实验编号： 4 **四川师大《IOS》实验报告 2018** 年 **9** 月 **26** 日

### **计算机科学学院** 2016 级 4 班 实验名称： 自定制视图、文件、Playground可视化

姓名：\_\_蒋宇童\_\_\_\_\_\_\_ 学号：\_2016110419\_\_\_\_\_\_\_\_\_ 指导老师：\_\_李贵洋\_\_ 实验成绩:\_\_\_\_\_

**实验\_四\_ \_**自定制视图、文件、Playground可视化**\_**

1. 实验目的及要求
2. 掌握沙盒文件的操作；
3. 掌握自定制视图的定义及使用；
4. 掌握playground下可视化开发方法；
5. 实验内容
6. **文件缓存处理**
   1. 判断沙盒的Document目录下是否存在某文件夹，如果没有则新建一个该文件夹；
   2. 判断是否该文件夹下存在一个图片文件，如果存在该文件，读取该文件到一个图片对象中并进行显示，如果不存在则从网上下载一张图片并保存到该图片文件中；
7. **自定制视图**
   1. 从UIView中派生一个自定制的View；
   2. 绘制一个椭圆（或则自己喜欢的任何图形）；
   3. 新建视图对象并进行显示；
8. **代码版hello world（在视图控制器中加入代码）**
   1. 代码中生成label（outlet）和button；
   2. 将label和button加入根view中；
   3. button添加像self（target）发射action（selector）的操作；

实现clicked响应代码（selector）

1. 实验主要流程、基本操作或核心代码、算法片段（该部分如不够填写，请另加附页）
2. **文件缓存处理**
   1. 判断沙盒的Document目录下是否存在某文件夹，如果没有则新建一个该文件夹；
   2. 判断是否该文件夹下存在一个图片文件，如果存在该文件，读取该文件到一个图片对象中并进行显示，如果不存在则从网上下载一张图片并保存到该图片文件中；

* 程序代码：

/\*

1. 文件缓存处理：

a) 判断沙盒的Document目录下是否存在某文件夹，如果没有则新建一个该文件夹；

b) 判断是否该文件夹下存在一个图片文件，如果存在该文件，读取该文件到一个图片对象中并进行显示，如果不存在则从网上下载一张图片并保存到该图片文件中。

\*/

let fileManager=FileManager.default

if var docPath=fileManager.urls(for: .documentDirectory, in: .userDomainMask).first

{

docPath.appendPathComponent("3.jpg")

if !fileManager.fileExists(atPath: docPath.path)

{

let imageUrl=URL(string: "http://img.zcool.cn/community/01f09e577b85450000012e7e182cf0.jpg@1280w\_1l\_2o\_100sh.jpg")

let imageData=try Data(contentsOf: imageUrl!)

var image=UIImage(data: imageData)

docPath.appendPathComponent("3.jpg")

try imageData.write(to:docPath)

}

else

{

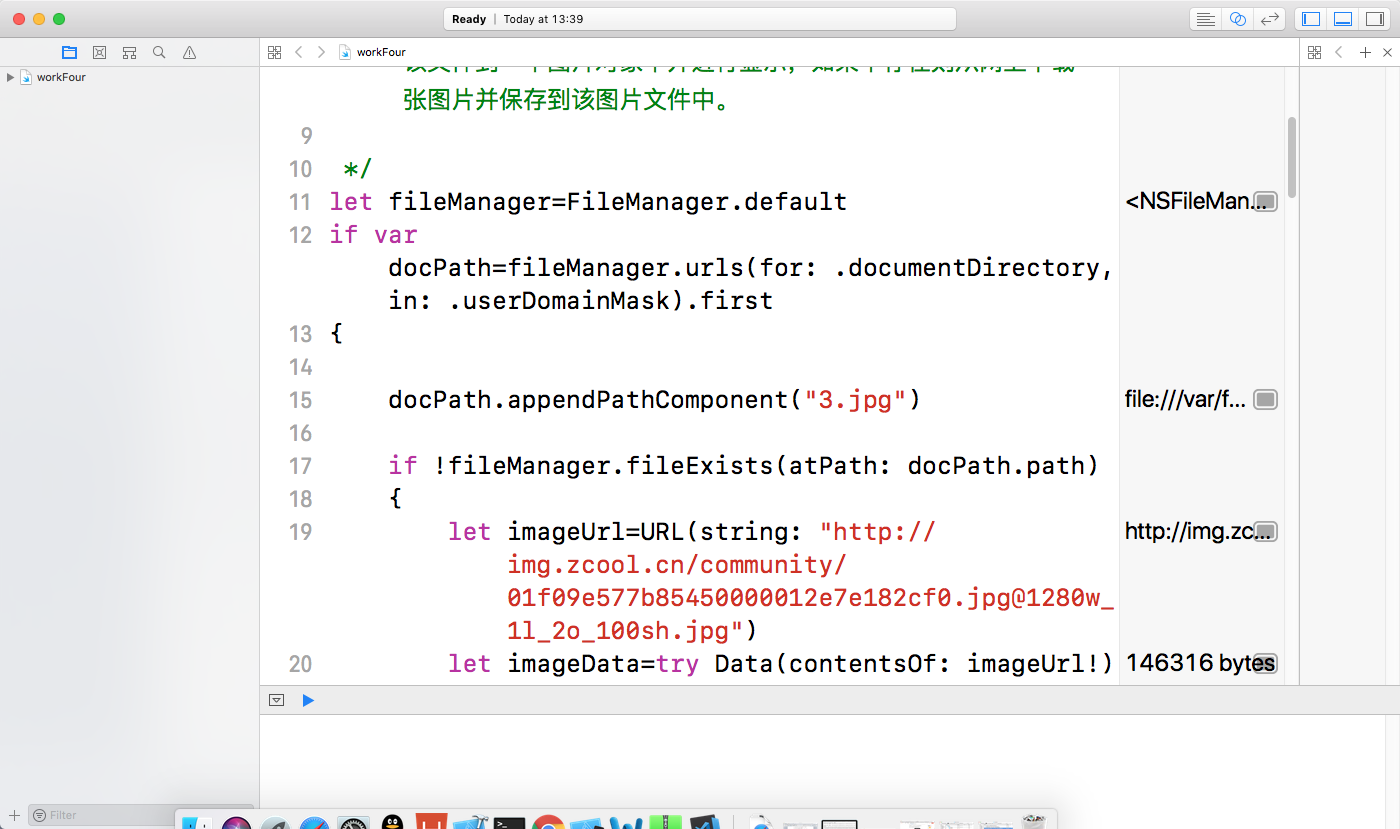
let imageData=try Data(contentsOf: docPath)

var image=UIImage(data: imageData)

}

}

* 运行结果：



1. **自定制视图**
   1. 从UIView中派生一个自定制的View；
   2. 绘制一个椭圆（或则自己喜欢的任何图形）；
   3. 新建视图对象并进行显示；

* 程序代码：

/\*

2. 自定制视图：

a) 从UIView中派生一个自定制的View；

b) 绘制一个椭圆（或则自己喜欢的任何图形）；

c) 新建视图对象并进行显示；

\*/

//做的是一个可以顺时针旋转的圆圈和一个椭圆

/\*

2. 自定制视图：

a) 从UIView中派生一个自定制的View；

b) 绘制一个椭圆（或则自己喜欢的任何图形）；

c) 新建视图对象并进行显示；

\*/

class Ellipse:UIView

{

override func draw(\_ rect: CGRect) {

let aPath = UIBezierPath(ovalIn: CGRect(x: 10, y: 10, width: 200, height: 300))

UIColor.blue.setFill()

}

}

class MyView:UIView

{

var beginAngle = Double.pi\*3/2 // 起点

var finishAngle = Double.pi\*3/2+Double.pi\*2/20 // 终点

override func draw(\_ rect: CGRect) {

let aPath = UIBezierPath(arcCenter:CGPoint(x: 150, y: 150), radius: 75, startAngle: (CGFloat)(beginAngle), endAngle: (CGFloat)(finishAngle), clockwise: true)

aPath.addLine(to:CGPoint(x: 150, y: 150))

aPath.close()

aPath.lineWidth = 5.0

UIColor.red.setFill()

aPath.fill()

finishAngle += Double.pi/20

}

}

class Controller:UIViewController

{

var lable:UILabel!

var timer: Timer?// 计时器

var myView=MyView(frame: CGRect(x: 100, y: 600, width: 300, height: 300))

var myEllipse=Ellipse()

@objc func reDrawView() {

myView.setNeedsDisplay() // 重绘界面

// 画完一圈后停止

if myView.finishAngle > myView.beginAngle+Double.pi\*2 {

timer?.invalidate() // 停止计时器

}

}

@IBAction func click()

{

print("jakds")

lable.text="i am click"

}

override func loadView() {

let view=UIView(frame:CGRect(x: 0, y: 0, width: 900, height: 900))

view.backgroundColor=UIColor.red

timer = Timer.scheduledTimer(timeInterval: 0.1, target: self,

selector:#selector(reDrawView), userInfo: nil, repeats: true)

view.addSubview(myView)

view.addSubview(myEllipse)

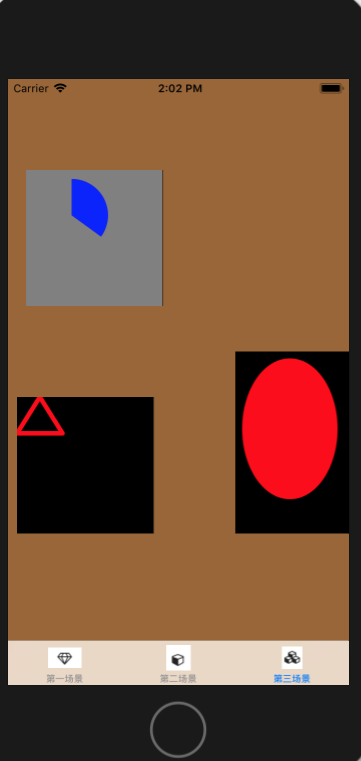
}

}

let tc=Controller()

PlaygroundPage.current.liveView=tc

* 运行结果：



1. **代码版hello world（在视图控制器中加入代码）**
   1. 代码中生成label（outlet）和button；
   2. 将label和button加入根view中；
   3. button添加像self（target）发射action（selector）的操作；

实现clicked响应代码（selector）

* 程序代码：

import UIKit

class FirstViewController: UIViewController {

var lable=UILabel(frame: CGRect(x: 130, y: 200, width: 100, height: 40))

override func viewDidLoad() {

super.viewDidLoad()

self.navigationItem.rightBarButtonItem=UIBarButtonItem(title: "go to >", style: UIBarButtonItemStyle.plain, target: self, action: #selector(click2))

self.title="第一场景"

view.backgroundColor=UIColor.purple

lable.backgroundColor=UIColor.brown

lable.textColor=UIColor.white

lable.textAlignment = .center

lable.text="蒋宇童"

lable.alpha=0.5

view.addSubview(lable)

let button=UIButton(frame: CGRect(x: 80, y: 300, width: 200, height: 50))

button.setTitle("变内容", for: .normal)

button.backgroundColor=UIColor.darkGray

button.addTarget(self, action: #selector(click), for: UIControlEvents.touchUpInside)

button.alpha=0.7

view.addSubview(button)

let button2=UIButton(frame: CGRect(x: 80, y: 400, width: 200, height: 50))

button2.setTitle("To第二场景", for: .normal)

button2.backgroundColor=UIColor.darkGray

button2.addTarget(self, action: #selector(click2), for: UIControlEvents.touchUpInside)

button.alpha=0.7

view.addSubview(button2)

// Do any additional setup after loading the view.

}

@IBAction func nextPage()

{

let sv=SecondeViewController()

self.navigationController?.pushViewController((sv), animated: true)

}

@IBAction func click()

{

lable.text="I am click"

}

@IBAction func click2()

{

let sv=SecondeViewController()

self.navigationController?.pushViewController((sv), animated: true)

//self.presentingViewController?.dismiss(animated: true, completion: nil)

}

override func didReceiveMemoryWarning() {

super.didReceiveMemoryWarning()

// Dispose of any resources that can be recreated.

}

/\*

// MARK: - Navigation

// In a storyboard-based application, you will often want to do a little preparation before navigation

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

// Get the new view controller using segue.destinationViewController.

// Pass the selected object to the new view controller.

}

\*/

}

* 运行结果：



1. 实验结果的分析与评价（该部分如不够填写，请另加附页）

**Github地址：https://github.com/jiangyutong/swiftWork/tree/master/代码**

通过注册的实验我知道了，如何判断一个文件是否存在沙盒里。并且知道怎么用代码去画自己想要的图形。在这次的实验中我是用定时器来画来一个圆，圆呢是用贝塞尔曲线画的。如何又画来三角形和椭圆。这次的实验我都用用的纯代码写的，自己写了一个bar和navigation来实现来一个多界面的应用。

注：实验成绩等级分为（90－100分）优，（80－89分）良，(70-79分)中，（60－69分）及格，（59分）不及格。