**12 CoreData**

**（1）AppDelegate**

import UIKit

import CoreData

// MARK: - Core Data stack

lazy var persistentContainer: NSPersistentContainer = {

//创建一个持久化容器，名称与工程名一致

let container = NSPersistentContainer(name: "Area")

container.loadPersistentStores(completionHandler: { (storeDescription, error) in

if let error = error as NSError? {

fatalError("Unresolved error \(error), \(error.userInfo)")

}

})

return container

}()

// MARK: - Core Data Saving support

func saveContext () {

let context = persistentContainer.viewContext

if context.hasChanges {

do {

try context.save()

} catch {

let nserror = error as NSError

fatalError("Unresolved error \(nserror), \(nserror.userInfo)")

}

}

}

}

**（2）AddAreaController**

class AddAreaController: UITableViewController,UIImagePickerControllerDelegate,UINavigationControllerDelegate {

var area: AreaMO!

var isVisited = false

@IBOutlet weak var tfName: UITextField!

@IBOutlet weak var tfProvince: UITextField!

@IBOutlet weak var tfPart: UITextField!

@IBOutlet weak var labelVisited: UILabel!

@IBOutlet weak var coverImageView: UIImageView!

@IBAction func saveTap(\_ sender: Any) {

//获取appDelegate，强制转换成工程的AppDelegate，可获得对它的引用

let appDelegate = UIApplication.shared.delegate as! AppDelegate

//可以获得CoreData持久化容器中的Context

area = AreaMO(context: appDelegate.persistentContainer.viewContext)

//设置相关的值

area.name = tfName.text

area.province = tfProvince.text

area.part = tfPart.text

area.isVisited = isVisited

//把图像转换为JPEG格式，0.7是图片的清晰度

if let imageData = UIImageJPEGRepresentation(coverImageView.image!, 0.7) {

area.image = NSData(data: imageData)

}

print("正在保存...")

//保存数据

appDelegate.saveContext()

//退出，返回到首页

performSegue(withIdentifier: "unwindToHomeList", sender: self)

}

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

if sender.tag == 8001 {

isVisited = true

labelVisited.text = "我来过"

} else {

isVisited = false

labelVisited.text = "没去过"

}

}

override func viewDidLoad() {

super.viewDidLoad()

}

}

**（3）AreaTableViewController**

import UIKit

import CoreData

class AreaTableViewController: UITableViewController,NSFetchedResultsControllerDelegate {

var areas: [AreaMO] = [ ]

var fc: NSFetchedResultsController<AreaMO>!

override func viewDidLoad() {

super.viewDidLoad()

fetchAllData2()

}

override func viewDidAppear(\_ animated: Bool) {

super.viewDidAppear(animated)

//调用获取数据的方法

//fetchAllData()

//整体刷新数据

//tableView.reloadData()

}

//当控制器开始处理内容变化时

func controllerWillChangeContent(\_ controller: NSFetchedResultsController<NSFetchRequestResult>) {

tableView.beginUpdates()

}

//当控制器已经处理完内容变更时

func controllerDidChangeContent(\_ controller: NSFetchedResultsController<NSFetchRequestResult>) {

tableView.endUpdates()

}

//内容发生变更时

func controller(\_ controller: NSFetchedResultsController<NSFetchRequestResult>, didChange anObject: Any, at indexPath: IndexPath?, for type: NSFetchedResultsChangeType, newIndexPath: IndexPath?) {

//变更类型筛选

switch type {

//删除数据

case .delete:

tableView.deleteRows(at: [indexPath!], with: .automatic)

//插入数据

case .insert:

tableView.insertRows(at: [newIndexPath!], with: .automatic)

//更新数据

case .update:

tableView.reloadRows(at: [indexPath!], with: .automatic)

default:

tableView.reloadData()

}

//数据已经发生变化，同步到数组

if let object = controller.fetchedObjects{

areas = object as! [AreaMO]

}

}

func fetchAllData2() {

//请求结果类型是AreaMO

let request: NSFetchRequest<AreaMO> = AreaMO.fetchRequest()

//NSSortDescriptor指定请求结果如何排序

let sd = NSSortDescriptor(key: "name", ascending: true)

request.sortDescriptors = [sd]

let appDelegate = UIApplication.shared.delegate as! AppDelegate

let context = appDelegate.persistentContainer.viewContext

//NSFetchedResultsController初始化后

fc = NSFetchedResultsController(fetchRequest: request, managedObjectContext: context, sectionNameKeyPath: nil, cacheName: nil)

//指定代理

fc.delegate = self

do {

try fc.performFetch()

//执行查询，指定代理

if let object = fc.fetchedObjects{

areas = object

}

} catch {

print(error)

}

}

//获取数据

/\*func fetchAllData(){

let appDelegate = UIApplication.shared.delegate as! AppDelegate

do {

//获取AreaMO此Enitity的所有条目

areas = try appDelegate.persistentContainer.viewContext.fetch(AreaMO.fetchRequest())

} catch {

print(error)

}

}\*/

// MARK: - Table view dalegate

override func tableView(\_ tableView: UITableView, editActionsForRowAt indexPath: IndexPath) -> [UITableViewRowAction]? {

let actionDel = UITableViewRowAction(style: .destructive, title: "删除") { (\_, indexPath) in

//从数据源中删除相应的行

//self.areas.remove(at: indexPath.row)

let appDelegate = UIApplication.shared.delegate as! AppDelegate

let context = appDelegate.persistentContainer.viewContext

//删除当前获取到的指定行的数据

context.delete(self.fc.object(at: indexPath))

//保存

appDelegate.saveContext()

//删除某一行刷新

//tableView.deleteRows(at: [indexPath], with: .fade)

//整体刷新

//tableView.reloadData()

}

}

**（4）DetailTableViewController**

import UIKit

class DetailTableViewController: UITableViewController {

@IBAction func close(segue: UIStoryboardSegue){

//反场前取回源控制器

let reviewVC = segue.source as! ReviewViewController

//得到源控制中rating中的值，并判断其中是否有值

if let rating = reviewVC.rating {

//更新模型中的rating值

self.area.rating = rating

//将当前ratingBtn中的图片设置为rating

self.ratingBtn.setImage(UIImage(named: rating), for: .normal)

}

let appDelegate = UIApplication.shared.delegate as! AppDelegate

appDelegate.saveContext()

}

override func viewDidLoad() {

super.viewDidLoad()

//预计行高

tableView.estimatedRowHeight = 40

//实际行高为自动适应

tableView.rowHeight = UITableViewAutomaticDimension

if let rating = area.rating{

self.ratingBtn.setImage(UIImage(named: rating), for: .normal)

}

}

**13 搜索**

**（1）AreaTableViewController**

class AreaTableViewController: UITableViewController,NSFetchedResultsControllerDelegate,UISearchResultsUpdating {

//定义搜索器控制器变量

var sc: UISearchController!

//定义个一个空数组保存搜索结果

var searchResult: [AreaMO] = []

override func viewDidLoad() {

super.viewDidLoad()

//参数为显示在哪个控制器里，参数如为nil，显示搜索条所在的当前tableView

sc = UISearchController(searchResultsController: nil)

//代理设置为当前控制器，由它负责更新搜索结果

sc.searchResultsUpdater = self

//显示在当前页眉

tableView.tableHeaderView = sc.searchBar

//搜索条背景不变暗

sc.dimsBackgroundDuringPresentation = false

//设置搜索条外观，为透明

sc.searchBar.searchBarStyle = .minimal

sc.searchBar.placeholder = "请输入地名进行搜索"

fetchAllData2()

}

func updateSearchResults(for searchController: UISearchController) {

//获取搜索栏中的文字

if var text = searchController.searchBar.text {

//去空格

text = text.trimmingCharacters(in: .whitespaces)

//筛选

searchFilter(text: text)

//刷新列表

tableView.reloadData()

}

}

//筛选器方法

func searchFilter(text: String) {

//swift数组自带filter方法，返回一个符合条件的新数组

searchResult = areas.filter({ (area) -> Bool in

//contains检测字符串中包含另一个字符串，localizedCaseInsensitiveContains不区分大小写

return (area.name!.localizedCaseInsensitiveContains(text))

})

}

override func tableView(\_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

//搜索条活动时，搜索结果是数据源

return sc.isActive ? searchResult.count : areas.count

}

override func tableView(\_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {

//as做类型转换，从一个类型转换到另一个类型

//as! 强制转换（失败app会崩溃，使用as!一般在转换的时候比较确定转换是成功的）

//as？安全转换（失败不会崩溃）

// 将 UITableViewCell 转换成 CumtomTableViewCell

let cell = tableView.dequeueReusableCell(withIdentifier: "cell", for: indexPath)as! CumtomTableViewCell

//搜索条活动时，数据源为搜索结果

let area = sc.isActive ? searchResult[indexPath.row] : areas[indexPath.row]

//cell中名称标签赋值

cell.nameLabel.text = area.name

//通过读取数组元素得到对应图片名称，设置imageView上的图片

cell.thumbImageView.image = UIImage(data: area.image as! Data)

//cell中省份标签赋值

cell.provinceLabel.text = area.province

//cell中地区标签赋值

cell.partLabel.text = area.part

//将图片的圆角半径设置成宽度的一半，调节layer.cornerRadius属性

cell.thumbImageView.layer.cornerRadius = cell.thumbImageView.frame.size.height/2

//让layer设置生效，将裁边选项值设为true

cell.thumbImageView.clipsToBounds = true

//重用单元格时判断是否被打钩

if area.isVisited{

cell.accessoryType = .checkmark

}else{

cell.accessoryType = .none

}

return cell

}

override func tableView(\_ tableView: UITableView, canEditRowAt indexPath: IndexPath) -> Bool {

//搜索时不可编辑，所以返回值要取反

return !sc.isActive

}

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

//判断转场的identifier是否与自定义的相等

if segue.identifier == "showAreaDetail"{

//获取转场目标的控制器

let dest = segue.destination as! DetailTableViewController

//设置目标控制器图像的名称，来自于area数组中的某一项，由当前tablevie中的行决定数组下标

dest.area = sc.isActive ? searchResult[(tableView.indexPathForSelectedRow?.row)!]

: areas[(tableView.indexPathForSelectedRow?.row)!]

}

}

//页面消失时调用该方法

override func viewWillDisappear(\_ animated: Bool) {

super.viewWillAppear(animated)

sc.isActive = false

}

}

**14 PageView**

**（1）ContentViewController**

import UIKit

class ContentViewController: UIViewController {

@IBOutlet weak var labelHeading: UILabel!

@IBOutlet weak var imageView: UIImageView!

@IBOutlet weak var labelFooter: UILabel!

@IBOutlet weak var pageControl: UIPageControl!

@IBOutlet weak var btnDone: UIButton!

//index用于保存当前页索引，其他三个用于数据传递

var index = 0

var heading = ""

var imageName = ""

var footer = ""

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

//创建一个UserDefaults实例

let defaluts = UserDefaults.standard

//存一个参数，键GuiderShow（引导已显示），值为true

defaluts.set(true, forKey: "GuiderShow")

//页面退场

dismiss(animated: true, completion: nil)

}

override func viewDidLoad() {

super.viewDidLoad()

//给控件赋值

labelFooter.text = footer

labelHeading.text = heading

imageView.image = UIImage(named: imageName)

pageControl.currentPage = index

//只要不是第三个索引页，“立即体验”按钮就隐藏

btnDone.isHidden = (index != 2)

}

}

**（2）GuiderViewController**

import UIKit

//遵从UIPageViewControllerDataSource协议

class GuiderViewController: UIPageViewController,UIPageViewControllerDataSource {

//添加头标签、尾标签、图片名的数组，用于创建各页面控制器

var headings = ["中国矿业大学","计算机科学与技术学院","卓越班"]

var images = ["eastgate","computer","ios"]

var footers = ["南湖校区","计算机专业","iOS应用开发技术"]

override func viewDidLoad() {

super.viewDidLoad()

//设置数据源为自身

dataSource = self

//创建第一个页面

if let startVC = vc(atIndex: 0){

setViewControllers([startVC], direction: .forward, animated: true, completion: nil)

}

}

func pageViewController(\_ pageViewController: UIPageViewController, viewControllerBefore viewController: UIViewController) -> UIViewController? {

//取出当前控制器的index

var index = (viewController as! ContentViewController).index

//索引+1后，返回下一个控制器

index += 1

return vc(atIndex:index)

}

func pageViewController(\_ pageViewController: UIPageViewController, viewControllerAfter viewController: UIViewController) -> UIViewController? {

var index = (viewController as! ContentViewController).index

//索引-1后，返回前一个控制器

index -= 1

return vc(atIndex:index)

}

func vc(atIndex: Int) -> ContentViewController? {

//if case判断index是否在合理的区间内

if case 0..<headings.count = atIndex{

//使用instantiateViewController初始化标识符为ContentController的控制器，并强制转化为ContentViewController类型

if let contentVC = storyboard?.instantiateViewController(withIdentifier: "ContentController") as? ContentViewController{

//为4个变量赋值

contentVC.heading = headings[atIndex]

contentVC.footer = footers[atIndex]

contentVC.imageName = images[atIndex]

contentVC.index = atIndex

return contentVC

}

}

return nil

}

/\*

//要显示的页数

func presentationCount(for pageViewController: UIPageViewController) -> Int {

return headings.count

}

//起始页的索引

func presentationIndex(for pageViewController: UIPageViewController) -> Int {

return 0

}

\*/

}

**（3）AreaTableViewController**

class AreaTableViewController: UITableViewController,NSFetchedResultsControllerDelegate,UISearchResultsUpdating {

override func viewDidAppear(\_ animated: Bool) {

super.viewDidAppear(animated)

//判断引导页是否已经显示过，如果已经显示过将不再展示

let defaults = UserDefaults.standard

if defaults.bool(forKey: "GuiderShow") {

return

}

//使用storyboard ID来初始化一个翻页控制器，并模态展示出来

if let pageVC = storyboard?.instantiateViewController(withIdentifier: "GuideController") as? GuiderViewController{

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

}

}

}