实验编号： 9 **四川师大《IOS》实验报告 2018** 年 **11** 月 **7** 日

### **计算机科学学院** 2016 级 4 班 实验名称： Gesture、UIAlertController、 ScrollView

姓名： 郭周倩 学号： 2016110413 指导老师：\_\_李贵洋\_\_ 实验成绩:\_\_\_\_\_

**实验 九 \_\_\_\_\_** Gesture、UIAlertController、 ScrollView **\_\_\_\_\_\_\_**

1. 实验目的及要求
2. 理解并掌握iOS多点手势识别的相关技术；
3. 实验要求
4. 认真填写实验报告，要求附加部分运行界面和主要代码；
5. 对设计好的程序，检查输出是否符合预期，如有错请分析错误原因并解决；
6. 实验内容
7. Gesture
   1. 分别采用代码随机位置大小生成和直接拖拽的方式产生多个视图；
   2. 采用简单的动画进行移动；
   3. 给视图加上阴影(layer)；
   4. 可全部清空子视图；
   5. 视图支持手势（pan移动、tap删除、pinch缩放、rotation旋转）；

提示：Pinch的scale属性可用于调整frame

rotation需要用transform属性实现

1. 实现UIAlertController交互
   1. 显示ActionSheet并进行交互；
   2. 显示Login Alert并进行交互；
2. 一个界面使用两个scrollView
   1. 在一个scrollView中可进行多张图片横屏滚动浏览(相册)，需要有pagecontrol进行提示；
   2. 在另一个scrollView中可放大缩小；

提示：需用delegate

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. Gesture
   1. 分别采用代码随机位置大小生成和直接拖拽的方式产生多个视图；
   2. 采用简单的动画进行移动；
   3. 给视图加上阴影(layer)；
   4. 可全部清空子视图；
   5. 视图支持手势（pan移动、tap删除、pinch缩放、rotation旋转）；

提示：Pinch的scale属性可用于调整frame

rotation需要用transform属性实现

* 程序代码：

import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {

        super.viewDidLoad()

        // Do any additional setup after loading the view, typically from a nib.

    }

    override func didReceiveMemoryWarning() {

        super.didReceiveMemoryWarning()

        // Dispose of any resources that can be recreated.

    }

    @IBAction func add(\_ sender: UIButton) {

        let x = Int(arc4random()) % Int(self.view.bounds.width)

        let y = Int(arc4random()) % Int(self.view.bounds.height)

        let label = UILabel(frame: CGRect(x: x, y: y, width: 50, height: 50))

        label.text = "A"

        label.textAlignment = .center

        label.backgroundColor =  colorLiteral(red: 1, green: 0.694814277, blue: 0.7656143615, alpha: 1)

        label.layer.shadowColor = UIColor.gray.cgColor

        label.layer.shadowOffset = CGSize(width: 10, height: 10)

        label.layer.shadowOpacity = 1

        let panRecognizer = UIPanGestureRecognizer(target: self, action: #selector(pan(recognizer:)))

        label.addGestureRecognizer(panRecognizer)

        let pinchRecognizer = UIPinchGestureRecognizer(target: self, action: #selector(pinch(recognizer:)))

        label.addGestureRecognizer(pinchRecognizer)

        label.isUserInteractionEnabled = true

        let tapRecognizer = UITapGestureRecognizer(target: self, action: #selector(tap(recognizer:)))

        label.addGestureRecognizer(tapRecognizer)

        tapRecognizer.numberOfTapsRequired = 2

        //旋转

        let rotationRecognizer = UIRotationGestureRecognizer(target: self, action: #selector(rotation(recognizer:)))

        label.addGestureRecognizer(rotationRecognizer)

        self.view.addSubview(label)

    }

    @objc func rotation(recognizer: UIRotationGestureRecognizer) {

        //设置旋转弧度

        recognizer.view?.transform = (recognizer.view?.transform.rotated(by: recognizer.rotation))!

        //将上一次弧度设置为1

        recognizer.rotation = 0

    }

    @objc func pan(recognizer: UIPanGestureRecognizer) {

        switch recognizer.state {

        case .changed:

            fallthrough

        case .ended:

            let translation = recognizer.translation(in: self.view)

            recognizer.view?.center.x += translation.x

            recognizer.view?.center.y += translation.y

            recognizer.setTranslation(.zero, in: self.view)

        default:

            break

        }

    }

    @objc func pinch(recognizer: UIPinchGestureRecognizer) {

        switch recognizer.state {

        case .changed:

            fallthrough

        case .ended:

            recognizer.view!.bounds.size = CGSize(width: recognizer.view!.bounds.width \* recognizer.scale, height: recognizer.view!.bounds.height \* recognizer.scale)

            recognizer.scale = 1

        default:

            break

        }

    }

    @objc func tap(recognizer: UITapGestureRecognizer) {

        if recognizer.state == .recognized {

            recognizer.view?.removeFromSuperview()

        }

    }

    @IBAction func move(\_ sender: UIButton) {

        for label in self.view.subviews {

            if label is UILabel {

                UIView.animate(withDuration: 1) {

                    let x = Int(arc4random()) % Int(self.view.bounds.width)

                    let y = Int(arc4random()) % Int(self.view.bounds.height)

                    label.center = CGPoint(x: x, y: y)

                }

            }

        }

    }

    @IBAction func deleteView(\_ sender: UIButton) {

        for label in self.view.subviews {

            if label is UILabel {

                label.removeFromSuperview()

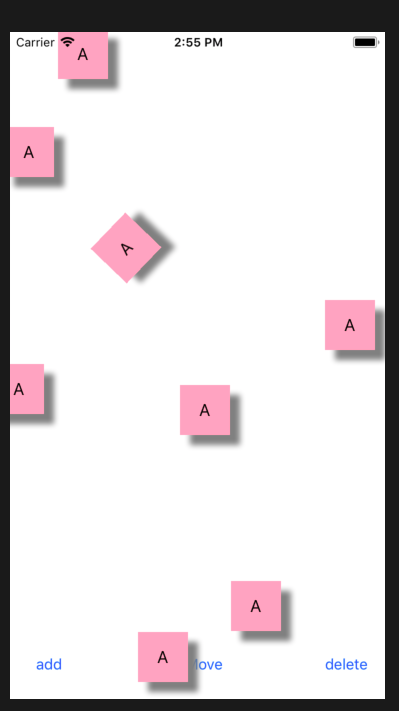
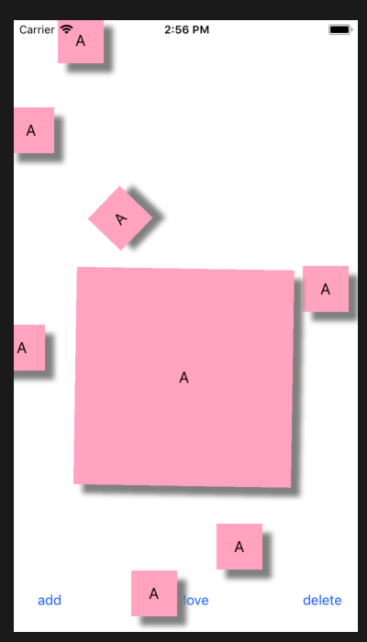
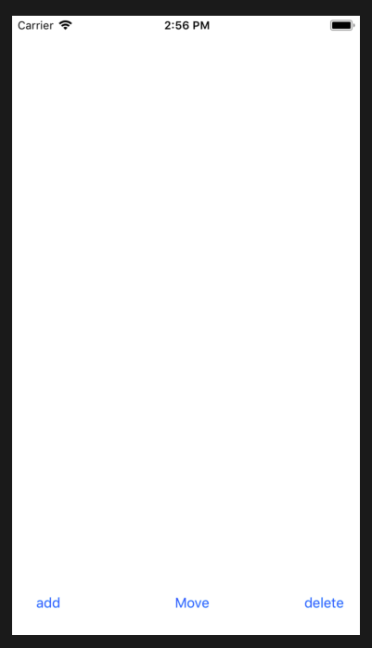
            }

        }

    }

}

* 运行结果：

1. 实现UIAlertController交互
   1. 显示ActionSheet并进行交互；
   2. 显示Login Alert并进行交互；

* 程序代码：

import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {

        super.viewDidLoad()

        // Do any additional setup after loading the view, typically from a nib.

    }

    override func didReceiveMemoryWarning() {

        super.didReceiveMemoryWarning()

        // Dispose of any resources that can be recreated.

    }

    @IBAction func actionSheet(\_ sender: UIButton) {

        let alert = UIAlertController(title: "ActionSheet", message: "choose Color?", preferredStyle: .actionSheet)

        alert.addAction(UIAlertAction(title: "red", style: .default, handler: { (action) in

            self.view.backgroundColor = UIColor.red

        }))

        alert.addAction(UIAlertAction(title: "yellow", style: .default, handler: { (action) in

            self.view.backgroundColor = UIColor.yellow

        }))

        alert.addAction(UIAlertAction(title: "blue", style: .destructive, handler: { (action) in

            self.view.backgroundColor = UIColor.blue

        }))

        alert.addAction(UIAlertAction(title: "White", style: .cancel, handler: { (action) in

            self.view.backgroundColor = UIColor.white

        }))

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

    }

    @IBAction func alert(\_ sender: UIButton) {

        let alert = UIAlertController(title: "alert", message: "Login message", preferredStyle: .alert)

        alert.addAction(UIAlertAction(title: "Login", style: .default, handler: { (action) in

            guard let username = alert.textFields?.first?.text, let password = alert.textFields?.last?.text else{

                return

            }

            print("username=\(username) password=\(password)")

        }))

        alert.addAction(UIAlertAction(title: "Cancel", style: .cancel, handler: { (action) in

            print("取消登陆！！！")

        }))

        alert.addTextField { (textField) in

            textField.placeholder = "your username"

        }

        alert.addTextField { (textField) in

            textField.placeholder = "your password"

            textField.isSecureTextEntry = true

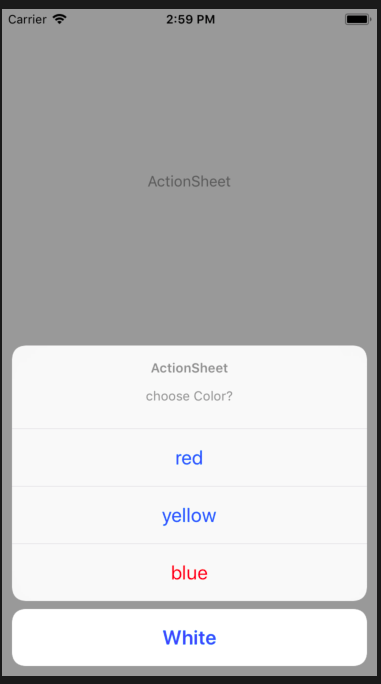
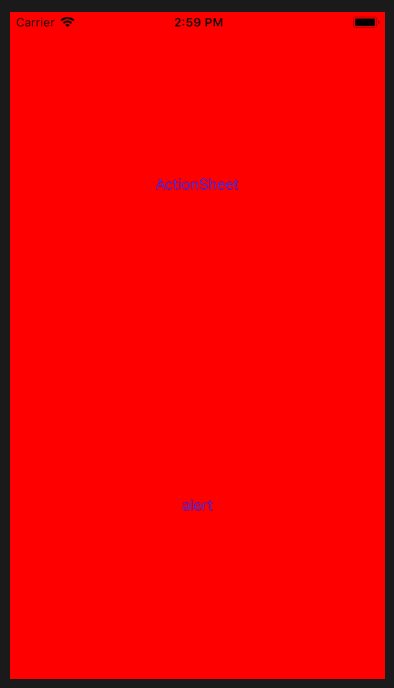
        }

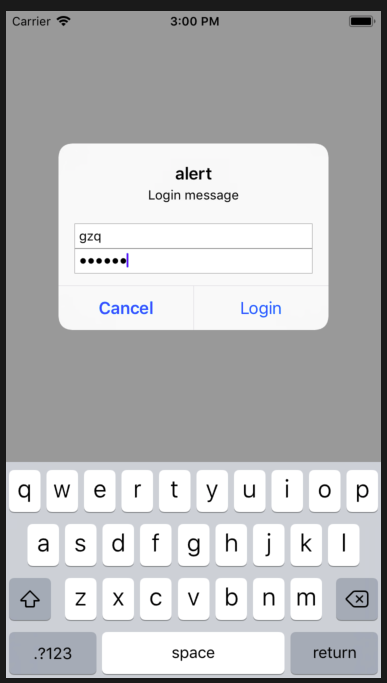
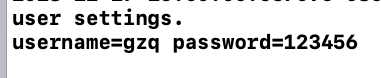
        present(alert, animated: true, completion: nil)

    }

}

* 运行结果：

1. 一个界面使用两个scrollView
   1. 在一个scrollView中可进行多张图片横屏滚动浏览(相册)，需要有pagecontrol进行提示；
   2. 在另一个scrollView中可放大缩小；

提示：需用delegate

* 程序代码：

import UIKit

class ViewController: UIViewController,UIScrollViewDelegate{

    @IBOutlet weak var scrollView1: UIScrollView!

    @IBOutlet weak var scrollView2: UIScrollView!

    @IBOutlet weak var pageControl: UIPageControl!

    override func viewDidLoad() {

        super.viewDidLoad()

        for i in 1...7{

            let imageViews = UIImageView(image: UIImage(named: "\(i)"))

            imageViews.contentMode = .scaleAspectFit

            imageViews.frame = CGRect(x: CGFloat(i-1) \* scrollView1.bounds.width, y: 0, width: scrollView1.bounds.width, height: scrollView1.bounds.height)

            scrollView1.addSubview(imageViews)

            scrollView1.isPagingEnabled = true

        }

        scrollView1.contentSize = CGSize(width: 7 \* scrollView1.bounds.width, height: scrollView1.bounds.height)

        scrollView1.delegate = self

        pageControl.currentPage = 0

        pageControl.numberOfPages = 7

        scrollView1.isUserInteractionEnabled = true

        scrollView1.showsHorizontalScrollIndicator = false

        let imageView = UIImageView(image: UIImage(named: "1"))

        scrollView2.addSubview(imageView)

        scrollView2.contentSize = imageView.bounds.size

        scrollView2.minimumZoomScale = 0.1

        scrollView2.maximumZoomScale = 5

        scrollView2.delegate = self

    }

    func scrollViewDidEndDecelerating(\_ scrollView: UIScrollView) {

        pageControl.currentPage = Int(scrollView1.contentOffset.x / scrollView1.bounds.width)

    }

    @IBAction func pageControlClicked(\_ sender: UIPageControl) {

        let rect = CGRect(x: CGFloat(pageControl.currentPage) \* scrollView1.bounds.width, y: 0, width: scrollView1.bounds.width, height: scrollView1.bounds.height)

        scrollView1.scrollRectToVisible(rect, animated: true)

    }

    func viewForZooming(in scrollView: UIScrollView) -> UIView? {

        return scrollView2.subviews.first

    }

    override func didReceiveMemoryWarning() {

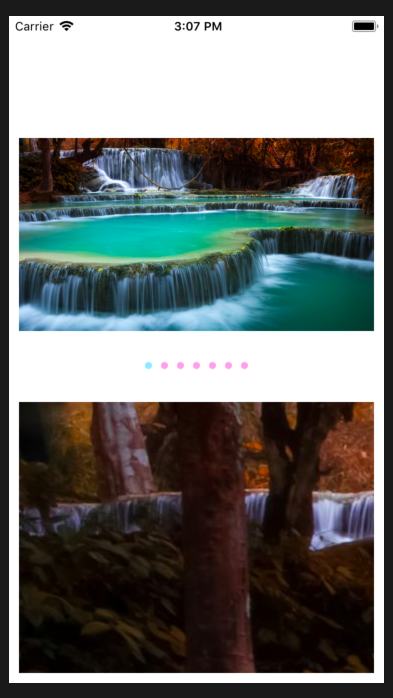
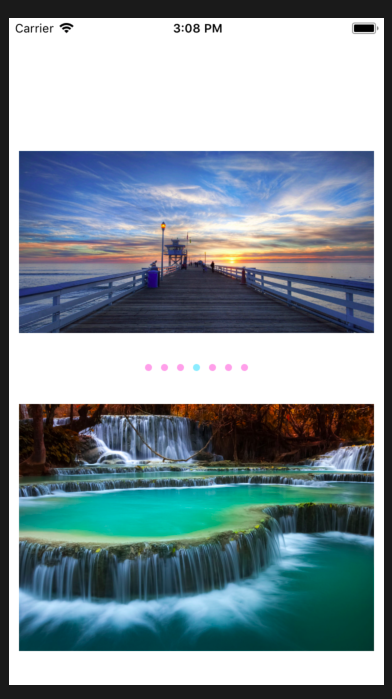
        super.didReceiveMemoryWarning()

        // Dispose of any resources that can be recreated.

    }

}

* 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：

本次实验主要是讲解iOS中的各个组件的用法，和效果，重点讲解的是：Gesture、alter、scrollView这三个。

Gesture是手势，就是我们对手机进行的一系列操作，如点击，旋转，双击以实现被点击对象的变化或者是事件的触发。

alter就是提示框，主要用于用户进行一些对象的改变时，确认自己所进行的操作。

scrollView就是滚动条，当我们加载的我图片太大时，就会自动的产生一个滚动条方便我们的预览。

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。