# Basic Exercises Part 3.1 UIImageView & UIButton

## UIImageView

* **A UIImageView** class is used to encapsulate an image in a view.

### **1.1 Create a UIImageView**

To create a UIImageView programmatically, all you need to do is create an instance of UIImageView:

//Swift

let imageView = UIImageView()

//Objective-C

UIImageView \*imageView = [[UIImageView alloc] init];

You can set the size and position of the UIImageView with a CGRect:

//Swift

imageView.frame = CGRect(x: 0, y: 0, width: 200, height: 200)

//Objective-C

imageView.frame = CGRectMake(0,0,200,200);

Or you can set the size during initialization:

//Swift

UIImageView(frame: CGRect(x: 0, y: 0, width: 200, height: 200))

//Objective-C

UIImageView \*imageView = [[UIImageView alloc] initWithFrame:CGRectMake(0,0,200,200);

//Alternative way of defining frame for UIImageView

UIImageView \*imageView = [[UIImageView alloc] init];

CGRect imageViewFrame = imageView.frame;

imageViewFrame.size.width = 200;

imageViewFrame.size.height = 200;

imageViewFrame.origin.x = 0;

imageViewFrame.origin.y = 0;

imageView.frame = imageViewFrame;

### **1.2 Assigning an image to a UIImageView**

You can assign an image to a UIImageView during initialization, or later using the image property:

//Swift

UIImageView(image: UIImage(named: "image1"))

UIImageView(image: UIImage(named: "image1"), highlightedImage: UIImage(named: "image2"))

imageView.image = UIImage(named: "image1")

//Objective-C

[[UIImageView alloc] initWithImage:[UIImage imageNamed:@"image1"];

[[UIImageView alloc] initWithImage:[UIImage imageNamed:@"image1"] highlightedImage:[UIImage imageNamed:@"image2"]];

imageView.image = [UIImage imageNamed:@"image1"];

### **1.3 Animating a UIImageView**

You can animate a UIImageView by quickly displaying images on it in a sequence using the UIImageView's animation properties:

imageView.animationImages = [UIImage(named: "image1")!,

UIImage(named: "image2")!,

UIImage(named: "image3")!,

UIImage(named: "image4")!,

UIImage(named: "image5")!,

UIImage(named: "image6")!,

UIImage(named: "image7")!,

UIImage(named: "image8")!]

imageView.animationDuration = 0.3

imageView.animationRepeatCount = 1

The animationImages property is an Array of UIImages that is run through from top to bottom when the animation is triggered.

The animationDuration property is a Double saying how many seconds the animation will run for.

The animationRepeatCount property is an Int that says how many times the animation will run.

To start and stop the animation, you can call the appropriate methods to do so:

imageView.startAnimating()

imageView.stopAnimating()

There is method isAnimating() which returns a Boolean value indicating whether the animation is running at a moment or not.

Please note that this's not a very efficient way to create animations: it's quite slow and resource-consuming. Consider using Layers or Sprites for better results

### **1.4 Adding a UIImageView to center of screen**

If you want to add a UIImageView to the center of the screen with width and height of 100 pixel, you need to set center x constraint and center y constraint to the superview from the UIImageView and width, height constraint to the UIIMageView. Here is the code.

UIImageView \*imageView = [[UIImageView alloc] initWithFrame:CGRectMake(0, 0, 100.0, 100.0);

imageView.self.translatesAutoresizingMaskIntoConstraints = NO;

[imageView addConstraint:[NSLayoutConstraint constraintWithItem:imageView attribute:NSLayoutAttributeWidth relatedBy:NSLayoutRelationEqual toItem:nil attribute:NSLayoutAttributeNotAnAttribute multiplier:1.0 constant:100.0]];

[imageView addConstraint:[NSLayoutConstraint constraintWithItem:imageView attribute:NSLayoutAttributeHeight relatedBy:NSLayoutRelationEqual toItem:nil attribute:NSLayoutAttributeNotAnAttribute multiplier:1.0 constant:100.0]];

[superview addConstraint:[NSLayoutConstraint constraintWithItem:imageView attribute:NSLayoutAttributeCenterX relatedBy:NSLayoutRelationEqual toItem:superview attribute:NSLayoutAttributeCenterX multiplier:1.0 constant:0]];

[superview addConstraint:[NSLayoutConstraint constraintWithItem:imageView attribute:NSLayoutAttributeCenterX relatedBy:NSLayoutRelationEqual toItem:superview attribute:NSLayoutAttributeCenterX multiplier:1.0 constant:0]];

### **1.5 Now try on Swift**

Repeat 1.4 but this time over Swift

### **1.6 Do this in storyboard**

Create a UIImageView in storyboard. Set an image placeholder, avoid crashes. Connect the image with the code.

// import UIKit class ViewController: UIViewController {

override func viewDidLoad() {

super.viewDidLoad()

}

@IBOutlet weak var simpleImageView: **UIImageView!**

override func didReceiveMemoryWarning() {

super.didReceiveMemoryWarning()

}

}

### **1.7. Find a nice web page where to download free images for your app**

Go visit:

<https://unsplash.com/s/photos/ios>

Other resources :

Flickr api. Flickr is a wonderful image sharing service that has a publicly accessible and dead- simple API for developers to use. With the API you can search for photos, add photos, comment on photos, and much more. To use the Flickr API, you need an API key. If you are doing a real project, I recommend you sign up for one here: [signup for Flickr](http://www.flickr.com/services/api/keys/apply/)

### **1.8. Mode property**

The content mode property of a view tells how its content should be laid out. In the interface builder, the various modes can be selected in the attributes inspector.

Try all the modes, both in code as well the IB.

mageView.contentMode = UIViewContentMode.scaleToFill  
imageView.contentMode = UIViewContentMode.scaleAspectFit  
imageView.contentMode = UIViewContentMode.scaleAspectFill  
imageView.contentMode = UIViewContentMode.redraw  
imageView.contentMode = UIViewContentMode.center  
imageView.contentMode = UIViewContentMode.top  
imageView.contentMode = UIViewContentMode.bottom  
imageView.contentMode = UIViewContentMode.left  
imageView.contentMode = UIViewContentMode.right  
imageView.contentMode = UIViewContentMode.topLeft  
imageView.contentMode = UIViewContentMode.topRight  
imageView.contentMode = UIViewContentMode.bottomLeft  
imageView.contentMode = UIViewContentMode.bottomRight

### **1.9. Upcoming. Under construction!!**

Cache images.

Download from API call.

## UIButton

* **Buttons** are used for handling user interactions. It intercepts the touch events and sends message to the target objetc.
* **Button Types.**
* UIButtonTypeRoundedRect
* UIButtonTypeCustom
* UIButtonTypeDetailDisclosure
* UIButtonTypeInfoLight
* UIButtonTypeInfoDark
* UIButtonTypeContactAdd
* **Important properties.**
* imageView
* titleLabel

### **1.1 Initialized**

UIButtons can be initialized in a frame:

**Swift**

let button = UIButton(frame: CGRect(x: x, y: y, width: width, height: height)

**Objective C**

UIButton \*button = [[UIButton alloc] initWithFrame:CGRectMake(x, y, width, height)];

A specific type of UIButton can be created like this:

**Swift**

let button = UIButton(type: .Custom)

**Objective C**

UIButton \*button = [UIButton buttonWithType:UIButtonTypeCustom];

where type is a UIButtonType:

enum UIButtonType : Int {

case Custom

case System

case DetailDisclosure

case InfoLight

case InfoDark

case ContactAdd

static var RoundedRect: UIButtonType { get }

}

### **1.2 Alerts (they have buttons)**

On the storyboard create a button and create an action relation with the next code:

@IBAction func showMessage(sender: UIButton) {

let alertController = UIAlertController(title: "Welcome to My First App", message: "Hello World", preferredStyle: UIAlertController.Style.alert)

alertController.addAction(UIAlertAction(title: "OK", style: UIAlertAction.Style.default, handler: nil))

present(alertController, animated: true, completion: nil)

}

Test the app.

### **1.3. Put all together.**

Create a table view (4 o 5 rows).

Make some space between the rows.

Change the background (randomly) for each row.

Pin a button in the center of each row.

Set an image for the button.

Tap the button and present an alert (can contain the row number you taped or the name of the button image)

BONUS

Do all programmatically

Do it use Swift and Objective C.

At the end you would get something like image below:

A picture containing drawing

Description automatically generated