

iPhone App Dev

Lesson 3

Resource

<https://github.com/bryanttang/iOS-Class-2015-9>

Practice

✓ Show your Converter

Summary

- Review
- Class (Advance)
- UIView
- UIViewController
- Gesture
- Animation

Basic Class

Calculator

Attributes

result
M1
M2
M3

Function

-Add
-Sub
-Cross
-Div

Class(Advance)

- Example `[UIView alloc]`

NSObject + alloc:

UIView + animateWithDuration: animations:

Class(Advance)

- Class Method

Declare: **+ methodName:**

Implement: **+ methodName:(id)params{**

 }

Class(Advance)

- Example: TranslateHelper

ContentHelper

Attributes

Function

-(NSString)ContentTranslateCN: EN: PT:
-(Bool)ContentIsPhoneNumber:
-(Bool)ContentIsEmail:

Character

- Self and Super

Car

Bus

Car's super is Object

Bus's super is Car

Car's self is Car

Bus's self is Bus

Character

- Example:

```
self.color = [UIColor blueColor];
```

```
[super init];
```

Setter & Getter

{ Car.color = [UIColor redColor];
UIColor *color = Car.color

Instance Property

- What is **Setter** and **Getter** exactly?

- **Getter**

```
1 - (NSString *) something
2 {
3     return something;
4 }
```

- **Setter**

```
5 - (void) setSomething: (NSString*) newSomething
6 {
8     something = newSomething;
9 }
```

Instance Property

something is one of property inside **ObjectA**

For: @property (strong) NSString *something;

id a = **ObjectA.something**;

ObjectA.something = otherthing;

For: NSString *something;

id a = **ObjectA.something**;

ObjectA.something = otherthing;

Variable Ownership

- Strong and Weak?

`NSString *name = @"Jobs"`



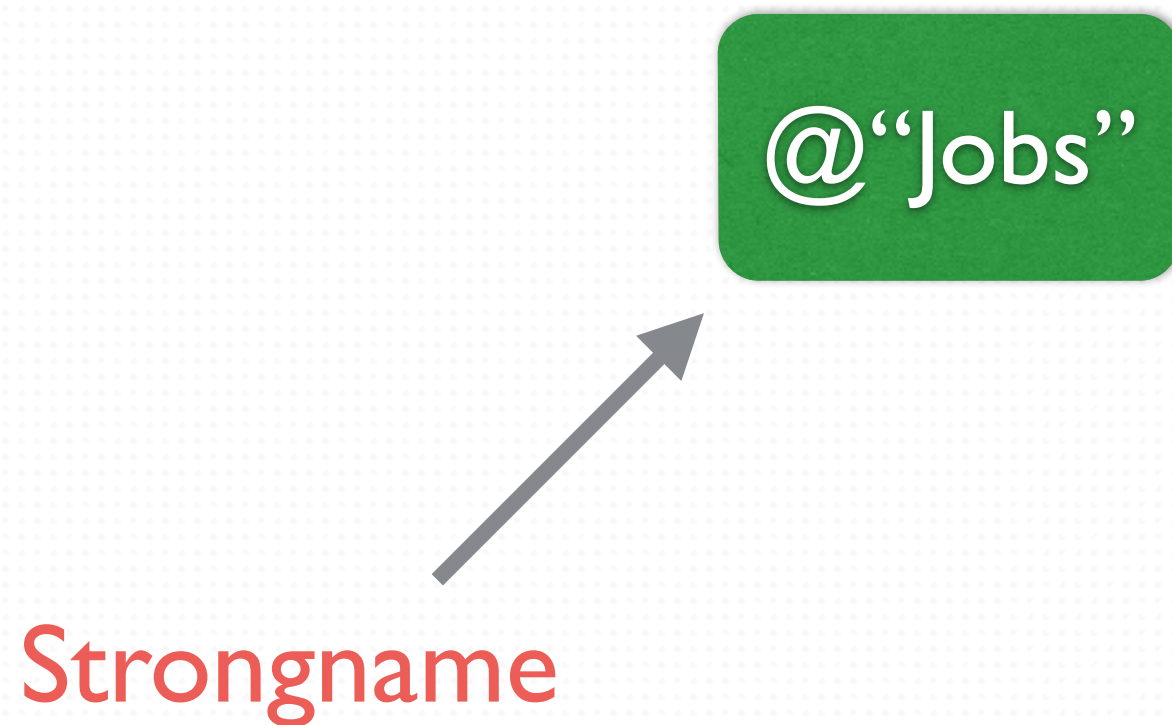
Pointer



Memory

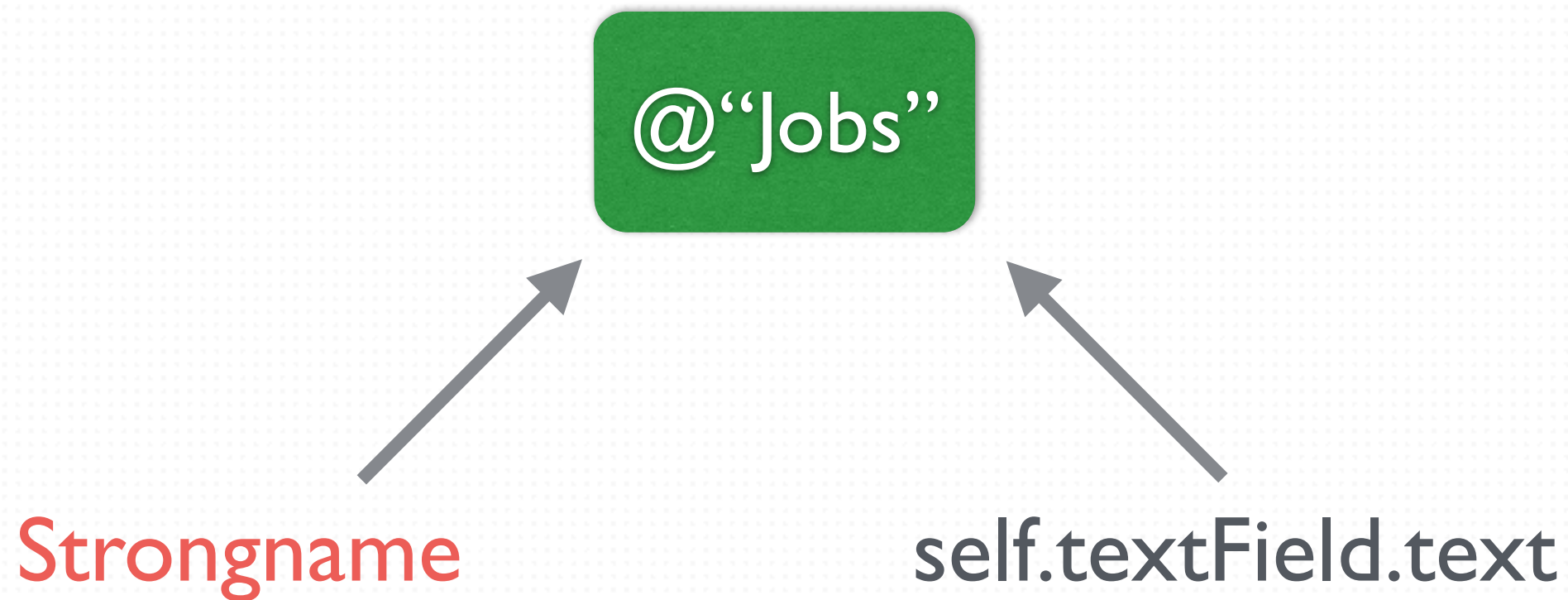
Variable Ownership

- Strong



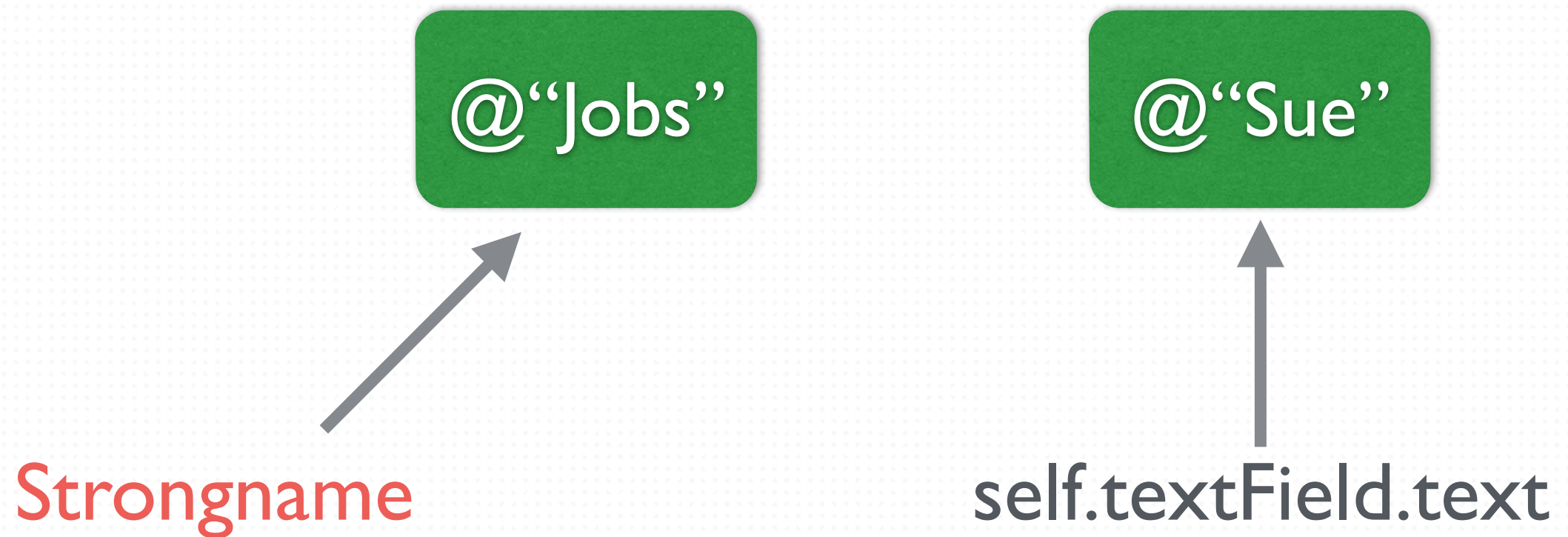
Variable Ownership

- Strong



Variable Ownership

- Strong



Variable Ownership

● Strong

@""

Strongname

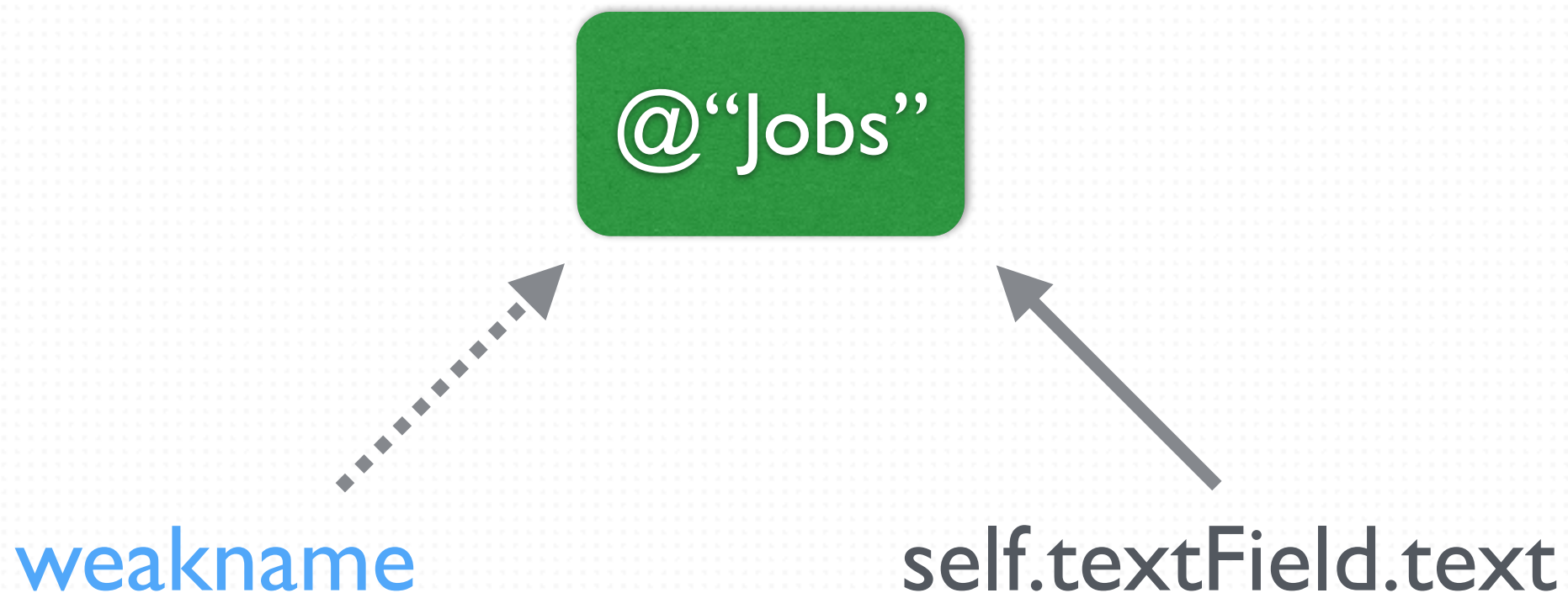
~~@“Jobs”~~

@“Sue”

self.textField.text

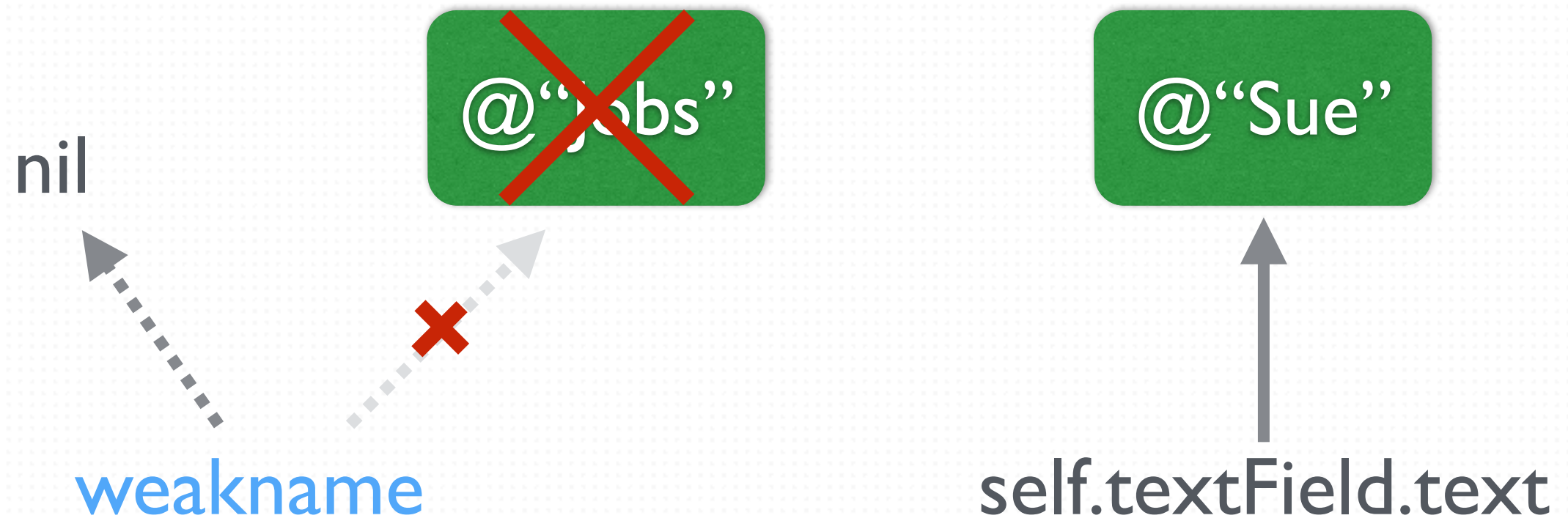
Variable Ownership

- Weak



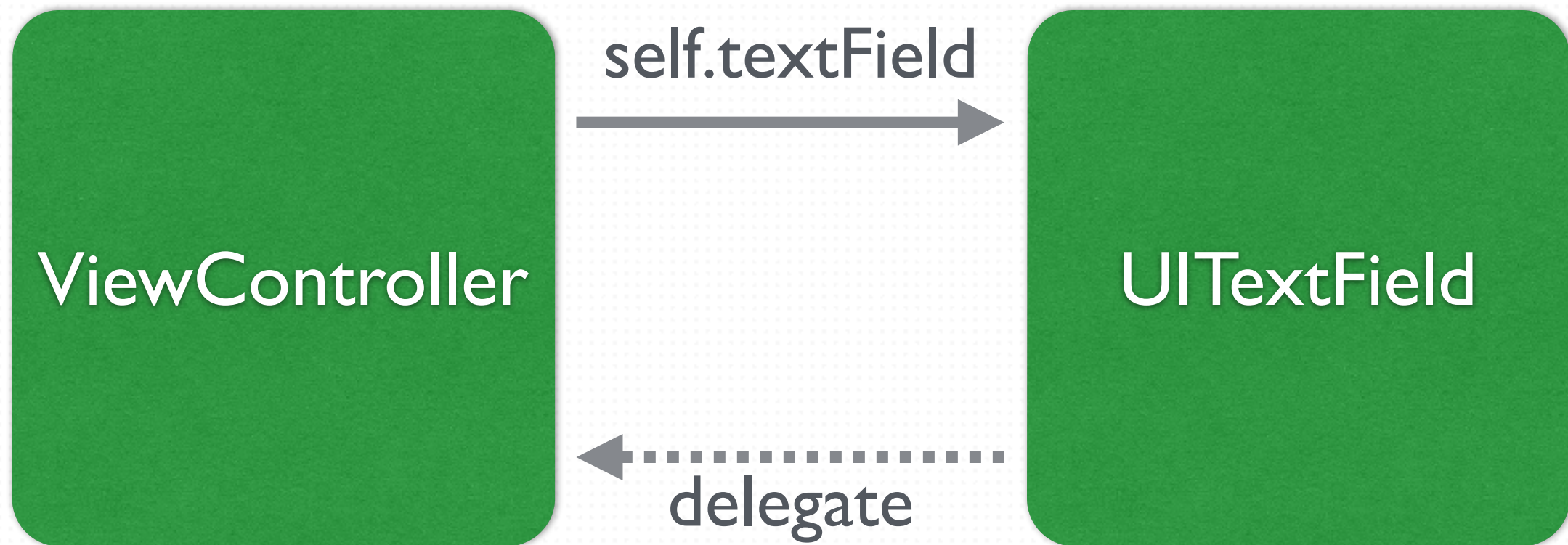
Variable Ownership

● Weak



Variable Ownership

- Example

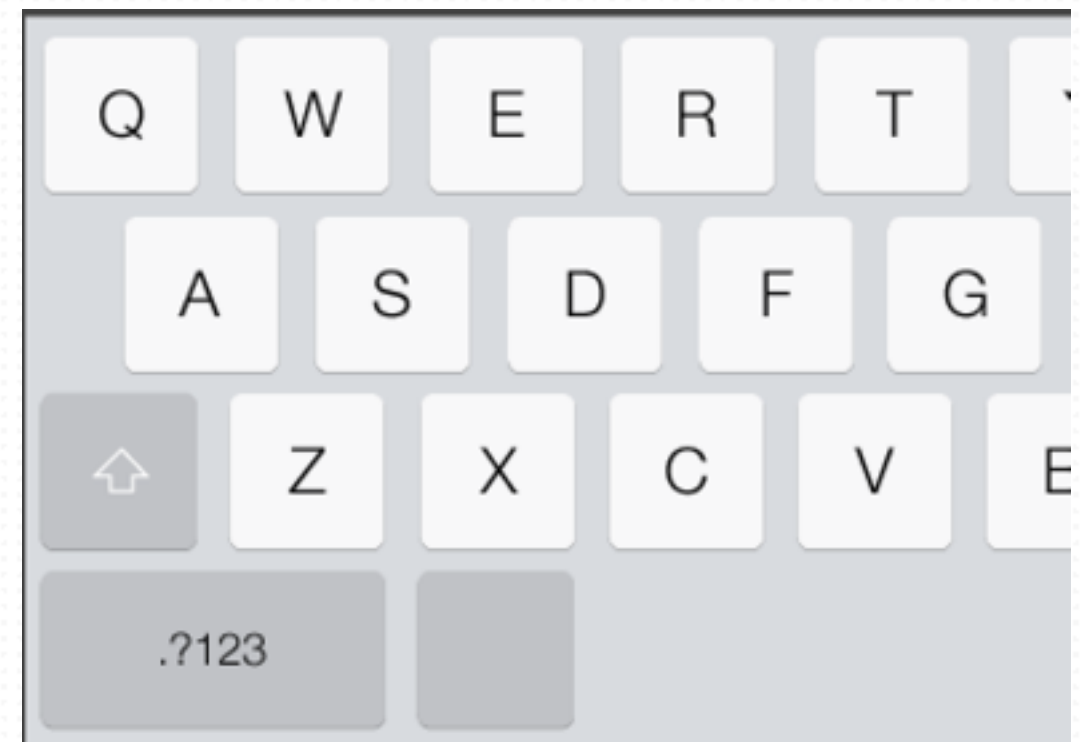
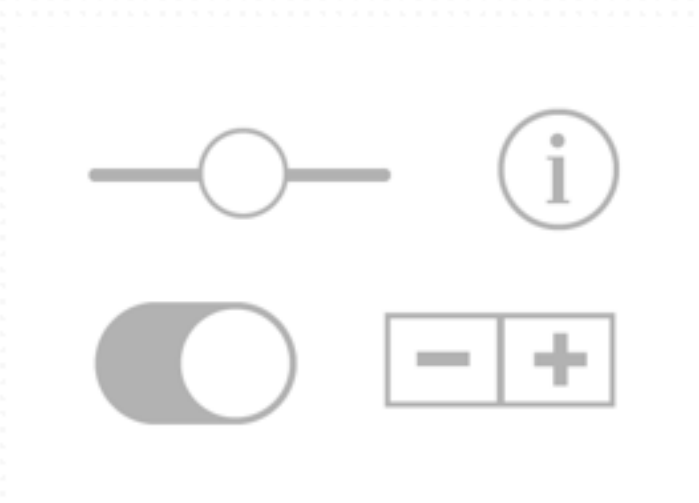
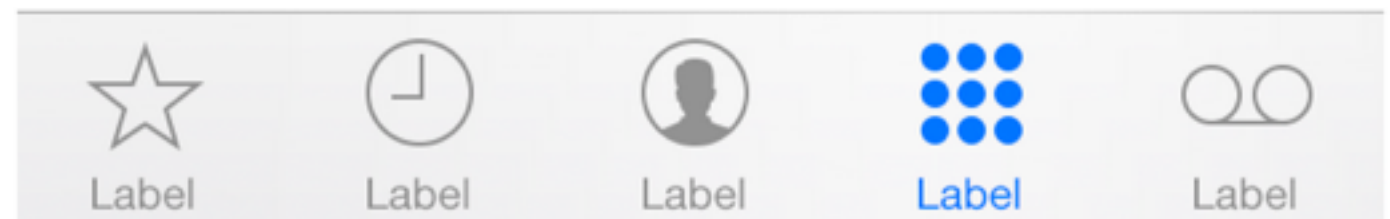
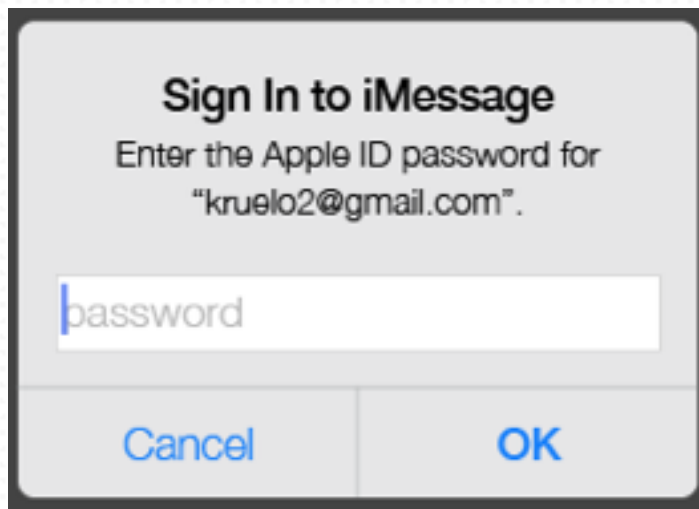


UI

UI is what you see

UIView

- What is UIView?



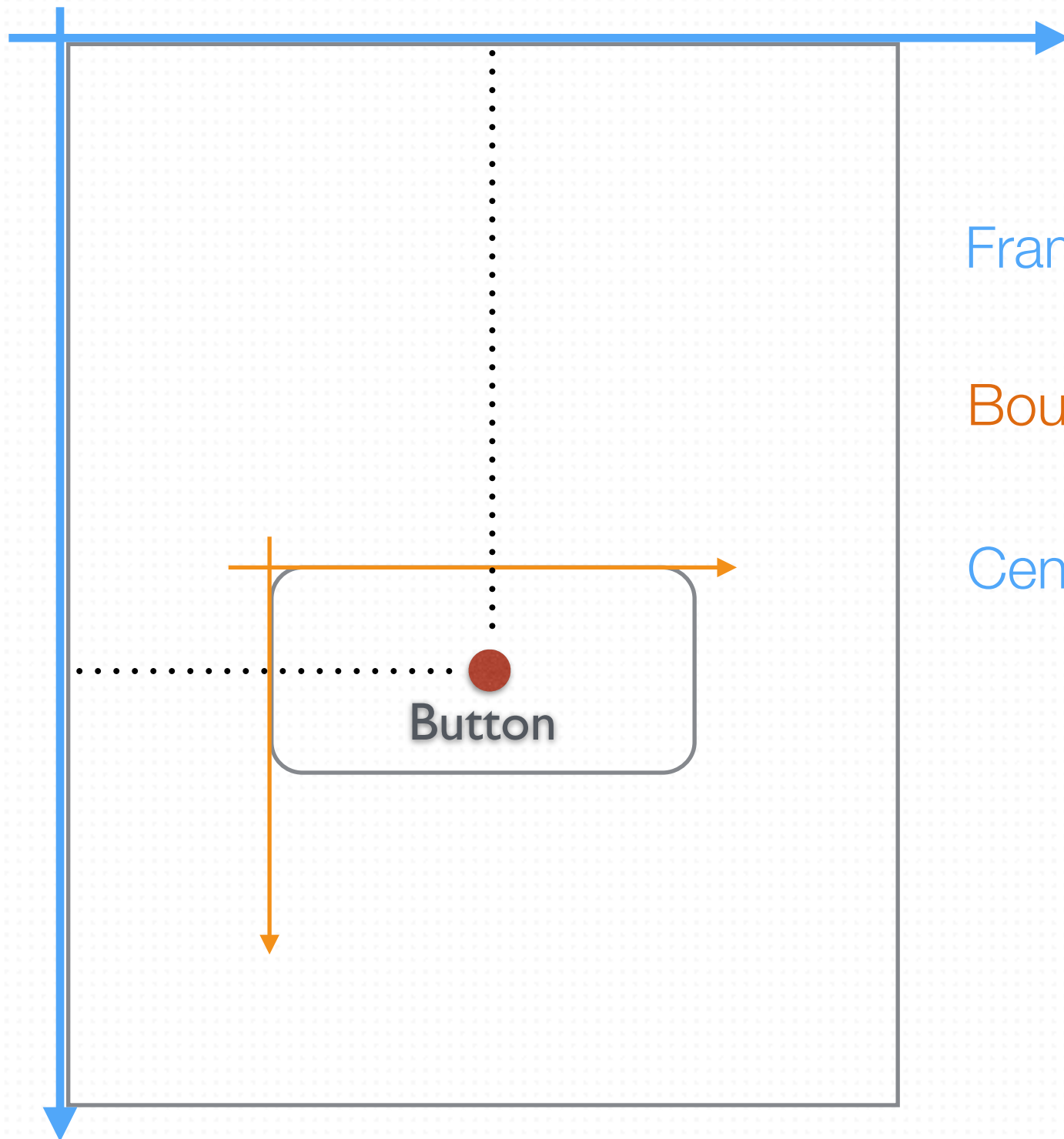
UIView

- UIButton
- UILabel
- UITextField
- UIImageView
- UITableView
- ...

UIView-Attributes

- Frame (size, position), Bounds, Center
- Background color, alpha, Hidden
- Transform

Frame, Bounds, Center



Frame: $(20, 100, \text{height}, \text{weight})$

Bound: $(0, 0, \text{height}, \text{weight})$

Center: $(20 + \text{weight}/2, 100 + \text{height}/2)$

UIView-Behavior

- Method:
 - addSubview:
 - animateWithDuration: animations:
- Event: -touchesBegan: withEvent:

UIButton

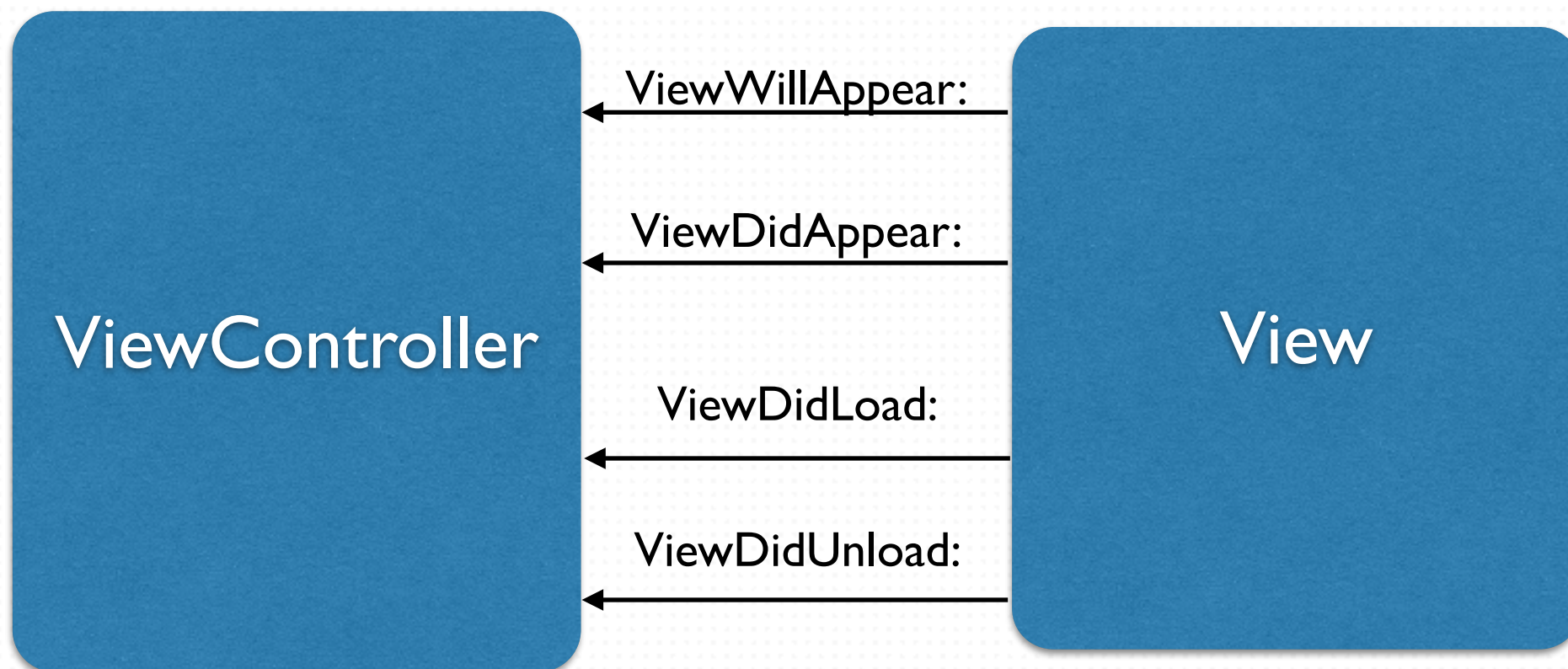
- UIView Attributes + Target (delegate)
- Method: -setTitle: forState:
- Event: -touchUpInside: , -touch

Controller

What would controller do?

UIView & UIViewController

- How a ViewController manage a view cycle?



UIView & UIViewController

- How a ViewController control a view?



UIView & UIViewController

- Example: Button

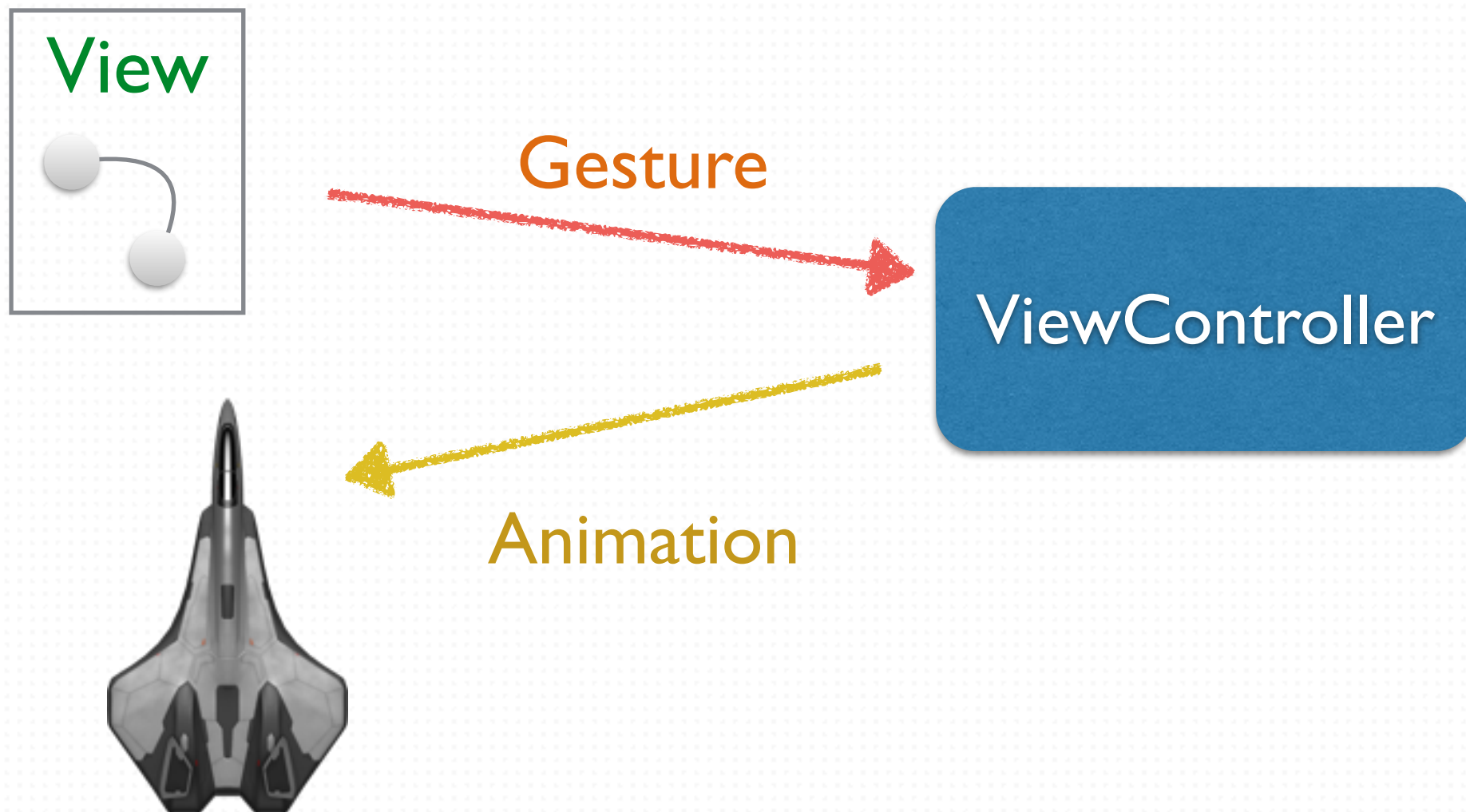


Gesture Recognition

Demo

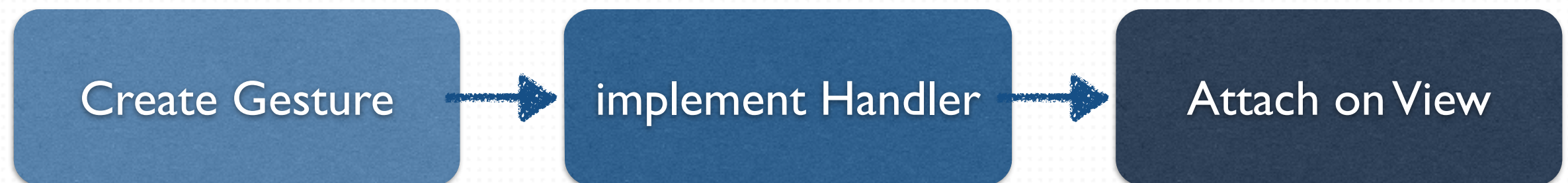
UIView & UIViewController

- Example: Gesture on



UIView & UIViewController

- How to?



Gesture Recognizer

● Tap

```
1 UITapGestureRecognizer *tapGesture = [[UITapGestureRecognizer alloc]
initWithTarget:self action:@selector(tapGestureHandler:)] ;
2 tapGesture.numberOfTapsRequired = 2;
3 [button addGestureRecognizer:tapGesture];

1 - (void) tapGestureHandler: (UIGestureRecognizer*) gestureRecognizer
2 {
3     NSLog(@"Tap Gesture Triggered. %d fingers tapped.",
gestureRecognizer.numberOfTouches);
4 }
```

@selector

@selector(sendMessage:to:)

- (void)sendMessage:(id)msg to:(id)somebody

@selector

@selector(helloWorld)

- (void)helloWorld

@selector(helloWorld:)

- (void)helloWorld:(id)param**

Gesture Recognizer

● Long Press

```
UILongPressGestureRecognizer *longPressGesture =  
[[UILongPressGestureRecognizer alloc] initWithTarget:self  
action:@selector(longPressHandler:)];  
  
longPressGesture.minimumPressDuration = 2.0; (2 Seconds)  
  
[button addGestureRecognizer:longPressGesture];
```

Gesture Recognizer

- Swipe

```
UISwipeGestureRecognizer *swipeGesture = [[UISwipeGestureRecognizer alloc]  
initWithTarget:self action:@selector(swipeGestureHandler)];
```

```
swipeGesture.direction = UISwipeGestureRecognizerDirectionLeft;
```

```
[self.view addGestureRecognizer:swipeGesture];
```

Gesture Recognizer

● Rotation

```
1 UIRotationGestureRecognizer *rotationGesture =  
[[UIRotationGestureRecognizer alloc] initWithTarget:self  
action:@selector(rotationGestureHandler)];  
2 [self.view addGestureRecognizer:rotationGesture];
```

```
1 - (void)rotationGestureHandler:  
(UIRotationGestureRecognizer*) gestureRecognizer  
2 {  
3     float degree = gestureRecognizer.rotation * 180 / M_PI;  
4     NSLog(@"Rotating: %fdeg", degree);  
5 }
```

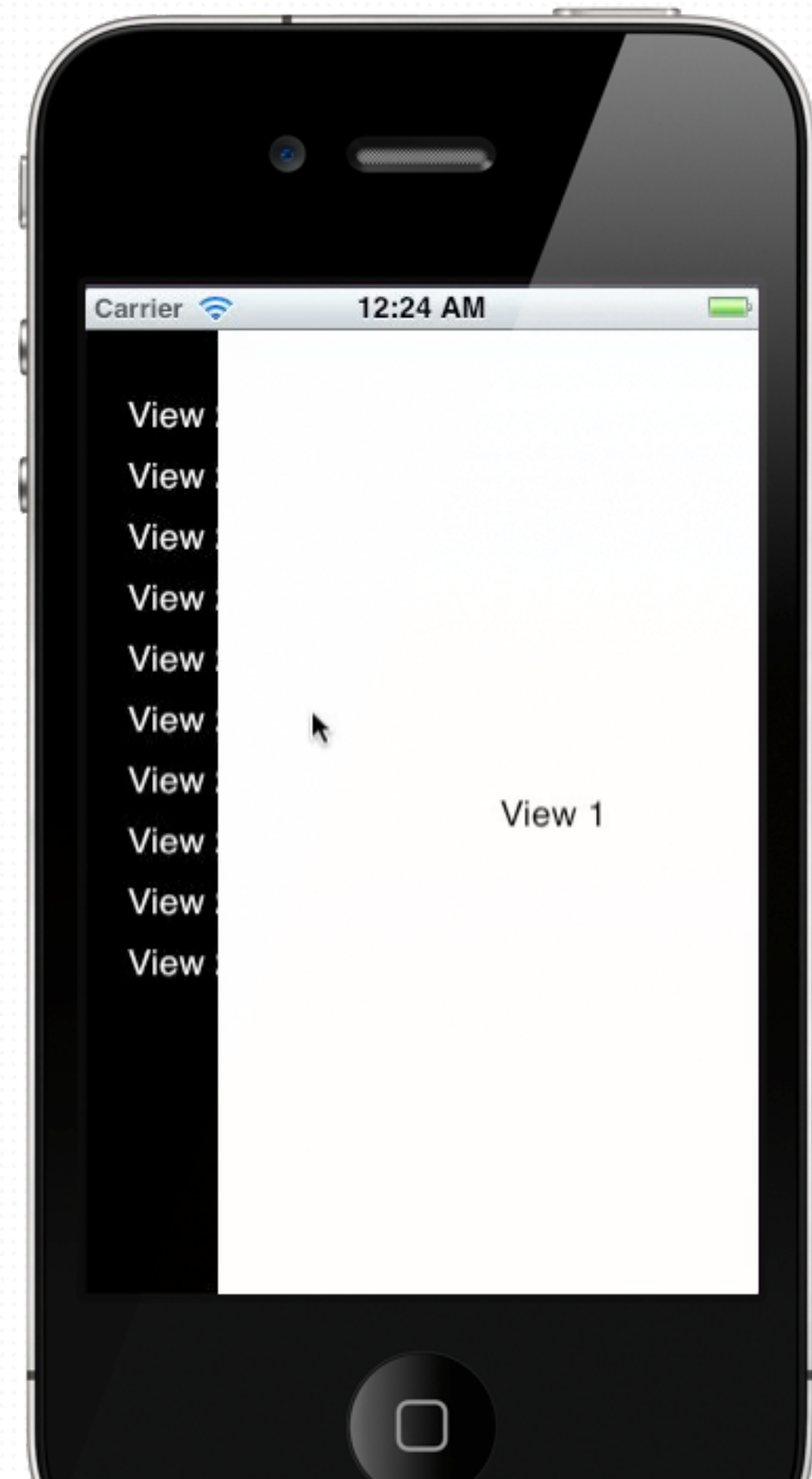
Gesture Recognizer

● Pan

```
1 UIPanGestureRecognizer *panGesture = [[UIPanGestureRecognizer alloc]
initWithTarget:self action:@selector(panGestureHandler:)];
2 [self.view addGestureRecognizer:panGesture];

1 - (void)panGestureHandler:(UIPanGestureRecognizer*)gestureRecognizer
2 {
3     NSString *translation = NSStringFromCGPoint([gestureRecognizer
translationInView:self.view]);
4     NSString *velocity = NSStringFromCGPoint([gestureRecognizer
velocityInView:self.view]);
5     NSLog(@"translation: %@, velocity: %@", translation, velocity);
6 }
```

View Panning



View Panning

● ViewDidLoad

```
1 - (void) viewDidLoad
2 {
3     [super viewDidLoad];
4
5     UIPanGestureRecognizer *panGesture = [[UIPanGestureRecognizer alloc]
initWithTarget:self action:@selector(panGestureHandler:)];
6     [self.view addGestureRecognizer:panGesture];
7 }
```

View Panning

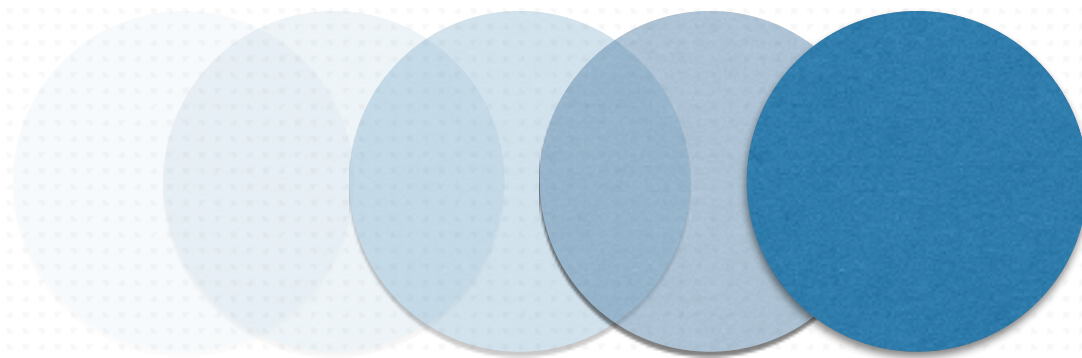
- Handling panning gesture

```
1 - (void) panGestureHandler: (UIPanGestureRecognizer*) gesture
2 {
3     CGPoint translation = [gesture translationInView:self.view];
4     NSLog(@"%@", NSStringFromCGPoint(translation));
5
6     CGRect frame = self.fgView.frame;
7
8     // gesture ended.
9     if (gesture.state == UIGestureRecognizerStateEnded)
10    {
11        frame.origin.x = 0;
12    } else {
13        frame.origin.x = frame.origin.x + translation.x;
14    }
15
16    // transform the frame.
17    self.fgView.frame = frame;
18
19    [gesture setTranslation:CGPointZero inView:self.view];
20 }
```


Animation

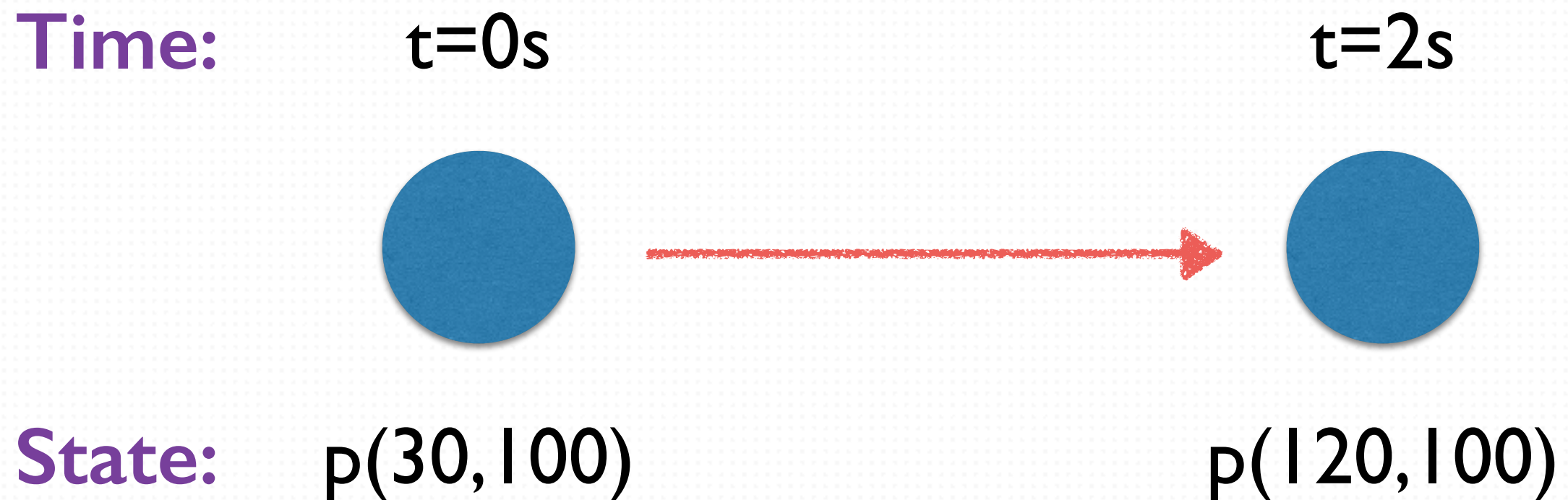
Animation

- Give UIView a life - Move!



Animation

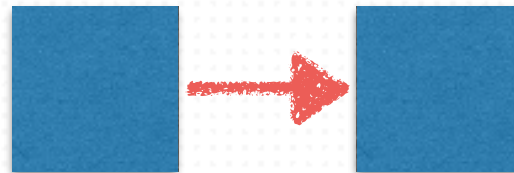
- UIView transit from one state to other state



Animation

- State

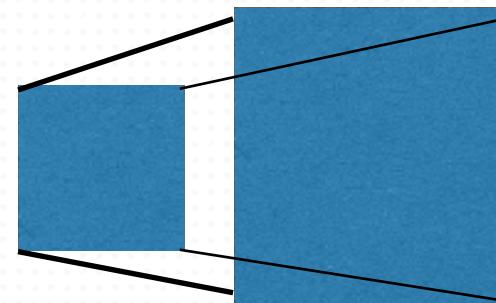
- Position



- Color



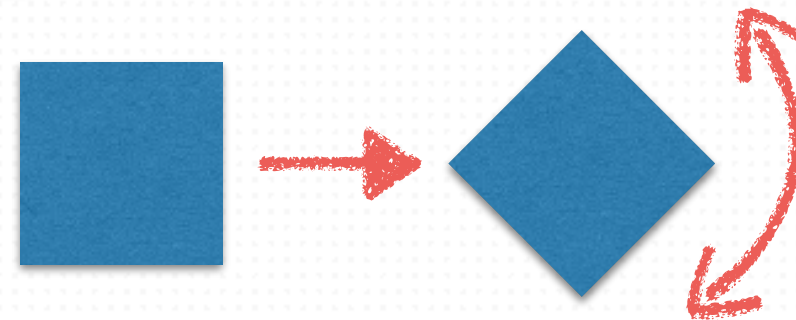
- Scale



- Opacity



- Rotation



Animation

- Implement an animation on UIView

```
view.frame = CGRectMake(0,200, 40, 40);
```

```
[UIView animateWithDuration:0.4 animations:^(  
    //UIView that wants to have animation  
    view.frame = CGRectMake(100, 200, 40,40 );  
)];
```

Animation

- Multistage animation (Nested animation)

```
[UIView animateWithDuration:0.5
    delay:0.0
    options:UIViewAnimationOptionBeginFromCurrentState
    animations:^(
        ##### first animation ###
    )
    completion:^(BOOL finished){[UIView animateWithDuration:0.5
        delay:0.0
        options:UIViewAnimationOptionBeginFromCurrentState
        animations:^(
            ##### second animation ###
        )
        completion:^(BOOL finished){### and so on.. ##
        }];}];
```

Animation

- Transform
 - Translation
 - Rotation
 - Scale

Animation

- Make Translation Matrix

CGAffineTransform CGAffineTransformMakeTranslation (CGFloat tx, CGFloat ty);

- Make Rotation Matrix

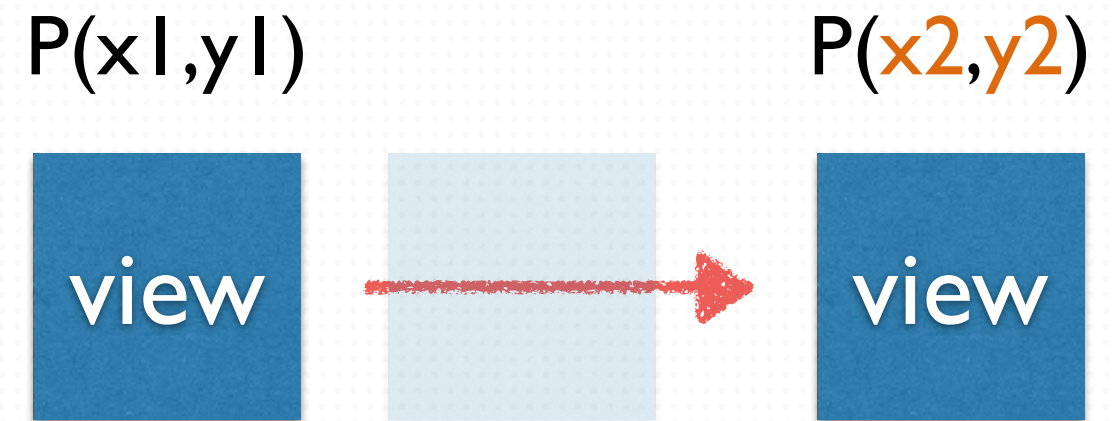
CGAffineTransform CGAffineTransformMakeRotation (CGFloat angle);

- Make Scale Matrix

CGAffineTransform CGAffineTransformMakeScale (CGFloat sx, CGFloat sy);

Animation

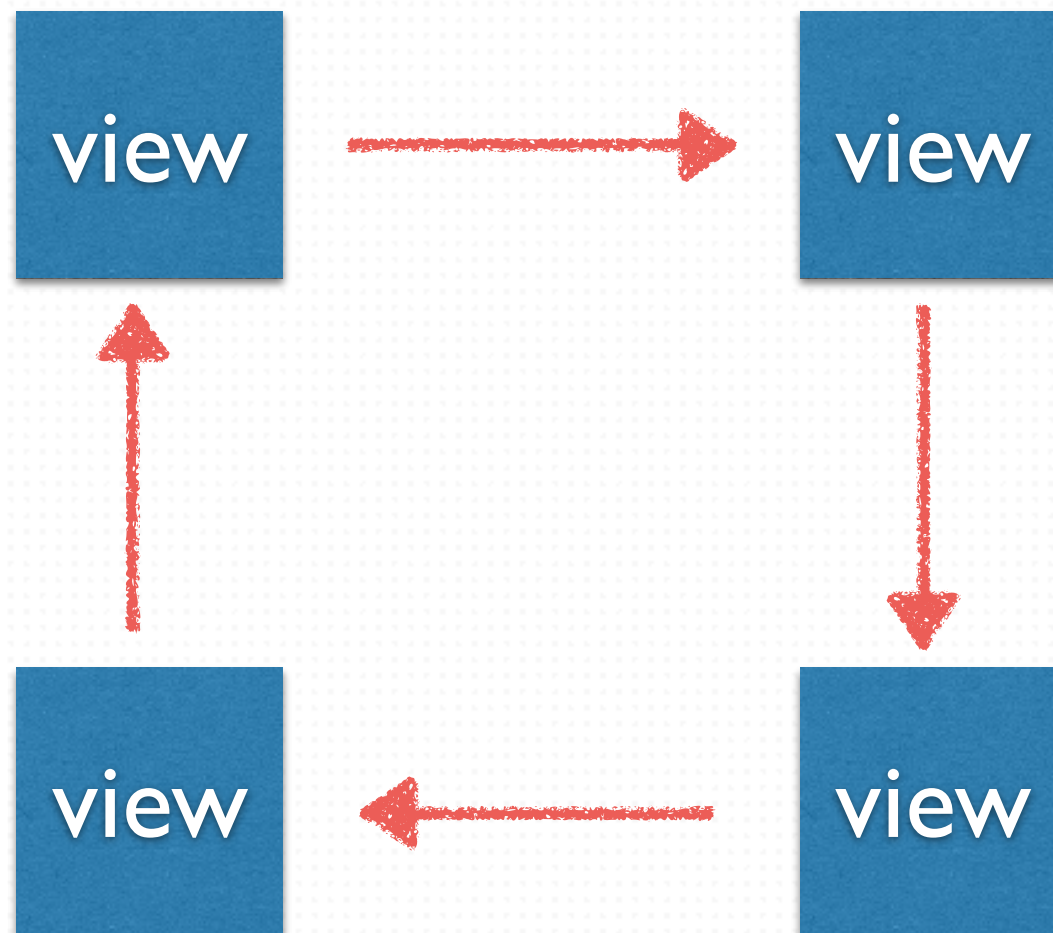
- Translation a UIView



```
view.transform = CGAffineTransformMakeTranslation(x2, y2);
```

Animation

- How about?



Practice



Exercise

- ✓ Think about how to make use of gestures.
- ✓ Design an app with **gesture features** and **animation**.
- ✓ Present it to the class in next lesson.