Swift Study 08

2017. 01.07

Swift 문법 & ios

- Selector

- UIGestureRecognizer

Selector

- Objective-C는 C의 함수 포인터와 유사한 개념으로 'SEL' 이라는 데이터 타입을 지원
- @selector 지시어와 임의의 메소드 이름을 사용하여 값을 설정하여 특정함수를 가르키게 함.
- Swift 3.0부터는 #selector(함수) 형태로 선언하여 해당함수를 가르킴

```
//생성할 클래스 Test
class Test {
    public init(target: Any?, action: Selector?)
}
```

```
//특정 클래스에 정의된 함수
func add(_ number:int){
    //덧셈연산
}

//Test 클래스에게 동작에 필요한 add()함수 지정
Test(target: self, action: #selector(self.add(_:)))
```

UIGestureRecognizer

- UIView의 제스처(동작행위)에 대해 이벤트리스너 클래스 (recognizer)
- 제스처의 swipe(방향지시) / pan(drag) / tab / rotate 등 다양한 서브클래스 존재

Figure 1-1 A gesture recognizer attached to a view

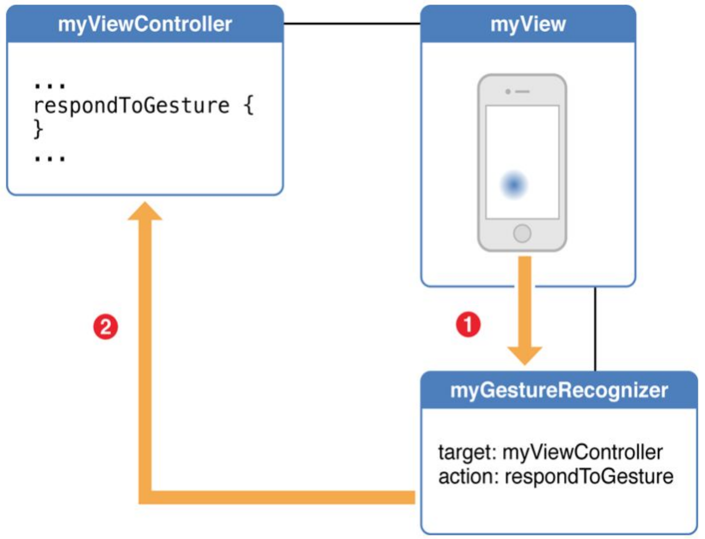


Table 1-1 Gesture recognizer classes of the UIKit framework

Gesture	UIKit class
Tapping (any number of taps)	`UITapGestureRecognizer`
Pinching in and out (for zooming a view)	`UIPinchGestureRecognizer`
Panning or dragging	`UIPanGestureRecognizer`
Swiping (in any direction)	`UISwipeGestureRecognizer`
Rotating (fingers moving in opposite directions)	`UIRotationGestureRecognizer`
Long press (also known as "touch and hold")	`UILongPressGestureRecognizer`

UIGestureRecognizer

- discrete(구분된) / contunuous(연석적) 두가지 종류의 형태로 상태방식 존재

Figure 1-2 Discrete and continuous gestures

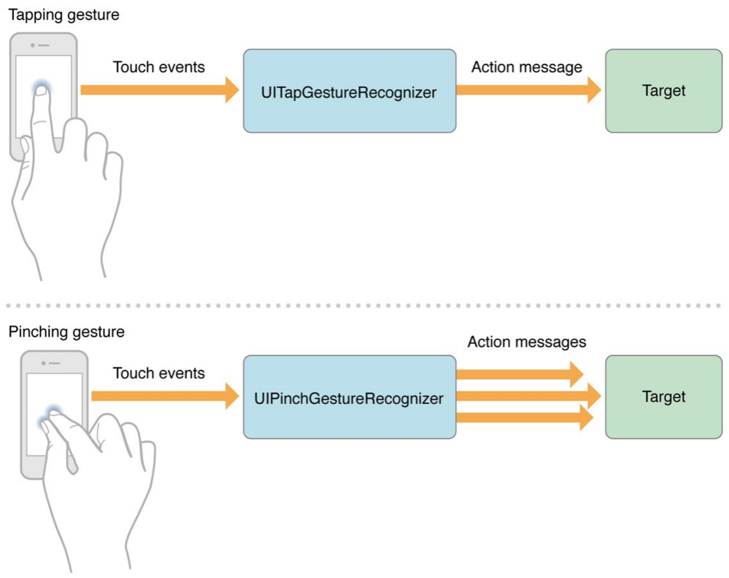
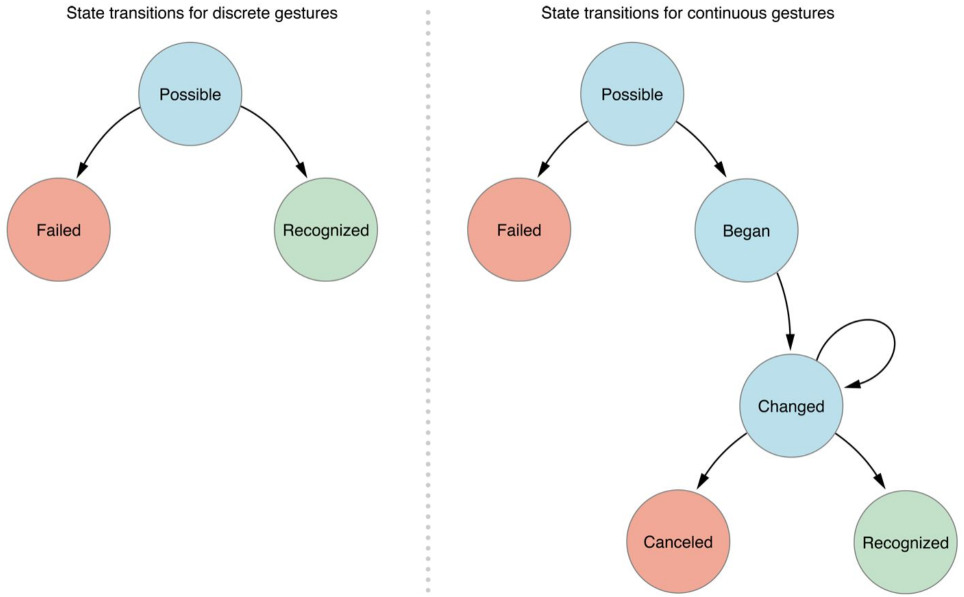


Figure 1-3 State machines for gesture recognizers

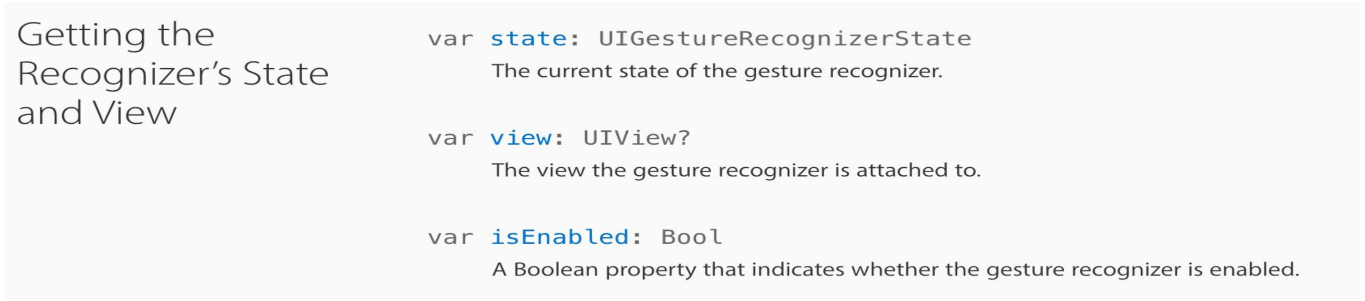

State transitions for discrete gestures
Possible
Failed
Recognized
State transitions for continuous gestures
Possible
Failed
Began
Changed
Canceled
Recognized

UIGestureRecognizer

The concrete subclasses of `UIGestureRecognizer` are the following:

• `UITapGestureRecognizer`	터치에 대한 제스처 감지기
• `UIPinchGestureRecognizer`	확대/축소에 대한 제스처 감지기(주로 두손가락 오므리고/펴기 제스처)
• `UIRotationGestureRecognizer`	회전(각도)에 대한 제스처 감지기(주로 두손가락으로 돌리는 제스처)
• `UISwipeGestureRecognizer`	방향에 대한 제스처 감지기(슬라이식 손가락 제스처)
• `UIPanGestureRecognizer`	특정 이동좌표에 대한 제스처 감지기
• `UIScreenEdgePanGestureRecognizer`	화면 가장자리에 대한 제스처 감지기
• `UILongPressGestureRecognizer`	긴 터치에 대한 제스처 감지기

UIGestureRecognizer

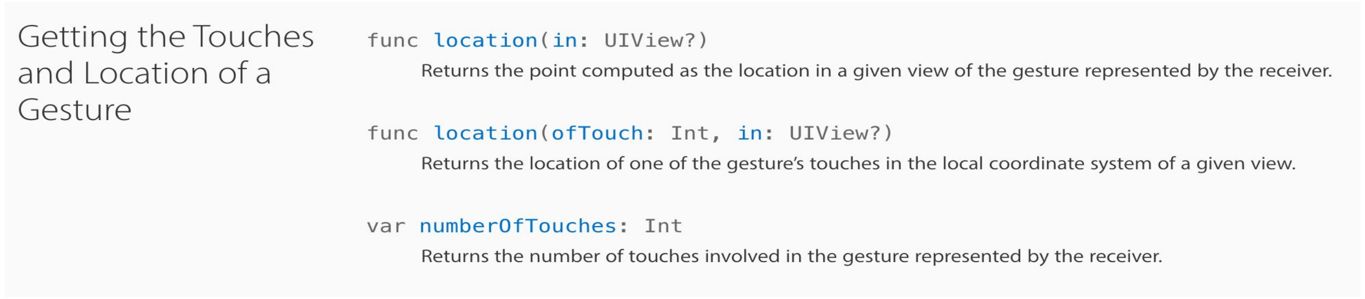


Getting the Recognizer's State and View

```
var state: UIGestureRecognizerState
```
The current state of the gesture recognizer.

```
var view: UIView?
```
The view the gesture recognizer is attached to.

```
var isEnabled: Bool
```
A Boolean property that indicates whether the gesture recognizer is enabled.

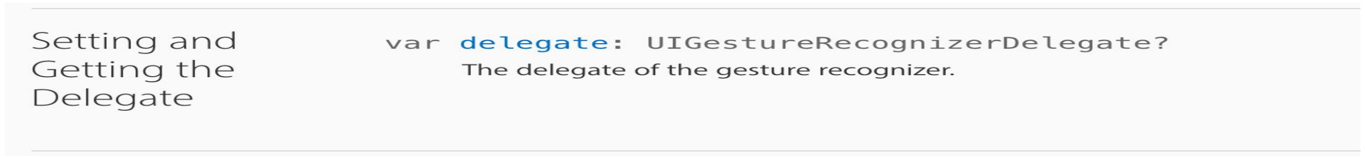


Getting the Touches and Location of a Gesture

```
func location(in: UIView?)
```
Returns the point computed as the location in a given view of the gesture represented by the receiver.

```
func location(ofTouch: Int, in: UIView?)
```
Returns the location of one of the gesture's touches in the local coordinate system of a given view.

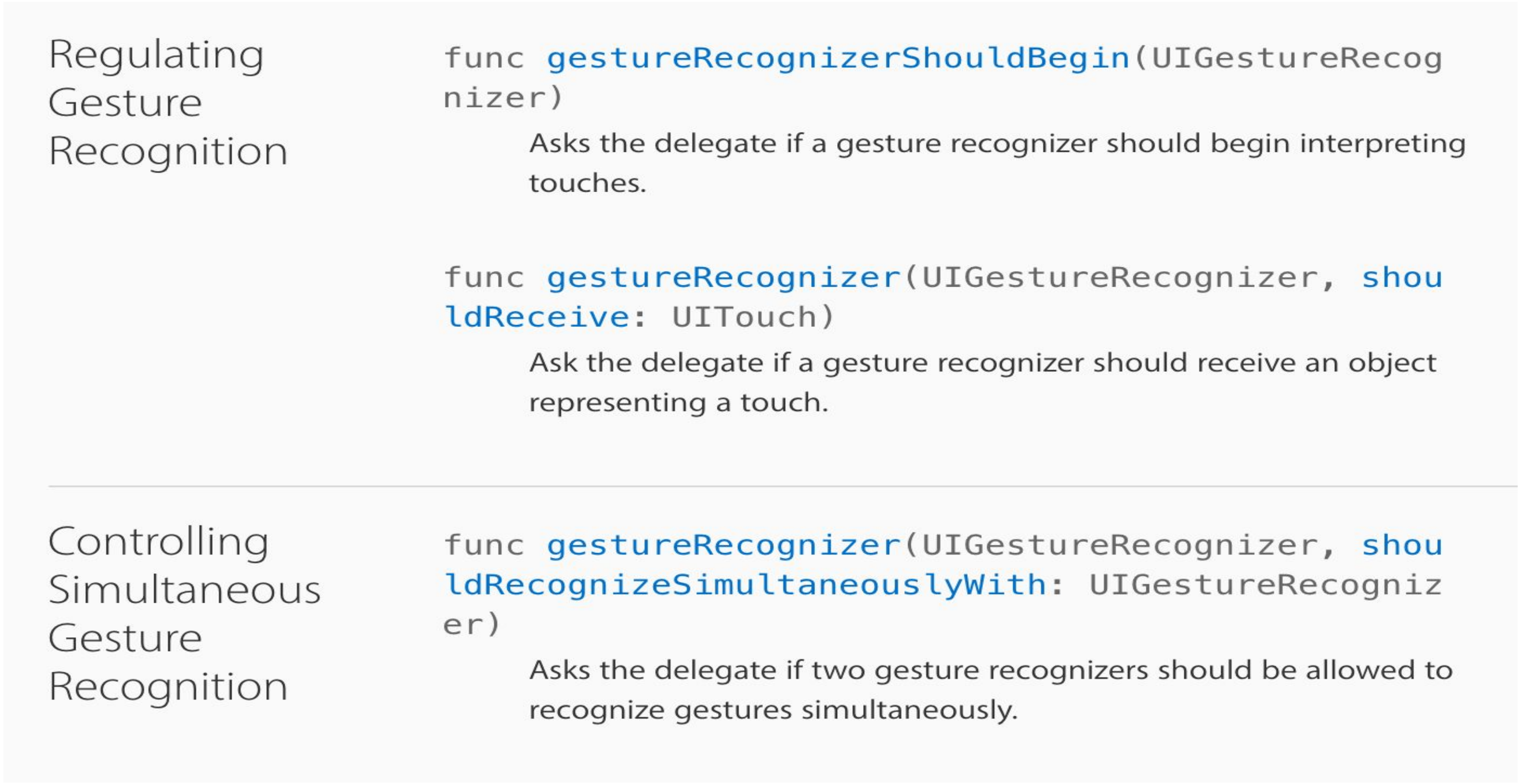
```
var numberOfTouches: Int
```
Returns the number of touches involved in the gesture represented by the receiver.



Setting and Getting the Delegate

```
var delegate: UIGestureRecognizerDelegate?
```
The delegate of the gesture recognizer.

UIGestureRecognizerDelegate



Regulating Gesture Recognition

```
func gestureRecognizerShouldBegin(UIGestureRecog
nizer)
```

Asks the delegate if a gesture recognizer should begin interpreting touches.

```
func gestureRecognizer(UIGestureRecognizer, shou
ldReceive: UITouch)
```

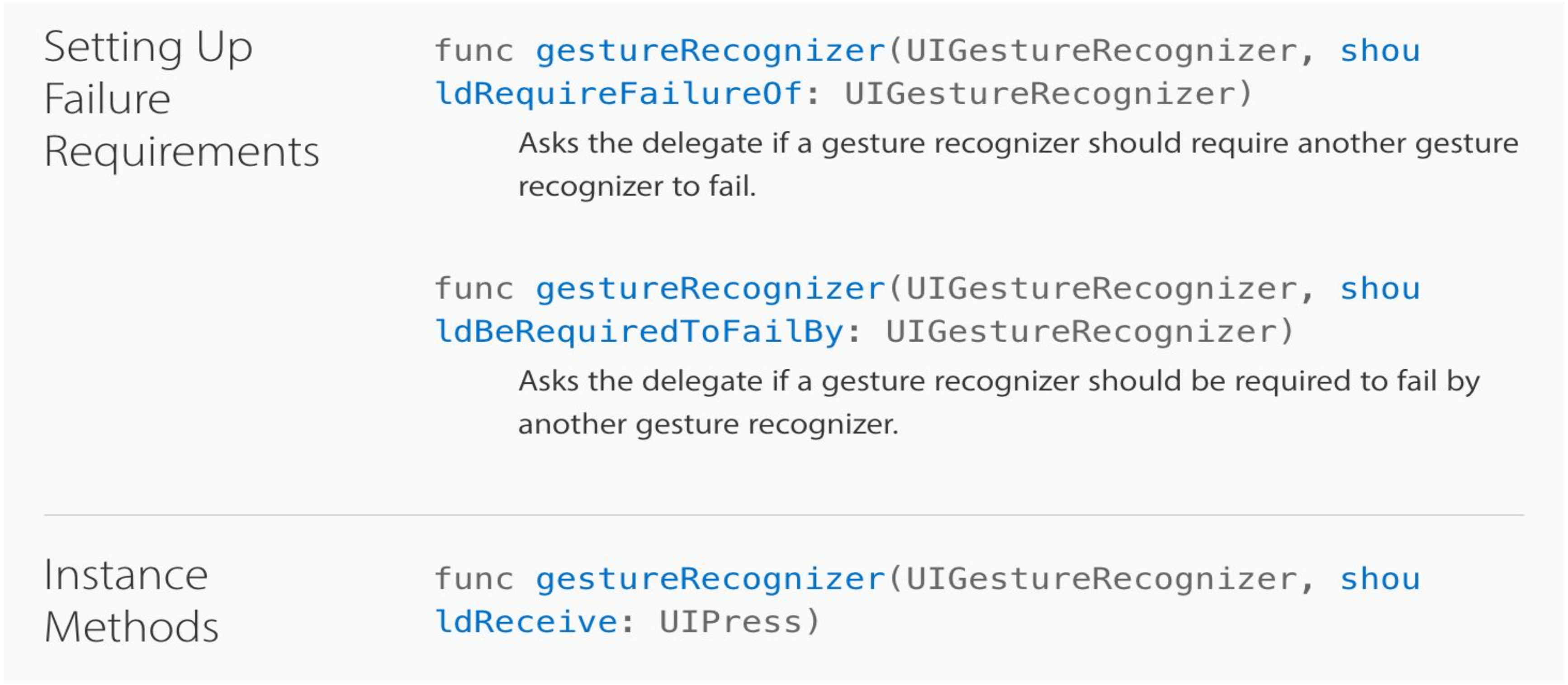
Ask the delegate if a gesture recognizer should receive an object representing a touch.

Controlling Simultaneous Gesture Recognition

```
func gestureRecognizer(UIGestureRecognizer, shou
ldRecognizeSimultaneouslyWith: UIGestureRecogniz
er)
```

Asks the delegate if two gesture recognizers should be allowed to recognize gestures simultaneously.

UIGestureRecognizerDelegate



Setting Up Failure Requirements

```
func gestureRecognizer(UIGestureRecognizer, shou
ldRequireFailureOf: UIGestureRecognizer)
```

Asks the delegate if a gesture recognizer should require another gesture recognizer to fail.

```
func gestureRecognizer(UIGestureRecognizer, shou
ldBeRequiredToFailBy: UIGestureRecognizer)
```

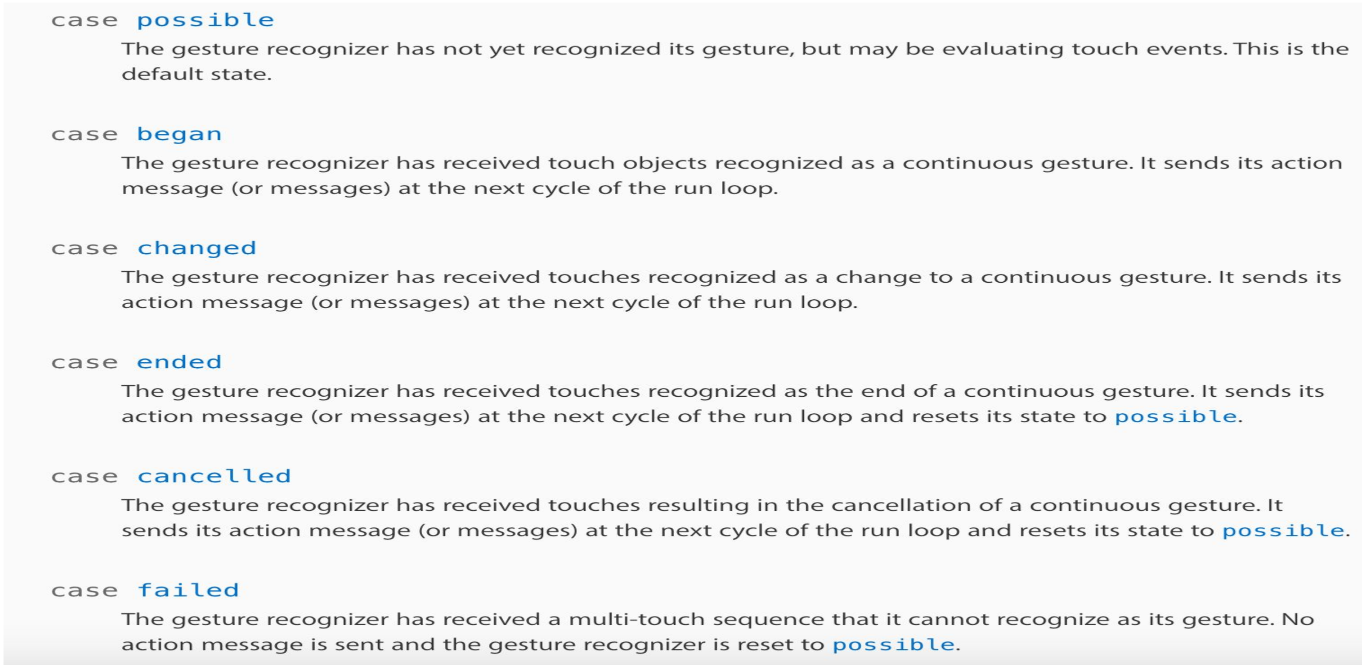
Asks the delegate if a gesture recognizer should be required to fail by another gesture recognizer.

Instance Methods

```
func gestureRecognizer(UIGestureRecognizer, shou
ldReceive: UIPress)
```

UIGestureRecognizerState

- UIGestureRecognizer의 상태값
- enum 타입값 형태



`case possible`
The gesture recognizer has not yet recognized its gesture, but may be evaluating touch events. This is the default state.

`case began`
The gesture recognizer has received touch objects recognized as a continuous gesture. It sends its action message (or messages) at the next cycle of the run loop.

`case changed`
The gesture recognizer has received touches recognized as a change to a continuous gesture. It sends its action message (or messages) at the next cycle of the run loop.

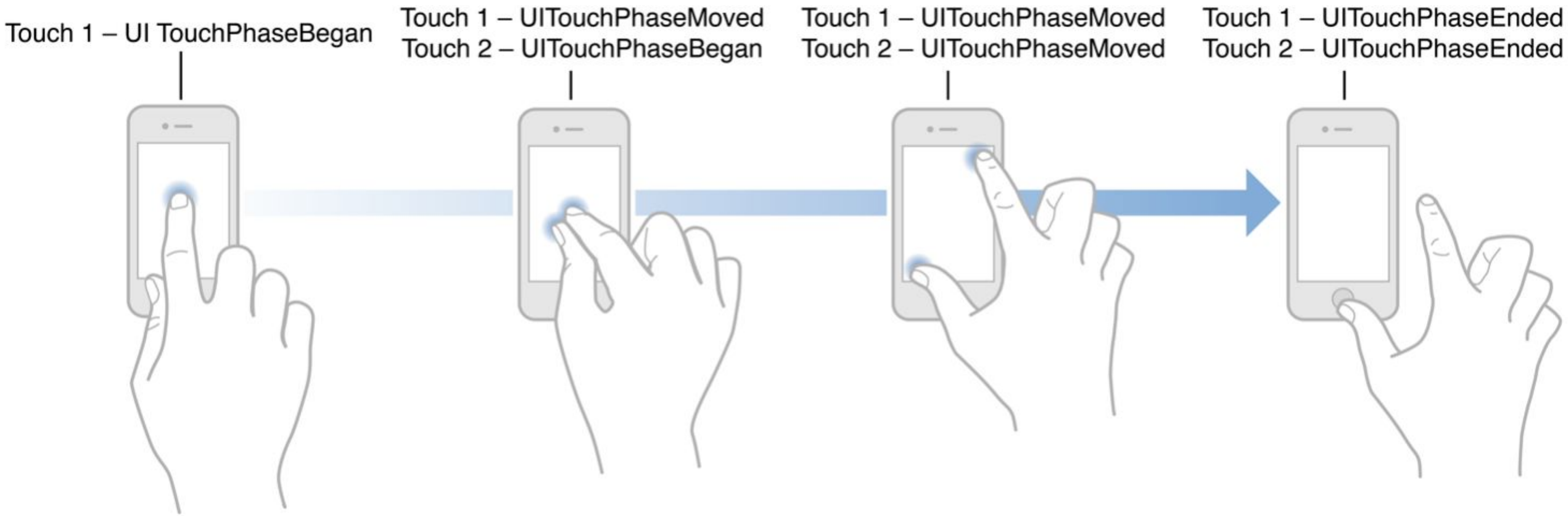
`case ended`
The gesture recognizer has received touches recognized as the end of a continuous gesture. It sends its action message (or messages) at the next cycle of the run loop and resets its state to `possible`.

`case cancelled`
The gesture recognizer has received touches resulting in the cancellation of a continuous gesture. It sends its action message (or messages) at the next cycle of the run loop and resets its state to `possible`.

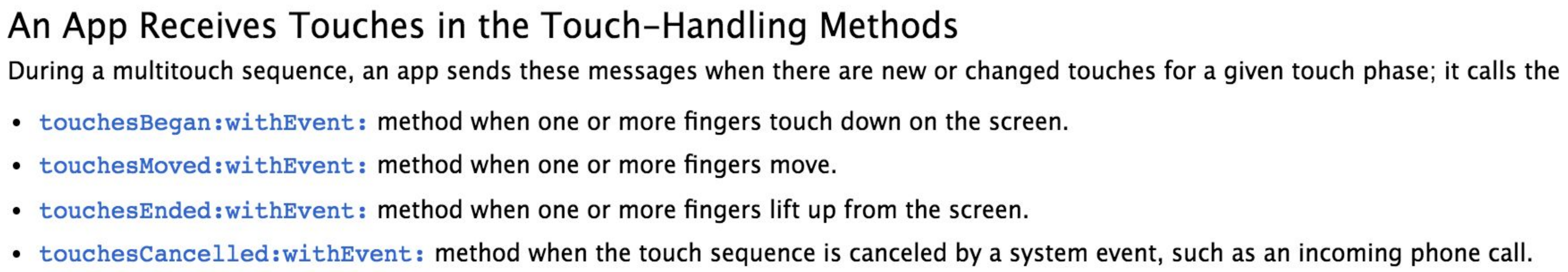
`case failed`
The gesture recognizer has received a multi-touch sequence that it cannot recognize as its gesture. No action message is sent and the gesture recognizer is reset to `possible`.

UIGestureRecognizer - phase

Figure 1-4 A multitouch sequence and touch phases


Touch 1 – UI TouchPhaseBegan
Touch 1 – UITouchPhaseMoved
Touch 2 – UITouchPhaseBegan
Touch 1 – UITouchPhaseMoved
Touch 2 – UITouchPhaseMoved
Touch 1 – UITouchPhaseEnded
Touch 2 – UITouchPhaseEnded

UIGestureRecognizer - phase



An App Receives Touches in the Touch-Handling Methods

During a multitouch sequence, an app sends these messages when there are new or changed touches for a given touch phase; it calls the

- `touchesBegan:withEvent:` method when one or more fingers touch down on the screen.
- `touchesMoved:withEvent:` method when one or more fingers move.
- `touchesEnded:withEvent:` method when one or more fingers lift up from the screen.
- `touchesCancelled:withEvent:` method when the touch sequence is canceled by a system event, such as an incoming phone call.

UIGestureRecognizer - 구현방식

1) UIGestureRecognizer를 생성 및 action에 대한 정의

2) 생성한 UIGestureRecognizer를 해당 UIView에 등록

```
// 1) UIGestureRecognizer 생성 및 action 적용
let swipeUp = UISwipeGestureRecognizer(target: self, action: 실행할 특정함수 )

// UISwipeGestureRecognizer에 대한 direction 지정 (up에 대한 제스처 감지기)
swipeUp.direction = UISwipeGestureRecognizerDirection.up

// 2) UIGestureRecognizer를 화면에 등록 (메인화면에 up swipe에 대한 제스처감지기 등록)
self.view.addGestureRecognizer(swipeUp)
```

* swipeGesture 경우 특정방향에 direction값 제공
* 제스처 발생시 등록된 특정함수를 호출

UIGestureRecognizer - 예제코드

- 4방향에 대한 swipeGestureRecoginzer 등록

```
let directions: [UISwipeGestureRecognizerDirection] = [.right, .left, .up, .down]

for direction in directions {
        let swipe = UISwipeGestureRecognizer(
                                target: self,
                                action: #selector(ViewController.respondSwipeGesture(_:))
         )
        swipe.direction = direction
        self.view.addGestureRecognizer(swipe)
}
```

UIGestureRecognizer - 예제코드

- 4방향의 제스처 발생시 정의한 특정함수 respondSwipeGesture(_:) 실행

```
func respondSwipeGesture(_ gestrue:UISwipeGestureRecognizer){
        upImageView.image = imageUp[0]
        downImageView.image = imageDown[0]
        leftImageView.image = imageLeft[0]
        rightImageView.image = imageRight[0]

        switch gestrue.direction {
                case UISwipeGestureRecognizerDirection.up: upImageView.image = imageUp[1]
                case UISwipeGestureRecognizerDirection.down: downImageView.image = imageDown[1]
                case UISwipeGestureRecognizerDirection.left: leftImageView.image = imageLeft[1]
                case UISwipeGestureRecognizerDirection.right: rightImageView.image = imageRight[1]
                default: break
        }
}
```