Simplifying Touch Event Handling with Gesture Recognizers

Let us do it for you

Brad Moore and Josh Shaffer
iPhone Frameworks Engineers

Touch Interfaces

Easy to use

Easy to Use

- Direct manipulation

Easy to Use

- Direct manipulation
- Common gestures and behaviors

Easy to Use

- Direct manipulation
- Common gestures and behaviors
  - Tap

Easy to Use

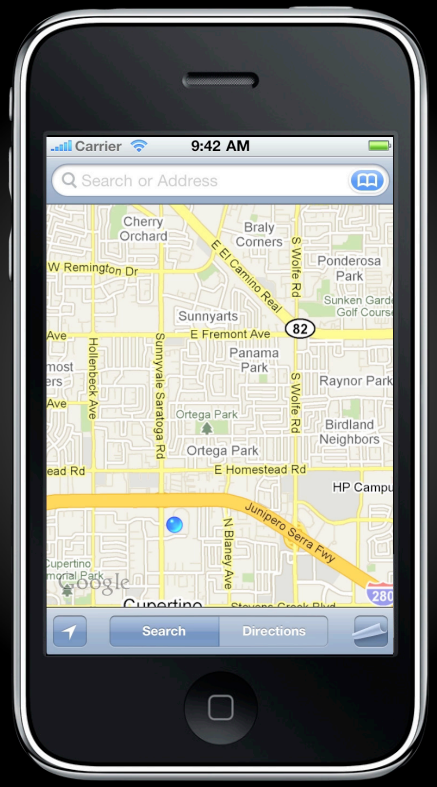
- Direct manipulation
- Common gestures and behaviors
  - Tap
  - Pinch

Easy to Use

- Direct manipulation
- Common gestures and behaviors
  - Tap
  - Pinch
  - Swipe

Easy to Use

- Direct manipulation
- Common gestures and behaviors
  - Tap
  - Pinch
  - Swipe
  - Pan



Easy to Use

- Direct manipulation
- Common gestures and behaviors
  - Tap
  - Pinch
  - Swipe
  - Pan
  - Press-and-hold

Touch Interfaces

- Easy to use
  - Direct manipulation
  - Common gestures and behaviors

Touch Interfaces

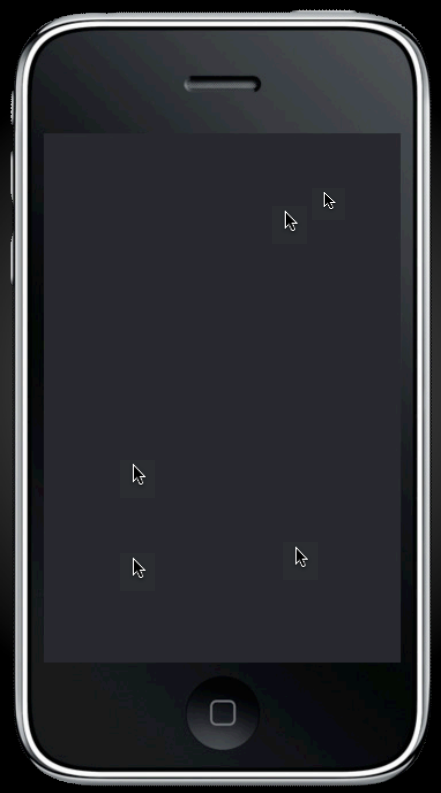
- Easy to use
  - Direct manipulation
  - Common gestures and behaviors
- Hard to write

Hard to Write

- Limited precision

Hard to Write

- Limited precision
- Many simultaneous inputs



Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
  - Tap

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
  - Tap
  - Double-tap

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
  - Tap
  - Double-tap
  - Pinch

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
  - Tap
  - Double-tap
  - Pinch
  - Pan

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
  - Tap
  - Double-tap
  - Pinch
  - Pan
  - Long-press

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
  - Tap
  - Double-tap
  - Pinch
  - Pan
  - Long-press
  - Tap-and-a-half

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
- Tempting non-solutions

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
- Tempting non-solutions
  - Wait

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
- Tempting non-solutions
  - Wait
  - Guess

Hard to Write

- Limited precision
- Many simultaneous inputs
- Inherent ambiguity
- Tempting non-solutions
  - Wait
  - Guess
  - Give up

Touch Interfaces

- Easy to use
  - Direct manipulation
  - Common gestures and behaviors
- Hard to write
  - Limited precision
  - Many simultaneous inputs
  - Inherent ambiguity

UIGestureRecognizer

Making it easy for developers

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Touch Handling

- One UITouch per finger
- UIView hit testing
- Responder delivery

Touch Handling

```
@interface MyView : UIView {
  UITouch *trackedTouch;
  CGPoint startPoint;
}

- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event
{
   if (trackedTouch == nil) {
      trackedTouch = [touches anyObject];
      startPoint = [trackedTouch locationInView:self];
   }
}
```

Touch Handling

```
- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event
{
   CGPoint currentPoint = [trackedTouch locationInView:self];
   if (currentPoint.x - startPoint.x > MIN_SWIPE_X_THRESHOLD &&
       ABS(currentPoint.y - startPoint.y) < MAX_SWIPE_Y_THRESHOLD)) {
        NSLog(@”Seems like a swipe.”)
    }
}

- (void)touchesEnded:(NSSet *)touches withEvent:(UIEvent *)event
{
  if (trackedTouch && [touches containsObject:trackedTouch])
     trackedTouch = nil;
}
```

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Gesture Handling

- Instantiate and configure a predefined UIGestureRecognizer
- Designate one or more handlers
- Add recognizer to a view

Gesture Handling

```
- (id)initWithFrame:(CGRect)frame
{
   if ((self = [super initWithFrame:frame]) == nil)
      return nil;

   UISwipeGestureRecognizer *swipe = [[UISwipeGestureRecognizer alloc]
      initWithTarget:self action:@selector(swipeRecognized:)];
   [self addGestureRecognizer:swipe];
   [swipe release];

   return self;
}
```

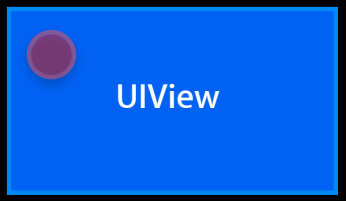
Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

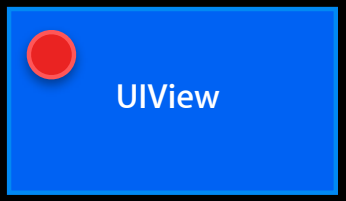
Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

How It Works

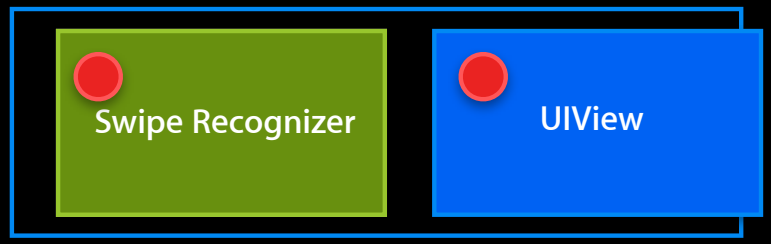


How It Works



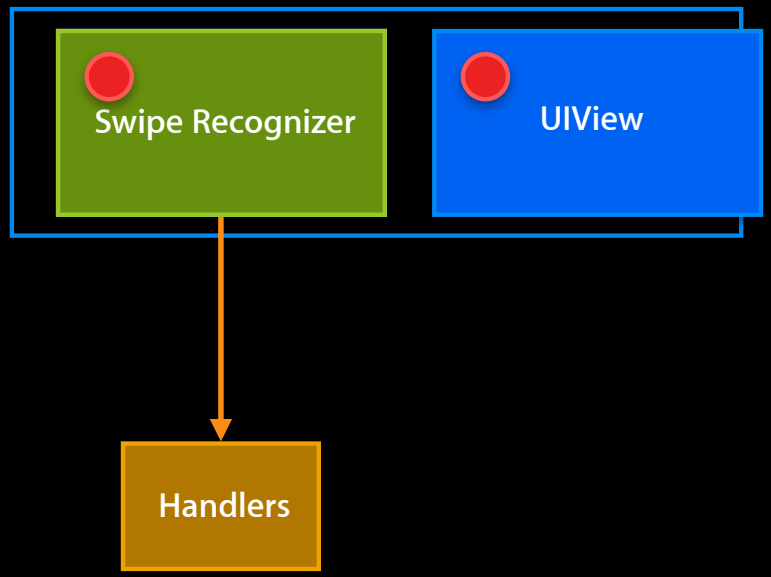
How It Works

Multicast touch delivery



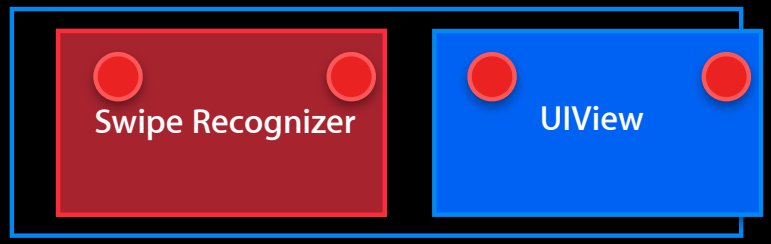
How It Works

Success



How It Works

Failure



How It Works

Independent analysis

How It Works

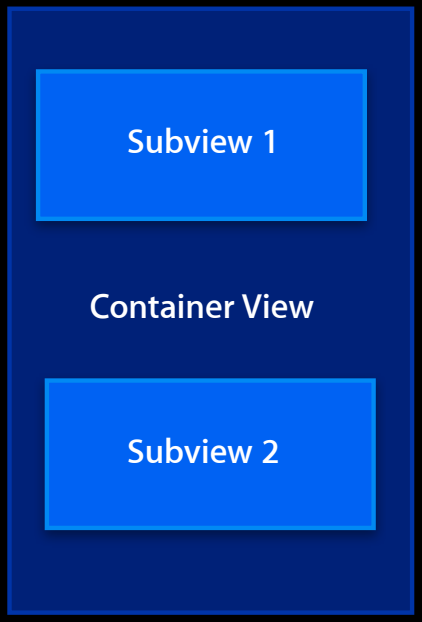
Independent analysis

How It Works

Independent analysis

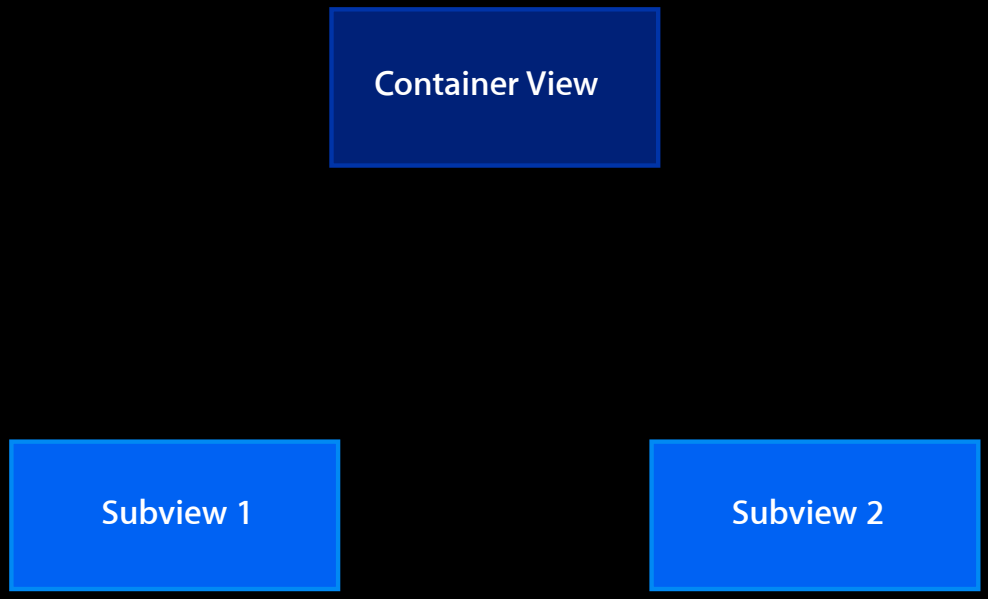
How It Works

Contextual analysis



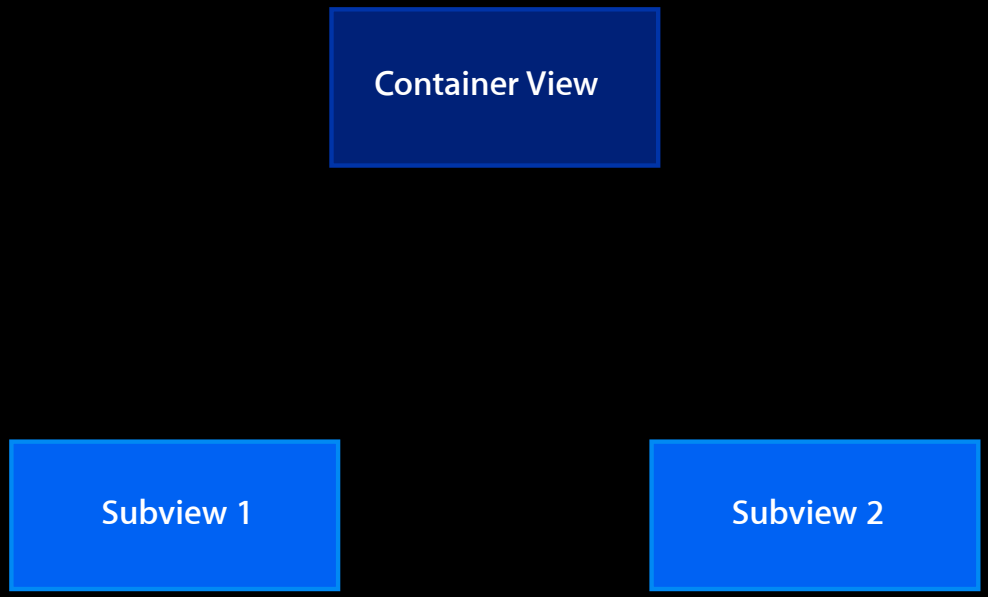
How It Works

Contextual analysis



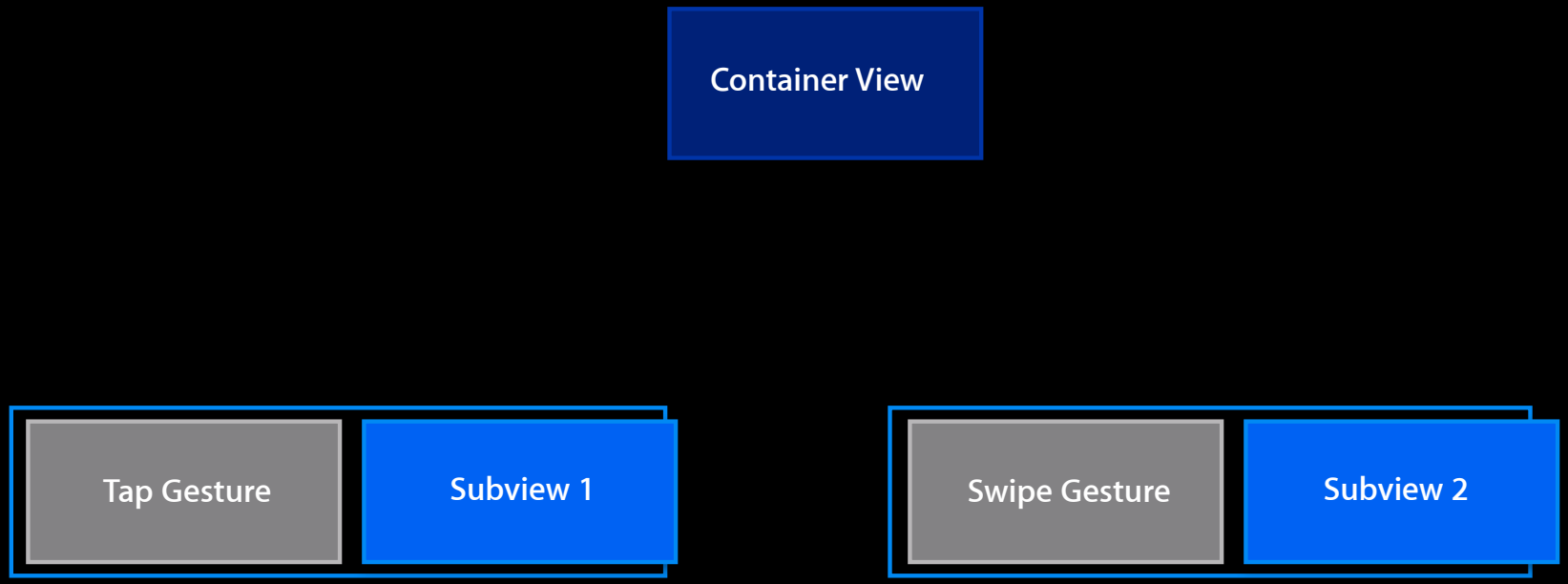
How It Works

Contextual analysis



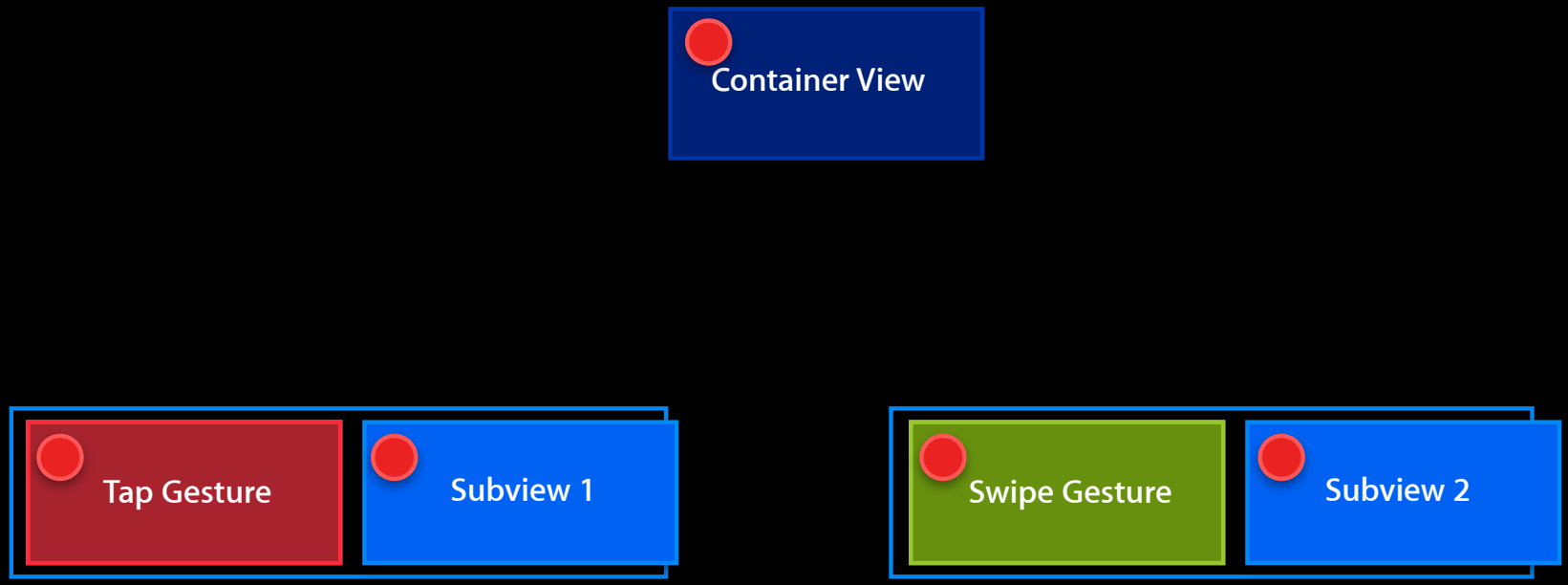
How It Works

Contextual analysis



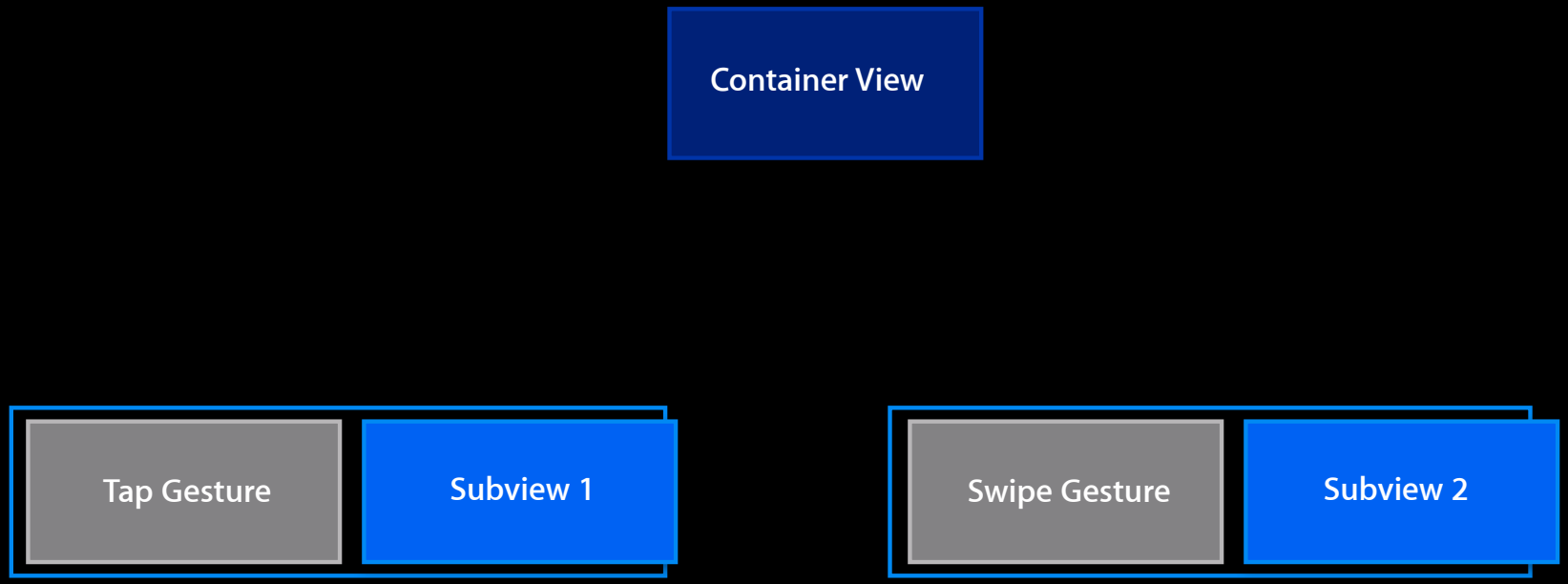
How It Works

Contextual analysis



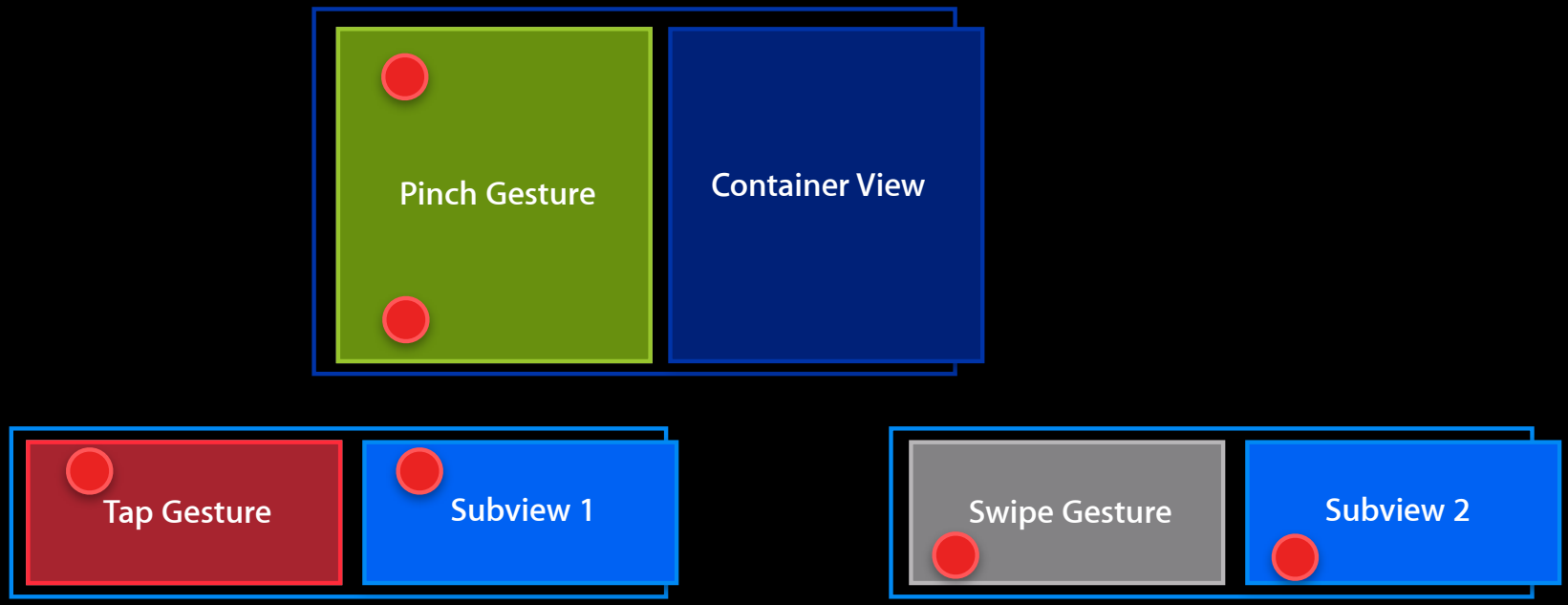
How It Works

Beyond one view



How It Works

Beyond one view



How It Works

- Multicast touch delivery
- Independent analysis
- Contextual processing
- Across multiple views

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

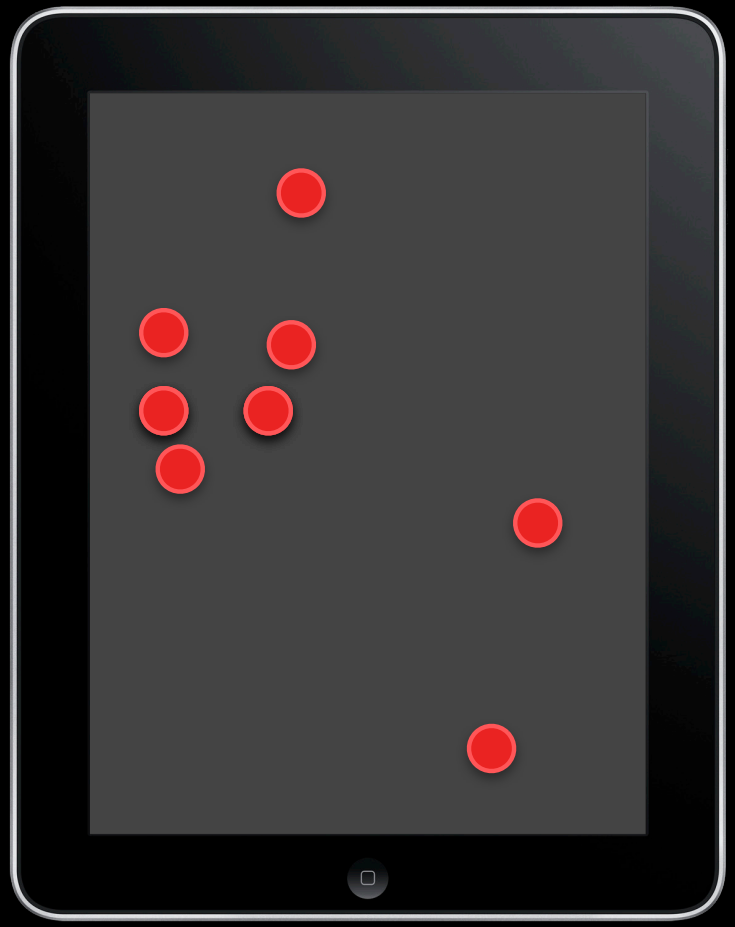
Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Using the API

UIGestureRecognizer

- Abstract base class
- Many concrete subclasses
  - UITapGestureRecognizer
  - UIPinchGestureRecognizer
  - UISwipeGestureRecognizer
  - UIPanGestureRecognizer
  - UILongPressGestureRecognizer
  - UIRotationGestureRecognizer
- Custom subclasses encouraged



Establishing Handlers

UIGestureRecognizer

- Notifies of recognition via target/action pairs

```
- (id)initWithTarget:(id)target action:(SEL)action;
- (void)addTarget:(id)target action:(SEL)action;
- (void)removeTarget:(id)target action:(SEL)action;
```

Establishing Handlers

UIGestureRecognizer

- Notifies of recognition via target/action pairs

```
- (id)initWithTarget:(id)target action:(SEL)action;
- (void)addTarget:(id)target action:(SEL)action;
- (void)removeTarget:(id)target action:(SEL)action;
```

- Actions take recognizer as argument

```
- (void)gestureRecognized:(UIGestureRecognizer *)recognizer
{
    // Do something
}
```

Establishing Handlers

UIGestureRecognizer

- Notifies of recognition via target/action pairs

```
- (id)initWithTarget:(id)target action:(SEL)action;
- (void)addTarget:(id)target action:(SEL)action;
- (void)removeTarget:(id)target action:(SEL)action;
```

- Actions take recognizer as argument

```
- (void)gestureRecognized:(UIGestureRecognizer *)recognizer
{
    // Do something
}
```

Handling Gestures

Location

- On-screen location
  - `- (CGPoint)locationInView:(UIView*)view;`

Handling Gestures

Location

- On-screen location
  - `- (CGPoint)locationInView:(UIView*)view;`

- Detailed touch information
  - `- (NSUInteger)numberOfTouches;`
  - `- (CGPoint)locationOfTouch:(NSUInteger)touchIndex inView:(UIView*)view;`

Handling Gestures

State

```
@property(nonatomic,readonly) UIGestureRecognizerState state;
```

- Just for bookkeeping
  - `UIGestureRecognizerStatePossible`
  - `UIGestureRecognizerStateFailed`

Handling Gestures

State

```
@property(nonatomic,readonly) UIGestureRecognizerState state;
```

- Just for bookkeeping
  - `UIGestureRecognizerStatePossible`
  - `UIGestureRecognizerStateFailed`
- Discrete recognizers
  - `UIGestureRecognizerStateRecognized`

Handling Gestures

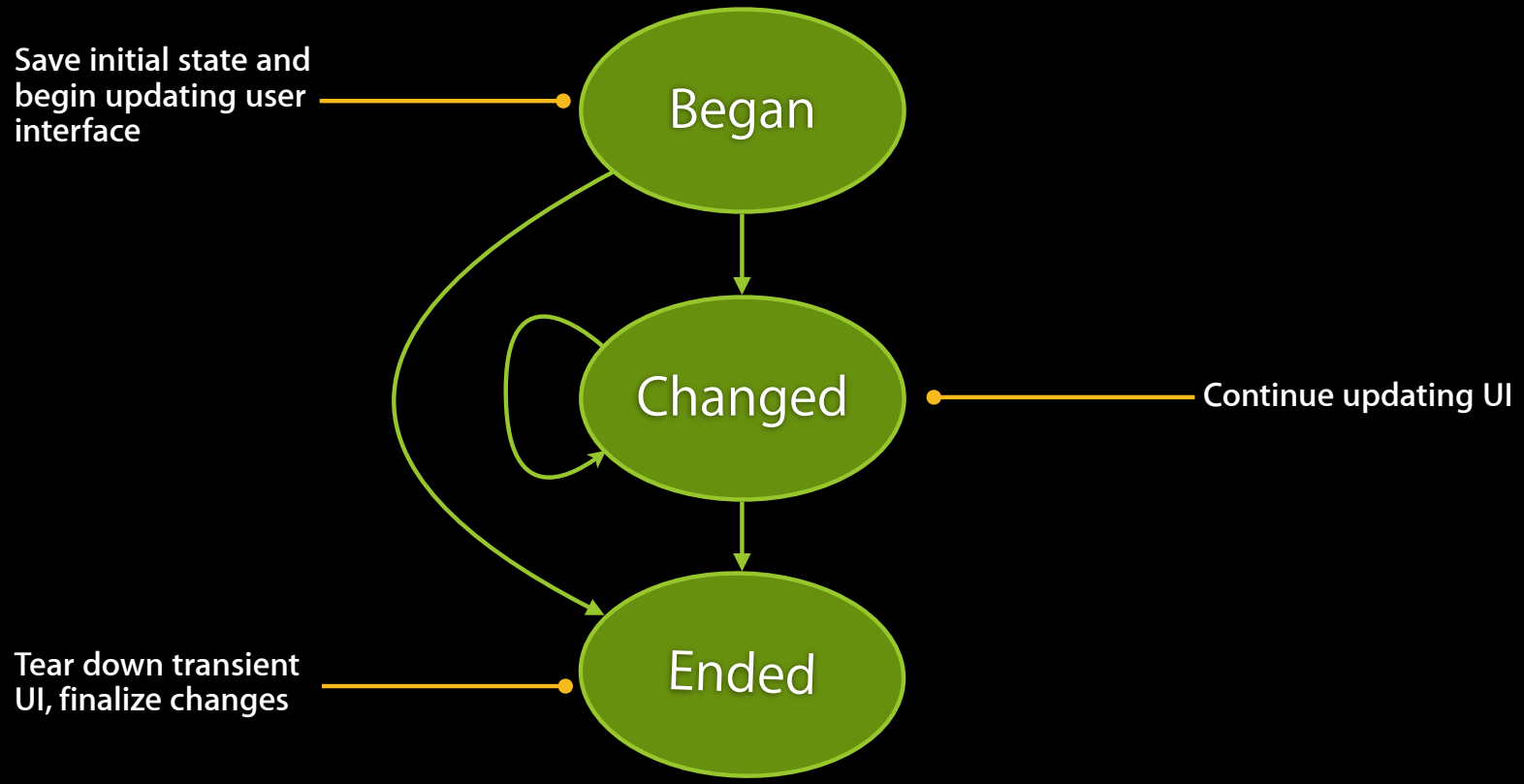
State

```
@property(nonatomic,readonly) UIGestureRecognizerState state;
```

- Just for bookkeeping
  - `UIGestureRecognizerStatePossible`
  - `UIGestureRecognizerStateFailed`
- Discrete recognizers
  - `UIGestureRecognizerStateRecognized`
- Continuous recognizers
  - `UIGestureRecognizerStateBegan`
  - `UIGestureRecognizerStateChanged`
  - `UIGestureRecognizerStateEnded`
  - `UIGestureRecognizerStateCancelled`

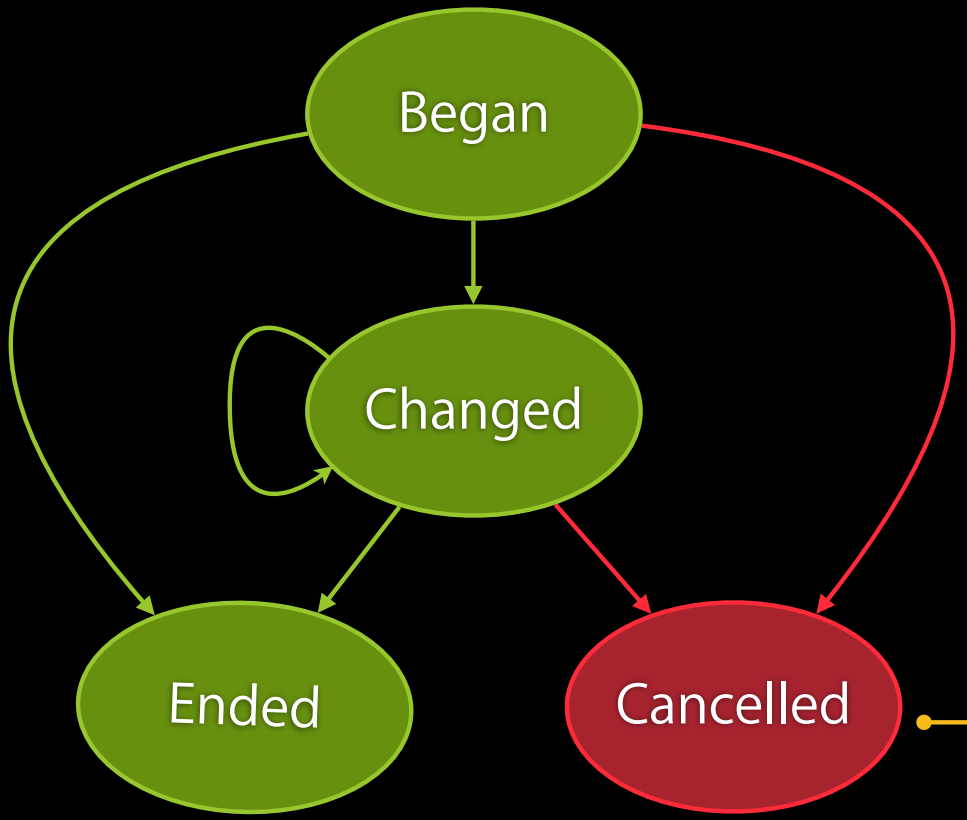
Handling Gestures

Continuous state



Handling Gestures

Continuous state



Handling Gestures

Continuous state

```
- (void)handleLongPress:(UIGestureRecognizer *)recognizer
{
    UIView *view = recognizer.view;
    CGPoint currentLocation = [recognizer locationInView:view.superview];

    switch (recognizer.state) {
        case UIGestureRecognizerStateBegan:
            startLocation = view.center;
            centerOffset = PointDifference(currentLocation, startLocation);
            [self beginJiggling:view];
            break;
        case UIGestureRecognizerStateChanged:
            view.center = PointSum(currentLocation, centerOffset);
            break;
```

```
- (void)handleLongPress:(UIGestureRecognizer *)recognizer {
        ...

        case UIGestureRecognizerStateEnded:
            view.center = PointSum(currentLocation, centerOffset);
            [self endJiggling:view];
            break;
        case UIGestureRecognizerStateCancelled:
            view.center = startLocation;
            [self endJiggling:view];
            break;
    }
}
```

Handling Gestures

Specialized state

- Subclasses usually have additional state
  - Appropriate to the gesture
    - `[UIPinchGestureRecognizer scale]`
    - `[UIPanGestureRecognizer translationInView:]`

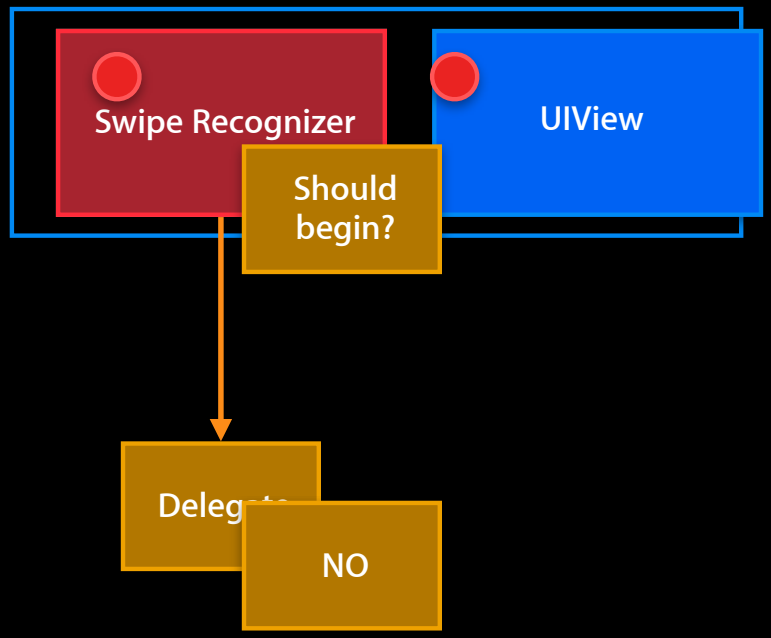
Configuring Gestures

Delegate

```
@property (nonatomic,assign) id <UIGestureRecognizerDelegate> delegate;


@protocol UIGestureRecognizerDelegate

...

- (BOOL)gestureRecognizerShouldBegin:(UIGestureRecognizer *)gestureRecognizer;

- (BOOL)gestureRecognizer:(UIGestureRecognizer *)gestureRecognizer
       shouldReceiveTouch:(UITouch *)touch;

@end
```

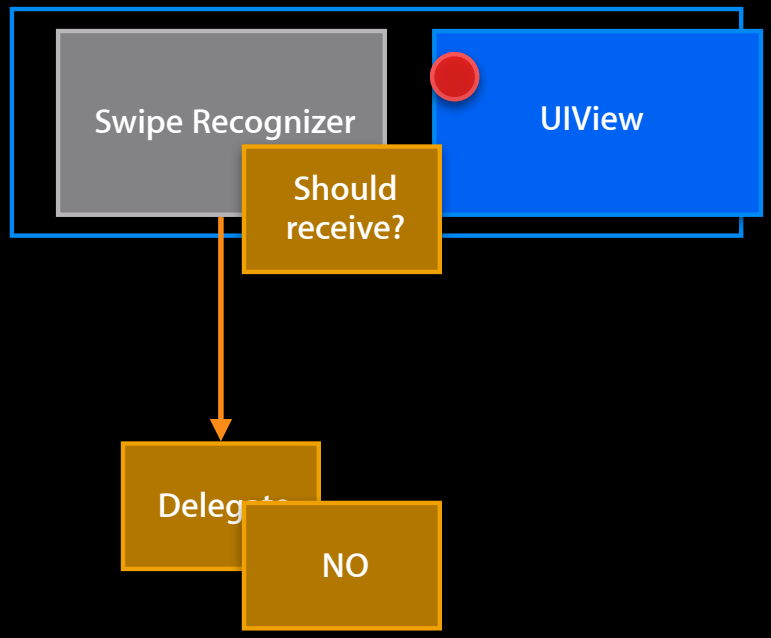
Configuring Gestures

Delegate



Configuring Gestures

Delegate



Configuring Gestures

Subclasses

- Built-in set highly configurable

```
@interface UITapGestureRecognizer

...

@property (nonatomic) NSUInteger  numberOfTapsRequired;
@property (nonatomic) NSUInteger  numberOfTouchesRequired;


@end
```

Configuring Gestures

Subclasses

- Built-in set highly configurable

```
@interface UIPanGestureRecognizer

...

@property (nonatomic) NSUInteger minimumNumberOfTouches;
@property (nonatomic) NSUInteger maximumNumberOfTouches;


@end
```

Configuring Gestures

Subclasses

- Built-in set highly configurable

```
@interface UILongPressGestureRecognizer

...

@property (nonatomic) NSInteger      numberOfTapsRequired;
@property (nonatomic) NSInteger      numberOfTouchesRequired;
@property (nonatomic) CFTimeInterval minimumPressDuration;
@property (nonatomic) CGFloat        allowableMovement;

@end
```

Configuring Gestures

Subclasses

- Built-in set highly configurable
  - To a fault?
- Please exercise restraint!
  - Consistency
  - Discoverability

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Demo

Adding Gesture Recognizers

Josh Shaffer
As seen on TV

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

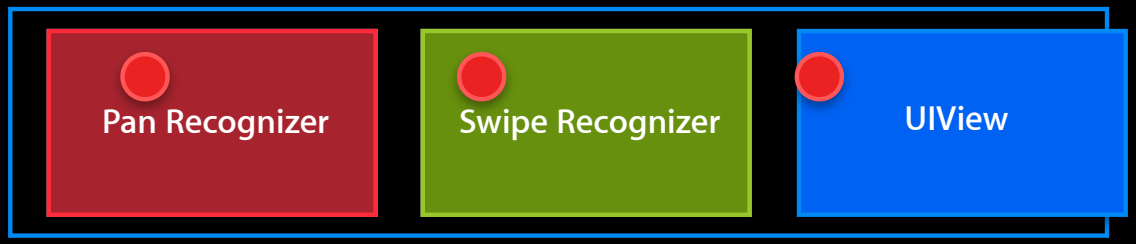
Conflict Resolution

Gestures in conflict

- A superposition of possibilities
  - But there can only be one
  - First-to-recognize wins

Conflict Resolution

Gestures in conflict



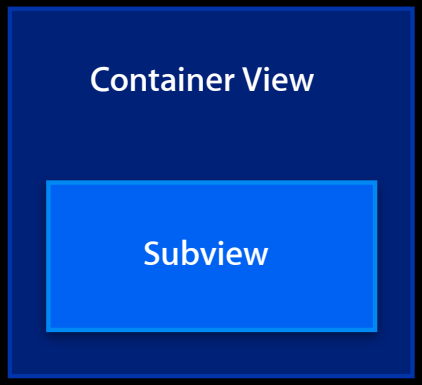
Conflict Resolution

Gestures in conflict

- A superposition of possibilities
  - But there can only be one
  - First-to-recognize wins
- Tie breakers
  - Deepest view
  - Most recently added

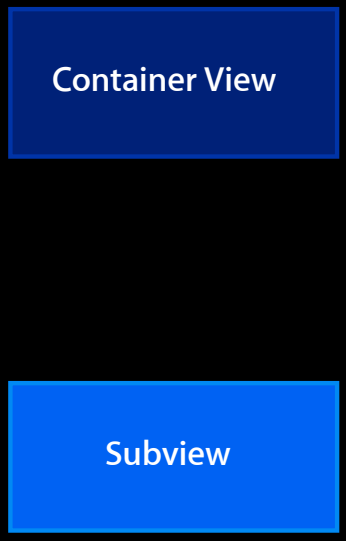
Conflict Resolution

Precedence



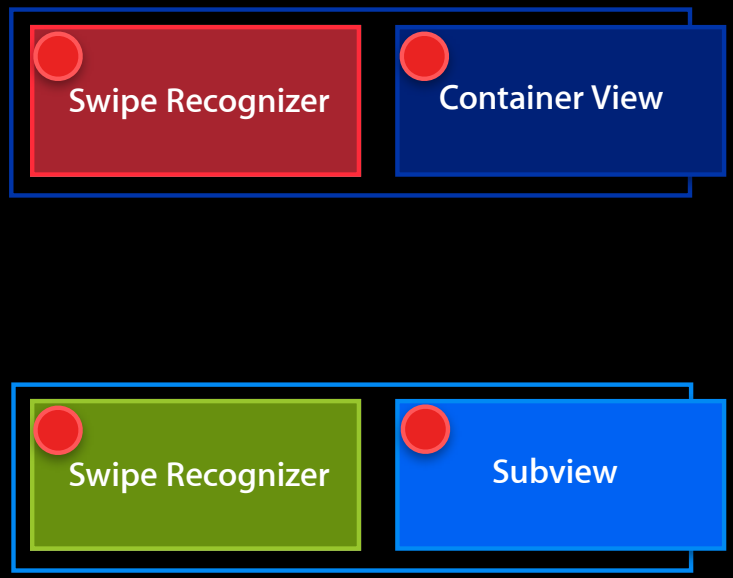
Conflict Resolution

Precedence



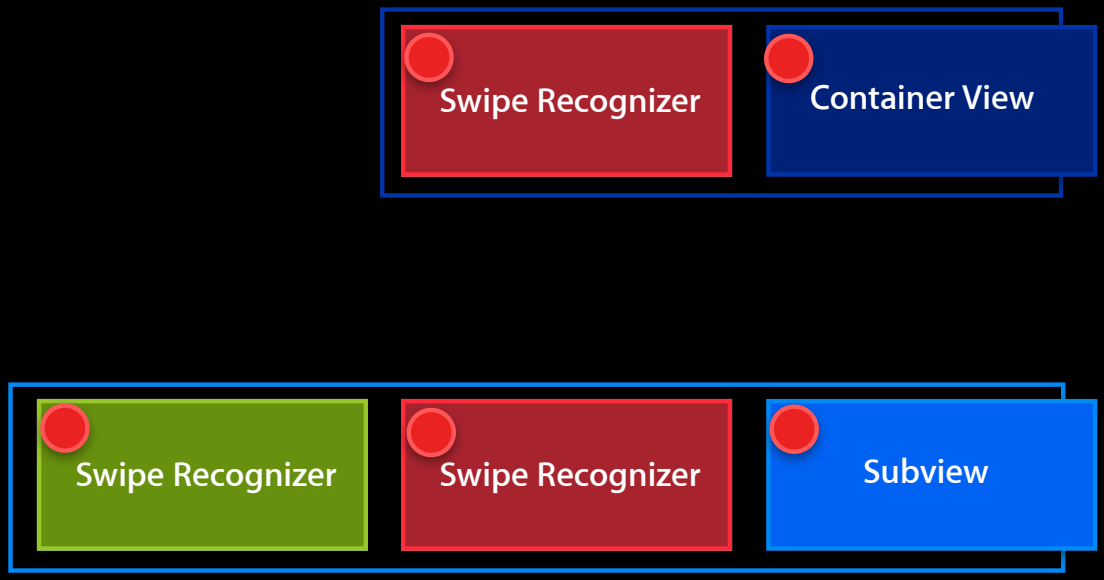
Conflict Resolution

Precedence



Conflict Resolution

Precedence



Conflict Resolution

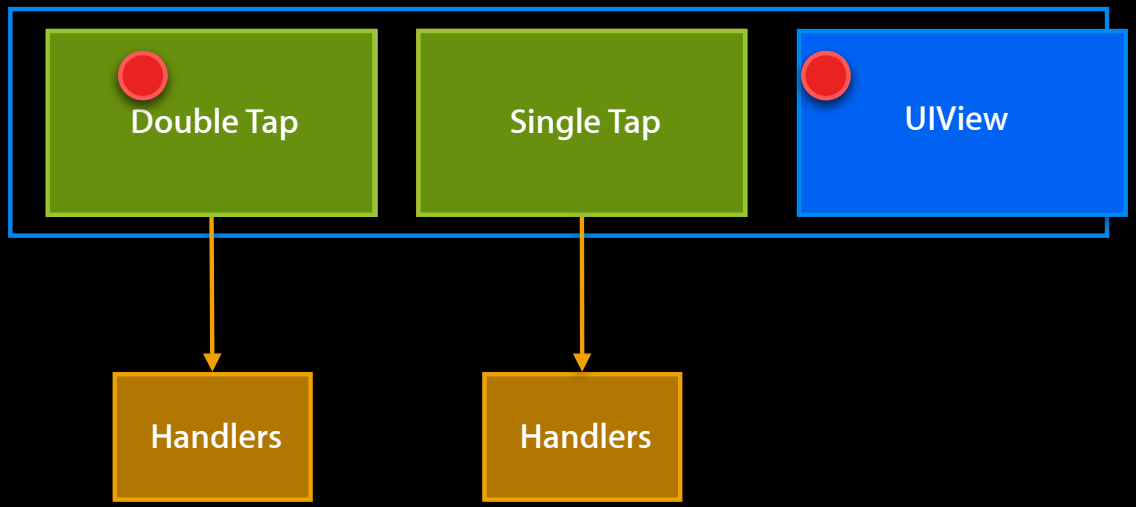
Exceptions to exclusivity

- Dependent gestures
  - e.g., tap, double-tap

Conflict Resolution

Dependent gestures

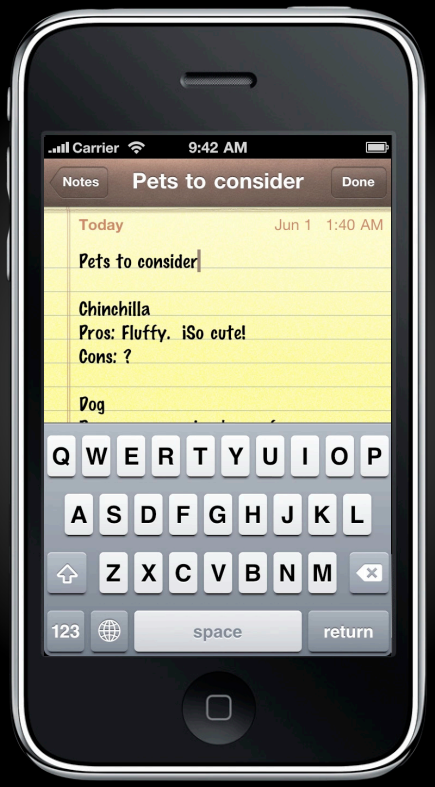
- Reasonable default behavior



Conflict Resolution

Dependent gestures

- Fires once for each gesture
- Great for stackable actions



Conflict Resolution

Dependent gestures

- Fires once for each gesture
- Great for stackable actions
- Poor for nonstackable actions

Conflict Resolution

Dependent gestures

- For nonstackable actions

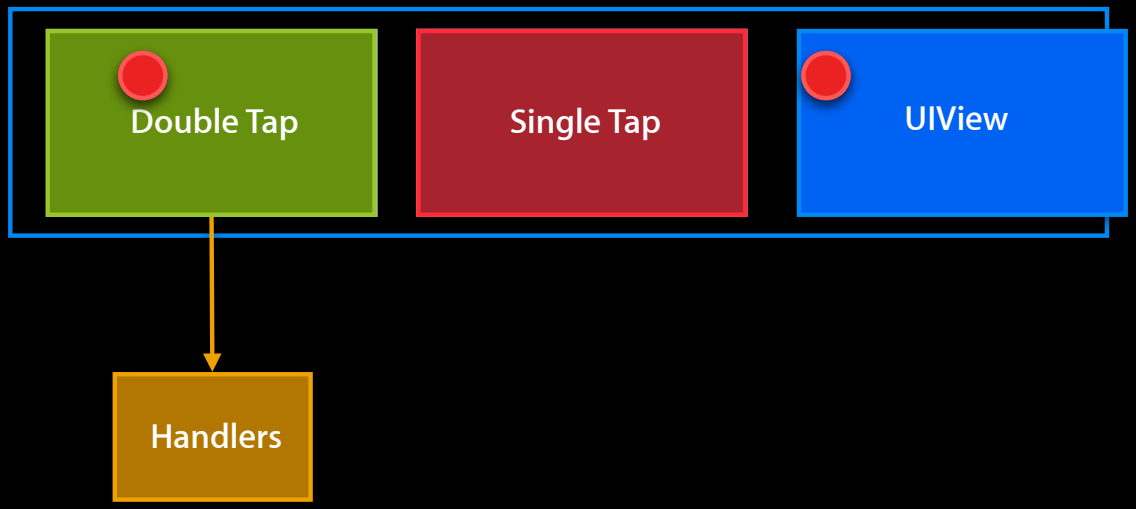
```
@interface UIGestureRecognizer
...
- (void)requireGestureRecognizerToFail:(UIGestureRecognizer *)recognizer;
@end
```

- Dependee waits for dependent to fail

```
[singleTap requireGestureRecognizerToFail:doubleTap];
```

Conflict Resolution

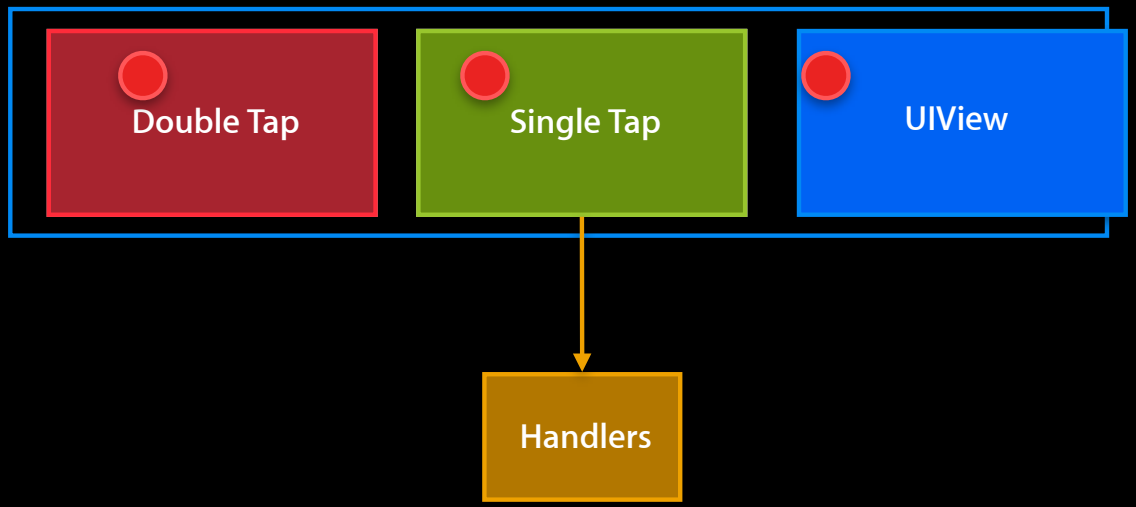
Dependent gestures



Conflict Resolution

Dependent gestures

- Introduces latency!



Conflict Resolution

Exceptions to exclusivity

- Dependent gestures
  - e.g., tap, double-tap
- Compatible gestures
  - e.g., rotate, pinch

Conflict Resolution

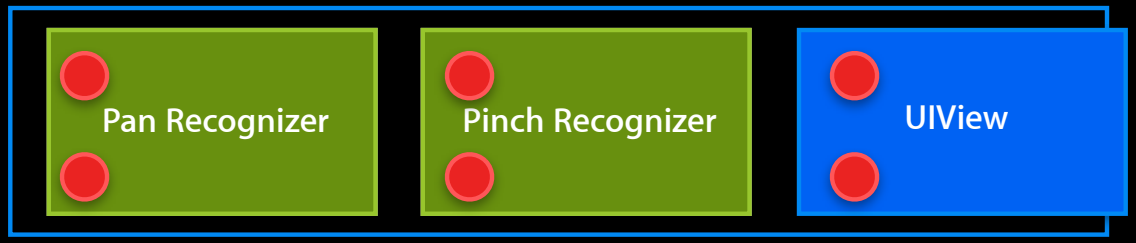
Compatible gestures

- Built-ins never assume compatibility
  - But delegate can override

```
@protocol UIGestureRecognizerDelegate

...

- (BOOL)gestureRecognizer:(UIGestureRecognizer *)gestureRecognizer
        shouldRecognizeSimultaneouslyWithGestureRecognizer:(UIGestureRecognizer *)
                                                           recognizer;


@end
```

Conflict Resolution

Gestures in conflict



Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Hybrid Event Handling

- Designed for mix-and-match
  - Easily add gestures to existing apps
  - Be ready for cancelled touches
- Not a replacement for raw events
  - Builds upon, exposes raw touches
  - There's no piano gesture
- Stay for the next session

Topics

- Touch handling
- Gesture handling
- How it works
- Using the API
- Conflict resolution
- Hybrid event handling

Demo

Non-exclusive gestures

Josh Shaffer
UIKit God

Gesture Recognizers

- Why
  - Less code to write!
  - Work to handle gestures, not to detect them
  - Achieve consistency
- How
  - Instantiate concrete recognizer
  - Set target/action pairs
  - Configure with properties and delegate
  - Attach to a view

More Information

Bill Dudney
Application Frameworks Evangelist
dudney@apple.com

Documentation
Gesture Recognition
http://developer.apple.com/iphone/library/documentation/General/Conceptual/iPadProgrammingGuide/GestureSupport/GestureSupport.html

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

Related Sessions

Advanced Gesture Recognition	Pacific Heights Wednesday 4:30PM

Labs

Gesture Recognition Lab	Application Frameworks Lab A Thursday 2:00PM

WWDC10

The last slide after the logo is intentionally left blank for all