Making the Most of Multi-Touch on iOS

Know the system. Customize. Get great results.

Session 118

Ken Kocienda
Principal Engineer, iOS Apps and Frameworks

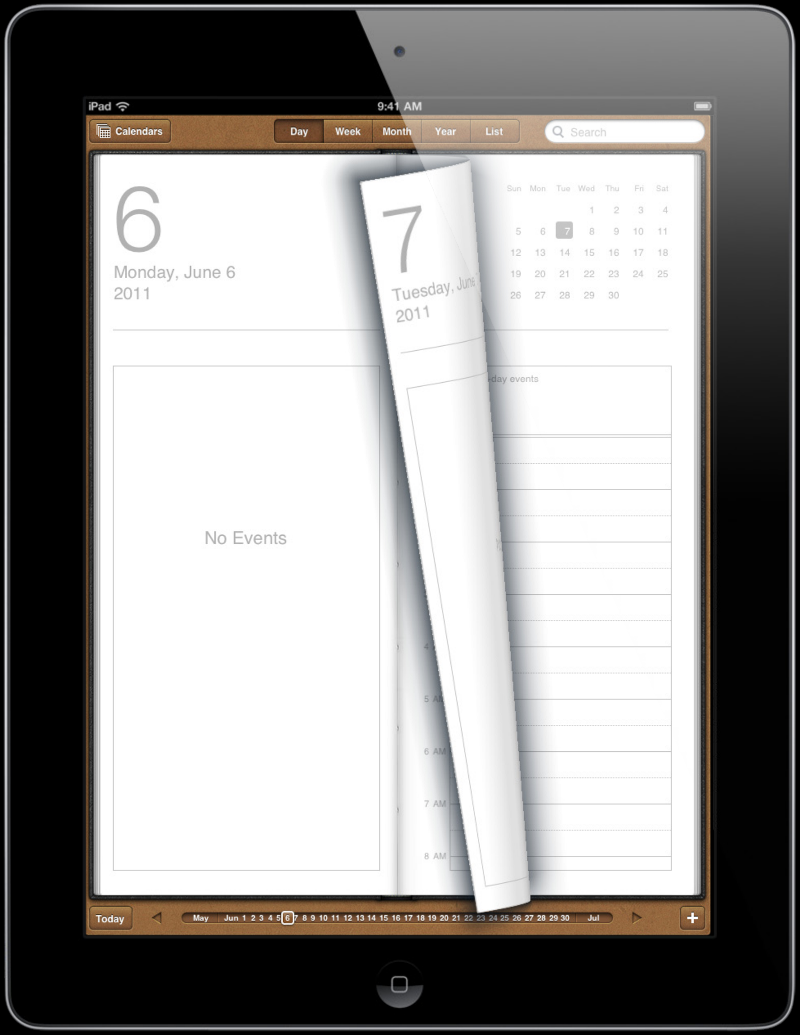
These are confidential sessions—please refrain from streaming, blogging, or taking pictures

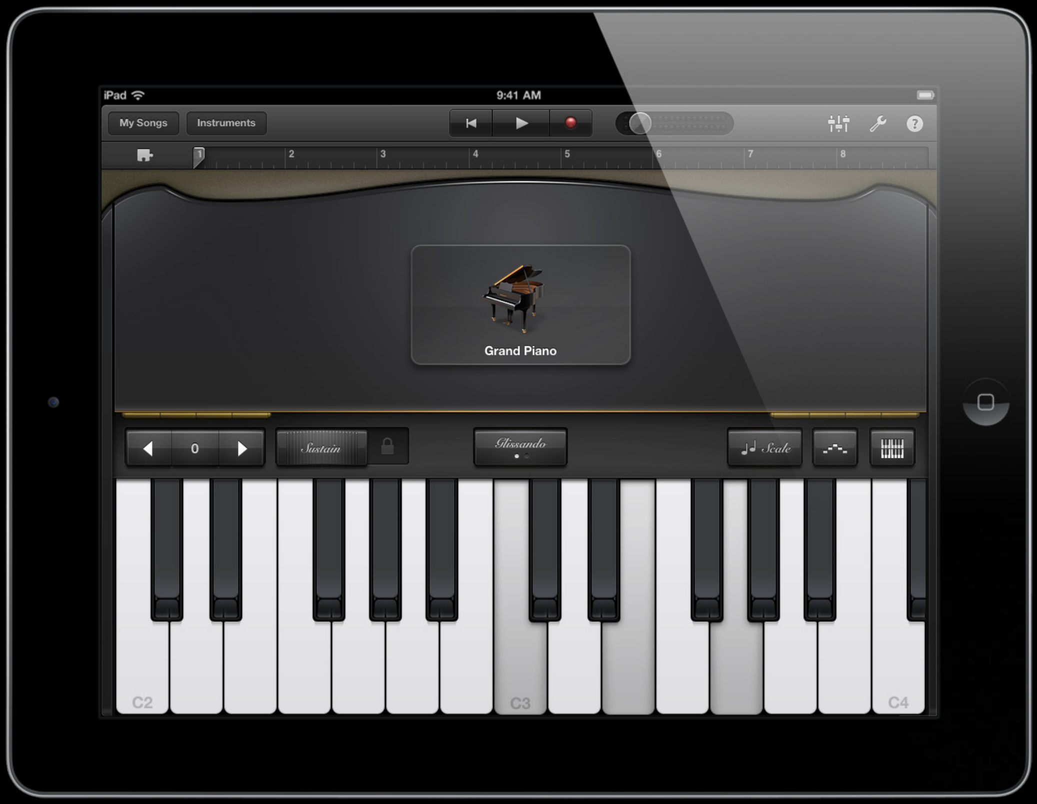
iOS Multi-Touch

Direct manipulation

and fun!

Natural ^ interaction model


iPad
9:41 AM
Calendars
Day
Week
Month
Year
List
Search
6
Monday, June 6
2011
No Events
7
Tuesday, June
2011
Sun Mon Tue Wed Thu Fri Sat
day events
Today
May
Jun
Jul
+


iPad
9:41 AM
My Songs
Instruments
1
2
3
4
5
6
7
8
Grand Piano
0
Sustain
Glissando
Scale
C2
C3
C4

iPad
9:41 AM

Talk about Multi-Touch in your apps

Help you direct your effort

Session Overview

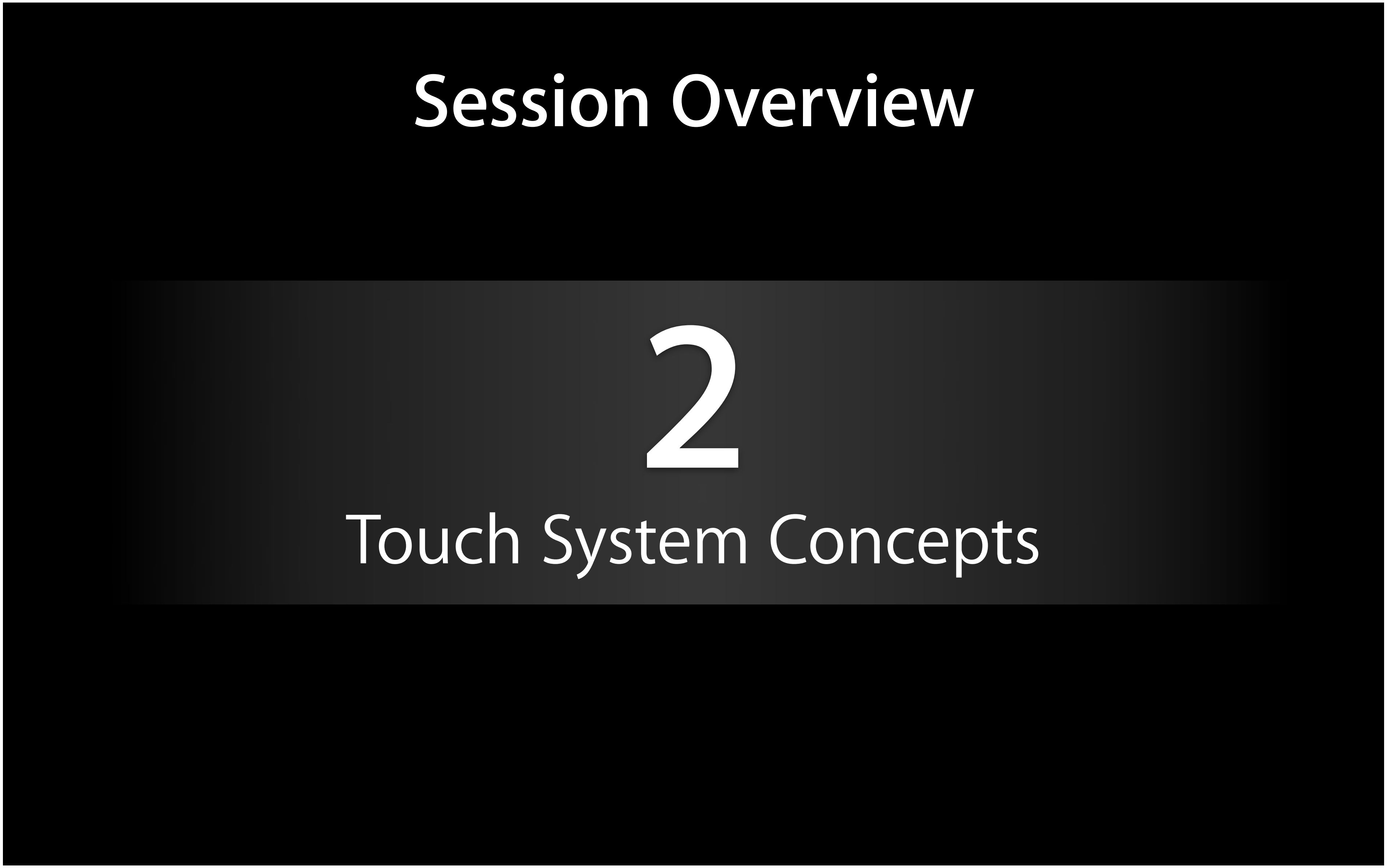
4

Big Ideas

Session Overview

1

Multi-Touch Strategies



Session Overview

2

Touch System Concepts

Session Overview

3

Touch System Tasks

Session Overview

4

Interacting with the Rest of iOS

Related Sessions from 2010

Gesture Recognition	iTunes U: Session 120
Advanced Gesture Recognition	iTunes U: Session 121

4

Big Ideas

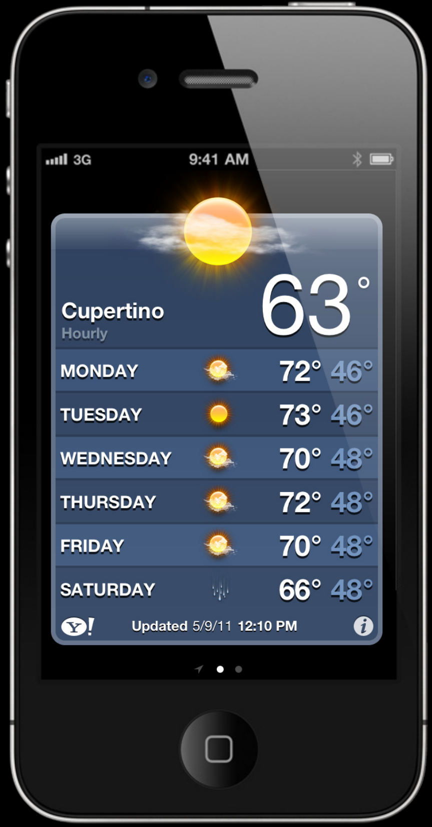
1

Multi-Touch Strategies

Four general approaches

Multi-Touch Strategy

Ignore multiple touches



Cupertino
Hourly

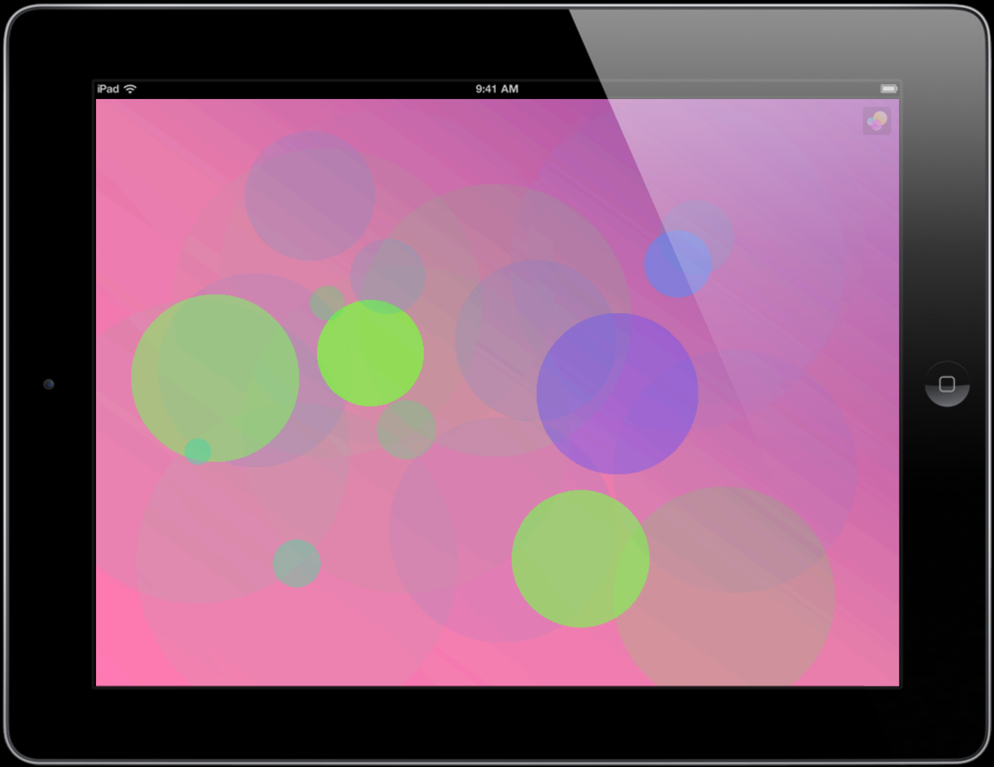
63°

MONDAY	72°	46°
TUESDAY	73°	46°
WEDNESDAY	70°	48°
THURSDAY	72°	48°
FRIDAY	70°	48°
SATURDAY	66°	48°

Updated 5/9/11 12:10 PM

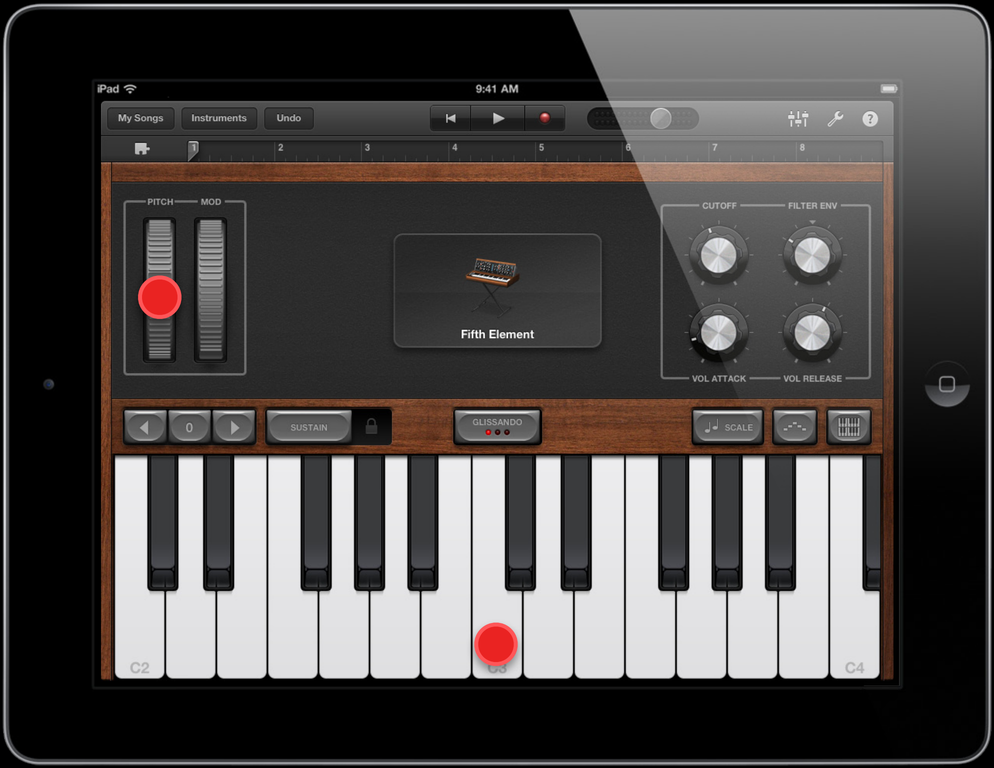
Multi-Touch Strategy

Handle multiple touches independently



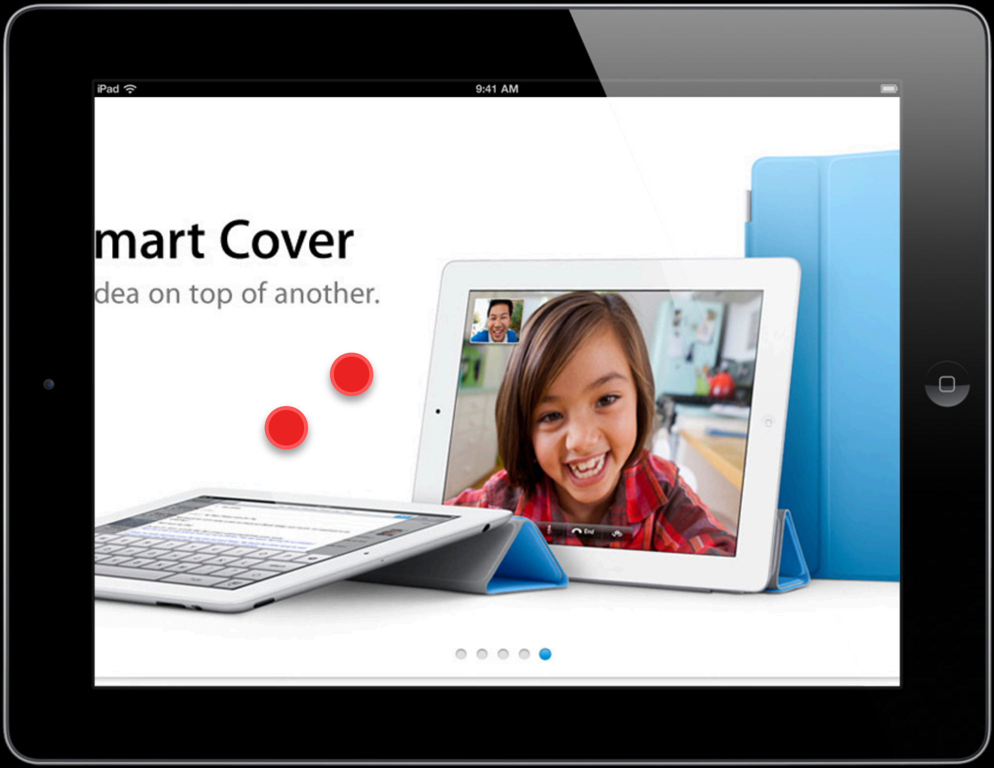
Multi-Touch Strategy

Handle multiple touches together



Multi-Touch Strategy

Handle multiple touches as a gesture



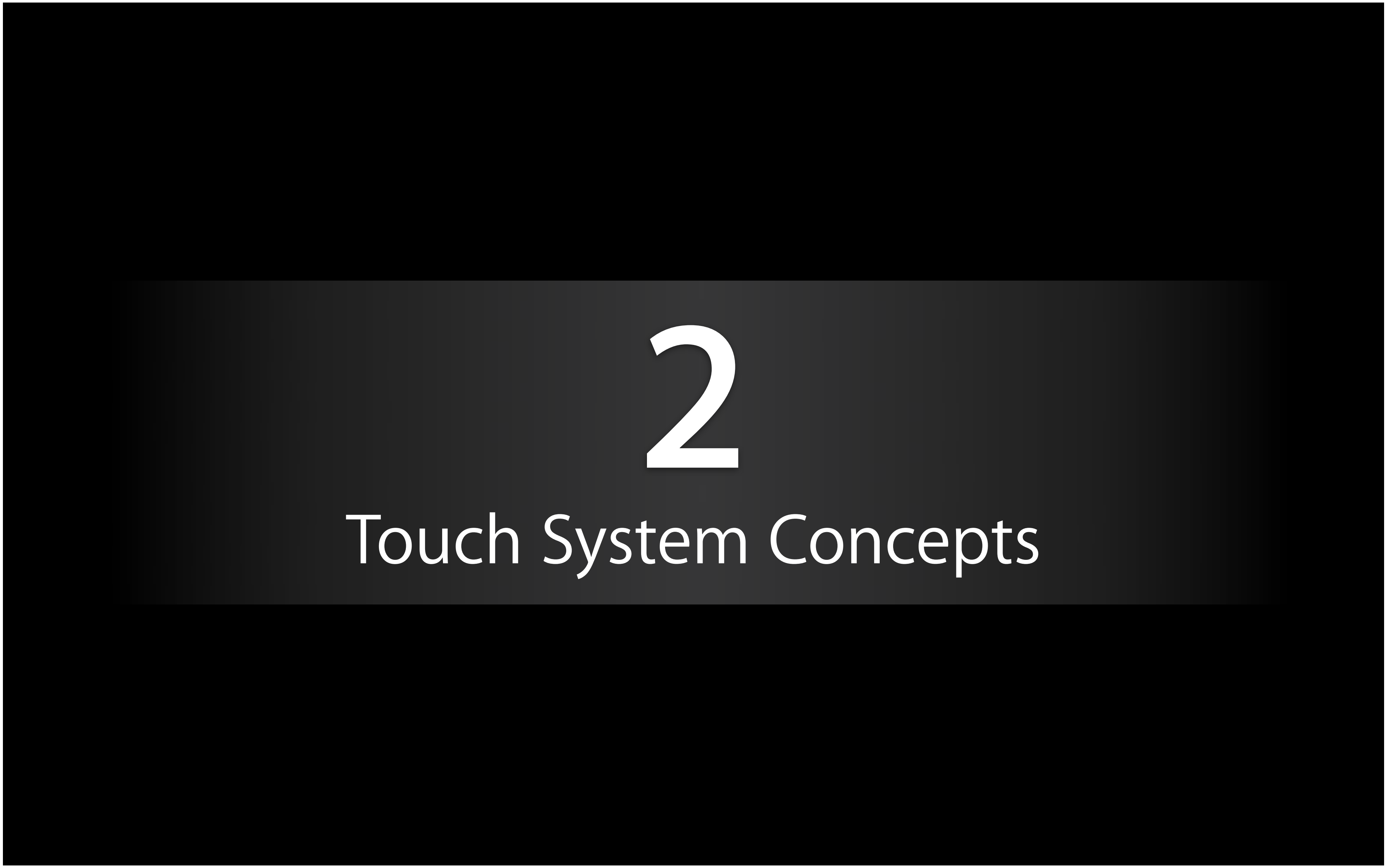
Multi-Touch Strategy

- Ignore
- Handle independently
- Handle together
- Handle as a gesture

- Mix and match

1

Multi-Touch Strategies



2

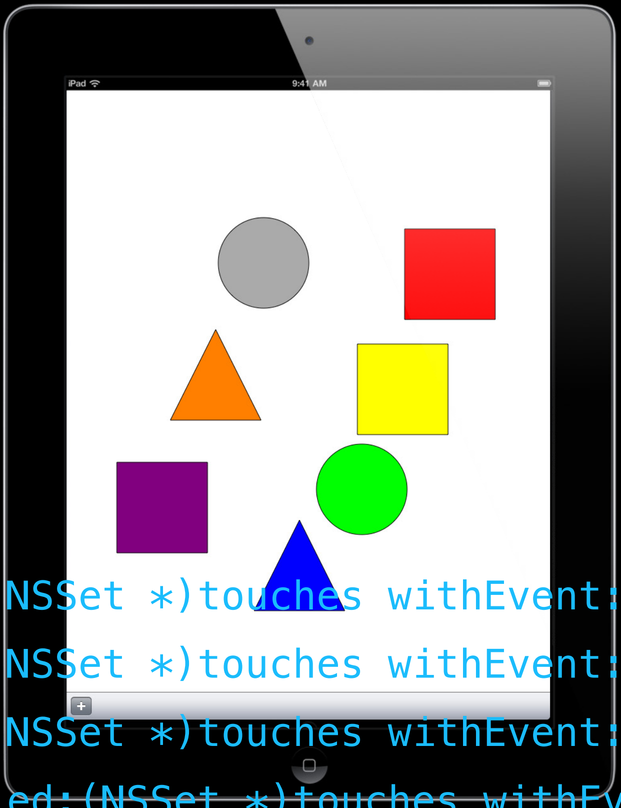
Touch System Concepts

Touch System Classes

- UITouch
- UIEvent
- UIApplication
- UIWindow
- UIViewController
- UIView
- UIResponder
- UIGestureRecognizer

Touch processing

Demo App



```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event;
- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event;
- (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event;
- (void)touchesCancelled:(NSSet *)touches withEvent:(UIEvent *)event;
```

Demo

Touch processing: direct manipulation

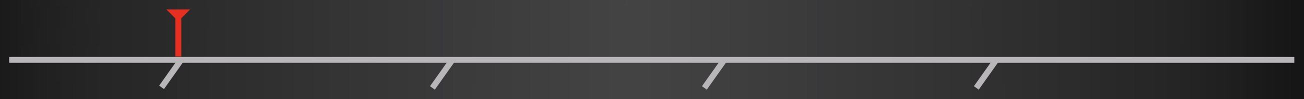
Processing a Single Touch

Touch comes down…

Find hit-test view

- Special work for UIApplication
  - Deepest view in hierarchy
  - Finds the view “under the touch”
  - All about view containment
  - Does not use first responder
- Touch and view are linked

Processing a Single Touch

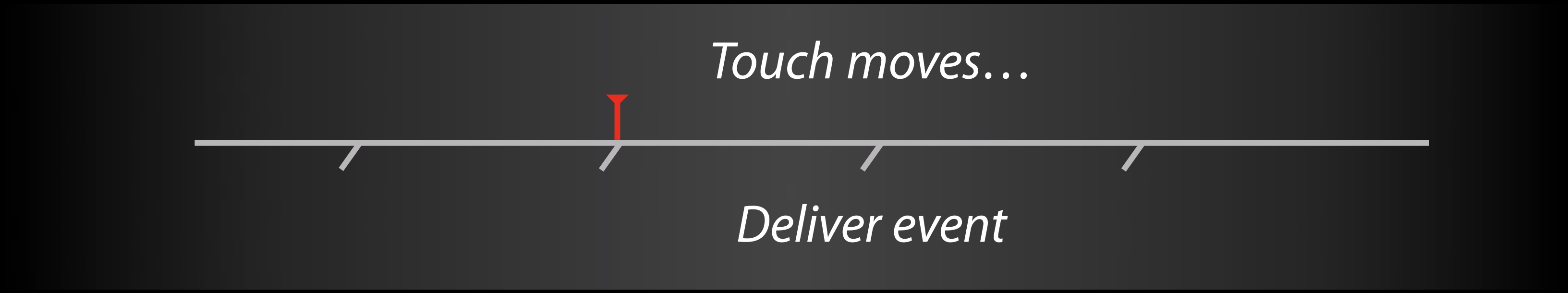


Touch comes down…

Deliver event

- Event delivery
  - UIApplication/UIWindow `-sendEvent:`
  - Hit-test view receives event
  - Gets `-touchesBegan:withEvent:`

Processing a Single Touch

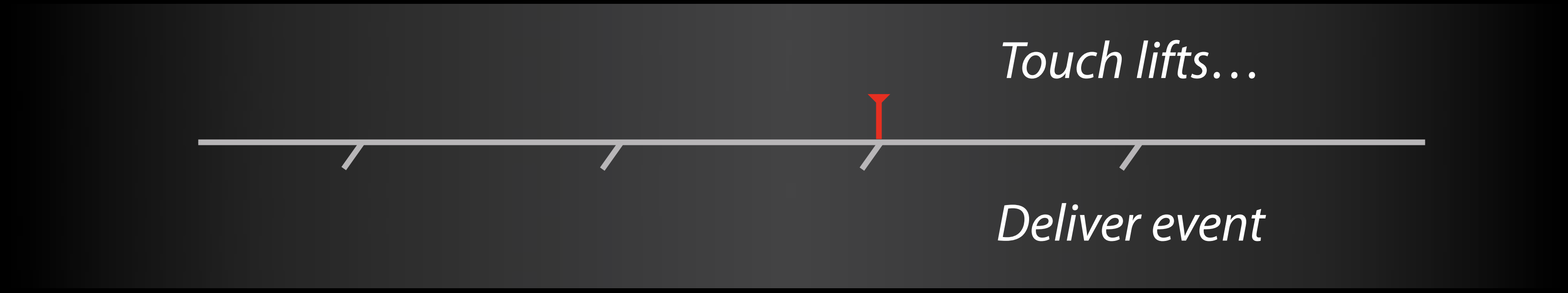


Touch moves…

Deliver event

- Event delivery is the same
  - Runs `-sendEvent:`
  - Hit-test view receives event
  - Gets `-touchesMoved:withEvent:`

Processing a Single Touch

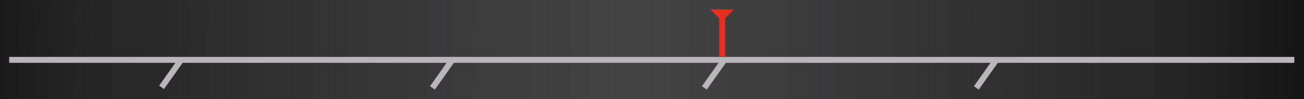


Touch lifts…

Deliver event

- Event delivery is the same
  - Runs `-sendEvent:`
  - Hit-test view receives event
  - Gets `-touchesEnded:withEvent:`

Processing a Single Touch

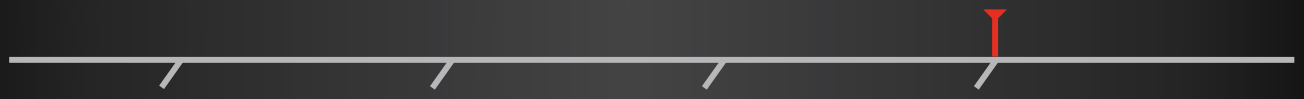


Touch cancelled…

Deliver event

- Event delivery is the same
  - Runs `-sendEvent:`
  - Hit-test view receives event
  - Gets `-touchesCancelled:withEvent:`

Processing a Single Touch

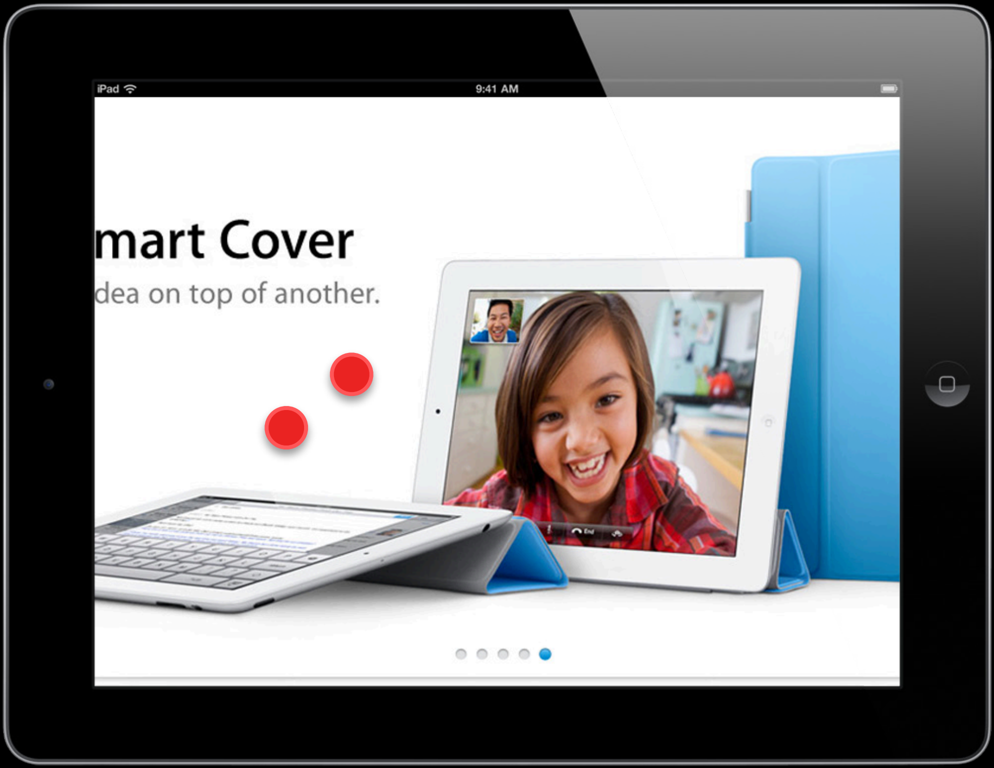


- Summary
  - Find hit-test view on touch start
  - Hit-test view gets all events

Processing a gesture

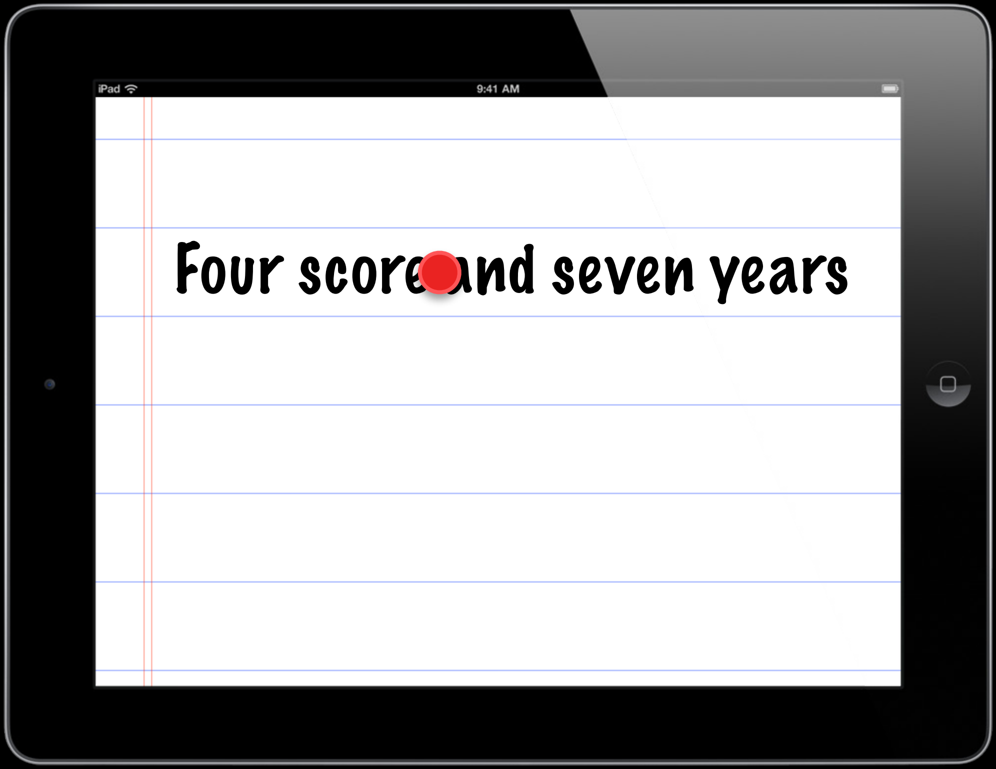
Touch handlers or gestures?

Gesture Recognizers Might Be Better



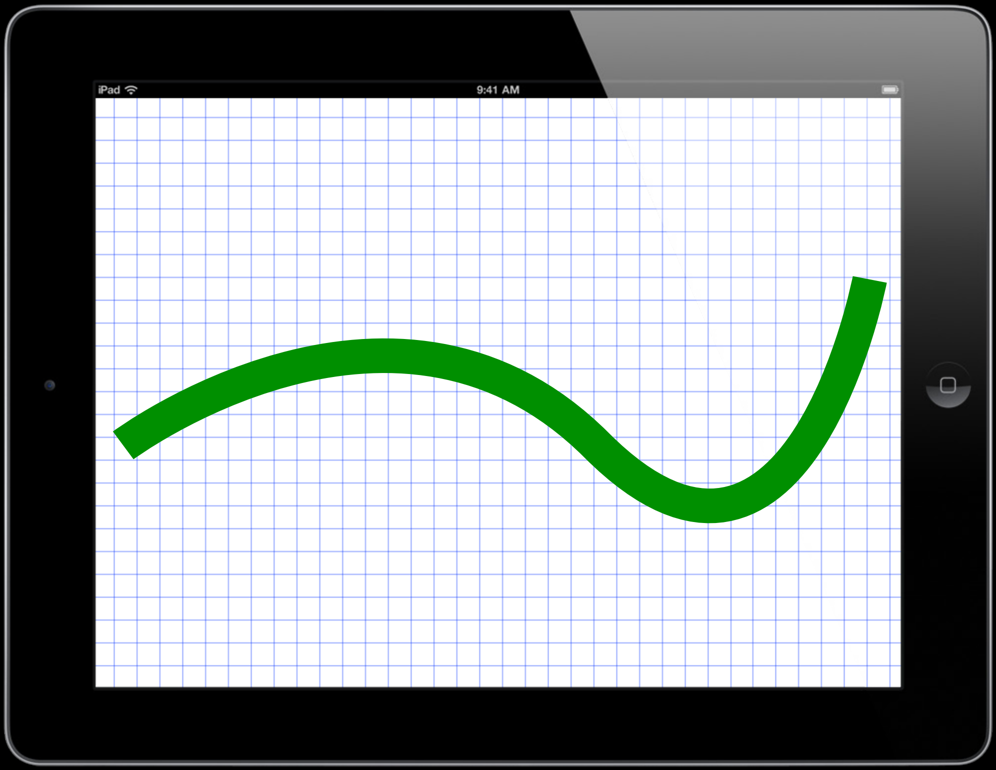
Gesture Recognizers Might Be Better

Custom delete gesture



Touch Handlers Might Be Better

Porting software



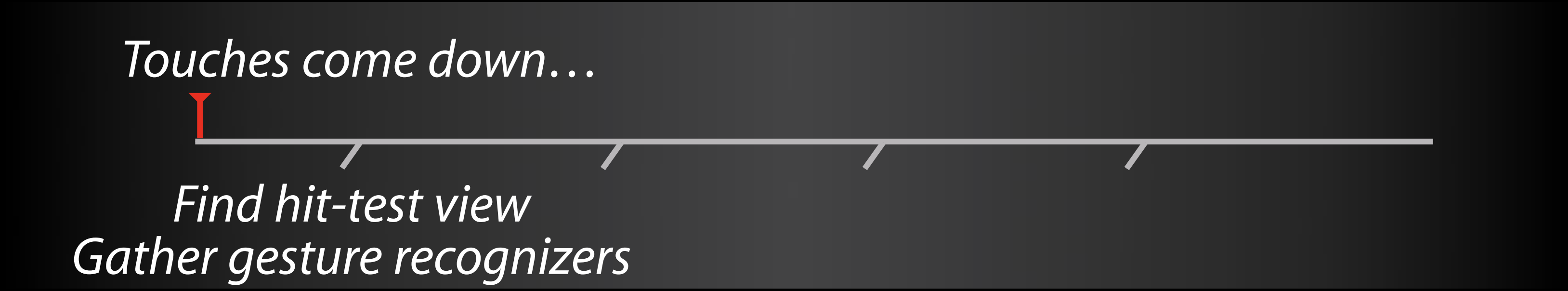
Six or half dozen?

Processing a gesture

Demo

Gesture processing: direct manipulation plus zoom

Processing a Gesture

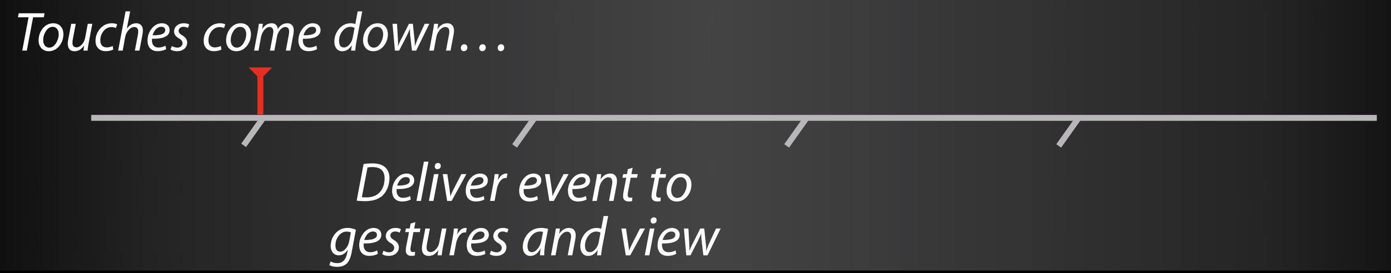


Touches come down…

Find hit-test view
Gather gesture recognizers

- Special work for UIApplication
  - Gesture gathering
  - Start with hit-test view
  - Go up view hierarchy
  - Priority: deepest view, first added

Processing a Gesture



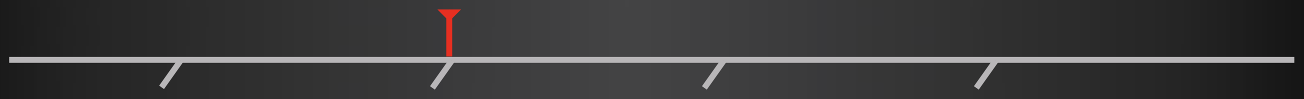
Touches come down…

Deliver event to gestures and view

- Event delivery is two-tracked
  - Gestures tested for recognition
  - Hit-test view receives event
  - Gets `–touchesBegan:withEvent:`

Processing a Gesture

Touches move…



Deliver event to gestures and view

- Event delivery is two-tracked
  - Gesture recognizes
  - View gets `–touchesCancelled:withEvent:`
  - Gesture runs its handler

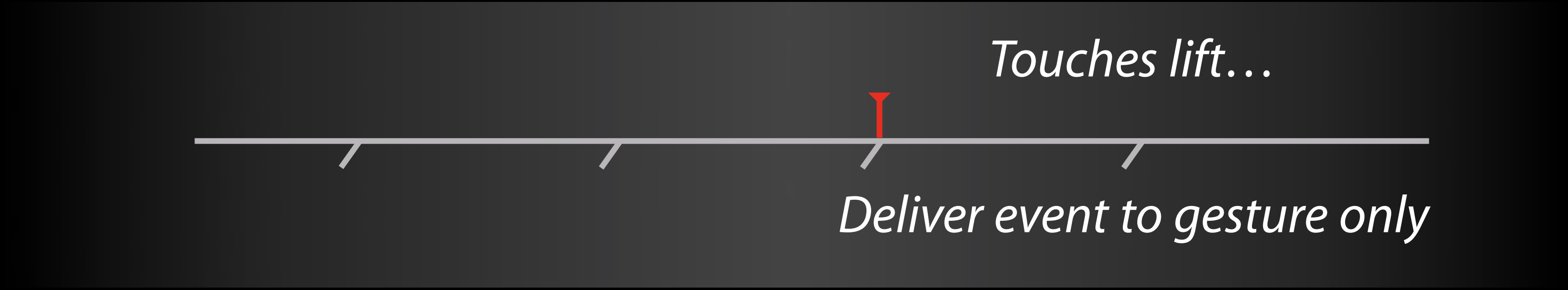
Processing a Gesture

Touches move again…

Deliver event to gesture only

- Event delivery is single-tracked
  - Hit-test view no longer receives event
  - Only gesture receives event
  - Gesture runs its handler

Processing a Gesture



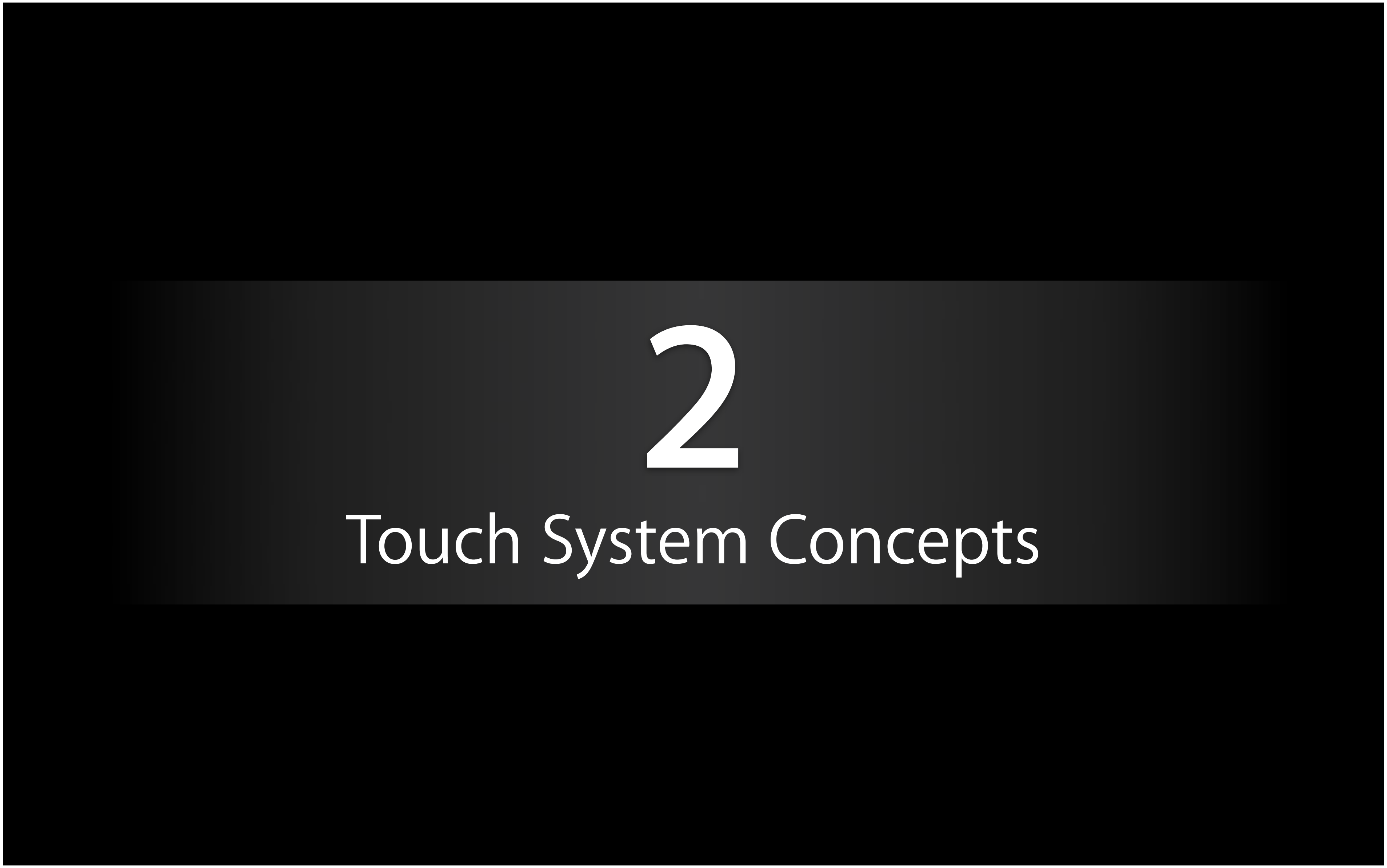
Touches lift…

Deliver event to gesture only

- Event delivery is the same
  - Hit-test view does not receives event
  - Only gesture receives event
  - Gesture runs its handler

Touch handlers and gestures

Work well together



2

Touch System Concepts

3

Touch System Tasks

Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

Implementing Direct Manipulation

With touch handlers

```
– (void)touchesBegan:(NSSet ∗)touches withEvent:(UIEvent ∗)event;
– (void)touchesMoved:(NSSet ∗)touches withEvent:(UIEvent ∗)event;
– (void)touchesEnded:(NSSet ∗)touches withEvent:(UIEvent ∗)event;
– (void)touchesCancelled:(NSSet ∗)touches withEvent:(UIEvent ∗)event;
```

Implementing Direct Manipulation

With gesture recognizers

```
- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event;
- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event;
- (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event;
- (void)touchesCancelled:(NSSet *)touches withEvent:(UIEvent *)event;
```

UIPanGestureRecognizer

UIPinchGestureRecognizer

Implementing Direct Manipulation

With gesture recognizers

```
- (void)pinch:(UIPinchGestureRecognizer *)pinch {
    switch (pinch.state) {
        case UIGestureRecognizerStatePossible: { break; }
        case UIGestureRecognizerStateBegan: { break; }
        case UIGestureRecognizerStateChanged: { break; }
        case UIGestureRecognizerStateEnded: { break; }
        case UIGestureRecognizerStateFailed: { break; }
        case UIGestureRecognizerStateCancelled: { break; }
    }
}
```

Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

Picking an Event Handler

Touches and the responder chain

- Touches do not go to first responder
- You can still use the responder chain
- Higher level object handle event

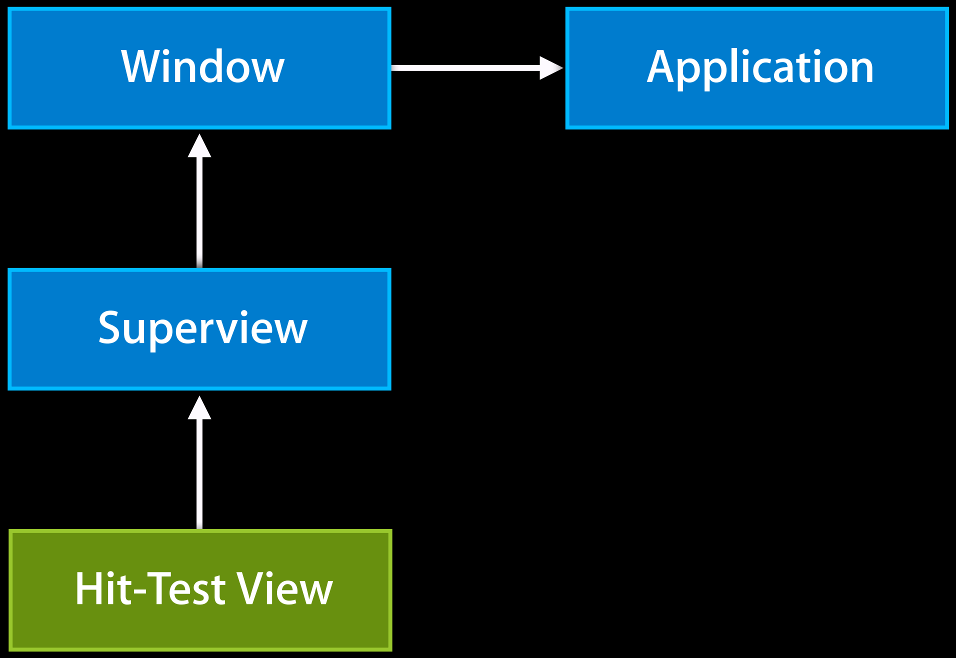
Picking an Event Handler

The responder chain

Hit-Test View

Picking an Event Handler

The responder chain



Window

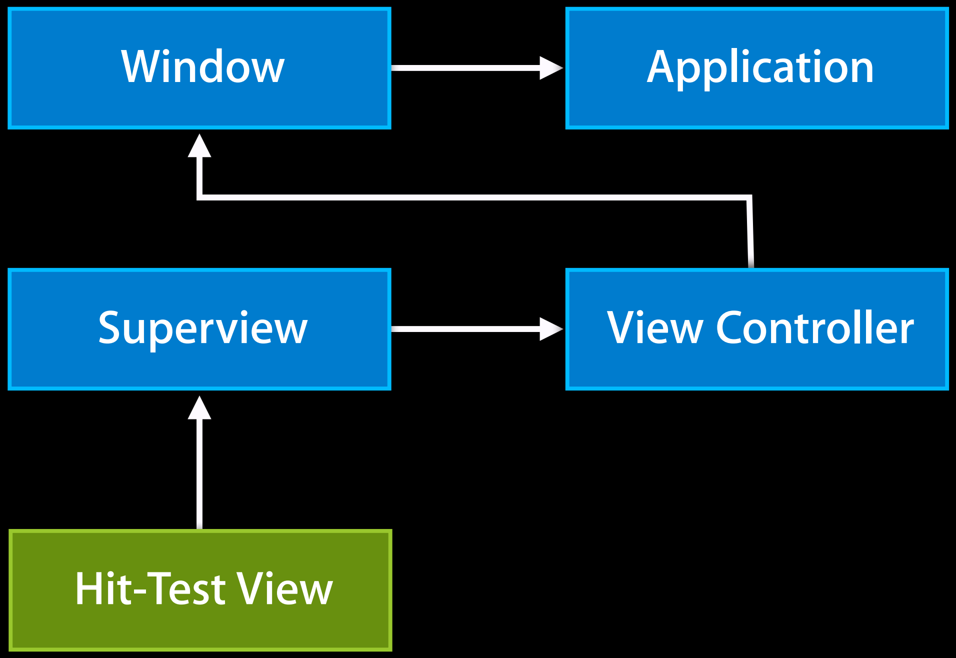
Application

Superview

Hit-Test View

Picking an Event Handler

The responder chain


Window

Application

Superview

View Controller

Hit-Test View

Picking an Event Handler

Gestures and the responder chain

- Gestures do not use responder chain
- View containment
- Gestures attached to views
- Gestures use target-action
- Gesture-controller pattern

Demo

Implementing direct manipulation and picking an event handler

Touch System Tasks

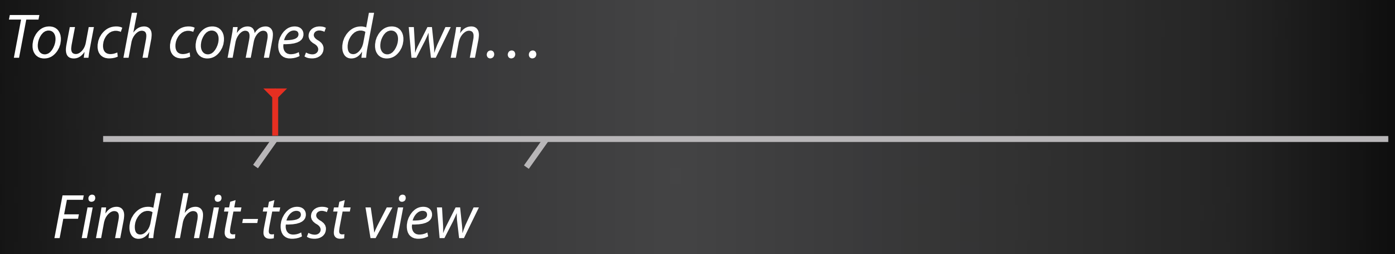
- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

Changing event flow is about…

…changing the hit-test view



Touch comes down…

Find hit-test view

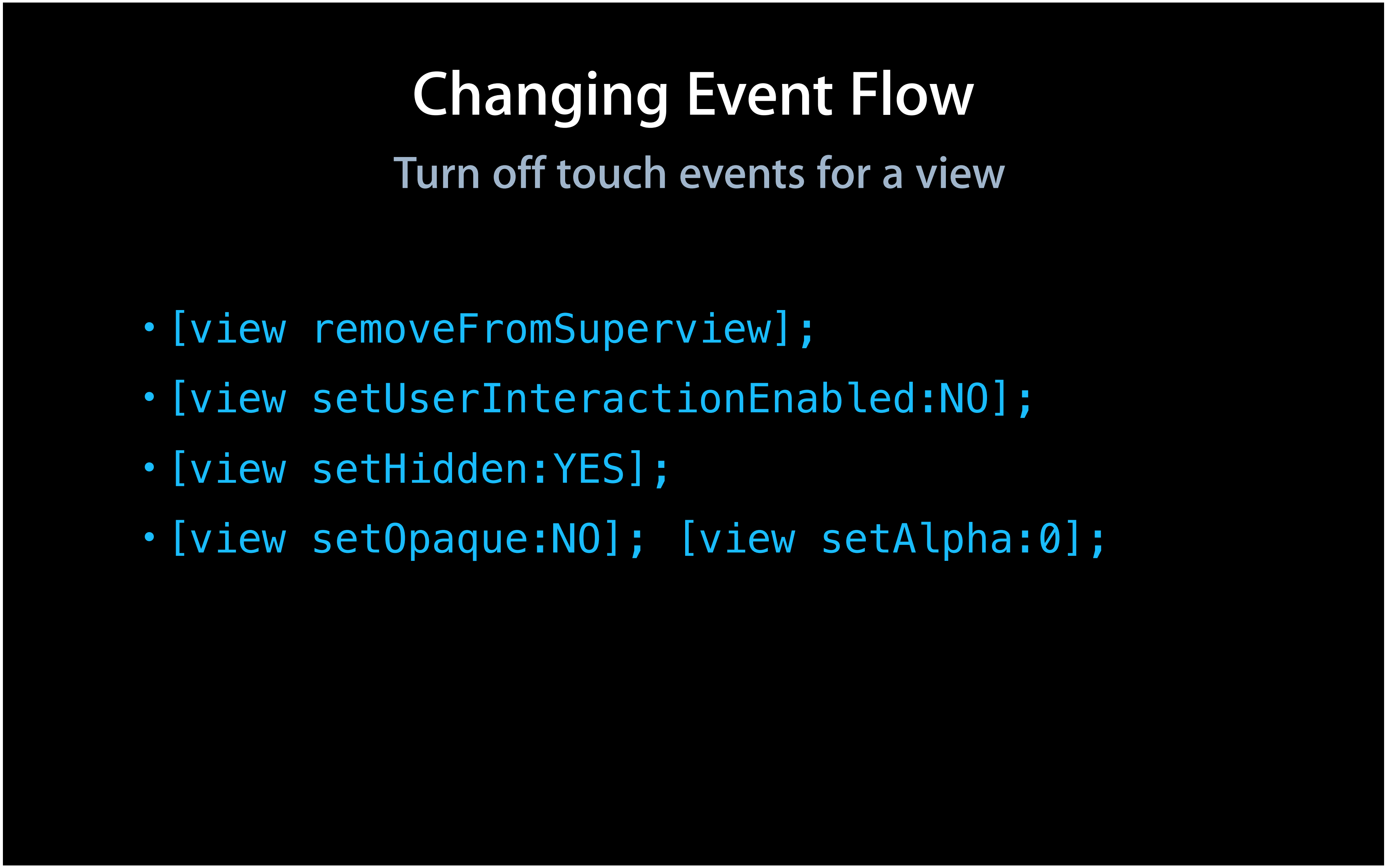
Changing Event Flow

Peek at all events

```
- (void)sendEvent:(UIEvent *)event;
```

UIApplication and UIWindow

❌ Think again



Changing Event Flow

Turn off touch events for a view

- `[view removeFromSuperview];`
- `[view setUserInteractionEnabled:NO];`
- `[view setHidden:YES];`
- `[view setOpaque:NO]; [view setAlpha:0];`

Changing Event Flow

Turn off touch events for entire app

UIApplication

```
myApp = [UIApplication sharedApplication];
[myApp beginIgnoringInteractionEvents];
…
[myApp endIgnoringInteractionEvents];
```

Make sure to balance the calls

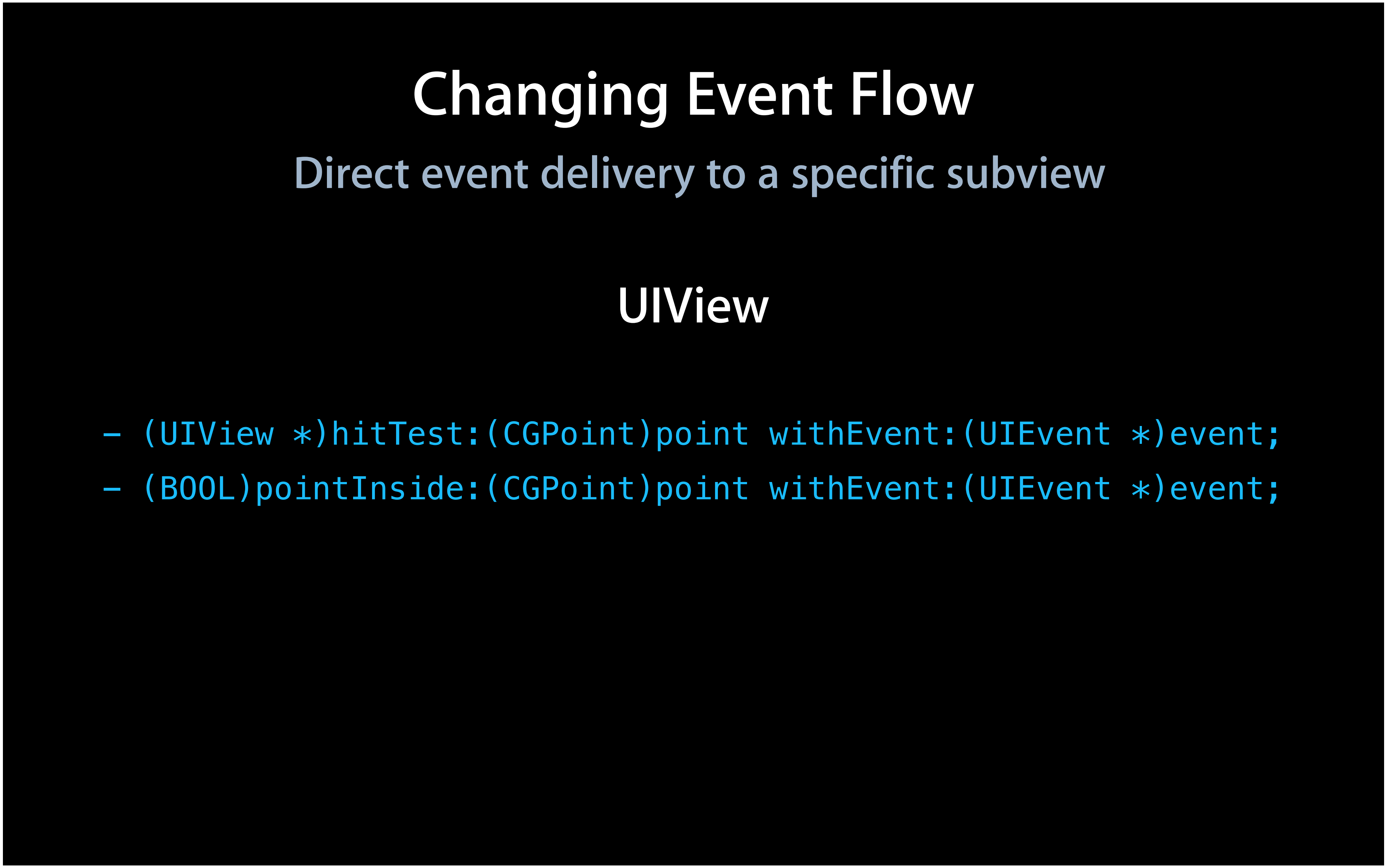
Changing Event Flow

Touches during animations

iOS 5+

Disables interaction only for animating view

Touches are eaten



Changing Event Flow

Direct event delivery to a specific subview

UIView

```
- (UIView *)hitTest:(CGPoint)point withEvent:(UIEvent *)event;
- (BOOL)pointInside:(CGPoint)point withEvent:(UIEvent *)event;
```

Changing Event Flow

Direct event delivery to a specific subview

UIView

```
- (UIView *)hitTest:(CGPoint)point withEvent:(UIEvent *)event;


- (BOOL)pointInside:(CGPoint)point withEvent:(UIEvent *)event;
```

Changing Event Flow

Direct event delivery to a specific subview

UIView

```
- (UIView *)hitTest:(CGPoint)point withEvent:(UIEvent *)event {
        // algorithmic test
}
- (BOOL)pointInside:(CGPoint)point withEvent:(UIEvent *)event {
        // geometric test
}
```

Changing Event Flow

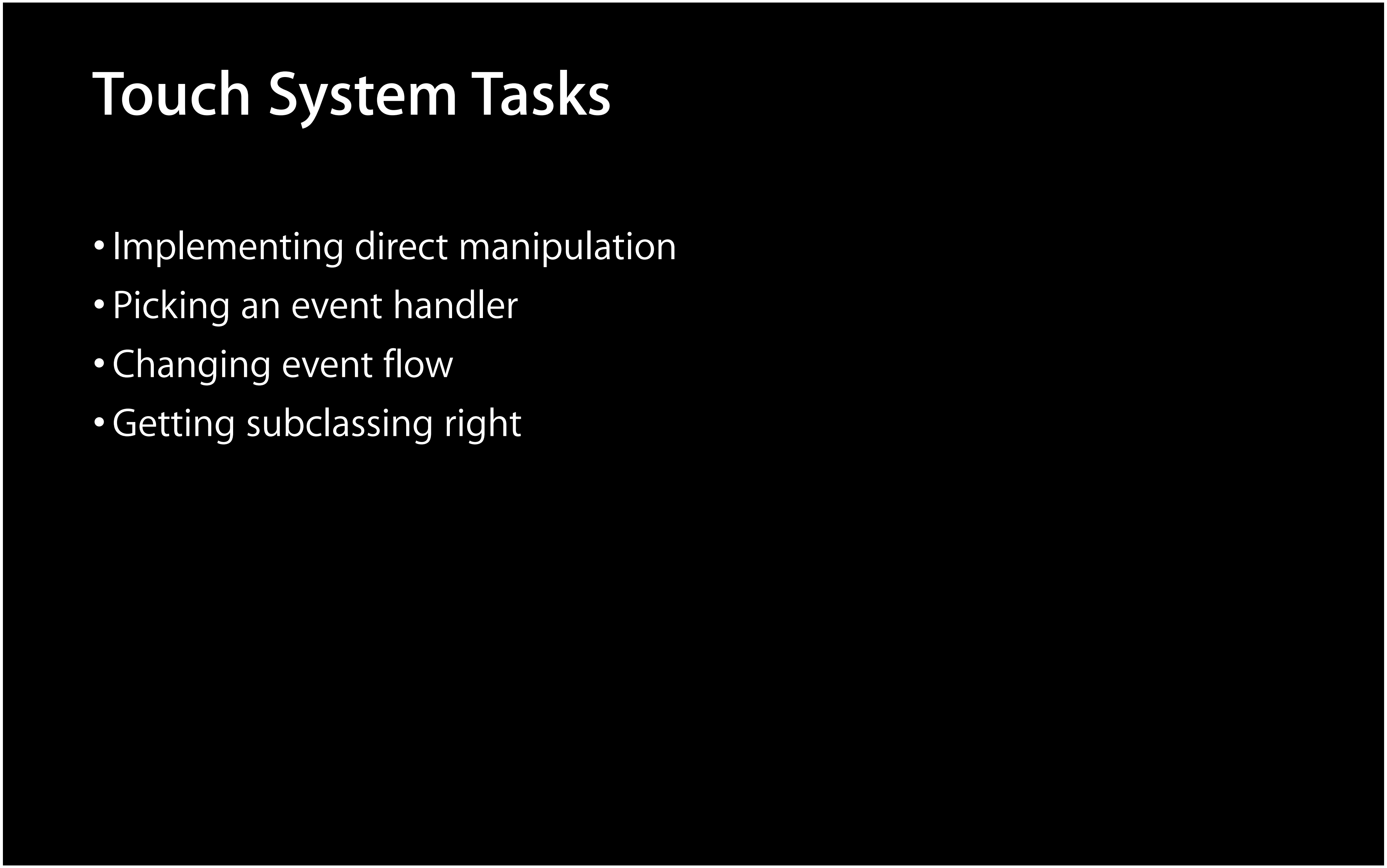
Direct event delivery to a specific subview

UIView

```
- (UIView *)hitTest:(CGPoint)point withEvent:(UIEvent *)event {
        // algorithmic test
}
- (BOOL)pointInside:(CGPoint)point withEvent:(UIEvent *)event {
        // geometric test
        CGRect rect = CGRectInset(self.bounds, -20, -20);
        return CGRectContainsPoint(rect, point);
}
```

Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right



Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

Getting Subclassing Right

Subclassing UIKit responder class

- Implement all the touch-handling methods
- Do not draw in event methods
- Do not forward events yourself

```
- (void)touchesBegan:(NSSet *)touches... {
        [[self nextResponder] touchesBegan…];
}
```

✗

Getting Subclassing Right

Subclassing UIKit responder class

- Implement all the touch-handling methods
- Do not draw in event methods
- Do not forward events yourself

```
- (void)touchesBegan:(NSSet *)touches... {
        [super touchesBegan…];
}
```

✓

Getting Subclassing Right

Subclass UIView or UIControl?

- Personal preference
- UIControl gives you common extras
- Target-action pattern
- Event-handling: `touchUpInside`

Getting Subclassing Right

Subclass an existing UIControl?

- ✗ Not recommended (generally)
- ✓ Delegates and notifications

Getting Subclassing Right

Subclassing UIGestureRecognizer

- Look at UIKit-provided classes
- Look at UIGestureRecognizerSubclass
- Keep gestures simple

Touch System Tasks

- Implementing direct manipulation
- Picking an event handler
- Changing event flow
- Getting subclassing right

3

Touch System Tasks

4

Interacting with the Rest of iOS

For users…

iOS devices become your app

For you…

Deliver a great experience to users

Work and play well with others

Scenarios for Canceling Touches

- Phone call
- Alert
- Locking device
- Multitasking switcher bar

- Multitasking gestures 5

What do you need to do?

Be a good multitasking citizen

Implement these!

`touchesCancelled`

`UIGestureRecognizerStateCancelled`

Strategies for Canceling Touches

- Canceling is like ending
- Leave in provisional state
- Undo

Demo

Canceling touches

Strategies for Canceling Touches

- Canceling is like ending
- Leave in provisional state
- Undo

4

Interacting with the Rest of iOS

4

Big Ideas

Make the most of your effort

WWDC2011