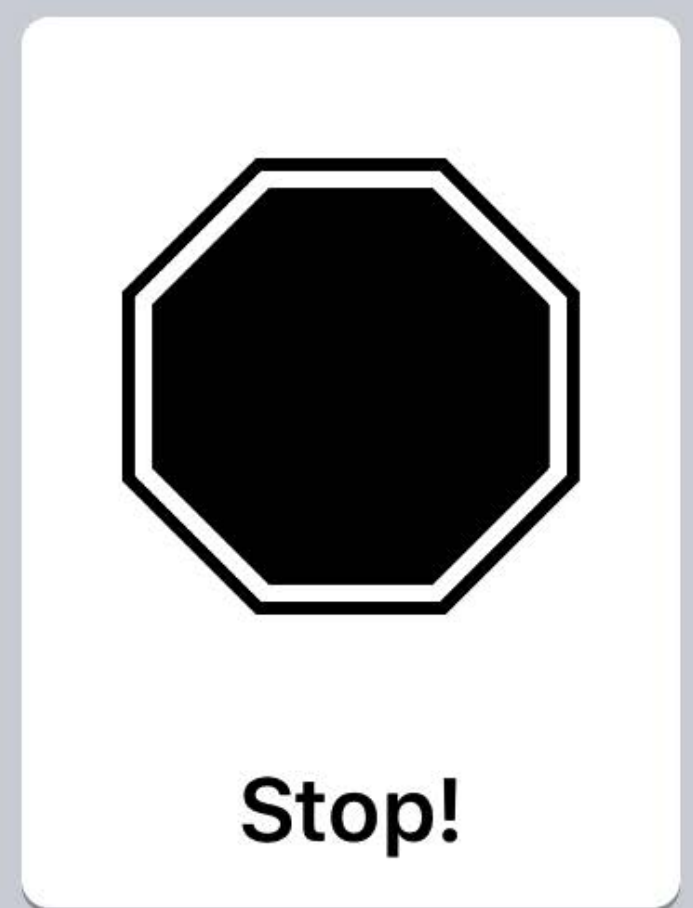
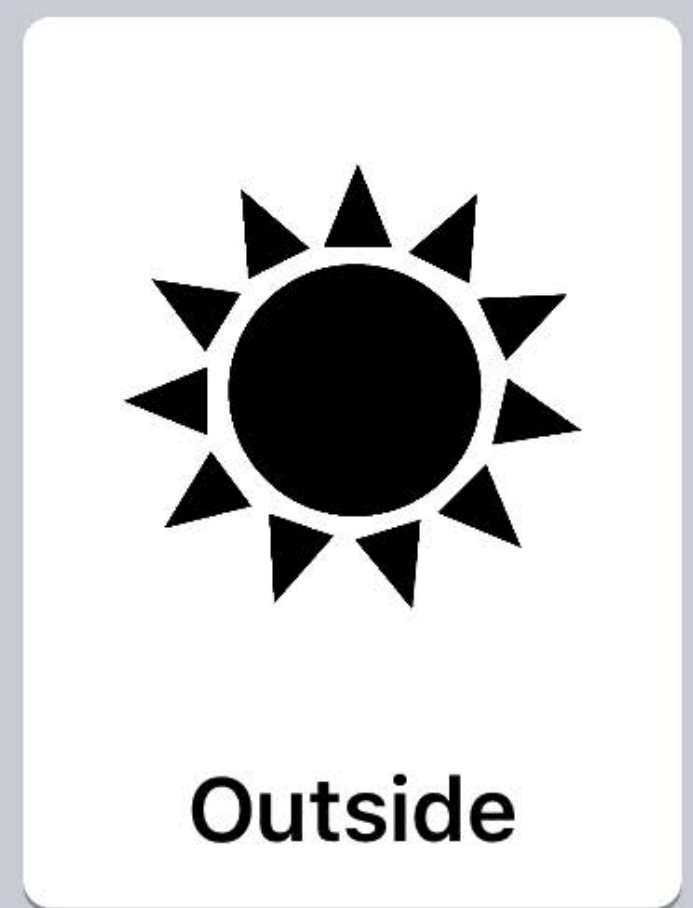
Stop!



UIInputViewController

UIInputView

Send



Providing a Custom Input View

```
class ConversationViewController: UITableViewController, UITextViewDelegate {
```


Providing a Custom Input View

```
class ConversationViewController: UITableViewController, UITextViewDelegate {  
  
    private let customInputView = AnimalInputView()  
  
    override var canBecomeFirstResponder: Bool {  
        return true  
    }  
}
```

Providing a Custom Input View

```
class ConversationViewController: UITableViewController, UITextViewDelegate {  
  
    private let customInputView = AnimalInputView()  
  
    override var canBecomeFirstResponder: Bool {  
        return true  
    }  
  
    override var inputView: UIInputView? {  
        // Return an instance of our custom UIInputView subclass  
        return customInputView  
    }  
  
    // ... other code ...  
}
```



9:41 AM

100%



Kate



iMessage
Today 7:59 PM

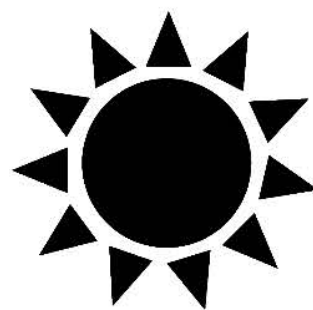
How's my pup?



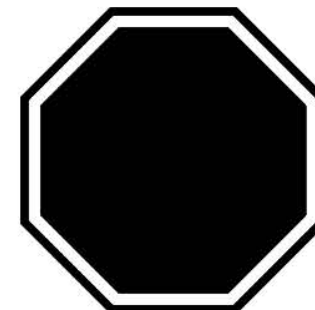
iMessage



Food



Outside



Stop!



Demo

Converting to a keyboard extension

James Magahern, iOS Keyboards

Converting to a Keyboard Extension

Converting to a Keyboard Extension

Create a new target

Converting to a Keyboard Extension

Create a new target

Set up your `UIInputViewController` subclass

Converting to a Keyboard Extension

Create a new target

Set up your `UIInputViewController` subclass

Users can now enable keyboard from your settings bundle



New APIs in iOS 11



NEW

New APIs in iOS 11



Selected text

New APIs in iOS 11

NEW

Selected text

`documentIdentifier` handle

New APIs in iOS 11



NEW

Selected text

`documentIdentifier` handle

Ability to query for full access

Other Third Party Keyboard APIs

Other Third Party Keyboard APIs

Incorporate the system input menu



Other Third Party Keyboard APIs

Incorporate the system input menu

Personalization with the supplementary lexicon



Other Third Party Keyboard APIs

Incorporate the system input menu

Personalization with the supplementary lexicon

Multilingual support



Designing for User Trust

Designing for User Trust

Privacy

Designing for User Trust

Privacy

Enhance with user data

Designing for User Trust

Privacy

Enhance with user data

Requesting full access

Full Access and Privacy

Full Access and Privacy

Value in not asking for full access

Full Access and Privacy

Value in not asking for full access

Communicating with your main app

Full Access and Privacy

Value in not asking for full access

Communicating with your main app

Networking

Full Access and Privacy

Value in not asking for full access

Communicating with your main app

Networking

Current location

Full Access and Privacy

Value in not asking for full access

Communicating with your main app

Networking

Current location

Address book

Full Access and Privacy

Value in not asking for full access

Communicating with your main app

Networking

Current location

Address book

Keyboard needs to work without it

Summary

Design your app with the keyboard in mind

Use advanced traits to enhance the user's experience

Building keyboard extensions is a lot easier than you think

More Information

<https://developer.apple.com/wwdc17/242>

Related Sessions

Introducing Password AutoFill for Apps

WWDC 2017

Localizing for Xcode 9

WWDC 2017

Increase Usage of Your App With Proactive Suggestions

WWDC 2016

Internationalization Best Practices

WWDC 2016

Mysteries of Auto Layout, Part 1

WWDC 2015

Labs

Cocoa Touch and Haptics Lab

Technology Lab C

Fri 12:00PM–1:50PM

