References

Event Handling Guide for iOS, Apple documentation

Beginning iPhone 3 Development, Mark & LaMarche, Chapters 13 & 15
Responders

Event is sent to view it occurs in

If it does not handle event it is passed on to super view (or controller)

Views and Controllers are responders
Responder methods

- (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

touches contains on UITouch object for each finger on screen
UITouch

Getting the Location of Touches
– locationInView:
– previousLocationInView:
  view (property)
  window (property)

Getting Touch Attributes
  tapCount (property)
  timestamp (property)
  phase (property)

Getting a Touch Object’s Gesture Recognizers
  gestureRecognizers (property)

Phases
UITouchPhaseBegan
UITouchPhaseMoved
UITouchPhaseStationary
UITouchPhaseEnded
UITouchPhaseCancelled
**UIEvent**

Getting theTouches for an Event
- allTouches
- touchesForView:
- touchesForWindow:

Getting Event Attributes
- timestamp (property)

Getting the Event Type
- type (property)
- subtype (property)

Getting the Touches for a Gesture Recognizer
- touchesForGestureRecognizer:

Types
- UIEventTypeTouches
- UIEventTypeMotion
- UIEventTypeRemoteControl

Subtypes
- UIEventSubtypeNone
- UIEventSubtypeMotionShake
- UIEventSubtypeRemoteControlPlay
- UIEventSubtypeRemoteControlPause
- etc.
Taps

- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
    NSUInteger numberTaps = [[touches anyObject] tapCount];
}

Counts as many taps as user does
Touches

- (void)touchesBegan:(NSSet *)Touches withEvent:(UIEvent *)event {
    NSUInteger numberTouchies = [[Touches count];
}

Up to 10 touches at once
Swipe 1

Measure how long from start to end of touch events

If touch ends in here - horizontal swipe

Start Swipe

Min x change required

Max Y change
Sample Code

- (void)touchesBegan:(NSSet *)touches withEvent:(UIEvent *)event {
    UITouch *touch = [touches anyObject];
    gestureStartPoint = [touch locationInView:self.view];
}

- (void)touchesMoved:(NSSet *)touches withEvent:(UIEvent *)event {
    UITouch *touch = [touches anyObject];
    CGPoint currentPosition = [touch locationInView:self.view];

    CGFloat deltaX = fabsf(gestureStartPoint.x - currentPosition.x);
    CGFloat deltaY = fabsf(gestureStartPoint.y - currentPosition.y);

    if (deltaX >= kMinimumGestureLength && deltaY <= kMaximumVariance) {
        tapsLabel.text = @"Horizontal swipe detected";
    }
}
## Gesture Recognizers

<table>
<thead>
<tr>
<th>Gesture</th>
<th>UIKit class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tapping (any number of taps)</td>
<td>UITapGestureRecognizer</td>
</tr>
<tr>
<td>Pinching in and out (for zooming a view)</td>
<td>UIPinchGestureRecognizer</td>
</tr>
<tr>
<td>Panning or dragging</td>
<td>UIPanGestureRecognizer</td>
</tr>
<tr>
<td>Swiping (in any direction)</td>
<td>UISwipeGestureRecognizer</td>
</tr>
<tr>
<td>Rotating (fingers moving in opposite directions)</td>
<td>UIRotationGestureRecognizer</td>
</tr>
<tr>
<td>Long press (&quot;touch and hold&quot;)</td>
<td>UILongPressGestureRecognizer</td>
</tr>
</tbody>
</table>
Swipe

- (void) viewDidLoad {
    UISwipeGestureRecognizer *swipeGesture = [[UISwipeGestureRecognizer alloc] initWithTarget:self action:@selector(handleSwipeGesture:)];
    [self.view addGestureRecognizer:swipeGesture];
    [swipeGesture release];
}

- (IBAction) handleSwipeGesture: (UISwipeGestureRecognizer *) sender {
    NSLog(@"state %i", sender.state);
    touchesLabel.text = @"swiper";
}
Gesture Recognizer States

UIGestureRecognizerStatePossible
    Start state of gesture

UIGestureRecognizerStateBegan
UIGestureRecognizerStateChanged
    In middle of gesture

UIGestureRecognizerStateEnded
UIGestureRecognizerStateCancelled
UIGestureRecognizerStateFailed
UIGestureRecognizerStateRecognized = UIGestureRecognizerStateEnded
Pan Gesture Example

- (void) viewDidLoad {
    UIPanGestureRecognizer *panGesture = [[UIPanGestureRecognizer alloc]
        initWithTarget:self action:@selector
        (handlePanGesture:)];
    [self.view addGestureRecognizer:panGesture];
    [panGesture release];
}

- (IBAction) handlePanGesture: (UIPanGestureRecognizer *) sender {
    NSLog(@"state %i", sender.state);
    CGPoint location = [sender locationInView:self.view];
    NSLog(@"Location x: %f, y: %f", location.x, location.y );
    NSLog(@"Number of touches %i", [sender numberOfTouches] );
}
Gesture Features

Have one gesture recognizer fail before another can start analyzing touch events.

Prevent other gesture recognizers from analyzing a specific multitouch sequence or a touch object in that sequence.

 Permit two gesture recognizers to operate simultaneously.
Custom Gesture Recognizers

Can create gesture recognizer for custom gestures
Motion Events
Shake Events

Shake motion events are sent to first responder

- (BOOL)canBecomeFirstResponder {
    return YES;
}

- (void)viewDidLoad {
    [self becomeFirstResponder];
}

- (void)viewDidLoad {
    [self becomeFirstResponder];
}
Motion Events

- (void)motionBegan:(UIEventSubtype)motion withEvent:(UIEvent *)event
  {
    NSLog(@"motion began");
  }

- (void)motionEnded:(UIEventSubtype)motion withEvent:(UIEvent *)event
  {
    NSLog(@"motion ended");
  }

- (void)motionCancelled:(UIEventSubtype)motion withEvent:(UIEvent *)event
  {
    NSLog(@"touches cancelled");
  }
Orientation

UIDevice * device = [UIDevice currentDevice];
[device beginGeneratingDeviceOrientationNotifications];

a bit later

UIDeviceOrientation orientation = device.orientation;

when done

[device endGeneratingDeviceOrientationNotifications];
Possible Orientations

UIDeviceOrientationUnknown
UIDeviceOrientationPortrait
UIDeviceOrientationPortraitUpsideDown
UIDeviceOrientationLandscapeLeft
UIDeviceOrientationLandscapeRight
UIDeviceOrientationFaceUp
UIDeviceOrientationFaceDown
Accelerometer

If use accelerometer must state so in Plist before submitting to Apple
  Required device capabilities
    accelerometer

Single instance of UIAccelerometer
  Set frequency of data
  Set delegate
#define kAccelerometerFrequency        20.0 //Hz

UIAccelerometer* theAccelerometer = [UIAccelerometer sharedAccelerometer];
theAccelerometer.updateInterval = 1 / kAccelerometerFrequency;
theAccelerometer.delegate = self;
Delegate method

- (void)accelerometer:(UIAccelerometer *)accelerometer
didAccelerate:(UIAcceleration *)acceleration
{
    UIAccelerationValue x, y, z;
    x = acceleration.x;
    y = acceleration.y;
    z = acceleration.z;

    NSLog(@"X: %f, Y: %f, Z: %f", x, y, z);
}

UIAcceleration only contains x, y, z
When Done

```
UILocalization* theAccelerometer = [UILocalization sharedAccelerometer];
theAccelerometer.delegate = nil;
```
## Update Intervals

<table>
<thead>
<tr>
<th>Event frequency (Hz)</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>10–20</td>
<td>Determining the current orientation of the device.</td>
</tr>
<tr>
<td>30–60</td>
<td>Suitable for games and other applications that use the accelerometers for real-time user input.</td>
</tr>
<tr>
<td>70–100</td>
<td>Suitable for applications that need to detect high-frequency motion.</td>
</tr>
</tbody>
</table>
Core Motion

Framework for motion data