//

// PaperVC.m

// MaestroEdu

//

// Created by jesse on 2017/7/9.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperVC.h"

#import "PaperRightItem.h"

#import "PaperSonTVC.h"

#import "PaperOrganizedView.h"

#import "FTPopOverMenu.h"

#import "QuestionBankVC.h"

@interface PaperVC ()<VTMagicViewDataSource,VTMagicViewDelegate>

{

PaperRightItem \*\_rightItem;

NSMutableArray \*\_subjectArrs;

PaperOrganizedView \*\_oranizedView;

}

@property (nonatomic,strong) NSMutableArray \*testType;

@end

@implementation PaperVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

self.testType = @[].mutableCopy;

\_subjectArrs = @[].mutableCopy;

self.view.backgroundColor = OVERALLCOLOR;

self.navigationItem.leftBarButtonItem = [UIBarButtonItem barButtonItemWithTitle:@"试卷" titleColor:[UIColor blackColor] andBarButtonItemBlock:^{

}];

PaperRightItem \*rightItem = [PaperRightItem viewFromNIB];

//点击学科

//点击筛选

rightItem.clickSift = ^(UIButton \*sender) {

[self showSiftListWith:sender];

};

\_rightItem.subjectImage.hidden = YES;

\_rightItem = rightItem;

self.navigationItem.rightBarButtonItem = [[UIBarButtonItem alloc] initWithCustomView:rightItem];

self.magicView.navigationColor = [UIColor clearColor];

self.magicView.sliderColor = BORDERCOLOR;

self.magicView.sliderWidth = 60;

self.magicView.layoutStyle = VTLayoutStyleCenter;

self.magicView.switchStyle = VTSwitchStyleDefault;

self.magicView.navigationHeight = 40.f;

self.magicView.itemScale = 1.1;

self.magicView.dataSource = self;

self.magicView.delegate = self;

self.magicView.needPreloading = NO;

//配置 弹出菜单

FTPopOverMenuConfiguration \*configuration = [FTPopOverMenuConfiguration defaultConfiguration];

configuration.textAlignment = NSTextAlignmentCenter;

configuration.tintColor = BORDERCOLOR;

//配置右下角 组 view

PaperOrganizedView \*oranizedView = [PaperOrganizedView viewFromNIB];

\_oranizedView = oranizedView;

oranizedView.clickViewBlock = ^{

QuestionBankVC \*questionBankVC = kStoryboardVC(@"QuestionBank", @"QuestionBankVCSBID");

questionBankVC.paperComeIn = YES;

questionBankVC.showTitle = @"组卷";

[self.navigationController pushViewController:questionBankVC animated:YES];

};

[oranizedView.topBtn setBackgroundImage:IMG(@"icon\_Set\_volume") forState:UIControlStateNormal];

oranizedView.center = CGPointMake(SCREEN\_WIDTH - 70, SCREEN\_HEIGHT - 64 - 49 - 80);

[self.view addSubview:oranizedView];

[DefaultCenter addObserver:self selector:@selector(subjectRefresh) name:kChangeSubjectNofity object:nil];

[self requestTestType];

[self requestSubjectData];

}

-(void)viewWillAppear:(BOOL)animated

{

[super viewWillAppear:animated];

[self reqeustNumData];

}

- (void)reqeustNumData

{

[JXHttpTool POST:@"allTestStatistics" parameters:@{} success:^(id responseObject) {

[\_oranizedView.oranizedNum setTitle:[NSString stringWithFormat:@"已组卷：%@套",responseObject[@"CompleteCount"]?:EMPTY\_STRING] forState:UIControlStateNormal];

} failure:nil];

}

- (void)requestTestType

{

[JXHttpTool POST:@"testType" parameters:@{} success:^(id responseObject) {

self.testType = [NSMutableArray arrayWithArray:responseObject[@"outList"]];

[self.magicView reloadData];

} failure:nil];

}

-(void)subjectRefresh

{

//刷新logo

\_rightItem.subjectImage.image = IMG(KACCOUNTTOOL.subjectImg);

}

- (void)requestSubjectData

{

[JXHttpTool POST:@"subjectList" parameters:@{} success:^(id responseObject) {

NSMutableArray \*tempArr1 = [NSMutableArray arrayWithArray:responseObject[@"outList"]];

NSMutableArray \*tempArr = @[].mutableCopy;

for (NSDictionary \*dict in tempArr1) {

NSLog(@"%@",KACCOUNTTOOL.Faculty);

if ([dict[@"Faculty"] containsString:@"部"]) {

NSString \*temp = [dict[@"Faculty"] stringByReplacingOccurrencesOfString:@"部" withString:EMPTY\_STRING];

if ([temp isEqualToString:KACCOUNTTOOL.Faculty]) {

[tempArr addObject:dict];

}

}

}

\_subjectArrs = tempArr;

[self showSubjectList];

} failure:nil];

}

- (void)showSubjectList

{

NSMutableArray \*tempArr = @[].mutableCopy;

for (NSDictionary \*dict in \_subjectArrs) {

[tempArr addObject:dict[@"SubjectsName"]];

}

NSString \*showSubject;

//默认第一个

if (!KACCOUNTTOOL.subject) {

showSubject = tempArr[0];

KACCOUNTTOOL.subject = [NSString stringWithFormat:@"%@",\_subjectArrs[0][@"SubjectsId"]];

KACCOUNTTOOL.subjectImg = [NSString stringWithFormat:@"%@",\_subjectArrs[0][@"SubjectsName"]];

[KACCOUNTTOOL saveUserAccount];

} else {

showSubject = KACCOUNTTOOL.subjectImg;

}

\_\_weak \_\_typeof(&\*\_rightItem)weakSelf\_rightItem = \_rightItem;

\_\_weak \_\_typeof(&\*\_subjectArrs)weakSelf\_subjectArrs = \_subjectArrs;

\_rightItem.subjectImage.hidden = NO;

\_rightItem.subjectImage.image = IMG(KACCOUNTTOOL.subjectImg);

\_rightItem.clickSubject = ^(UIButton \*sender) {

[FTPopOverMenu showForSender:sender

withMenuArray:tempArr

imageArray:nil

doneBlock:^(NSInteger selectedIndex) {

weakSelf\_rightItem.subjectImage.image = IMG(tempArr[selectedIndex]);

//切换科目 刷新

KACCOUNTTOOL.subject = [NSString stringWithFormat:@"%@",weakSelf\_subjectArrs[selectedIndex][@"SubjectsId"]];

KACCOUNTTOOL.subjectImg = [NSString stringWithFormat:@"%@",weakSelf\_subjectArrs[selectedIndex][@"SubjectsName"]];

[KACCOUNTTOOL saveUserAccount];

[DefaultCenter postNotificationName:kChangeSubjectNofity object:nil];

NSLog(@"done block. do something. selectedIndex : %ld", (long)selectedIndex);

} dismissBlock:^{

NSLog(@"user canceled. do nothing.");

}];

};

[DefaultCenter postNotificationName:kChangeSubjectNofity object:nil];

}

- (void)showSiftListWith:(UIButton \*)sender

{

NSArray \*images = @[@"全部", @"未发布", @"待批改", @"已完成"];

[FTPopOverMenu showForSender:sender

withMenuArray:images

imageArray:nil

doneBlock:^(NSInteger selectedIndex) {

[DefaultCenter postNotificationName:kChangePaperTypeNotify object:[NSString stringWithFormat:@"%ld",(long)selectedIndex - 1]];

NSLog(@"done block. do something. selectedIndex : %ld", (long)selectedIndex);

} dismissBlock:^{

NSLog(@"user canceled. do nothing.");

}];

}

- (NSArray<NSString \*> \*)menuTitlesForMagicView:(VTMagicView \*)magicView {

NSMutableArray \*temp = @[].mutableCopy;

for (NSDictionary \*dict in self.testType) {

[temp addObject:dict[@"Name"]];

}

return temp;

}

- (UIButton \*)magicView:(VTMagicView \*)magicView menuItemAtIndex:(NSUInteger)itemIndex {

static NSString \*itemIdentifier = @"itemIdentifier";

UIButton \*menuItem = [magicView dequeueReusableItemWithIdentifier:itemIdentifier];

if (!menuItem) {

menuItem = [UIButton buttonWithType:UIButtonTypeCustom];

[menuItem setTitleColor:[UIColor colorWithHexString:@"#9a9a9a"] forState:UIControlStateNormal];

[menuItem setTitleColor:BORDERCOLOR forState:UIControlStateSelected];

menuItem.titleLabel.font = FONT(15);

}

return menuItem;

}

- (UIViewController \*)magicView:(VTMagicView \*)magicView viewControllerAtPage:(NSUInteger)pageIndex {

static NSString \*recomVCId = @"paperSonTVC.ID";

PaperSonTVC \*paperSonTVC = [magicView dequeueReusablePageWithIdentifier:recomVCId];

if (!paperSonTVC) {

paperSonTVC = [[PaperSonTVC alloc] initWithStyle:UITableViewStyleGrouped];

}

paperSonTVC.pageIndex = pageIndex;

return paperSonTVC;

}

@end

//

// PaperSonTVC.m

// MaestroEdu

//

// Created by 张明辉 on 2017/7/10.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperSonTVC.h"

#import "PaperSonCell.h"

#import "PaperListVC.h"

#import "PaperListFinashTVC.h"

#import "PaperStructureTVC.h"

#import "PaperSonListModel.h"

@interface PaperSonTVC ()

{

NSString \*\_inIsstate;

}

@end

@implementation PaperSonTVC

- (void)viewDidLoad {

[super viewDidLoad];

WEAKSELF

kTableViewFuckingInit(weakSelf\_SC.tableView, requestListData);

\_inIsstate = @"";

[DefaultCenter addObserver:self selector:@selector(refreshPaperType:) name:kChangePaperTypeNotify object:nil];

}

- (void)refreshPaperType:(NSNotification \*)notification

{

if ([notification.object isEqualToString:@"-1"]) {

\_inIsstate = @"";

} else {

\_inIsstate = notification.object;

}

kTableViewHeaderAction(self.tableView);

[self requestListData];

}

- (void)viewWillAppear:(BOOL)animated

{

[super viewWillAppear:animated];

kTableViewHeaderAction(self.tableView);

[self requestListData];

}

- (void)requestListData

{

[JXHttpTool POST:@"selTestPaperApp" parameters:@{@"incurrentPage":self.tableView.CurrentPage?:@"1",@"inshowCount":@"20",@"inAttributeType":@(self.pageIndex)?:@"",@"inIsstate":\_inIsstate?:@""} success:^(id responseObject) {

kTableViewAfterDataAction(self.tableView, responseObject, [PaperSonListModel parse:responseObject[@"outList"]]);

} failure:nil];

}

-(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

{

return self.tableView.tableViewDatas.count;

}

- (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

{

PaperSonCell \*cell = [PaperSonCell cellWithNibTableView:tableView andReuseID:@"PaperSonCellSBID"];

[cell addDataWith:self.tableView.tableViewDatas[indexPath.row]];

return cell;

}

-(void)tableView:(UITableView \*)tableView didSelectRowAtIndexPath:(NSIndexPath \*)indexPath

{

PaperSonListModel \*model = self.tableView.tableViewDatas[indexPath.row];

if ([model.Isstate isEqualToNumber:@2]) {

//已完成

PaperListFinashTVC \*finashTVC = [[PaperListFinashTVC alloc] init];

finashTVC.inId = [NSString stringWithFormat:@"%@",model.Id?:EMPTY\_STRING];

finashTVC.showTitle = @"练习回顾";

[self.navigationController pushViewController:finashTVC animated:YES];

} else {

PaperListVC \*listVC = [[PaperListVC alloc] init];

if ([model.Isstate isEqualToNumber:@0]) {

//未发布

listVC.isUnRelease = YES;

} else if ([model.Isstate isEqualToNumber:@1]) {

//待批改

listVC.isUnCorrect = YES;

}

listVC.inId = [NSString stringWithFormat:@"%@",model.Id?:EMPTY\_STRING];

[self.navigationController pushViewController:listVC animated:YES];

}

}

@end

//

// PaperListVC.m

// MaestroEdu

//

// Created by 张明辉 on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperListVC.h"

#import "PaperListCell.h"

#import "PaperListFinashModel.h"

#import "PaperListFinashSonModel.h"

#import "PaperBrowseVC.h"

#import "PaperAttributeTVC.h"

#import "PaperCorrectConfigVT.h"

@interface PaperListVC ()<UITableViewDelegate,UITableViewDataSource>

@property (nonatomic,strong) UITableView \*tableView;

@property (nonatomic,strong) NSArray \*datas;

@end

@implementation PaperListVC

- (void)viewDidLoad {

[super viewDidLoad];

self.tableView = [[UITableView alloc] initWithFrame:CGRectMake(0, 0, SCREEN\_WIDTH, SCREEN\_HEIGHT - 64 - 60) style:UITableViewStylePlain];

self.tableView.backgroundColor = OVERALLCOLOR;

self.tableView.delegate = self;

self.tableView.dataSource = self;

self.tableView.estimatedRowHeight = 400;

self.tableView.separatorStyle = UITableViewCellSeparatorStyleNone;

[self.view addSubview:self.tableView];

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

bottomBtn.frame = CGRectMake(0, CGRectGetMaxY(self.tableView.frame), 155, 36);

bottomBtn.centerX = SCREEN\_WIDTH \* 0.5;

[bottomBtn setTitleColor:BORDERCOLOR forState:UIControlStateNormal];

[bottomBtn setBackgroundImage:IMG(@"border\_code") forState:UIControlStateNormal];

if (self.isUnRelease) {

//未发布

self.title = @"试题列表";

[bottomBtn setTitle:@"发布" forState:UIControlStateNormal];

[bottomBtn setTapActionWithBlock:^{

PaperAttributeTVC \*attr = kStoryboardVC(@"Paper", @"PaperAttributeTVCSBID");

attr.inId = self.inId;

attr.isBorrowListComeIn = YES;

[self.navigationController pushViewController:[JXBottomBtnController bottomBtnControllerWith:attr bottomTitle:@"下一步" andPara:nil] animated:YES];

}];

} else if (self.isUnCorrect) {

//待批改

self.title = @"试题列表";

[bottomBtn setTitle:@"批改" forState:UIControlStateNormal];

[bottomBtn setTapActionWithBlock:^{

PaperCorrectConfigVT \*config = [[PaperCorrectConfigVT alloc] init];

config.inAttributeId = self.inId;

[self.navigationController pushViewController:config animated:YES];

}];

}

[self.view addSubview:bottomBtn];

[self requestData];

}

- (void)requestData

{

NSDictionary \*para;

if (self.isUnCorrect) {

//待批改

para = @{@"inId":self.inId?:EMPTY\_STRING,@"inIsstate":@"1"};

} else if (self.isUnRelease) {

//未发布

para = @{@"inId":self.inId?:EMPTY\_STRING,@"inIsstate":@"0"};

}

[JXHttpTool POST:@"appTestListHomeWork" parameters:para success:^(id responseObject) {

self.datas = [PaperListFinashModel parse:responseObject[@"outList"]];

[self.tableView reloadData];

} failure:nil];

}

-(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

{

return self.datas.count;

}

-(UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

{

PaperListCell \*cell = [PaperListCell cellWithNibTableView:tableView andReuseID:@"PaperListCellID"];

[cell addDataWith:self.datas[indexPath.row] andRow:indexPath.row isFinash:NO];

cell.clickBtnBlock = ^(NSInteger row) {

PaperBrowseVC \*paperBrowseVC = [[PaperBrowseVC alloc] init];

paperBrowseVC.inId = self.inId?:EMPTY\_STRING;

paperBrowseVC.incurrentPage = [NSString stringWithFormat:@"%@",[[self.datas[indexPath.row] Tests][row] Number]];

paperBrowseVC.paperBrowerListComeIn = YES;

NSMutableArray \*arr = @[].mutableCopy;

for (PaperListFinashSonModel \*model in [self.datas[indexPath.row] Tests]) {

[arr addObject:model.Number];

}

paperBrowseVC.numbers = arr;

paperBrowseVC.currentPageUI = row;

[self.navigationController pushViewController:paperBrowseVC animated:YES];

};

return cell;

}

@end

//

// PaperListFinashTVC.m

// MaestroEdu

//

// Created by jesse on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperListFinashTVC.h"

#import "PaperListCell.h"

#import "PaperListUnSubmitStudentCell.h"

#import "PaperListFinashSituationCell.h"

#import "PaperListFinashModel.h"

#import "PaperListNoCompleteModel.h"

#import "PaperListNoCompleteSonModel.h"

#import "PaperBrowseVC.h"

#import "PaperListFinashSonModel.h"

#import "PaperListAverageListModel.h"

@interface PaperListFinashTVC ()

@property (nonatomic,strong) NSArray \*datas;

@property (nonatomic,strong) NSArray \*noCompleteDatas;

@property (nonatomic,strong) PaperListAverageListModel \*averageListModel;

@end

@implementation PaperListFinashTVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

self.title = @"试题列表";

if (self.showTitle) {

self.title = self.showTitle;

}

self.tableView.estimatedRowHeight = 400;

self.tableView.separatorStyle = UITableViewCellSeparatorStyleNone;

[self requestData];

}

- (void)requestData

{

[JXHttpTool POST:@"appTestListHomeWork" parameters:@{@"inId":self.inId?:EMPTY\_STRING,@"inIsstate":@"2"} success:^(id responseObject) {

self.datas = [PaperListFinashModel parse:responseObject[@"outList"]];

self.noCompleteDatas = [PaperListNoCompleteModel parse:responseObject[@"outNoComplete"]];

self.averageListModel = [PaperListAverageListModel parse:responseObject[@"AverageList"]];

[self.tableView reloadData];

} failure:nil];

}

-(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

{

return self.datas.count + 2;

}

-(UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

{

if (indexPath.row == self.datas.count) {

//未提交学生

PaperListUnSubmitStudentCell \*cell = [PaperListUnSubmitStudentCell cellWithNibTableView:tableView andReuseID:@"PaperListUnSubmitStudentCellSBID"];

[cell addDataWith:[self.noCompleteDatas[0] Students]];

return cell;

} else if (indexPath.row == self.datas.count + 1) {

//最底部的

PaperListFinashSituationCell \*cell = [PaperListFinashSituationCell cellWithNibTableView:tableView andReuseID:@"PaperListFinashSituationCellSBID"];

[cell addDataWith:self.averageListModel];

return cell;

}

//大圈圈

PaperListCell \*cell = [PaperListCell cellWithNibTableView:tableView andReuseID:@"PaperListCellID"];

cell.clickBtnBlock = ^(NSInteger row) {

PaperBrowseVC \*paperBrowseVC = [[PaperBrowseVC alloc] init];

paperBrowseVC.inId = self.inId?:EMPTY\_STRING;

paperBrowseVC.incurrentPage = [NSString stringWithFormat:@"%@",[[self.datas[indexPath.row] Tests][row] Number]];

paperBrowseVC.paperBrowerListComeIn = YES;

NSMutableArray \*arr = @[].mutableCopy;

for (PaperListFinashSonModel \*model in [self.datas[indexPath.row] Tests]) {

[arr addObject:model.Number];

}

paperBrowseVC.numbers = arr;

paperBrowseVC.currentPageUI = row;

[self.navigationController pushViewController:paperBrowseVC animated:YES];

};

[cell addDataWith:self.datas[indexPath.row] andRow:indexPath.row isFinash:YES];

return cell;

}

@end

//

// PaperCorrectConfigVT.m

// MaestroEdu

//

// Created by 张明辉 on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperCorrectConfigVT.h"

#import "PaperCorrectConfigSonTVC.h"

#import "PaperListVC.h"

#import "PaperBrowseModel.h"

#import "PaperBrowseVC.h"

@interface PaperCorrectConfigVT ()<VTMagicViewDataSource,VTMagicViewDelegate>

{

int \_currentPage;

}

@end

@implementation PaperCorrectConfigVT

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

\_currentPage = 0;

self.title = @"试题批改设置";

self.view.backgroundColor = OVERALLCOLOR;

self.magicView.navigationColor = [UIColor clearColor];

self.magicView.sliderColor = BORDERCOLOR;

self.magicView.sliderWidth = 60;

self.magicView.layoutStyle = VTLayoutStyleCenter;

self.magicView.switchStyle = VTSwitchStyleDefault;

self.magicView.navigationHeight = 40.f;

self.magicView.itemScale = 1.1;

self.magicView.dataSource = self;

self.magicView.delegate = self;

self.magicView.needPreloading = NO;

[self.magicView reloadData];

UIView \*bottomView = [[UIView alloc] initWithFrame:CGRectMake(0, SCREEN\_HEIGHT - 64 - 60, SCREEN\_WIDTH, 60)];

bottomView.backgroundColor = OVERALLCOLOR;

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

bottomBtn.frame = CGRectMake(0, 5, 105, 36);

bottomBtn.centerX = bottomView.width \* 0.5;

[bottomBtn setTitleColor:BORDERCOLOR forState:UIControlStateNormal];

[bottomBtn setBackgroundImage:IMG(@"border\_code") forState:UIControlStateNormal];

[bottomBtn setTitle:@"开始批改" forState:UIControlStateNormal];

[bottomBtn setTapActionWithBlock:^{

NSArray \*arr = @[@"题型",@"试卷"];

PaperBrowseVC \*paperBrowseVC = [[PaperBrowseVC alloc] init];

paperBrowseVC.paperCorrectComeIn = YES;

paperBrowseVC.inType = arr[\_currentPage];

paperBrowseVC.inAttributeId = self.inAttributeId;

[self.navigationController pushViewController:paperBrowseVC animated:YES];

// kPushBottomBtnController(@"Paper", @"PaperCorrectTVCSBID", @"下一页", nil);

}];

[bottomView addSubview:bottomBtn];

[self.view addSubview:bottomView];

}

- (NSArray<NSString \*> \*)menuTitlesForMagicView:(VTMagicView \*)magicView {

return @[@"按题型批改",@"按试卷批改"];

}

- (UIButton \*)magicView:(VTMagicView \*)magicView menuItemAtIndex:(NSUInteger)itemIndex {

static NSString \*itemIdentifier = @"itemIdentifier";

UIButton \*menuItem = [magicView dequeueReusableItemWithIdentifier:itemIdentifier];

if (!menuItem) {

menuItem = [UIButton buttonWithType:UIButtonTypeCustom];

[menuItem setTitleColor:[UIColor colorWithHexString:@"#9a9a9a"] forState:UIControlStateNormal];

[menuItem setTitleColor:BORDERCOLOR forState:UIControlStateSelected];

menuItem.titleLabel.font = FONT(15);

}

return menuItem;

}

- (UIViewController \*)magicView:(VTMagicView \*)magicView viewControllerAtPage:(NSUInteger)pageIndex {

static NSString \*recomVCId = @"PaperCorrectConfigSonTVC.ID";

PaperCorrectConfigSonTVC \*paperCorrectConfigSonTVC = [magicView dequeueReusablePageWithIdentifier:recomVCId];

if (!paperCorrectConfigSonTVC) {

paperCorrectConfigSonTVC = kStoryboardVC(@"Paper", @"PaperCorrectConfigSonTVCSBID");

}

\_currentPage = (int)pageIndex;

paperCorrectConfigSonTVC.pageIndex = pageIndex;

paperCorrectConfigSonTVC.inAttributeId = self.inAttributeId;

return paperCorrectConfigSonTVC;

}

@end

//

// PaperAttributeTVC.m

// MaestroEdu

//

// Created by jesse on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperAttributeTVC.h"

#import "PaperStructureTVC.h"

#import "FlyDatePickerView.h"

#import "MineStudentManagerTVC.h"

@interface PaperAttributeTVC ()

{

NSMutableArray \*\_Grades;

NSString \*\_selectedGrade;//选择年级

NSString \*\_selectedGradeName;//选择年级名称

NSMutableArray \*\_testType;

NSString \*\_selectedTestType;//选择试卷类型

NSString \*\_selectStudentJson;//选择的发布对象 学生

NSString \*\_selectedClassJson;//选择班级

NSString \*\_publicTimeStr;//发布日期

}

@property (weak, nonatomic) IBOutlet UILabel \*subject;

@property (weak, nonatomic) IBOutlet UITextField \*showTitle;

@property (weak, nonatomic) IBOutlet UITextField \*titleContent;

@property (weak, nonatomic) IBOutlet UITextField \*paperPeople;

@property (weak, nonatomic) IBOutlet UIButton \*publicNow;

@property (weak, nonatomic) IBOutlet UIButton \*publicTime;

@property (weak, nonatomic) IBOutlet UILabel \*grade;

@property (weak, nonatomic) IBOutlet UILabel \*type;

@property (weak, nonatomic) IBOutlet UILabel \*publicObj;

@property (weak, nonatomic) IBOutlet UILabel \*classess;

@property (weak, nonatomic) IBOutlet UILabel \*classessOrStudent;

@property (weak, nonatomic) IBOutlet UILabel \*typeDetail;

@end

@implementation PaperAttributeTVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

\_Grades = @[].mutableCopy;

self.subject.text = KACCOUNTTOOL.subjectImg?:EMPTY\_STRING;

[self requestGradeList];

[self requestTestType];

}

- (IBAction)publicNowBtn:(UIButton \*)sender {

self.publicNow.selected = YES;

self.publicTime.selected = NO;

}

- (IBAction)selectTimePublic:(UIButton \*)sender {

self.publicNow.selected = NO;

self.publicTime.selected = YES;

[FlyDatePickerView showInView:nil dateModel:UIDatePickerModeDateAndTime completionBlock:^(NSDate \*date) {

NSDateFormatter\* formatter = [[NSDateFormatter alloc] init];

[formatter setDateFormat:@"yyyy-MM-dd HH:MM:SS"];

// \_publicTimeStr = [NSString stringWithFormat:@"%ld", (long)[date timeIntervalSince1970]]; //时间戳

\_publicTimeStr = [formatter stringFromDate:date];

}];

}

- (void)requestGradeList

{

[JXHttpTool POST:@"gradeList" parameters:@{} success:^(id responseObject) {

NSLog(@"%@",responseObject);

NSMutableArray \*grades = [NSMutableArray arrayWithArray: responseObject[@"outList"]];

NSMutableArray \*tempArr = @[].mutableCopy;

for (NSDictionary \*dict in grades) {

if ([dict[@"Faculty"] containsString:@"部"]) {

NSString \*temp = [dict[@"Faculty"] stringByReplacingOccurrencesOfString:@"部" withString:EMPTY\_STRING];

if ([temp isEqualToString:KACCOUNTTOOL.Faculty]) {

[tempArr addObject:dict];

}

}

}

\_Grades = tempArr;

} failure:^(id error) {

NSLog(@"%@",error);

}];

}

- (void)requestTestType

{

[JXHttpTool POST:@"testType" parameters:@{} success:^(id responseObject) {

\_testType = [NSMutableArray arrayWithArray:responseObject[@"outList"]];

} failure:nil];

}

- (void)clickBottomBtn

{

NSString \*publicObjStr;

NSString \*publicObjJsonStr;

if ([self.classessOrStudent.text isEqualToString:@"选择班级"]) {

publicObjStr = @"0";

publicObjJsonStr = \_selectedClassJson;

} else {

publicObjStr = @"1";

publicObjJsonStr = \_selectStudentJson;

}

NSString \*inReleaseDate;

if (self.publicNow.selected) {

//立即发布

NSDate \*senddate=[NSDate date];

NSDateFormatter \*dateformatter=[[NSDateFormatter alloc] init];

[dateformatter setDateFormat:@"YYYY-MM-dd HH:mm:ss"];

inReleaseDate = [dateformatter stringFromDate:senddate];

} else {

if (!\_publicTimeStr || [\_publicTimeStr isEqualToString:EMPTY\_STRING]) {

SHOWERROR(@"请选择发布时间");

return;

}

inReleaseDate = \_publicTimeStr;

}

if (self.isBorrowListComeIn) {

//从试卷列表大圈圈进入

[JXHttpTool POST:@"upRelease" parameters:@{@"inId":self.inId?:EMPTY\_STRING,@"inReleaseDate":inReleaseDate?:EMPTY\_STRING,@"inReleaseType":publicObjStr?:EMPTY\_STRING,@"inClassObj":publicObjJsonStr?:EMPTY\_STRING} success:^(id responseObject) {

NSLog(@"%@",responseObject);

SHOWSUCCESSAFTERONESECOND(responseObject[MSG]);

[self.navigationController popToRootViewControllerAnimated:NO];

[ROOTTABBARVC setSelectedIndex:2];

} failure:nil];

return;

}

NSString \*inAttributeJson;

if ([\_selectedTestType isEqualToString:@"1"]) {

kJudgeSameShowError(self.typeDetail.text, @"请选择", @"请选择分类");

inAttributeJson = @{@"radioType":self.typeDetail.text?:EMPTY\_STRING}.mj\_JSONString;

} else {

inAttributeJson = @{}.mj\_JSONString;

}

[JXHttpTool POST:@"addTestAttribute" parameters:@{@"inTitle":self.showTitle.text?:EMPTY\_STRING,@"inDescribe":self.titleContent.text?:EMPTY\_STRING,@"inGradeId":\_selectedGrade?:EMPTY\_STRING,@"inGradeName":\_selectedGradeName?:EMPTY\_STRING,@"inSubjectId":KACCOUNTTOOL.subject?:EMPTY\_STRING,@"inSubjectName":KACCOUNTTOOL.subjectImg?:EMPTY\_STRING,@"inProblemMaker":self.paperPeople.text?:EMPTY\_STRING,@"inAttributeType":\_selectedTestType?:EMPTY\_STRING,@"inAttributeJson":inAttributeJson,@"inIsstate":@"1",@"inReleaseDate":inReleaseDate?:EMPTY\_STRING,@"inReleaseType":publicObjStr?:EMPTY\_STRING,@"inClassObj":publicObjJsonStr?:EMPTY\_STRING} success:^(id responseObject) {

PaperStructureTVC \* paperStructureTVC = [[PaperStructureTVC alloc] init];

paperStructureTVC.title = @"试题结构";

paperStructureTVC.inReleaseDate = inReleaseDate;

paperStructureTVC.inReleaseType = publicObjStr;

paperStructureTVC.inClassObj = publicObjJsonStr;

paperStructureTVC.inId = [NSString stringWithFormat:@"%@",responseObject[@"Id"]];

[self.navigationController pushViewController:[JXBottomBtnController bottomBtnControllerWith: paperStructureTVC bottomTitle:@"发布" andPara:nil] animated:YES];

} failure:nil];

}

-(void)tableView:(UITableView \*)tableView didSelectRowAtIndexPath:(NSIndexPath \*)indexPath

{

[super tableView:tableView didSelectRowAtIndexPath:indexPath];

if (indexPath.row == 3) {

//选择年级

if (\_Grades.count == 0) {

return;

}

//点击年级

NSMutableArray \*temp = @[].mutableCopy;

for (NSDictionary \*dict in \_Grades) {

[temp addObject:dict[@"GradeName"]];

}

[JXShowSelectTool showSelectWithArr:temp andDefaultStr:temp[0] complete:^(NSString \*selectedStr, NSInteger num) {

self.grade.text = selectedStr;

\_selectedGrade = \_Grades[num][@"GradeId"];

\_selectedGradeName = \_Grades[num][@"GradeName"];

}];

} else if (indexPath.row == 4) {

//选择类型

NSMutableArray \*temp = @[].mutableCopy;

for (NSDictionary \*dict in \_testType) {

[temp addObject:dict[@"Name"]];

}

[JXShowSelectTool showSelectWithArr:temp andDefaultStr:temp[0] complete:^(NSString \*selectedStr, NSInteger num) {

self.type.text = selectedStr;

\_selectedTestType = \_testType[num][@"NameKey"];

[self.tableView reloadData];

}];

} else if (indexPath.row == 5) {

//选择分类

NSArray \*types = @[@"月考",@"期中",@"期末"];

[JXShowSelectTool showSelectWithArr:types andDefaultStr:types[0] complete:^(NSString \*selectedStr, NSInteger num) {

self.typeDetail.text = selectedStr;

}];

} else if (indexPath.row == 8) {

//选择发布对象

NSArray \*arr = @[@"班级",@"学生"];

[JXShowSelectTool showSelectWithArr:arr andDefaultStr:arr[0] complete:^(NSString \*selectedStr, NSInteger num) {

self.publicObj.text = selectedStr;

self.classessOrStudent.text = [NSString stringWithFormat:@"选择%@",selectedStr];

self.classess.text = @"请选择";

\_selectStudentJson = EMPTY\_STRING;

\_selectedClassJson = EMPTY\_STRING;

}];

} else if (indexPath.row == 9) {

if ([self.classessOrStudent.text isEqualToString:@"选择班级"]) {

//点击班级

MineStudentManagerTVC \*studentManager = kStoryboardVC(@"Mine", @"MineStudentManagerTVCSBID");

studentManager.paperAttributeShowClassess = YES;

studentManager.showTitle = @"选择班级";

studentManager.clickFinishSelect = ^(NSArray \*arr) {

NSLog(@"%@",arr);

NSMutableArray \*temp = @[].mutableCopy;

NSMutableArray \*nameArr = @[].mutableCopy;

for (NSDictionary \*dict in arr) {

[temp addObject:dict[@"ClassId"]];

[nameArr addObject:dict[@"ClassName"]];

}

\_selectedClassJson = temp.mj\_JSONString;

self.classess.text = [nameArr componentsJoinedByString:@"、"];

NSLog(@"%@",\_selectedClassJson);

};

JXBottomBtnController \*vc = [JXBottomBtnController bottomBtnControllerWith:studentManager bottomTitle:@"下一步" andPara:nil];

[self.navigationController pushViewController:vc animated:YES];

} else {

//选择学生

MineStudentManagerTVC \*studentManager = kStoryboardVC(@"Mine", @"MineStudentManagerTVCSBID");

studentManager.paperAttribute = YES;

studentManager.showTitle = @"选择学生";

studentManager.clickFinishSelect = ^(NSArray \*arr) {

NSLog(@"%@",arr);

NSMutableArray \*temp = @[].mutableCopy;

NSMutableArray \*nameArr = @[].mutableCopy;

for (NSDictionary \*dict in arr) {

[temp addObject:@{@"SId":dict[@"StudentId"]?:EMPTY\_STRING,@"CId":dict[@"ClassId"]?:EMPTY\_STRING}];

[nameArr addObject:dict[@"StudentName"]];

}

\_selectStudentJson = temp.mj\_JSONString;

self.classess.text = [nameArr componentsJoinedByString:@"、"];

NSLog(@"%@",\_selectStudentJson);

};

JXBottomBtnController \*vc = [JXBottomBtnController bottomBtnControllerWith:studentManager bottomTitle:@"下一步" andPara:nil];

[self.navigationController pushViewController:vc animated:YES];

}

}

}

-(CGFloat)tableView:(UITableView \*)tableView heightForRowAtIndexPath:(NSIndexPath \*)indexPath

{

if (indexPath.row == 5) {

if ([\_selectedTestType isEqualToString:@"1"]) {

//选择了阶段考试之后

return [super tableView:tableView heightForRowAtIndexPath:indexPath];

} else {

return 0.0001f;

}

}

if (self.isBorrowListComeIn) {

if (indexPath.row == 0

|| indexPath.row == 1

|| indexPath.row == 2

|| indexPath.row == 3

|| indexPath.row == 4

|| indexPath.row == 5

|| indexPath.row == 6

) {

return 0.0001f;

}

}

return [super tableView:tableView heightForRowAtIndexPath:indexPath];

}

@end

//

// PaperStructureTVC.m

// MaestroEdu

//

// Created by jesse on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperStructureTVC.h"

#import "PaperStructureCell.h"

#import "PaperStructureModel.h"

@interface PaperStructureTVC ()

@property (nonatomic,strong) NSMutableArray \*datas;

@end

@implementation PaperStructureTVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

self.tableView.separatorStyle = UITableViewCellSeparatorStyleNone;

self.tableView.tableFooterView = [[UIView alloc] init];

self.tableView.estimatedRowHeight = 200;

self.tableView.rowHeight = UITableViewAutomaticDimension;//设置cell的高度为自动计算，只有才xib或者storyboard上自定义的cell才会生效，而且需要设置好约束

[self requestData];

}

- (void)requestData

{

[JXHttpTool POST:@"selScoreCart" parameters:@{} success:^(id responseObject) {

self.datas = [NSMutableArray arrayWithArray:[PaperStructureModel parse:responseObject[@"outList"]]];

[self.tableView reloadData];

} failure:nil];

}

-(void)clickBottomBtn

{

NSMutableArray \*arr = [PaperStructureModel mj\_keyValuesArrayWithObjectArray:self.datas];

for (NSMutableDictionary \*dict in arr) {

[dict removeObjectForKey:@"everyPaper"];

for (NSMutableDictionary \*dictSon in dict[@"Tests"]) {

[dictSon removeObjectForKey:@"everyQuestion"];

if (![dictSon[@"Score"] isEqualToString:EMPTY\_STRING]) {

if ([dictSon[@"Score"] containsString:@"."]) {

if (![[dictSon[@"Score"] componentsSeparatedByString:@"."][1] isEqualToString:@"5"]) {

//小数点后不是5的话

SHOWERROR(@"分数应为0.5的倍数")

return;

}

}

}

}

}

[JXHttpTool POST:@"addScore" parameters:@{@"TestList":arr.mj\_JSONString} success:^(id responseObject) {

[JXHttpTool POST:@"EndComplete" parameters:@{} success:^(id responseObject) {

// [JXHttpTool POST:@"upRelease" parameters:@{@"inId":self.inId?:EMPTY\_STRING,@"inReleaseDate":self.inReleaseDate?:EMPTY\_STRING,@"inReleaseType":self.inReleaseType?:EMPTY\_STRING,@"inClassObj":self.inClassObj?:EMPTY\_STRING} success:^(id responseObject) {

// NSLog(@"2233333322");

SHOWSUCCESSAFTERONESECOND(responseObject[MSG]);

[self.navigationController popToRootViewControllerAnimated:NO];

[ROOTTABBARVC setSelectedIndex:2];

// } failure:nil];

} failure:nil];

} failure:nil];

}

-(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

{

return self.datas.count;

}

- (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

{

PaperStructureCell \*cell = [PaperStructureCell cellWithNibTableView:tableView andReuseID:@"PaperStructureCellSBID"];

[cell addDataWith:self.datas[indexPath.row]];

cell.clickSortBtn = ^{

//点击第几题

NSMutableArray \*temp = @[].mutableCopy;

for (int i = 0; i < self.datas.count; i ++) {

[temp addObject:[NSString stringWithFormat:@"%d",i + 1]];

}

[JXShowSelectTool showSelectWithArr:temp andDefaultStr:temp[0] complete:^(NSString \*selectedStr, NSInteger num) {

for (PaperStructureModel \*model in self.datas) {

if ([selectedStr isEqualToString:[NSString stringWithFormat:@"%@",model.Sort]]) {

//选择的数字 跟 原本数字一样 换sort

NSNumber \*tempNum = [self.datas[indexPath.row] Sort];

((PaperStructureModel \*)self.datas[indexPath.row]).Sort = model.Sort;

model.Sort = tempNum;

}

}

self.datas = [self.datas sortedArrayUsingSelector:@selector(compareSort:)].mutableCopy;

[self.tableView reloadData];

}];

};

return cell;

}

@end

//

// PaperBrowseVC.m

// MaestroEdu

//

// Created by jesse on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperBrowseVC.h"

#import "PaperObjectView.h"

#import "FTPopOverMenu.h"

#import "PaperBrowseModel.h"

#import "PaperBrowseProblemModel.h"

#import "PaperBrowseQuestionsModel.h"

#import "PaperBrowseOptionModel.h"

#import "SunOrderArray.h"

#import "PaperBrowseCollectListView.h"

#import "UISlipUpTool.h"

#import "HomeCollectCreateAndEdit.h"

#import "PaperCorrectMistakeTVC.h"

#import "SnShareTool.h"

@interface PaperBrowseVC ()<UIScrollViewDelegate>

{

NSInteger \_currentPage;

NSMutableArray \*\_allDatasArr;

PaperBrowseModel \*\_currentModel;

int \_homeWordCurrentPage;

BOOL \_isLastNum;

}

//最大的scrollview

@property (nonatomic,strong) UIScrollView \*scrollView;

@property (nonatomic,strong) PaperBrowseModel \*model;

@end

@implementation PaperBrowseVC

- (void)viewDidLoad {

[super viewDidLoad];

self.title = @"试题浏览";

\_currentPage = 0;

\_homeWordCurrentPage = 1;

\_allDatasArr = @[].mutableCopy;

self.navigationItem.rightBarButtonItem = [UIBarButtonItem barButtonItemWithImage:@"general\_icon\_more" andBarButtonItemBlock:^{

[self showList];

}];

self.scrollView = [[UIScrollView alloc] initWithFrame:self.view.bounds];

self.scrollView.delegate = self;

self.scrollView.pagingEnabled = YES;

[self.view addSubview:self.scrollView];

if (self.paperCorrectComeIn) {

//试卷批改进入

self.scrollView.scrollEnabled = NO;

self.title = @"试卷批改";

self.navigationItem.rightBarButtonItem = nil;

}

[self reqeustAllData];

}

- (void)collectPaper

{

PaperBrowseCollectListView \*listView = [PaperBrowseCollectListView viewFromNIB];

listView.clickCommitBlock = ^(NSNumber \*inFavId) {

//点击提交收藏

[JXHttpTool POST:@"addFavorites" parameters:@{@"inPlatformId":KACCOUNTTOOLOUTUSER.PlatformNumber,@"inFavId":inFavId?:EMPTY\_STRING,@"inTestId":\_currentModel.TestId?:EMPTY\_STRING} success:^(id responseObject) {

\_currentModel.IsFavo = @1;

[UISlipUpTool disMiss];

SHOWSUCCESSAFTERONESECOND(responseObject[MSG]);

} failure:nil];

};

listView.clickCreatCollectBlock = ^{

//点击创建收藏夹

[UISlipUpTool disMiss];

HomeCollectCreateAndEdit \*collectionVC = kStoryboardVC(@"Home", @"HomeCollectCreateAndEditSBID");

collectionVC.isAddCollect = YES;

[self.navigationController pushViewController:collectionVC animated:YES];

};

[UISlipUpTool slipUpWithView:listView andSpeed:0.5];

}

- (void)cancelCollectPaper

{

NSLog(@"%@",\_currentModel);

[JXHttpTool POST:@"cancelFavorites" parameters:@{@"inPlatformId":KACCOUNTTOOLOUTUSER.PlatformNumber,@"inFavId":\_currentModel.FavId?:EMPTY\_STRING,@"inTestId":\_currentModel.TestId?:EMPTY\_STRING} success:^(id responseObject) {

SHOWSUCCESSAFTERONESECOND(responseObject[MSG]);

\_currentModel.IsFavo = @0;

} failure:nil];

}

- (void)showList {

NSArray \*images;

if (![\_currentModel.IsFavo isEqualToNumber:@0]) {

//显示取消收藏

images = @[@"取消收藏", @"我要纠错",@"分享"];

} else {

//显示收藏

images = @[@"收藏", @"我要纠错",@"分享"];

}

[FTPopOverMenu showFromSenderFrame:CGRectMake(self.view.frame.size.width - 40, 20, 40, 40)

withMenuArray:images

imageArray:nil

doneBlock:^(NSInteger selectedIndex) {

if (selectedIndex == 0) {

//收藏

if (![\_currentModel.IsFavo isEqualToNumber:@0]) {

//显示取消收藏

[self cancelCollectPaper];

} else {

//显示收藏

[self collectPaper];

}

} else if (selectedIndex == 1) {

//我要纠错

NSString \*inTestId;

if (self.inChapterId || self.inKnowledgeId) {

inTestId = [NSString stringWithFormat:@"%@",\_currentModel.Id?:EMPTY\_STRING];

} else {

inTestId = [NSString stringWithFormat:@"%@",\_currentModel.TestId?:EMPTY\_STRING];

}

PaperCorrectMistakeTVC \*mistakeVC = kStoryboardVC(@"Paper", @"PaperCorrectMistakeTVCSBID");

mistakeVC.inTestId = inTestId;

[self.navigationController pushViewController:mistakeVC animated:YES];

} else if (selectedIndex == 2) {

//分享

NSString \*url = [NSString stringWithFormat:@"%@%@?inId=%@",kBaseURL,@"/TestInfoShow",self.inId?:EMPTY\_STRING];

[SnShareTool showShareWithImage:@"logo" url:url andTitle:@"因卓教育平台题目"];

}

NSLog(@"done block. do something. selectedIndex : %ld", (long)selectedIndex);

} dismissBlock:^{

NSLog(@"user canceled. do nothing.");

}];

}

- (void)reqeustAllData

{

if (self.inFavId) {

//收藏列表进入

[self requestCollectData];

} else if (self.selectPaperShowModels) {

//选择试题进入

\_allDatasArr = @[].mutableCopy;

for (int j = 0; j < self.selectPaperShowModels.count; j ++) {

PaperBrowseModel \*model = self.selectPaperShowModels[j];

[\_allDatasArr addObject:model];;

[self refreshDataWith:model andCount:j];

}

\_currentModel = \_allDatasArr[\_currentPage];

} else if (self.paperBrowerListComeIn){

//点击试题列表 大圈圈 进入

if (self.numbers) {

SunOrderArray \*postOrderArray = [[SunOrderArray alloc] init];

for (int i = 0 ; i < self.numbers.count ; i ++) {

[JXHttpTool POST:@"testPaperTaskOne" parameters:@{@"inId":self.inId?:EMPTY\_STRING,@"incurrentPage":self.numbers[i]?:EMPTY\_STRING} success:^(id responseObject) {

if ([responseObject[@"outList"] count] == 0) {

return ;

}

[postOrderArray addArray:@[[PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"][0]]] WithIndex:i];

if (self.numbers.count == [postOrderArray allDatas].count) {

//便利完之后

\_allDatasArr = @[].mutableCopy;

for (int j = 0 ; j < [[postOrderArray allDatas] count]; j ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:[postOrderArray allDatas][j]];

[\_allDatasArr addObject:model];

[self refreshDataWith:model andCount:j];

}

\_currentPage = self.currentPageUI;

[self.scrollView scrollRectToVisible:CGRectMake(\_currentPage \* SCREEN\_WIDTH, 0, SCREEN\_WIDTH, SCREEN\_HEIGHT) animated:NO];

\_currentModel = \_allDatasArr[\_currentPage];

}

} failure:nil];

}

}

} else if (self.paperCorrectComeIn) {

//试卷批改进入

[JXHttpTool POST:@"selTestCorrecting" parameters:@{@"inAttributeId":self.inAttributeId?:EMPTY\_STRING,@"incurrentPage":@(1),@"inType":self.inType?:EMPTY\_STRING} success:^(id responseObject) {

if ([responseObject[@"Page"][@"TotalPage"] isEqualToString:@"0"]) {

//批改完 退出

// SHOWERRORAFTERONESECOND(responseObject[MSG]);

[self.navigationController popToRootViewControllerAnimated:YES];

return ;

}

if ([responseObject[@"outList"] count] == 0) {

return ;

}

//下一页的批改

[\_allDatasArr removeAllObjects];

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, SCREEN\_HEIGHT);

[self.scrollView removeAllSubviews];

if ([responseObject[@"Page"][@"TotalPage"] isEqualToString:@"1"]) {

\_isLastNum = YES;

} else {

\_isLastNum = NO;

}

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"]];

[\_allDatasArr addObject:model];

// if (![[NSString stringWithFormat:@"%@",responseObject[@"Page"][@"CurrentPage"]] isEqualToString:responseObject[@"Page"][@"TotalPage"] ]) {

// //不是最后一页的时候

// \_homeWordCurrentPage ++;

// [self reqeustAllData];

// return;

// }

for (int j = 0; j < \_allDatasArr.count; j ++) {

PaperBrowseModel \*model = \_allDatasArr[j];

[self refreshDataWith:model andCount:j];

}

\_currentModel = \_allDatasArr[\_currentPage];

}failure:nil];

} else if (self.studyHistoryComeIn){

//做学习历史 ,

[JXHttpTool POST:@"testPaperTaskOne" parameters:@{@"inId":self.inId?:EMPTY\_STRING,@"incurrentPage":@(\_homeWordCurrentPage)} success:^(id responseObject) {

if ([responseObject[@"outList"] count] == 0) {

return ;

}

for (int i = 0 ; i < [responseObject[@"outList"] count]; i ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"][i]];

[\_allDatasArr addObject:model];

}

if (![responseObject[@"Page"][@"CurrentPage"] isEqualToNumber:responseObject[@"Page"][@"TotalPage"] ]) {

//最后一页的时候

\_homeWordCurrentPage ++;

[self reqeustAllData];

return;

}

for (int j = 0; j < \_allDatasArr.count; j ++) {

PaperBrowseModel \*model = \_allDatasArr[j];

[self refreshDataWith:model andCount:j];

}

\_currentModel = \_allDatasArr[\_currentPage];

} failure:nil];

} else {

//选择范围进入

[self requestQuestionData];

}

}

- (void)requestCollectData

{

//试题列表

[JXHttpTool POST:@"getFaQestionList" parameters:@{@"inPlatformId":self.inPlatformId?:EMPTY\_STRING,@"inFavId":self.inFavId?:EMPTY\_STRING,@"inshowCount":@"5",@"incurrentPage":@"1"} success:^(id responseObject) {

SunOrderArray \*postOrderArray = [[SunOrderArray alloc] init];

for (int i = 0; i < [self.allNum integerValue]; i ++) {

[JXHttpTool POST:@"getFaQestionList" parameters:@{@"inPlatformId":self.inPlatformId?:EMPTY\_STRING,@"inFavId":self.inFavId?:EMPTY\_STRING,@"inshowCount":@"1",@"incurrentPage":[NSString stringWithFormat:@"%d",i + 1]} success:^(id responseObj) {

if ([responseObj[@"outList"] count] == 0) {

return ;

}

[postOrderArray addArray:@[[PaperBrowseModel mj\_objectWithKeyValues:responseObj[@"outList"][0]]] WithIndex:i];

if ([self.allNum integerValue] == [postOrderArray allDatas].count) {

//便利完之后

\_allDatasArr = @[].mutableCopy;

for (int j = 0; j < [postOrderArray allDatas].count; j ++) {

PaperBrowseModel \*model = [postOrderArray allDatas][j];

[\_allDatasArr addObject:model];;

[self refreshDataWith:model andCount:j];

}

\_currentModel = \_allDatasArr[\_currentPage];

}

} failure:nil];

}

} failure:nil];

}

- (void)requestQuestionData

{

NSDictionary \*dict;

if (self.inChapterId) {

//选择的章节点

dict = @{@"inChapterId":self.inChapterId?:EMPTY\_STRING,@"incurrentPage":@"1",@"inshowCount":@"5"};

} else if (self.inKnowledgeId) {

//选择的知识点

dict = @{@"inKnowledgeId":self.inKnowledgeId?:EMPTY\_STRING,@"incurrentPage":@"1",@"inshowCount":@"5"};

}

[JXHttpTool POST:@"getPageQuestionsList" parameters:dict success:^(id responseObject) {

if ([responseObject[@"outList"] count] == 0) {

return ;

}

\_allDatasArr = @[].mutableCopy;

for (int i = 0 ; i < [responseObject[@"outList"] count]; i ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"][i]];

[\_allDatasArr addObject:model];

[self refreshDataWith:model andCount:i];

}

\_currentModel = \_allDatasArr[\_currentPage];

} failure:nil];

}

- (void)refreshDataWith:(PaperBrowseModel \*)model andCount:(int)count

{

//10道题

self.scrollView.contentSize = CGSizeMake((count + 1) \* SCREEN\_WIDTH, SCREEN\_HEIGHT);

//循环配置每道题的界面

PaperObjectView \*objView = [[PaperObjectView alloc] initWithFrame:CGRectMake(count \* SCREEN\_WIDTH, 0, SCREEN\_WIDTH, SCREEN\_HEIGHT) andDatas:model allNum:0 currentNum:0 isCorrect:self.paperCorrectComeIn?YES:NO isLastNum:\_isLastNum];

objView.clickNextPage = ^{

[self nextPage];

};

objView.clickCorrectBlock = ^(NSString \*inProObj) {

[JXHttpTool POST:@"updateTestCorrecting" parameters:@{@"inAttributeId":[\_currentModel.Pros[0] AttributeId]?:EMPTY\_STRING,@"inClassId":[\_currentModel.Pros[0] ClassId]?:EMPTY\_STRING,@"inStudentId":[\_currentModel.Pros[0] StudentId]?:EMPTY\_STRING,@"inTestId":[\_currentModel TestId]?:EMPTY\_STRING,@"inProObj":inProObj?:EMPTY\_STRING} success:^(id responseObject) {

SHOWSUCCESSAFTERONESECOND(responseObject[MSG]);

[self nextPage];

} failure:nil];

};

[self.scrollView addSubview:objView];

}

- (void)nextPage

{

if (\_currentPage != \_allDatasArr.count - 1) {

//不是最后一页

\_currentPage ++;

[self.scrollView scrollRectToVisible:CGRectMake(\_currentPage \* SCREEN\_WIDTH, 0, SCREEN\_WIDTH, SCREEN\_HEIGHT) animated:YES];

\_currentPage = self.scrollView.contentOffset.x/SCREEN\_WIDTH;

\_currentModel = \_allDatasArr[\_currentPage];

} else {

//最后一页

// [self.navigationController popToRootViewControllerAnimated:YES];

[self reqeustAllData];

}

}

#pragma uiscrollview delegate

-(void)scrollViewDidScroll:(UIScrollView \*)scrollView

{

\_currentPage = self.scrollView.contentOffset.x/SCREEN\_WIDTH;

\_currentModel = \_allDatasArr[\_currentPage];

}

@end

//

// QuestionBankVC.m

// MaestroEdu

//

// Created by jesse on 2017/7/9.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "QuestionBankVC.h"

#import "PaperOrganizedView.h"

#import "FTPopOverMenu.h"

#import "QuestionBankSonTVC.h"

@interface QuestionBankVC ()<VTMagicViewDataSource,VTMagicViewDelegate>

{

NSMutableArray \*\_subjectArrs;

PaperOrganizedView \*\_oranizedView;

}

@end

@implementation QuestionBankVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

\_subjectArrs = @[].mutableCopy;

self.view.backgroundColor = OVERALLCOLOR;

if (!self.paperComeIn) {

self.navigationItem.leftBarButtonItem = [UIBarButtonItem barButtonItemWithTitle:@"题库" titleColor:[UIColor blackColor] andBarButtonItemBlock:^{

}];

} else {

//从试卷进入

self.title = self.showTitle?:EMPTY\_STRING;

}

self.magicView.navigationColor = [UIColor clearColor];

self.magicView.sliderColor = BORDERCOLOR;

self.magicView.sliderWidth = 60;

self.magicView.layoutStyle = VTLayoutStyleCenter;

self.magicView.switchStyle = VTSwitchStyleDefault;

self.magicView.navigationHeight = 40.f;

self.magicView.itemScale = 1.1;

self.magicView.dataSource = self;

self.magicView.delegate = self;

self.magicView.needPreloading = NO;

[self.magicView reloadData];

//配置 弹出菜单

//请求学科信息

[self requestSubjectData];

[DefaultCenter addObserver:self selector:@selector(requestSubjectData) name:kChangeFacultyNofity object:nil];

if (!self.paperComeIn) {

//配置右下角 组 view

PaperOrganizedView \*oranizedView = [PaperOrganizedView viewFromNIB];

[oranizedView.topBtn setBackgroundImage:IMG(@"btn\_Collection") forState:UIControlStateNormal];

\_oranizedView = oranizedView;

oranizedView.clickViewBlock = ^{

//点击收藏

kPushStoryboardVC(@"Home", @"HomeCollectTVCSBID");

};

oranizedView.center = CGPointMake(SCREEN\_WIDTH - 70, SCREEN\_HEIGHT - 64 - 49 - 80);

[self.view addSubview:oranizedView];

}

}

-(void)viewWillAppear:(BOOL)animated

{

[super viewWillAppear:animated];

[self reqeustNumData];

}

- (void)reqeustNumData

{

[JXHttpTool POST:@"allTestStatistics" parameters:@{} success:^(id responseObject) {

[\_oranizedView.oranizedNum setTitle:[NSString stringWithFormat:@"已收藏：%@题",responseObject[@"FavoCount"]?:EMPTY\_STRING] forState:UIControlStateNormal];

} failure:nil];

}

- (void)requestSubjectData

{

[JXHttpTool POST:@"subjectList" parameters:@{} success:^(id responseObject) {

NSMutableArray \*tempArr1 = [NSMutableArray arrayWithArray:responseObject[@"outList"]];

NSMutableArray \*tempArr = @[].mutableCopy;

for (NSDictionary \*dict in tempArr1) {

NSLog(@"%@",KACCOUNTTOOL.Faculty);

if ([dict[@"Faculty"] containsString:@"部"]) {

NSString \*temp = [dict[@"Faculty"] stringByReplacingOccurrencesOfString:@"部" withString:EMPTY\_STRING];

if ([temp isEqualToString:KACCOUNTTOOL.Faculty]) {

[tempArr addObject:dict];

}

}

}

\_subjectArrs = tempArr;

[self showSubjectList];

} failure:nil];

}

- (void)showSubjectList

{

NSMutableArray \*tempArr = @[].mutableCopy;

for (NSDictionary \*dict in \_subjectArrs) {

[tempArr addObject:dict[@"SubjectsName"]];

}

NSString \*showSubject;

//默认第一个

if (!KACCOUNTTOOL.subject) {

showSubject = tempArr[0];

KACCOUNTTOOL.subject = [NSString stringWithFormat:@"%@",\_subjectArrs[0][@"SubjectsId"]];

KACCOUNTTOOL.subjectImg = [NSString stringWithFormat:@"%@",\_subjectArrs[0][@"SubjectsName"]];

[KACCOUNTTOOL saveUserAccount];

[self postNotificationWith:\_subjectArrs[0]];

} else {

showSubject = KACCOUNTTOOL.subjectImg;

}

self.navigationItem.rightBarButtonItem = [UIBarButtonItem barButtonItemWithImage:showSubject andBarButtonItemBlock:^{

[FTPopOverMenu showFromSenderFrame:CGRectMake(self.view.frame.size.width - 40, 20, 40, 40)

withMenuArray:tempArr

imageArray:nil

doneBlock:^(NSInteger selectedIndex) {

UIButton \*btn = (UIButton \*)self.navigationItem.rightBarButtonItem.customView;

[btn setBackgroundImage:IMG(tempArr[selectedIndex]) forState:UIControlStateNormal];

[btn setBackgroundImage:IMG(tempArr[selectedIndex]) forState:UIControlStateHighlighted];

NSLog(@"done block. do something. selectedIndex : %ld", (long)selectedIndex);

//切换科目 刷新

KACCOUNTTOOL.subject = [NSString stringWithFormat:@"%@",\_subjectArrs[selectedIndex][@"SubjectsId"]];

KACCOUNTTOOL.subjectImg = [NSString stringWithFormat:@"%@",\_subjectArrs[selectedIndex][@"SubjectsName"]];

[KACCOUNTTOOL saveUserAccount];

[self postNotificationWith:\_subjectArrs[selectedIndex]];

} dismissBlock:^{

NSLog(@"user canceled. do nothing.");

}];

}];

[DefaultCenter addObserver:self selector:@selector(subjectRefresh) name:kChangeSubjectNofity object:nil];

}

-(void)subjectRefresh

{

//刷新logo

UIButton \*btn = self.navigationItem.rightBarButtonItem.customView;

[btn setBackgroundImage:IMG(KACCOUNTTOOL.subjectImg) forState:UIControlStateNormal];

}

- (void)postNotificationWith:(NSDictionary \*)dict

{

[DefaultCenter postNotificationName:kChangeSubjectNofity object:nil];

}

//- (void)requestBookVersionData

//{

// [JXHttpTool POST:@"bookversionList" parameters:@{@"inSubjectsId":\_currentSubject?:EMPTY\_STRING} success:^(id responseObject) {

//

// } failure:nil];

//}

- (NSArray<NSString \*> \*)menuTitlesForMagicView:(VTMagicView \*)magicView {

return @[@"按章节",@"按知识点"];

}

- (UIButton \*)magicView:(VTMagicView \*)magicView menuItemAtIndex:(NSUInteger)itemIndex {

static NSString \*itemIdentifier = @"QuestionBankSonTVCitemIdentifier";

UIButton \*menuItem = [magicView dequeueReusableItemWithIdentifier:itemIdentifier];

if (!menuItem) {

menuItem = [UIButton buttonWithType:UIButtonTypeCustom];

[menuItem setTitleColor:[UIColor colorWithHexString:@"#9a9a9a"] forState:UIControlStateNormal];

[menuItem setTitleColor:BORDERCOLOR forState:UIControlStateSelected];

menuItem.titleLabel.font = FONT(15);

}

return menuItem;

}

- (UIViewController \*)magicView:(VTMagicView \*)magicView viewControllerAtPage:(NSUInteger)pageIndex {

static NSString \*recomVCId = @"QuestionBankSonTVC.ID";

QuestionBankSonTVC \*questionBankSonTVC = [magicView dequeueReusablePageWithIdentifier:recomVCId];

if (!questionBankSonTVC) {

questionBankSonTVC = [[QuestionBankSonTVC alloc] init];

}

questionBankSonTVC.pageIndex = pageIndex;

questionBankSonTVC.paperComeIn = self.paperComeIn;

return questionBankSonTVC;

}

@end

//

// QuestionBankSonTVC.m

// MaestroEdu

//

// Created by 张明辉 on 2017/7/10.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "QuestionBankSonTVC.h"

#import "RATreeView.h"

#import "RADataObject.h"

#import "RATableViewCell.h"

#import "SunOrderArray.h"

#import "QuestionBankSonModel.h"

#import "PaperBrowseVC.h"

#import "QuestionSelectVC.h"

@interface QuestionBankSonTVC ()<RATreeViewDelegate, RATreeViewDataSource>

{

NSString \*\_subjectID;

NSString \*\_currentVersion;

}

@property (strong, nonatomic) NSMutableArray \*data;

@property (weak, nonatomic) RATreeView \*treeView;

@property (strong, nonatomic) UIBarButtonItem \*editButton;

@property (nonatomic,strong) NSMutableArray \*versionArrs;//教材版本

@property (nonatomic,strong) NSMutableArray \*bookArrs;//八年级上册

@property (nonatomic,strong) NSMutableArray \*chapterArrs;//第一章

@end

@implementation QuestionBankSonTVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

self.data = @[].mutableCopy;

self.versionArrs = @[].mutableCopy;

self.bookArrs = @[].mutableCopy;

self.chapterArrs = @[].mutableCopy;

// [self loadData];

RATreeView \*treeView = [[RATreeView alloc] initWithFrame:self.view.bounds];

\_\_weak \_\_typeof(&\*treeView)weakSelf\_treeView = treeView;

[treeView.scrollView addHeaderRefresh:^{

[self subjectRefresh];

[weakSelf\_treeView.scrollView endHeaderRefresh];

}];

treeView.delegate = self;

treeView.dataSource = self;

treeView.treeFooterView = [UIView new];

treeView.separatorStyle = RATreeViewCellSeparatorStyleNone;

[treeView setBackgroundColor:OVERALLCOLOR];

self.treeView = treeView;

self.treeView.frame = self.view.bounds;

self.treeView.autoresizingMask = UIViewAutoresizingFlexibleWidth | UIViewAutoresizingFlexibleHeight;

[self.view insertSubview:treeView atIndex:0];

[self.navigationController setNavigationBarHidden:NO];

self.navigationItem.title = NSLocalizedString(@"Things", nil);

[self.treeView registerNib:[UINib nibWithNibName:NSStringFromClass([RATableViewCell class]) bundle:nil] forCellReuseIdentifier:NSStringFromClass([RATableViewCell class])];

[DefaultCenter addObserver:self selector:@selector(subjectRefresh) name:kChangeSubjectNofity object:nil];

[self subjectRefresh];

}

- (void)subjectRefresh

{

//为了按知识点查询有数据

if (!KACCOUNTTOOL.subject || [KACCOUNTTOOL.subject isEqualToString:EMPTY\_STRING]) {

return;

}

[self.data removeAllObjects];

[self.treeView reloadData];

\_subjectID = KACCOUNTTOOL.subject?:EMPTY\_STRING;

[JXHttpTool POST:@"bookversionList" parameters:@{@"inSubjectsId":\_subjectID} success:^(id responseObject) {

self.versionArrs = [NSMutableArray arrayWithArray:responseObject[@"outVersionList"]];

//刷新教材

[self refreshVersionWithRow:0];

} failure:nil];

}

- (void)viewWillAppear:(BOOL)animated

{

[super viewWillAppear:animated];

[self subjectRefresh];

self.treeView.frame = self.view.bounds;

}

#pragma mark TreeView Delegate methods

- (CGFloat)treeView:(RATreeView \*)treeView heightForRowForItem:(id)item

{

return 44;

}

- (BOOL)treeView:(RATreeView \*)treeView canEditRowForItem:(id)item

{

return NO;

}

- (void)treeView:(RATreeView \*)treeView willExpandRowForItem:(id)item

{

RADataObject \*dataObject = item;

dataObject.isOpen = YES;

RATableViewCell \*cell = (RATableViewCell \*)[treeView cellForItem:item];

[cell setChangeImageWith:dataObject.type isOpen:YES animated:YES andChildrens:!(dataObject.children.count==0)];

[cell setAdditionButtonHidden:NO animated:YES];

}

- (void)treeView:(RATreeView \*)treeView willCollapseRowForItem:(id)item

{

RADataObject \*dataObject = item;

dataObject.isOpen = NO;

RATableViewCell \*cell = (RATableViewCell \*)[treeView cellForItem:item];

[cell setChangeImageWith:dataObject.type isOpen:NO animated:YES andChildrens:!(dataObject.children.count==0)];

[cell setAdditionButtonHidden:YES animated:YES];

}

-(void)treeView:(RATreeView \*)treeView didSelectRowForItem:(id)item

{

//如果点击的是选择教材

RADataObject \*dataObject = item;

if ([dataObject.name isEqualToString:\_currentVersion]) {

NSMutableArray \*temp = @[].mutableCopy;

for (NSDictionary \*dict in self.versionArrs) {

[temp addObject:dict[@"VersionName"]];

}

[JXShowSelectTool showSelectWithArr:temp andDefaultStr:temp[0] complete:^(NSString \*selectedStr, NSInteger num) {

[self refreshVersionWithRow:num];

}];

}

}

- (void)treeView:(RATreeView \*)treeView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowForItem:(id)item

{

if (editingStyle != UITableViewCellEditingStyleDelete) {

return;

}

RADataObject \*parent = [self.treeView parentForItem:item];

NSInteger index = 0;

if (parent == nil) {

index = [self.data indexOfObject:item];

NSMutableArray \*children = [self.data mutableCopy];

[children removeObject:item];

self.data = [children copy];

} else {

index = [parent.children indexOfObject:item];

[parent removeChild:item];

}

[self.treeView deleteItemsAtIndexes:[NSIndexSet indexSetWithIndex:index] inParent:parent withAnimation:RATreeViewRowAnimationRight];

if (parent) {

[self.treeView reloadRowsForItems:@[parent] withRowAnimation:RATreeViewRowAnimationNone];

}

}

#pragma mark TreeView Data Source

- (UITableViewCell \*)treeView:(RATreeView \*)treeView cellForItem:(id)item

{

RADataObject \*dataObject = item;

// if (dataObject.type == 2) {

//

// }

// [self.data indexOfObject:dataObject];

NSInteger level = [self.treeView levelForCellForItem:item];

// NSInteger numberOfChildren = [dataObject.children count];

// NSString \*detailText = [NSString localizedStringWithFormat:@"Number of children %@", [@(numberOfChildren) stringValue]];

// BOOL expanded = [self.treeView isCellForItemExpanded:item];

RATableViewCell \*cell = [self.treeView dequeueReusableCellWithIdentifier:NSStringFromClass([RATableViewCell class])];

[cell setupWithTitle:dataObject.name detailText:EMPTY\_STRING level:level additionButtonHidden:YES andType:dataObject.type andChildrens:!(dataObject.children.count==0) isOpen:dataObject.isOpen hidenLine:0];//!expanded];

cell.selectionStyle = UITableViewCellSelectionStyleNone;

cell.selectedQuestionTapAction = ^(id sender) {

if (dataObject.type == 0

|| dataObject.type == 1) {

return ;

}

if (!self.paperComeIn) {

PaperBrowseVC \*paperBrowseVC = [[PaperBrowseVC alloc] init];

if (self.pageIndex == 0) {

//选择的章节点

paperBrowseVC.inChapterId = dataObject.Id;

} else {

//选择的知识点

paperBrowseVC.inKnowledgeId = dataObject.Id;

}

[self.navigationController pushViewController:paperBrowseVC animated:YES];

} else {

//选择试题

QuestionSelectVC \*question = kStoryboardVC(@"QuestionBank", @"QuestionSelectVCSBID");

if (self.pageIndex == 0) {

//选择的章节点

question.inChapterId = dataObject.Id;

} else {

//选择的知识点

question.inKnowledgeId = dataObject.Id;

}

[self.navigationController pushViewController:question animated:YES];

}

};

return cell;

}

- (NSInteger)treeView:(RATreeView \*)treeView numberOfChildrenOfItem:(id)item

{

if (item == nil) {

return [self.data count];

}

RADataObject \*data = item;

return [data.children count];

}

- (id)treeView:(RATreeView \*)treeView child:(NSInteger)index ofItem:(id)item

{

RADataObject \*data = item;

if (item == nil) {

return [self.data objectAtIndex:index];

}

return data.children[index];

}

- (void)refreshVersionWithRow:(NSInteger)row

{

//刷新教材版本

\_currentVersion = self.versionArrs[row][@"VersionName"];

[self refreshVersionDetail:self.versionArrs[row][@"VersionId"]];

}

- (void)requestKonwledgeData

{

//请求章节

[JXHttpTool POST:@"konwledgeList" parameters:@{@"inSubjectsId":KACCOUNTTOOL.subject?:EMPTY\_STRING} success:^(id responseObject) {

//最后一个的话

NSMutableArray \*temp = @[].mutableCopy;

for (NSDictionary \*dict in responseObject[@"outkonwledgeList"][@"ChildList"]) {

[temp addObject:[QuestionBankSonModel mj\_objectWithKeyValues:dict]];

}

self.chapterArrs = temp;

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self loadData];

});

} failure:nil];

}

- (void)refreshVersionDetail:(NSString \*)inVersionId

{

//刷新 八年级上册

[JXHttpTool POST:@"textBookList" parameters:@{@"inSubjectsId":\_subjectID,@"inVersionId":inVersionId?:EMPTY\_STRING} success:^(id responseObject) {

self.bookArrs = [NSMutableArray arrayWithArray:responseObject[@"outBookList"]];

if (self.pageIndex == 0) {

//章节点

SunOrderArray \*postOrderArray = [[SunOrderArray alloc] init];

for (int i = 0; i < self.bookArrs.count ; i ++) {

//请求章节

[JXHttpTool POST:@"chapterList" parameters:@{@"inTextBookId":self.bookArrs[i][@"TextBookId"]?:EMPTY\_STRING} success:^(id responseObject) {

[postOrderArray addArray:@[responseObject[@"outChapterList"]] WithIndex:i];

if (self.bookArrs.count == [[postOrderArray allDatas] count]) {

//最后一个的话

NSMutableArray \*temp = @[].mutableCopy;

for (NSDictionary \*dict in [postOrderArray allDatas]) {

[temp addObject:[QuestionBankSonModel mj\_objectWithKeyValues:dict]];

}

self.chapterArrs = temp;

dispatch\_async(dispatch\_get\_main\_queue(), ^{

[self loadData];

});

}

} failure:nil];

}

} else {

//知识点

[self requestKonwledgeData];

}

} failure:nil];

}

#pragma mark - Helpers

- (void)getDataWithRADataObject:(RADataObject \*)raData andArr:(NSArray <QuestionBankSonModel \*> \*)arr andType:(int)type

{

NSMutableArray \*tempArr = @[].mutableCopy;

for (int i = 0; i < arr.count; i ++) {

QuestionBankSonModel \*currentModel = arr[i];

RADataObject \*phone = [RADataObject dataObjectWithName:currentModel.Name

children:nil andType:type andId:currentModel.Id];

[tempArr addObject:phone];

if ([arr[i] ChildList] && [[arr[i] ChildList] count] > 0) {

[self getDataWithRADataObject:phone andArr:[arr[i] ChildList] andType:type + 1];

}

}

if (raData) {

raData.children = tempArr;

} else {

[self.data addObjectsFromArray:tempArr];

}

}

- (void)loadData

{

//教材选择

RADataObject \*phone = [RADataObject dataObjectWithName:\_currentVersion

children:nil andType:1 andId:nil];

if (self.pageIndex == 0) {

//添加教材版本

self.data = [NSMutableArray arrayWithObjects:phone, nil];

[self getDataWithRADataObject:nil andArr:self.chapterArrs andType:1];

} else {

[self getDataWithRADataObject:nil andArr:self.chapterArrs andType:2];

}

[self.treeView reloadData];

}

@end

//

// QuestionSelectVC.m

// MaestroEdu

//

// Created by 张明辉 on 2017/7/14.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "QuestionSelectVC.h"

#import "FTPopOverMenu.h"

#import "QuestionSelectCell.h"

#import "PaperBrowseModel.h"

#import "QuestionAllSelectedVC.h"

#import "PaperBrowseVC.h"

@interface QuestionSelectVC ()<UITableViewDelegate,UITableViewDataSource>

{

NSString \*\_inTitleType;//题型

NSString \*\_inFacilityValue;//难易程度 难易度 asc,desc asc:难度由低到高 desc:难度由高到低

NSInteger \_allTempNum;

}

@property (weak, nonatomic) IBOutlet UIButton \*intelligentSort;

@property (weak, nonatomic) IBOutlet UIImageView \*intelligentSortImage;

@property (weak, nonatomic) IBOutlet UIButton \*allTypeSort;

@property (weak, nonatomic) IBOutlet UIImageView \*allTypeSortImage;

@property (weak, nonatomic) IBOutlet UITableView \*tableView;

@property (weak, nonatomic) IBOutlet UIButton \*underBtn;

@property (nonatomic,strong) NSMutableArray \*quTypeList;

@property (nonatomic,strong) NSMutableArray \*selectedDataList;

@end

@implementation QuestionSelectVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

self.tableView.delegate = self;

self.tableView.dataSource = self;

self.tableView.estimatedRowHeight = 300;

self.tableView.tableFooterView = [[UIView alloc] init];

self.quTypeList = @[].mutableCopy;

[self.quTypeList addObject:@"全部题型"];

self.selectedDataList = @[].mutableCopy;

\_inTitleType = EMPTY\_STRING;

\_inFacilityValue = EMPTY\_STRING;

[self questQuTypeList];

WEAKSELF

kTableViewFuckingInit(weakSelf\_SC.tableView, requestData);

[self requestData];

}

- (IBAction)clickBottom:(UIButton \*)sender {

QuestionAllSelectedVC \*allSelectedVC = kStoryboardVC(@"QuestionBank", @"QuestionAllSelectedVCSBID");

// allSelectedVC.selectedDataList = self.selectedDataList;

[self.navigationController pushViewController:allSelectedVC animated:YES];

}

- (void)questQuTypeList

{

[JXHttpTool POST:@"quTypeList" parameters:@{@"inSubjectName":KACCOUNTTOOL.subjectImg?:EMPTY\_STRING} success:^(id responseObject) {

[self.quTypeList addObjectsFromArray:responseObject[@"outList"]];

} failure:nil];

}

- (void)requestData

{

[self.selectedDataList removeAllObjects];

NSMutableDictionary \*dict;

if (self.inChapterId) {

//选择的章节点

dict = @{@"inChapterId":self.inChapterId?:EMPTY\_STRING,@"incurrentPage":self.tableView.CurrentPage,@"inshowCount":@(INT\_MAX),@"inTitleType":\_inTitleType?:EMPTY\_STRING,@"inFacilityValue":\_inFacilityValue?:EMPTY\_STRING}.mutableCopy;

} else if (self.inKnowledgeId) {

//选择的知识点

dict = @{@"inKnowledgeId":self.inKnowledgeId?:EMPTY\_STRING,@"incurrentPage":self.tableView.CurrentPage,@"inshowCount":@(INT\_MAX),@"inTitleType":\_inTitleType?:EMPTY\_STRING,@"inFacilityValue":\_inFacilityValue?:EMPTY\_STRING}.mutableCopy;

}

[JXHttpTool POST:@"getPageQuestionsList" parameters:dict success:^(id responseObject) {

NSMutableDictionary \*tempDict = dict.mutableCopy;

[tempDict setValue:EMPTY\_STRING forKey:@"inTitleType"];

[JXHttpTool POST:@"getPageQuestionsList" parameters:tempDict success:^(id response) {

NSMutableArray \*temp = @[].mutableCopy;

for (int i = 0 ; i < [responseObject[@"outList"] count]; i ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"][i]];

if ([model.IsCart isEqualToNumber:@1]) {

//已选择加入试题

[self.selectedDataList addObject:model];

}

[temp addObject:model];

}

NSMutableArray \*tempArr = @[].mutableCopy;

for (int i = 0 ; i < [response[@"outList"] count]; i ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:response[@"outList"][i]];

if ([model.IsCart isEqualToNumber:@1]) {

//已选择加入试题

[tempArr addObject:model];

}

}

\_allTempNum = tempArr.count;

[self.underBtn setTitle:[NSString stringWithFormat:@"已选%lu题",(unsigned long)\_allTempNum] forState:UIControlStateNormal];

kTableViewAfterDataAction(self.tableView,responseObject,temp);

}failure:nil];

} failure:nil tableView:self.tableView];

}

- (void)showIntelligentListWith:(UIButton \*)sender

{

NSArray \*images = @[@"智能排序",@"难度由高到低", @"难度由低到高"];

[FTPopOverMenu showForSender:sender

withMenuArray:images

imageArray:nil

doneBlock:^(NSInteger selectedIndex) {

[self.intelligentSort setTitle:images[selectedIndex] forState:UIControlStateNormal];

if (selectedIndex == 1) {

\_inFacilityValue = @"desc";

} else if (selectedIndex == 2) {

\_inFacilityValue = @"asc";

} else {

\_inFacilityValue = EMPTY\_STRING;

}

kTableViewHeaderAction(self.tableView);

[self requestData];

NSLog(@"done block. do something. selectedIndex : %ld", (long)selectedIndex);

} dismissBlock:^{

NSLog(@"user canceled. do nothing.");

}];

}

- (void)showAllTypeListWith:(UIButton \*)sender

{

[FTPopOverMenu showForSender:sender

withMenuArray:self.quTypeList

imageArray:nil

doneBlock:^(NSInteger selectedIndex) {

[self.allTypeSort setTitle:self.quTypeList[selectedIndex] forState:UIControlStateNormal];

if (selectedIndex != 0) {

\_inTitleType = self.quTypeList[selectedIndex];

} else {

\_inTitleType = EMPTY\_STRING;

}

kTableViewHeaderAction(self.tableView);

[self requestData];

NSLog(@"done block. do something. selectedIndex : %ld", (long)selectedIndex);

} dismissBlock:^{

NSLog(@"user canceled. do nothing.");

}];

}

- (IBAction)clickIntelligentBtn:(UIButton \*)sender {

[self showIntelligentListWith:sender];

[self.intelligentSort setTitleColor:BORDERCOLOR forState:UIControlStateNormal];

[self.allTypeSort setTitleColor:[UIColor darkGrayColor] forState:UIControlStateNormal];

self.allTypeSortImage.image = IMG(@"drop\_down\_arrow\_gray");

self.intelligentSortImage.image = IMG(@"drop\_down\_arrow");

}

- (IBAction)clickAllTypeBtn:(UIButton \*)sender {

[self showAllTypeListWith:sender];

[self.allTypeSort setTitleColor:BORDERCOLOR forState:UIControlStateNormal];

[self.intelligentSort setTitleColor:[UIColor darkGrayColor] forState:UIControlStateNormal];

self.intelligentSortImage.image = IMG(@"drop\_down\_arrow\_gray");

self.allTypeSortImage.image = IMG(@"drop\_down\_arrow");

}

-(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

{

return self.tableView.tableViewDatas.count;

}

- (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

{

QuestionSelectCell \*cell = [QuestionSelectCell cellWithNibTableView:tableView andReuseID:@"QuestionSelectCellSBID"];

cell.isSelected = NO;

\_\_weak \_\_typeof(&\*cell)weakSelf\_cell = cell;

cell.chooseThisCellBlock = ^{

[self clickSelectBtnWith:indexPath.row andCell:weakSelf\_cell];

};

[cell addDataWith:self.tableView.tableViewDatas[indexPath.row]];

return cell;

}

- (void)clickSelectBtnWith:(NSInteger)row andCell:(QuestionSelectCell \*)cell

{

PaperBrowseModel \*model = self.tableView.tableViewDatas[row];

if ([self.selectedDataList containsObject:self.tableView.tableViewDatas[row]]) {

[JXHttpTool POST:@"delCart" parameters:@{@"inCartId":model.CartId?:EMPTY\_STRING} success:^(id responseObject) {

cell.selectedBtn.selected = !cell.selectedBtn.selected;

[self.selectedDataList removeObject:self.tableView.tableViewDatas[row]];

\_allTempNum --;

[self.underBtn setTitle:[NSString stringWithFormat:@"已选%lu题",(unsigned long)\_allTempNum] forState:UIControlStateNormal];

} failure:nil];

} else {

[JXHttpTool POST:@"addCart" parameters:@{@"inTestId":model.Id?:EMPTY\_STRING,@"inTitleType":model.TitleType?:EMPTY\_STRING} success:^(id responseObject) {

cell.selectedBtn.selected = !cell.selectedBtn.selected;

model.CartId = responseObject[@"CardId"];

[self.selectedDataList addObject:self.tableView.tableViewDatas[row]];

\_allTempNum ++;

[self.underBtn setTitle:[NSString stringWithFormat:@"已选%lu题",(unsigned long)\_allTempNum] forState:UIControlStateNormal];

} failure:nil];

}

}

-(void)tableView:(UITableView \*)tableView didSelectRowAtIndexPath:(NSIndexPath \*)indexPath

{

[tableView clearSelectedRowsAnimated:YES];

PaperBrowseVC \*paperBrowseVC = [[PaperBrowseVC alloc] init];

paperBrowseVC.selectPaperShowModels = @[self.tableView.tableViewDatas[indexPath.row]];

[self.navigationController pushViewController:paperBrowseVC animated:YES];

}

@end

//

// QuestionAllSelectedVC.m

// MaestroEdu

//

// Created by 张明辉 on 2017/7/17.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "QuestionAllSelectedVC.h"

#import "QuestionSelectCell.h"

#import "PaperBrowseModel.h"

#import "PaperBrowseVC.h"

@interface QuestionAllSelectedVC ()<UITableViewDataSource,UITableViewDelegate>

@property (weak, nonatomic) IBOutlet UITableView \*tableView;

@property (weak, nonatomic) IBOutlet UIView \*bottomView;

@end

@implementation QuestionAllSelectedVC

- (void)viewDidLoad {

[super viewDidLoad];

// Do any additional setup after loading the view.

self.tableView.delegate = self;

self.tableView.dataSource = self;

self.tableView.estimatedRowHeight = 300;

WEAKSELF

kTableViewFuckingInit(weakSelf\_SC.tableView,requestData);

[self requestData];

}

- (void)requestData

{

[JXHttpTool POST:@"selCartList" parameters:@{@"inSchoolId":KACCOUNTTOOLOUTUSER.CurrentRole,@"incurrentPage":self.tableView.CurrentPage?:@"1"} success:^(id responseObject) {

NSMutableArray \*temp = @[].mutableCopy;

for (int i = 0 ; i < [responseObject[@"outList"] count]; i ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"][i]];

[temp addObject:model];

}

kTableViewAfterDataAction(self.tableView,responseObject,temp);

} failure:nil tableView:self.tableView];

}

- (IBAction)releaseAfter:(UIButton \*)sender {

[self.navigationController popToRootViewControllerAnimated:NO];

[ROOTTABBARVC setSelectedIndex:2];

}

- (IBAction)release:(UIButton \*)sender {

kPushBottomBtnController(@"Paper", @"PaperAttributeTVCSBID", @"下一步", nil);

}

-(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

{

return self.tableView.tableViewDatas.count;

}

- (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

{

QuestionSelectCell \*cell = [QuestionSelectCell cellWithNibTableView:tableView andReuseID:@"QuestionSelectCellSBID"];

cell.isSelected = YES;

cell.chooseThisCellBlock = ^{

//删除选择的试题

[JXHttpTool POST:@"delCart" parameters:@{@"inCartId":[self.tableView.tableViewDatas[indexPath.row] CartId]?:EMPTY\_STRING} success:^(id responseObject) {

SHOWSUCCESSAFTERONESECOND(responseObject[MSG]);

kTableViewHeaderAction(self.tableView);

[self requestData];

} failure:nil];

};

[cell addDataWith:self.tableView.tableViewDatas[indexPath.row]];

return cell;

}

-(void)tableView:(UITableView \*)tableView didSelectRowAtIndexPath:(NSIndexPath \*)indexPath

{

[JXHttpTool POST:@"selCartShow" parameters:@{@"inSchoolId":KACCOUNTTOOLOUTUSER.CurrentRole,@"incurrentPage":[NSString stringWithFormat:@"%ld",(long)indexPath.row + 1],@"inshowCount":@"1"} success:^(id responseObject) {

NSMutableArray \*temp = @[].mutableCopy;

for (int i = 0 ; i < [responseObject[@"outList"] count]; i ++) {

PaperBrowseModel \*model = [PaperBrowseModel mj\_objectWithKeyValues:responseObject[@"outList"][i]];

[temp addObject:model];

}

PaperBrowseVC \*paperBrowseVC = [[PaperBrowseVC alloc] init];

paperBrowseVC.selectPaperShowModels = temp;

[self.navigationController pushViewController:paperBrowseVC animated:YES];

} failure:nil tableView:self.tableView];

}

@end

//

// UIImage+vImage.m

//

// Copyright (c) 2012 Shuichi Tsutsumi. All rights reserved.

//

#import "UIImage+vImage.h"

#import <Accelerate/Accelerate.h>

#import "UIImage+Blend.h"

static int16\_t gaussianblur\_kernel[25] = {

1, 4, 6, 4, 1,

4, 16, 24, 16, 4,

6, 24, 36, 24, 6,

4, 16, 24, 16, 4,

1, 4, 6, 4, 1

};

static int16\_t edgedetect\_kernel[9] = {

-1, -1, -1,

-1, 8, -1,

-1, -1, -1

};

static int16\_t emboss\_kernel[9] = {

-2, 0, 0,

0, 1, 0,

0, 0, 2

};

static int16\_t sharpen\_kernel[9] = {

-1, -1, -1,

-1, 9, -1,

-1, -1, -1

};

static int16\_t unsharpen\_kernel[9] = {

-1, -1, -1,

-1, 17, -1,

-1, -1, -1

};

static uint8\_t backgroundColorBlack[4] = {0,0,0,0};

static unsigned char morphological\_kernel[9] = {

1, 1, 1,

1, 1, 1,

1, 1, 1,

};

//static unsigned char morphological\_kernel[25] = {

// 0, 1, 1, 1, 0,

// 1, 1, 1, 1, 1,

// 1, 1, 1, 1, 1,

// 1, 1, 1, 1, 1,

// 0, 1, 1, 1, 0,

//};

@implementation UIImage (vImage)

- (UIImage \*)gaussianBlur

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageConvolve\_ARGB8888(&src, &dest, NULL, 0, 0, gaussianblur\_kernel, 5, 5, 256, NULL, kvImageCopyInPlace);

memcpy(data, outt, n);

free(outt);

CGImageRef blurredImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* blurred = [UIImage imageWithCGImage:blurredImageRef];

CGImageRelease(blurredImageRef);

CGContextRelease(bmContext);

return blurred;

}

- (UIImage \*)edgeDetection

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageConvolve\_ARGB8888(&src, &dest, NULL, 0, 0, edgedetect\_kernel, 3, 3, 1, backgroundColorBlack, kvImageCopyInPlace);

memcpy(data, outt, n);

CGImageRef edgedImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* edged = [UIImage imageWithCGImage:edgedImageRef];

CGImageRelease(edgedImageRef);

free(outt);

CGContextRelease(bmContext);

return edged;

}

- (UIImage \*)emboss

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageConvolve\_ARGB8888(&src, &dest, NULL, 0, 0, emboss\_kernel, 3, 3, 1, NULL, kvImageCopyInPlace);

memcpy(data, outt, n);

free(outt);

CGImageRef embossImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* emboss = [UIImage imageWithCGImage:embossImageRef];

CGImageRelease(embossImageRef);

CGContextRelease(bmContext);

return emboss;

}

- (UIImage \*)sharpen

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageConvolve\_ARGB8888(&src, &dest, NULL, 0, 0, sharpen\_kernel, 3, 3, 1, NULL, kvImageCopyInPlace);

memcpy(data, outt, n);

free(outt);

CGImageRef sharpenedImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* sharpened = [UIImage imageWithCGImage:sharpenedImageRef];

CGImageRelease(sharpenedImageRef);

CGContextRelease(bmContext);

return sharpened;

}

- (UIImage \*)unsharpen

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageConvolve\_ARGB8888(&src, &dest, NULL, 0, 0, unsharpen\_kernel, 3, 3, 9, NULL, kvImageCopyInPlace);

memcpy(data, outt, n);

free(outt);

CGImageRef unsharpenedImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* unsharpened = [UIImage imageWithCGImage:unsharpenedImageRef];

CGImageRelease(unsharpenedImageRef);

CGContextRelease(bmContext);

return unsharpened;

}

- (UIImage \*)rotateInRadians:(float)radians

{

if (!(&vImageRotate\_ARGB8888))

return nil;

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {data, height, width, bytesPerRow};

Pixel\_8888 bgColor = {0, 0, 0, 0};

vImageRotate\_ARGB8888(&src, &dest, NULL, radians, bgColor, kvImageBackgroundColorFill);

CGImageRef rotatedImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* rotated = [UIImage imageWithCGImage:rotatedImageRef];

CGImageRelease(rotatedImageRef);

CGContextRelease(bmContext);

return rotated;

}

- (UIImage \*)dilate

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageDilate\_ARGB8888(&src, &dest, 0, 0, morphological\_kernel, 3, 3, kvImageCopyInPlace);

memcpy(data, outt, n);

free(outt);

CGImageRef dilatedImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* dilated = [UIImage imageWithCGImage:dilatedImageRef];

CGImageRelease(dilatedImageRef);

CGContextRelease(bmContext);

return dilated;

}

- (UIImage \*)erode

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

const size\_t n = sizeof(UInt8) \* width \* height \* 4;

void\* outt = malloc(n);

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {outt, height, width, bytesPerRow};

vImageErode\_ARGB8888(&src, &dest, 0, 0, morphological\_kernel, 3, 3, kvImageCopyInPlace);

memcpy(data, outt, n);

free(outt);

CGImageRef erodedImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* eroded = [UIImage imageWithCGImage:erodedImageRef];

CGImageRelease(erodedImageRef);

CGContextRelease(bmContext);

return eroded;

}

- (UIImage \*)dilateWithIterations:(int)iterations {

UIImage \*dstImage = self;

for (int i=0; i<iterations; i++) {

dstImage = [dstImage dilate];

}

return dstImage;

}

- (UIImage \*)erodeWithIterations:(int)iterations {

UIImage \*dstImage = self;

for (int i=0; i<iterations; i++) {

dstImage = [dstImage erode];

}

return dstImage;

}

- (UIImage \*)gradientWithIterations:(int)iterations {

UIImage \*dilated = [self dilateWithIterations:iterations];

UIImage \*eroded = [self erodeWithIterations:iterations];

UIImage \*dstImage = [dilated imageBlendedWithImage:eroded blendMode:kCGBlendModeDifference alpha:1.0];

return dstImage;

}

- (UIImage \*)tophatWithIterations:(int)iterations {

UIImage \*dilated = [self dilateWithIterations:iterations];

UIImage \*dstImage = [self imageBlendedWithImage:dilated blendMode:kCGBlendModeDifference alpha:1.0];

return dstImage;

}

- (UIImage \*)blackhatWithIterations:(int)iterations {

UIImage \*eroded = [self erodeWithIterations:iterations];

UIImage \*dstImage = [eroded imageBlendedWithImage:self blendMode:kCGBlendModeDifference alpha:1.0];

return dstImage;

}

- (UIImage \*)equalization

{

const size\_t width = self.size.width;

const size\_t height = self.size.height;

const size\_t bytesPerRow = width \* 4;

CGColorSpaceRef space = CGColorSpaceCreateDeviceRGB();

CGContextRef bmContext = CGBitmapContextCreate(NULL, width, height, 8, bytesPerRow, space, kCGBitmapByteOrderDefault | kCGImageAlphaPremultipliedFirst);

CGColorSpaceRelease(space);

if (!bmContext)

return nil;

CGContextDrawImage(bmContext, (CGRect){.origin.x = 0.0f, .origin.y = 0.0f, .size.width = width, .size.height = height}, self.CGImage);

UInt8\* data = (UInt8\*)CGBitmapContextGetData(bmContext);

if (!data)

{

CGContextRelease(bmContext);

return nil;

}

vImage\_Buffer src = {data, height, width, bytesPerRow};

vImage\_Buffer dest = {data, height, width, bytesPerRow};

vImageEqualization\_ARGB8888(&src, &dest, kvImageNoFlags);

CGImageRef destImageRef = CGBitmapContextCreateImage(bmContext);

UIImage\* destImage = [UIImage imageWithCGImage:destImageRef];

CGImageRelease(destImageRef);

CGContextRelease(bmContext);

return destImage;

}

@end

#import "UIImage+ImageEffects.h"

@import Accelerate;

#import <float.h>

@implementation UIImage (ImageEffects)

- (UIImage \*)applyLightEffect

{

UIColor \*tintColor = [UIColor colorWithWhite:1.0 alpha:0.3];

return [self applyBlurWithRadius:30 tintColor:tintColor saturationDeltaFactor:1.8 maskImage:nil];

}

- (UIImage \*)applyExtraLightEffect

{

UIColor \*tintColor = [UIColor colorWithWhite:0.97 alpha:0.82];

return [self applyBlurWithRadius:20 tintColor:tintColor saturationDeltaFactor:1.8 maskImage:nil];

}

- (UIImage \*)applyDarkEffect

{

UIColor \*tintColor = [UIColor colorWithWhite:0.11 alpha:0];

return [self applyBlurWithRadius:20 tintColor:tintColor saturationDeltaFactor:1.8 maskImage:nil];

}

- (UIImage \*)applyTintEffectWithColor:(UIColor \*)tintColor

{

const CGFloat EffectColorAlpha = 0.6;

UIColor \*effectColor = tintColor;

NSInteger componentCount = CGColorGetNumberOfComponents(tintColor.CGColor);

if (componentCount == 2) {

CGFloat b;

if ([tintColor getWhite:&b alpha:NULL]) {

effectColor = [UIColor colorWithWhite:b alpha:EffectColorAlpha];

}

}

else {

CGFloat r, g, b;

if ([tintColor getRed:&r green:&g blue:&b alpha:NULL]) {

effectColor = [UIColor colorWithRed:r green:g blue:b alpha:EffectColorAlpha];

}

}

return [self applyBlurWithRadius:10 tintColor:effectColor saturationDeltaFactor:-1.0 maskImage:nil];

}

- (UIImage \*)applyBlurWithRadius:(CGFloat)blurRadius tintColor:(UIColor \*)tintColor saturationDeltaFactor:(CGFloat)saturationDeltaFactor maskImage:(UIImage \*)maskImage

{

// Check pre-conditions.

if (self.size.width < 1 || self.size.height < 1) {

NSLog (@"\*\*\* error: invalid size: (%.2f x %.2f). Both dimensions must be >= 1: %@", self.size.width, self.size.height, self);

return nil;

}

if (!self.CGImage) {

NSLog (@"\*\*\* error: image must be backed by a CGImage: %@", self);

return nil;

}

if (maskImage && !maskImage.CGImage) {

NSLog (@"\*\*\* error: maskImage must be backed by a CGImage: %@", maskImage);

return nil;

}

CGRect imageRect = { CGPointZero, self.size };

UIImage \*effectImage = self;

BOOL hasBlur = blurRadius > \_\_FLT\_EPSILON\_\_;

BOOL hasSaturationChange = fabs(saturationDeltaFactor - 1.) > \_\_FLT\_EPSILON\_\_;

if (hasBlur || hasSaturationChange) {

UIGraphicsBeginImageContextWithOptions(self.size, NO, [[UIScreen mainScreen] scale]);

CGContextRef effectInContext = UIGraphicsGetCurrentContext();

CGContextScaleCTM(effectInContext, 1.0, -1.0);

CGContextTranslateCTM(effectInContext, 0, -self.size.height);

CGContextDrawImage(effectInContext, imageRect, self.CGImage);

vImage\_Buffer effectInBuffer;

effectInBuffer.data = CGBitmapContextGetData(effectInContext);

effectInBuffer.width = CGBitmapContextGetWidth(effectInContext);

effectInBuffer.height = CGBitmapContextGetHeight(effectInContext);

effectInBuffer.rowBytes = CGBitmapContextGetBytesPerRow(effectInContext);

UIGraphicsBeginImageContextWithOptions(self.size, NO, [[UIScreen mainScreen] scale]);

CGContextRef effectOutContext = UIGraphicsGetCurrentContext();

vImage\_Buffer effectOutBuffer;

effectOutBuffer.data = CGBitmapContextGetData(effectOutContext);

effectOutBuffer.width = CGBitmapContextGetWidth(effectOutContext);

effectOutBuffer.height = CGBitmapContextGetHeight(effectOutContext);

effectOutBuffer.rowBytes = CGBitmapContextGetBytesPerRow(effectOutContext);

if (hasBlur) {

CGFloat inputRadius = blurRadius \* [[UIScreen mainScreen] scale];

uint32\_t radius = floor(inputRadius \* 3. \* sqrt(2 \* M\_PI) / 4 + 0.5);

if (radius % 2 != 1) {

radius += 1; // force radius to be odd so that the three box-blur methodology works.

}

vImageBoxConvolve\_ARGB8888(&effectInBuffer, &effectOutBuffer, NULL, 0, 0, radius, radius, 0, kvImageEdgeExtend);

vImageBoxConvolve\_ARGB8888(&effectOutBuffer, &effectInBuffer, NULL, 0, 0, radius, radius, 0, kvImageEdgeExtend);

vImageBoxConvolve\_ARGB8888(&effectInBuffer, &effectOutBuffer, NULL, 0, 0, radius, radius, 0, kvImageEdgeExtend);

}

BOOL effectImageBuffersAreSwapped = NO;

if (hasSaturationChange) {

CGFloat s = saturationDeltaFactor;

CGFloat floatingPointSaturationMatrix[] = {

0.0722 + 0.9278 \* s, 0.0722 - 0.0722 \* s, 0.0722 - 0.0722 \* s, 0,

0.7152 - 0.7152 \* s, 0.7152 + 0.2848 \* s, 0.7152 - 0.7152 \* s, 0,

0.2126 - 0.2126 \* s, 0.2126 - 0.2126 \* s, 0.2126 + 0.7873 \* s, 0,

0, 0, 0, 1,

};

const int32\_t divisor = 256;

NSUInteger matrixSize = sizeof(floatingPointSaturationMatrix)/sizeof(floatingPointSaturationMatrix[0]);

int16\_t saturationMatrix[matrixSize];

for (NSUInteger i = 0; i < matrixSize; ++i) {

saturationMatrix[i] = (int16\_t)roundf(floatingPointSaturationMatrix[i] \* divisor);

}

if (hasBlur) {

vImageMatrixMultiply\_ARGB8888(&effectOutBuffer, &effectInBuffer, saturationMatrix, divisor, NULL, NULL, kvImageNoFlags);

effectImageBuffersAreSwapped = YES;

}

else {

vImageMatrixMultiply\_ARGB8888(&effectInBuffer, &effectOutBuffer, saturationMatrix, divisor, NULL, NULL, kvImageNoFlags);

}

}

if (!effectImageBuffersAreSwapped)

effectImage = UIGraphicsGetImageFromCurrentImageContext();

UIGraphicsEndImageContext();

if (effectImageBuffersAreSwapped)

effectImage = UIGraphicsGetImageFromCurrentImageContext();

UIGraphicsEndImageContext();

}

// Set up output context.

UIGraphicsBeginImageContextWithOptions(self.size, NO, [[UIScreen mainScreen] scale]);

CGContextRef outputContext = UIGraphicsGetCurrentContext();

CGContextScaleCTM(outputContext, 1.0, -1.0);

CGContextTranslateCTM(outputContext, 0, -self.size.height);

// Draw base image.

CGContextDrawImage(outputContext, imageRect, self.CGImage);

// Draw effect image.

if (hasBlur) {

CGContextSaveGState(outputContext);

if (maskImage) {

CGContextClipToMask(outputContext, imageRect, maskImage.CGImage);

}

CGContextDrawImage(outputContext, imageRect, effectImage.CGImage);

CGContextRestoreGState(outputContext);

}

// Add in color tint.

if (tintColor) {

CGContextSaveGState(outputContext);

CGContextSetFillColorWithColor(outputContext, tintColor.CGColor);

CGContextFillRect(outputContext, imageRect);

CGContextRestoreGState(outputContext);

}

// Output image is ready.

UIImage \*outputImage = UIGraphicsGetImageFromCurrentImageContext();

UIGraphicsEndImageContext();

return outputImage;

}

@end

//

// THDatePickerView.m

// rongyp-company

//

// Created by Apple on 2016/11/16.

// Copyright © 2016年 Apple. All rights reserved.

//

#import "THDatePickerView.h"

@interface THDatePickerView () <UIPickerViewDelegate, UIPickerViewDataSource>

@property (strong, nonatomic) UIPickerView \*pickerView; // 选择器

@property (strong, nonatomic) UIView \*toolView; // 工具条

@property (strong, nonatomic) UILabel \*titleLbl; // 标题

@property (strong, nonatomic) NSMutableArray \*dataArray; // 数据源

@property (copy, nonatomic) NSString \*selectStr; // 选中的时间

@property (strong, nonatomic) NSMutableArray \*yearArr; // 年数组

@property (strong, nonatomic) NSMutableArray \*monthArr; // 月数组

@property (strong, nonatomic) NSMutableArray \*dayArr; // 日数组

@property (strong, nonatomic) NSMutableArray \*hourArr; // 时数组

@property (strong, nonatomic) NSMutableArray \*minuteArr; // 分数组

@property (strong, nonatomic) NSArray \*timeArr; // 当前时间数组

@property (copy, nonatomic) NSString \*year; // 选中年

@property (copy, nonatomic) NSString \*month; //选中月

@property (copy, nonatomic) NSString \*day; //选中日

@property (copy, nonatomic) NSString \*hour; //选中时

@property (copy, nonatomic) NSString \*minute; //选中分

@end

#define THColorRGB(rgb) [UIColor colorWithRed:(rgb)/255.0 green:(rgb)/255.0 blue:(rgb)/255.0 alpha:1.0]

@implementation THDatePickerView

#pragma mark - init

/// 初始化

- (instancetype)initWithFrame:(CGRect)frame

{

self = [super initWithFrame:frame];

if (self) {

self.backgroundColor = [UIColor whiteColor];

self.dataArray = [NSMutableArray array];

[self.dataArray addObject:self.yearArr];

[self.dataArray addObject:self.monthArr];

[self.dataArray addObject:self.dayArr];

[self.dataArray addObject:self.hourArr];

[self.dataArray addObject:self.minuteArr];

[self configToolView];

[self configPickerView];

}

return self;

}

#pragma mark - 配置界面

/// 配置工具条

- (void)configToolView {

self.toolView = [[UIView alloc] init];

self.toolView.frame = CGRectMake(0, 0, self.frame.size.width, 44);

[self addSubview:self.toolView];

UIButton \*saveBtn = [[UIButton alloc] init];

saveBtn.frame = CGRectMake(self.frame.size.width - 50, 2, 40, 40);

[saveBtn setImage:[UIImage imageNamed:@"icon\_select1"] forState:UIControlStateNormal];

[saveBtn addTarget:self action:@selector(saveBtnClick) forControlEvents:UIControlEventTouchUpInside];

[self.toolView addSubview:saveBtn];

UIButton \*cancelBtn = [[UIButton alloc] init];

cancelBtn.frame = CGRectMake(10, 2, 40, 40);

[cancelBtn setImage:[UIImage imageNamed:@"icon\_revocation1"] forState:UIControlStateNormal];

[cancelBtn addTarget:self action:@selector(cancelBtnClick) forControlEvents:UIControlEventTouchUpInside];

[self.toolView addSubview:cancelBtn];

self.titleLbl = [[UILabel alloc] init];

self.titleLbl.frame = CGRectMake(60, 2, self.frame.size.width - 120, 40);

self.titleLbl.textAlignment = NSTextAlignmentCenter;

self.titleLbl.textColor = THColorRGB(34);

[self.toolView addSubview:self.titleLbl];

}

/// 配置UIPickerView

- (void)configPickerView {

self.pickerView = [[UIPickerView alloc] initWithFrame:CGRectMake(0, CGRectGetMaxY(self.toolView.frame), self.frame.size.width, self.frame.size.height - 44)];

self.pickerView.backgroundColor = [UIColor whiteColor];

self.pickerView.dataSource = self;

self.pickerView.delegate = self;

self.pickerView.showsSelectionIndicator = YES;

[self addSubview:self.pickerView];

}

- (void)setTitle:(NSString \*)title {

\_title = title;

self.titleLbl.text = title;

}

- (void)show {

self.year = self.timeArr[0];

self.month = [NSString stringWithFormat:@"%ld月", [self.timeArr[1] integerValue]];

self.day = [NSString stringWithFormat:@"%ld日", [self.timeArr[2] integerValue]];

self.hour = [NSString stringWithFormat:@"%ld时", [self.timeArr[3] integerValue]];

self.minute = self.minuteArr[self.minuteArr.count / 2];

[self.pickerView selectRow:[self.yearArr indexOfObject:self.year] inComponent:0 animated:YES];

/// 重新格式化转一下，是因为如果是09月/日/时，数据源是9月/日/时,就会出现崩溃

[self.pickerView selectRow:[self.monthArr indexOfObject:self.month] inComponent:1 animated:YES];

[self.pickerView selectRow:[self.dayArr indexOfObject:self.day] inComponent:2 animated:YES];

[self.pickerView selectRow:[self.hourArr indexOfObject:self.hour] inComponent:3 animated:YES];

[self.pickerView selectRow:self.minuteArr.count / 2 inComponent:4 animated:YES];

/// 刷新日

[self refreshDay];

}

#pragma mark - 点击方法

/// 保存按钮点击方法

- (void)saveBtnClick {

NSLog(@"点击了保存");

NSString \*month = self.month.length == 3 ? [NSString stringWithFormat:@"%ld", self.month.integerValue] : [NSString stringWithFormat:@"0%ld", self.month.integerValue];

NSString \*day = self.day.length == 3 ? [NSString stringWithFormat:@"%ld", self.day.integerValue] : [NSString stringWithFormat:@"0%ld", self.day.integerValue];

NSString \*hour = self.hour.length == 3 ? [NSString stringWithFormat:@"%ld", self.hour.integerValue] : [NSString stringWithFormat:@"0%ld", self.hour.integerValue];

NSString \*minute = self.minute.length == 3 ? [NSString stringWithFormat:@"%ld", self.minute.integerValue] : [NSString stringWithFormat:@"0%ld", self.minute.integerValue];

self.selectStr = [NSString stringWithFormat:@"%ld-%@-%@ %@:%@", [self.year integerValue], month, day, hour, minute];

if ([self.delegate respondsToSelector:@selector(datePickerViewSaveBtnClickDelegate:)]) {

[self.delegate datePickerViewSaveBtnClickDelegate:self.selectStr];

}

}

/// 取消按钮点击方法

- (void)cancelBtnClick {

NSLog(@"点击了取消");

if ([self.delegate respondsToSelector:@selector(datePickerViewCancelBtnClickDelegate)]) {

[self.delegate datePickerViewCancelBtnClickDelegate];

}

}

#pragma mark - UIPickerViewDelegate and UIPickerViewDataSource

/// UIPickerView返回多少组

- (NSInteger)numberOfComponentsInPickerView:(UIPickerView \*)pickerView{

return self.dataArray.count;

}

/// UIPickerView返回每组多少条数据

- (NSInteger)pickerView:(UIPickerView \*)pickerView numberOfRowsInComponent:(NSInteger)component{

return [self.dataArray[component] count] \* 200;

}

/// UIPickerView选择哪一行

- (void)pickerView:(UIPickerView \*)pickerView didSelectRow:(NSInteger)row inComponent:(NSInteger)component {

NSInteger time\_integerValue = [self.timeArr[component] integerValue];

switch (component) {

case 0: { // 年

NSString \*year\_integerValue = self.yearArr[row%[self.dataArray[component] count]];

if (year\_integerValue.integerValue < time\_integerValue) {

[pickerView selectRow:[self.dataArray[component] indexOfObject:self.timeArr[component]] inComponent:component animated:YES];

} else {

self.year = year\_integerValue;

/// 刷新日

[self refreshDay];

/// 根据当前选择的年份和月份获取当月的天数

NSString \*dayStr = [self getDayNumber:[self.year integerValue] month:[self.month integerValue]];

if (self.dayArr.count > [dayStr integerValue]) {

if (self.day.integerValue > [dayStr integerValue]) {

[pickerView selectRow:[self.dataArray[2] indexOfObject:[dayStr stringByAppendingString:@"日"]] inComponent:2 animated:YES];

self.day = [dayStr stringByAppendingString:@"日"];

}

}

}

} break;

case 1: { // 月

NSString \*month\_value = self.monthArr[row%[self.dataArray[component] count]];

// 如果选择年大于当前年 就直接赋值月

if ([self.year integerValue] > [self.timeArr[0] integerValue]) {

self.month = month\_value;

/// 刷新日

[self refreshDay];

/// 根据当前选择的年份和月份获取当月的天数

NSString \*dayStr = [self getDayNumber:[self.year integerValue] month:[self.month integerValue]];

if (self.dayArr.count > [dayStr integerValue]) {

if (self.day.integerValue > [dayStr integerValue]) {

[pickerView selectRow:[self.dataArray[2] indexOfObject:[dayStr stringByAppendingString:@"日"]] inComponent:2 animated:YES];

self.day = [dayStr stringByAppendingString:@"日"];

}

}

// 如果选择的年等于当前年，就判断月份

} else if ([self.year integerValue] == [self.timeArr[0] integerValue]) {

// 如果选择的月份小于当前月份 就刷新到当前月份

if (month\_value.integerValue < [self.timeArr[component] integerValue]) {

[pickerView selectRow:[self.dataArray[component] indexOfObject:[NSString stringWithFormat:@"%ld月", [self.timeArr[component] integerValue]]] inComponent:component animated:YES];

// 如果选择的月份大于当前月份，就直接赋值月份

} else {

self.month = month\_value;

/// 刷新日

[self refreshDay];

/// 根据当前选择的年份和月份获取当月的天数

NSString \*dayStr = [self getDayNumber:[self.year integerValue] month:[self.month integerValue]];

if (self.dayArr.count > dayStr.integerValue) {

if (self.day.integerValue > dayStr.integerValue) {

[pickerView selectRow:[self.dataArray[2] indexOfObject:[dayStr stringByAppendingString:@"日"]] inComponent:2 animated:YES];

self.day = [dayStr stringByAppendingString:@"日"];

}

}

}

}

} break;

case 2: { // 日

/// 根据当前选择的年份和月份获取当月的天数

NSString \*dayStr = [self getDayNumber:[self.year integerValue] month:[self.month integerValue]];

// 如果选择年大于当前年 就直接赋值日

NSString \*day\_value = self.dayArr[row%[self.dataArray[component] count]];

NSLog(@"%ld", self.dayArr.count);

if ([self.year integerValue] > [self.timeArr[0] integerValue]) {

if (self.dayArr.count <= [dayStr integerValue]) {

self.day = day\_value;

} else {

if (day\_value.integerValue <= [dayStr integerValue]) {

self.day = day\_value;

} else {

[pickerView selectRow:[self.dataArray[component] indexOfObject:[dayStr stringByAppendingString:@"日"]] inComponent:component animated:YES];

}

}

// 如果选择的年等于当前年，就判断月份

} else if ([self.year integerValue] == [self.timeArr[0] integerValue]) {

// 如果选择的月份大于当前月份 就直接复制

if ([self.month integerValue] > [self.timeArr[1] integerValue]) {

if (self.dayArr.count <= [dayStr integerValue]) {

self.day = day\_value;

} else {

if (day\_value.integerValue <= [dayStr integerValue]) {

self.day = day\_value;

} else {

[pickerView selectRow:[self.dataArray[component] indexOfObject:[dayStr stringByAppendingString:@"日"]] inComponent:component animated:YES];

}

}

// 如果选择的月份等于当前月份，就判断日

} else if ([self.month integerValue] == [self.timeArr[1] integerValue]) {

// 如果选择的日小于当前日，就刷新到当前日

if (day\_value.integerValue < [self.timeArr[component] integerValue]) {

[pickerView selectRow:[self.dataArray[component] indexOfObject:[NSString stringWithFormat:@"%ld日", time\_integerValue]] inComponent:component animated:YES];

// 如果选择的日大于当前日，就复制日

} else {

if (self.dayArr.count <= [dayStr integerValue]) {

self.day = self.dayArr[row%[self.dataArray[component] count]];

} else {

if ([self.dayArr[row%[self.dataArray[component] count]] integerValue] <= [dayStr integerValue]) {

self.day = self.dayArr[row%[self.dataArray[component] count]];

} else {

[pickerView selectRow:[self.dataArray[component] indexOfObject:[dayStr stringByAppendingString:@"日"]] inComponent:component animated:YES];

}

}

}

}

}

} break;

case 3: { // 时

// 如果选择年大于当前年 就直接赋值时

if ([self.year integerValue] > [self.timeArr[0] integerValue]) {

self.hour = self.hourArr[row%[self.dataArray[component] count]];

// 如果选择的年等于当前年，就判断月份

} else if ([self.year integerValue] == [self.timeArr[0] integerValue]) {

// 如果选择的月份大于当前月份 就直接复制时

if ([self.month integerValue] > [self.timeArr[1] integerValue]) {

self.hour = self.hourArr[row%[self.dataArray[component] count]];

// 如果选择的月份等于当前月份，就判断日

} else if ([self.month integerValue] == [self.timeArr[1] integerValue]) {

// 如果选择的日大于当前日，就直接复制时

if ([self.day integerValue] > [self.timeArr[2] integerValue]) {

self.hour = self.hourArr[row%[self.dataArray[component] count]];

// 如果选择的日等于当前日，就判断时

} else if ([self.day integerValue] == [self.timeArr[2] integerValue]) {

// 如果选择的时小于当前时，就刷新到当前时

if ([self.hourArr[row%[self.dataArray[component] count]] integerValue] < [self.timeArr[3] integerValue]) {

[pickerView selectRow:[self.dataArray[component] indexOfObject:self.timeArr[component]] inComponent:component animated:YES];

// 如果选择的时大于当前时，就直接赋值

} else {

self.hour = self.hourArr[row%[self.dataArray[component] count]];

}

}

}

}

} break;

case 4: { // 分

// 如果选择年大于当前年 就直接赋值时

// if ([self.year integerValue] > [self.timeArr[0] integerValue]) {

self.minute = self.minuteArr[row%[self.dataArray[component] count]];

// // 如果选择的年等于当前年，就判断月份

// } else if ([self.year integerValue] == [self.timeArr[0] integerValue]) {

// // 如果选择的月份大于当前月份 就直接复制时

// if ([self.month integerValue] > [self.timeArr[1] integerValue]) {

// self.minute = self.minuteArr[row%[self.dataArray[component] count]];

// // 如果选择的月份等于当前月份，就判断日

// } else if ([self.month integerValue] == [self.timeArr[1] integerValue]) {

// // 如果选择的日大于当前日，就直接复制时

// if ([self.day integerValue] > [self.timeArr[2] integerValue]) {

// self.minute = self.minuteArr[row%[self.dataArray[component] count]];

// // 如果选择的日等于当前日，就判断时

// } else if ([self.day integerValue] == [self.timeArr[2] integerValue]) {

// // 如果选择的时大于当前时，就直接赋值

// if ([self.hour integerValue] > [self.timeArr[3] integerValue]) {

// self.minute = self.minuteArr[row%[self.dataArray[component] count]];

// // 如果选择的时等于当前时,就判断分

// } else if ([self.hour integerValue] == [self.timeArr[3] integerValue]) {

// // 如果选择的分小于当前分，就刷新分

// if ([self.minuteArr[row%[self.dataArray[component] count]] integerValue] < [self.timeArr[4] integerValue]) {

// [pickerView selectRow:[self.dataArray[component] indexOfObject:self.timeArr[component]] inComponent:component animated:YES];

// // 如果选择分大于当前分，就直接赋值

// } else {

// self.minute = self.minuteArr[row%[self.dataArray[component] count]];

// }

// }

// }

// }

// }

} break;

default: break;

}

}

/// UIPickerView返回每一行数据

- (nullable NSString \*)pickerView:(UIPickerView \*)pickerView titleForRow:(NSInteger)row forComponent:(NSInteger)component{

return [self.dataArray[component] objectAtIndex:row%[self.dataArray[component] count]];

}

/// UIPickerView返回每一行的高度

- (CGFloat)pickerView:(UIPickerView \*)pickerView rowHeightForComponent:(NSInteger)component {

return 44;

}

/// UIPickerView返回每一行的View

-(UIView \*)pickerView:(UIPickerView \*)pickerView viewForRow:(NSInteger)row forComponent:(NSInteger)component reusingView:(UIView \*)view {

NSLog(@"%@", view);

UILabel \*titleLbl = [[UILabel alloc] initWithFrame:CGRectMake(0.0, 0.0, self.frame.size.width, 44)];

titleLbl.font = [UIFont systemFontOfSize:15];

titleLbl.textAlignment = NSTextAlignmentCenter;

titleLbl.text = [self.dataArray[component] objectAtIndex:row%[self.dataArray[component] count]];

return titleLbl;

}

- (void)pickerViewLoaded:(NSInteger)component row:(NSInteger)row{

NSUInteger max = 16384;

NSUInteger base10 = (max/2)-(max/2)%row;

[self.pickerView selectRow:[self.pickerView selectedRowInComponent:component] % row + base10 inComponent:component animated:NO];

}

/// 获取年份

- (NSMutableArray \*)yearArr {

if (!\_yearArr) {

\_yearArr = [NSMutableArray array];

for (int i = 1970; i < 2099; i ++) {

[\_yearArr addObject:[NSString stringWithFormat:@"%d年", i]];

}

}

return \_yearArr;

}

/// 获取月份

- (NSMutableArray \*)monthArr {

// NSDate \*today = [NSDate date];

// NSCalendar \*c = [NSCalendar currentCalendar];

// NSRange days = [c rangeOfUnit:NSCalendarUnitMonth inUnit:NSCalendarUnitYear forDate:today];

if (!\_monthArr) {

\_monthArr = [NSMutableArray array];

for (int i = 1; i <= 12; i ++) {

[\_monthArr addObject:[NSString stringWithFormat:@"%d月", i]];

}

}

return \_monthArr;

}

/// 获取当前月的天数

- (NSMutableArray \*)dayArr {

if (!\_dayArr) {

\_dayArr = [NSMutableArray array];

for (int i = 1; i <= 31; i ++) {

[\_dayArr addObject:[NSString stringWithFormat:@"%d日", i]];

}

}

return \_dayArr;

}

/// 获取小时

- (NSMutableArray \*)hourArr {

if (!\_hourArr) {

\_hourArr = [NSMutableArray array];

for (int i = 0; i < 24; i ++) {

[\_hourArr addObject:[NSString stringWithFormat:@"%d时", i]];

}

}

return \_hourArr;

}

/// 获取分钟

- (NSMutableArray \*)minuteArr {

if (!\_minuteArr) {

\_minuteArr = [NSMutableArray array];

for (int i = 0; i <= 55; i ++) {

if (i % 5 == 0) {

[\_minuteArr addObject:[NSString stringWithFormat:@"%d分", i]];

continue;

}

}

}

return \_minuteArr;

}

// 获取当前的年月日时

- (NSArray \*)timeArr {

if (!\_timeArr) {

\_timeArr = [NSArray array];

NSDateFormatter \*formatter = [[NSDateFormatter alloc] init];

[formatter setDateFormat:@"yyyy年,MM月,dd日,HH时,mm分"];

NSDate \*date = [NSDate date];

NSString \*time = [formatter stringFromDate:date];

\_timeArr = [time componentsSeparatedByString:@","];

}

return \_timeArr;

}

// 比较选择的时间是否小于当前时间

- (int)compareDate:(NSString \*)date01 withDate:(NSString \*)date02{

int ci;

NSDateFormatter \*df = [[NSDateFormatter alloc]init];

[df setDateFormat:@"yyyy年,MM月,dd日,HH时,mm分"];

NSDate \*dt1 = [[NSDate alloc] init];

NSDate \*dt2 = [[NSDate alloc] init];

dt1 = [df dateFromString:date01];

dt2 = [df dateFromString:date02];

NSComparisonResult result = [dt1 compare:dt2];

switch (result) {

//date02比date01大

case NSOrderedAscending: ci=1;break;

//date02比date01小

case NSOrderedDescending: ci=-1;break;

//date02=date01

case NSOrderedSame: ci=0;break;

default: NSLog(@"erorr dates %@, %@", dt2, dt1);break;

}

return ci;

}

- (void)refreshDay {

NSMutableArray \*arr = [NSMutableArray array];

for (int i = 1; i < [self getDayNumber:self.year.integerValue month:self.month.integerValue].integerValue + 1; i ++) {

[arr addObject:[NSString stringWithFormat:@"%d日", i]];

}

[self.dataArray replaceObjectAtIndex:2 withObject:arr];

[self.pickerView reloadComponent:2];

}

- (NSString \*)getDayNumber:(NSInteger)year month:(NSInteger)month{

NSArray \*days = @[@"31", @"28", @"31", @"30", @"31", @"30", @"31", @"31", @"30", @"31", @"30", @"31"];

if (2 == month && 0 == (year % 4) && (0 != (year % 100) || 0 == (year % 400))) {

return @"29";

}

return days[month - 1];

}

//

// PaperObjectView.m

// MaestroEdu

//

// Created by jesse on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperObjectView.h"

#import "PaperObjectTrunkView.h"

#import "NSString+size.h"

#import "PaperChooseObjectView.h"

#import "PaperAnswerAnalysisView.h"

#import "PaperCorrectAnswerView.h"

#import "PaperBrowseModel.h"

#import "PaperBrowseProblemModel.h"

#import "PaperBrowseQuestionsModel.h"

#import "PaperBrowseOptionModel.h"

//解答题

#import "PaperAnswerPhotoView.h"

//填空题

#import "PaperInsertTypeView.h"

//带切换的

#import "PaperSwitchPageView.h"

#import "PaperAnswerPhotoShowView.h"

#import "PaperAnswerAndRemarksView.h"

@interface PaperObjectView ()

{

//上下间隙

CGFloat gap;

//1.题干

PaperObjectTrunkView \*\_trunkView;

//2.选项

UIView \*\_chooseBgView;

//3.解析 加分割线

PaperAnswerAnalysisView \*\_paperAnswerView;

//4.答案

PaperCorrectAnswerView \*\_correctView;

int \_currentSwitchPage;

int \_allNumPage;

UILabel \*\_stuAnswerLabel;

UIButton \*\_stuAnswerBtn;

BOOL \_isLastNum;

}

@property (nonatomic,strong) UIScrollView \*scrollView;

@end

@implementation PaperObjectView

-(instancetype)initWithFrame:(CGRect)frame andDatas:(id)model allNum:(int)allNum currentNum:(int)currentNum isCorrect:(BOOL)isCorrect isLastNum:(BOOL)isLastNum

{

if (self = [super initWithFrame:frame]) {

gap = 20;

\_currentSwitchPage = currentNum;

\_allNumPage = allNum;

\_isLastNum = isLastNum;

self.scrollView = [[UIScrollView alloc] initWithFrame:self.bounds];

[self addSubview:self.scrollView];

/\*\*

单选题、多选题、填空题、判断题、解答题

1.单选、多选、选择题 （一种UI）

2.填空题 （一个填空 一种UI 多个小问 下边带切换UI）

3.简答题（拍照上传）（一种UI）

\*/

//直接显示

if ([model isKindOfClass:[PaperBrowseModel class]]) {

PaperBrowseModel \*tempModel = model;

if ([tempModel.IsHide isEqualToNumber:@0]) {

//不显示小问

if ([[tempModel.outProList[0] QuType] isEqualToString:@"单选题"]

|| [[tempModel.outProList[0] QuType] isEqualToString:@"多选题"]

|| [[tempModel.outProList[0] QuType] isEqualToString:@"判断题"]) {

//单选、多选、选择题 （一种UI）

[self setupOneChooseUIWithData:tempModel andAllNum:allNum];

} else if ([[tempModel.outProList[0] QuType] isEqualToString:@"解答题"]) {

//解答题（拍照上传）（一种UI）

[self setupAnswerUIWithData:tempModel andAllNum:allNum];

} else if ([[tempModel.outProList[0] QuType] isEqualToString:@"填空题"]) {

//填空题 卓师不允许输入（一种UI）

[self setupinsertTypeUIWithData:tempModel andAllNum:allNum];

}

if (isCorrect) {

//解答题的话 批改的话

[self setupAnswerUIWithData:tempModel andAllNum:allNum];

}

} else {

//显示多个小问带切换

[self setupSwitchUIWith:tempModel];

}

} else if ([model isKindOfClass:[PaperBrowseOptionModel class]]) {

//在小问里显示

PaperBrowseOptionModel \*tempModel = model;

if ([[tempModel QuType] isEqualToString:@"单选题"]

|| [[tempModel QuType] isEqualToString:@"多选题"]

|| [[tempModel QuType] isEqualToString:@"判断题"]) {

//单选、多选、选择题 （一种UI）

[self setupOneChooseUIWithData:tempModel andAllNum:allNum];

} else if ([[tempModel QuType] isEqualToString:@"解答题"]) {

//解答题（拍照上传）（一种UI）

[self setupAnswerUIWithData:tempModel andAllNum:allNum];

} else if ([[tempModel QuType] isEqualToString:@"填空题"]) {

//填空题 卓师不允许输入（一种UI）

[self setupinsertTypeUIWithData:tempModel andAllNum:allNum];

}

//批改试卷

if (tempModel.StudentAnswer && ![tempModel.StudentAnswer isEqualToString:EMPTY\_STRING]) {

//批改UI

[self setupAnswerUIWithData:tempModel andAllNum:allNum];

}

}

}

return self;

}

//allNum 为了展示第几个小问

-(instancetype)initWithFrame:(CGRect)frame andDatas:(id)model allNum:(int)allNum{

return [self initWithFrame:frame andDatas:model allNum:allNum currentNum:0 isCorrect:NO isLastNum:NO];

}

-(instancetype)initWithFrame:(CGRect)frame andDatas:(id)model {

return [self initWithFrame:frame andDatas:model allNum:0];

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*带多个小问\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- (void)setupSwitchUIWith:(PaperBrowseModel \*)model

{

//显示多个小问带切换

PaperSwitchPageView \*pageView = [PaperSwitchPageView viewFromNIB];

pageView.frame = self.scrollView.bounds;

pageView.isLastNum = \_isLastNum;

pageView.clickNextPage = ^{

!self.clickNextPage ?: self.clickNextPage();

};

[pageView setupWithData:model];

pageView.heightBlock = ^(CGFloat height) {

};

[self.scrollView addSubview:pageView];

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*填空题\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- (void)setupinsertTypeUIWithData:(id)model andAllNum:(int)allNum

{

NSString \*showTitle;

if ([model isKindOfClass:[PaperBrowseModel class]]) {

//直接显示的话

PaperBrowseModel \*tempModel = model;

showTitle = tempModel.Title;

} else if ([model isKindOfClass:[PaperBrowseOptionModel class]]) {

//显示小问的话

PaperBrowseOptionModel \*tempModel = model;

showTitle = [NSString stringWithFormat:@"%@/%d",tempModel.Sort,allNum];

}

//题干

PaperInsertTypeView \*insertTypeView = [PaperInsertTypeView viewFromNIB];

insertTypeView.frame = CGRectMake(0, 0, SCREEN\_WIDTH, SCREEN\_HEIGHT \* 0.5);

[insertTypeView addDataWith:showTitle];

[self.scrollView addSubview:insertTypeView];

//答案解析

[self refreshAnwserAndAnalysisWith:model andRect:insertTypeView.frame isChoose:NO isPaperCorrect:NO];

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*解答题\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- (void)setupAnswerUIWithData:(id)model andAllNum:(int)allNum

{

NSString \*showTitle;

NSString \*showType;

NSString \*answerUrl;

NSString \*StudentAnswer;

if ([model isKindOfClass:[PaperBrowseModel class]]) {

//直接显示的话

PaperBrowseModel \*tempModel = model;

showTitle = tempModel.Title;

showType = tempModel.TitleType;

answerUrl = [tempModel.outProList[0] Answer];

if (tempModel.Pros && tempModel.Pros.count != 0) {

StudentAnswer = [tempModel.Pros[0] StudentAnswer];

}

} else if ([model isKindOfClass:[PaperBrowseOptionModel class]]) {

//显示小问的话

PaperBrowseOptionModel \*tempModel = model;

showTitle = tempModel.QuTitle;

showType = [NSString stringWithFormat:@"%@/%d",(\_currentSwitchPage != 0)?[NSString stringWithFormat:@"%d",\_currentSwitchPage]:tempModel.Sort,allNum];

answerUrl = [tempModel Answer];

if (tempModel.StudentAnswer && ![tempModel.StudentAnswer isEqualToString:EMPTY\_STRING]) {

StudentAnswer = [tempModel StudentAnswer];

}

}

//题干数据

NSString \*trunkStr = showTitle;

NSAttributedString \* attrStr = [[NSAttributedString alloc] initWithData:[trunkStr dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

//1.题干

PaperObjectTrunkView \*trunkView = [PaperObjectTrunkView viewFromNIB];

//类型

trunkView.showTitle.font = FONT(15);

trunkView.showTitle.text = showType;

//题干

trunkView.trunkLabel.attributedText = attrStr;

CGFloat trunkViewHeight = [attrStr boundingRectWithSize:CGSizeMake(SCREEN\_WIDTH - 20, CGFLOAT\_MAX) options:NSStringDrawingUsesLineFragmentOrigin context:nil].size.height;//针对富文本有专门的计算动态高度的方法，与nsstring 大同小异；

[self.scrollView addSubview:trunkView];

\_trunkView = trunkView;

[self refreshTrunkChooseAndAnswerWith:trunkView and:trunkViewHeight isCorrect:YES];

//如果是试卷批改的话

UIButton \*btn;

UILabel \*label;

if (StudentAnswer && ![StudentAnswer isEqualToString:EMPTY\_STRING]) {

//显示答案照片

label = [[UILabel alloc] initWithFrame:CGRectMake(20, CGRectGetMaxY(\_trunkView.frame) + 20, SCREEN\_WIDTH, 25)];

label.text = @"学生答案";

label.font = FONT(15);

\_stuAnswerLabel = label;

[self.scrollView addSubview:label];

btn = [UIButton buttonWithType:UIButtonTypeCustom];

[btn sd\_setBackgroundImageWithURL:URLWITHSTRING(StudentAnswer) forState:UIControlStateNormal];

btn.frame = CGRectMake(20, CGRectGetMaxY(label.frame) + 10, SCREEN\_WIDTH - 40, (SCREEN\_WIDTH - 40) \* 0.57);

\_stuAnswerBtn = btn;

[self.scrollView addSubview:btn];

// NSAttributedString \* attrStrShowPhoto = [[NSAttributedString alloc] initWithData:[StudentAnswer dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

//

//

// PaperAnswerPhotoShowView \*photoView = [PaperAnswerPhotoShowView viewFromNIB];

// photoView.origin = CGPointMake(30, CGRectGetMaxY(\_trunkView.frame) + gap);

// photoView.showLabel.attributedText = attrStrShowPhoto;

// [self.scrollView addSubview:photoView];

//

//

// [self refreshTrunkChooseAndAnswerWith:trunkView and:layout];

//

// photoView.origin = CGPointMake(30, CGRectGetMaxY(\_trunkView.frame) + gap);

//

// photoView.showLabel.size = CGSizeMake(showPhotoLayout.textBoundingSize.width, showPhotoLayout.textBoundingSize.height + 50) ;

// photoView.height = photoView.showLabel.size.height;

}

//

////////////////

// \_trunkView.frame = CGRectMake(10,10, SCREEN\_WIDTH - 20, trunkViewHeight);

if (btn) {

label.frame = CGRectMake(20, CGRectGetMaxY(\_trunkView.frame) + 20, SCREEN\_WIDTH, 25);

btn.frame = CGRectMake(20, CGRectGetMaxY(label.frame) + 10, SCREEN\_WIDTH - 40, (SCREEN\_WIDTH - 40) \* 0.57);

}

[self refreshAnwserAndAnalysisWith:model andRect:btn?btn.frame:\_trunkView.frame isChoose:NO isPaperCorrect:btn?YES:NO];

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*单选\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

- (void)refreshAllViewsOriginY

{

//刷新选择ABCD

\_chooseBgView.origin = CGPointMake(0, CGRectGetMaxY(\_trunkView.frame) + gap);

//刷新解析

\_paperAnswerView.origin = CGPointMake(0, CGRectGetMaxY(\_chooseBgView.frame));

//刷新答案

\_correctView.origin = CGPointMake(0, CGRectGetMaxY(\_paperAnswerView.frame));

//刷新滚动区域

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, CGRectGetMaxY(\_correctView.frame) + 64);

!self.getNewHeightBlock ?: self.getNewHeightBlock();

}

- (void)setupOneChooseUIWithData:(id)model andAllNum:(int)allNum

{

NSString \*showTitle;

NSString \*showType;

NSArray \*chooseArrs;

BOOL isJudgePaper;

NSString \*answer;

if ([model isKindOfClass:[PaperBrowseModel class]]) {

//直接显示的话

PaperBrowseModel \*tempModel = model;

showTitle = tempModel.Title;

showType = tempModel.TitleType;

chooseArrs = [tempModel.outProList[0] outOptList];

if ([[tempModel.outProList[0] QuType] isEqualToString:@"判断题"]) {

isJudgePaper = YES;

} else {

isJudgePaper = NO;

}

answer = [tempModel.outProList[0] Answer];

} else {

//显示小问的话

PaperBrowseOptionModel \*tempModel = model;

showTitle = tempModel.QuTitle;

showType = [NSString stringWithFormat:@"%@/%d",tempModel.Sort,allNum];

chooseArrs = [tempModel outOptList];

if ([tempModel.QuType isEqualToString:@"判断题"]) {

isJudgePaper = YES;

} else {

isJudgePaper = NO;

}

answer = [tempModel Answer];

}

//题干数据

NSString \*trunkStr = showTitle;//@"1.一个正方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积1.一个正方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？1.一个正方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？1.一个正方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？方形，边长是5米，求面积？？是5米，求面积？方形，边长是5米，求面积？？cacacacacacaacacacacacacac";

NSAttributedString \* attrStr = [[NSAttributedString alloc] initWithData:[trunkStr dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

//1.题干

PaperObjectTrunkView \*trunkView = [PaperObjectTrunkView viewFromNIB];

//类型

trunkView.showTitle.text = showType;

//题干

trunkView.trunkLabel.attributedText = attrStr;

CGFloat trunkViewHeight = [attrStr boundingRectWithSize:CGSizeMake(SCREEN\_WIDTH - 60, CGFLOAT\_MAX) options:NSStringDrawingUsesLineFragmentOrigin context:nil].size.height;//针对富文本有专门的计算动态高度的方法，与nsstring 大同小异；

[self.scrollView addSubview:trunkView];

\_trunkView = trunkView;

//2.选项

UIView \*chooseBgView = [[UIView alloc] initWithFrame:CGRectMake(0, CGRectGetMaxY(trunkView.frame) + gap, SCREEN\_WIDTH, 900)];

chooseBgView.backgroundColor = [UIColor clearColor];

\_chooseBgView = chooseBgView;

[self.scrollView addSubview:chooseBgView];

#warning 暂定4个选项 为什么+30 我也不造 BUG

CGFloat chooseViewHeight = 0;

if (isJudgePaper) {

//判断题

NSArray \*arr = @[@"对",@"错"];

for (int i = 0; i < 2; i ++) {

PaperChooseObjectView \*chooseView = [PaperChooseObjectView viewFromNIB];

NSAttributedString \* tempAttr = [[NSAttributedString alloc] initWithData:[EMPTY\_STRING dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

chooseView.backgroundColor = [UIColor clearColor];

chooseView.origin = CGPointMake(0, i \* chooseView.height + 8);

chooseViewHeight = chooseView.height;

chooseView.whichChoose.text = arr[i];

chooseView.whichContent.attributedText = tempAttr;

chooseView.height = 55;

if ([chooseView.whichChoose.text isEqualToString:answer]) {

chooseView.chooseBtn.selected = YES;

}

[chooseBgView addSubview:chooseView];

}

//选项的真实高度

chooseBgView.height = chooseViewHeight \* [arr count];

} else {

//单 多选

CGFloat chooseAllHeight = 0.0;

for (int i = 0; i < [chooseArrs count]; i ++) {

PaperChooseObjectView \*chooseView = [PaperChooseObjectView viewFromNIB];

NSAttributedString \* tempAttr = [[NSAttributedString alloc] initWithData:[[chooseArrs[i] Content] dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

CGFloat height = [tempAttr boundingRectWithSize:CGSizeMake(SCREEN\_WIDTH - 87 - 30 - 12 - 19 - 8, CGFLOAT\_MAX) options:NSStringDrawingUsesLineFragmentOrigin context:nil].size.height;

chooseView.height = height;

chooseView.backgroundColor = [UIColor clearColor];

chooseView.origin = CGPointMake(0, chooseAllHeight + height \* 0.5);

chooseViewHeight = chooseView.height;

chooseView.whichChoose.text = [chooseArrs[i] OpNum];

chooseView.whichContent.attributedText = tempAttr;

if (chooseView.height > 50) {

chooseView.chooseBtnHeight.constant = chooseView.height;

} else {

chooseView.chooseBtnHeight.constant = 50;

}

if ([chooseView.whichChoose.text isEqualToString:answer]) {

chooseView.chooseBtn.selected = YES;

}

[chooseBgView addSubview:chooseView];

chooseAllHeight += chooseView.chooseBtnHeight.constant + 8;

}

//选项的真实高度

chooseBgView.height = chooseAllHeight;//chooseViewHeight \* [chooseArrs count];

}

[self refreshAnwserAndAnalysisWith:model andRect:\_chooseBgView.frame isChoose:YES isPaperCorrect:NO];

[self refreshTrunkChooseAndAnswerWith:trunkView and:trunkViewHeight isCorrect:NO];

//\*\*\*\*\*\*\*\*\*\*\*\*刷新其他

//刷新出题干之外的所有originy

[self refreshAllViewsOriginY];

}

//单选，解答题 公用

- (void)refreshTrunkChooseAndAnswerWith:(PaperObjectTrunkView \*)trunkView and:(CGFloat)height isCorrect:(BOOL)isCorrect

{

trunkView.trunkLabel.size = CGSizeMake(SCREEN\_WIDTH - 60, height) ;

//\*\*\*\*\*\*\*\*\*\*\*\*刷新题干

CGFloat trunkHeight = height;//[trunkStr sizeWithFont:FONT(15) Size:CGSizeMake(SCREEN\_WIDTH - 60, MAXFLOAT)].height;

trunkView.backgroundColor = [UIColor clearColor];

trunkView.frame = CGRectMake(0, 15, SCREEN\_WIDTH - 20, 37 + trunkHeight);

\_\_weak \_\_typeof(&\*trunkView)weakSelf\_TrunkView = trunkView;

trunkView.clickMoreBlock = ^(UIButton \*sender) {

//点击查看更多

sender.selected = !sender.selected;

if (sender.selected) {

//设置真正的题干高度

//37是 后来新增的 选择题 以及 间隙

weakSelf\_TrunkView.trunkLabelHeight.constant = trunkHeight + 37;

weakSelf\_TrunkView.height = trunkHeight + weakSelf\_TrunkView.clickMoreBtn.height + 37 + 37;

} else {

weakSelf\_TrunkView.trunkLabelHeight.constant = 175;

weakSelf\_TrunkView.height = 175 + 37 + weakSelf\_TrunkView.clickMoreBtn.height;

}

//刷新出题干之外的所有originy

if (isCorrect) {

//批改的话

\_stuAnswerLabel.origin = CGPointMake(20, CGRectGetMaxY(\_trunkView.frame) + 20);

\_stuAnswerBtn.origin = CGPointMake(20, CGRectGetMaxY(\_stuAnswerLabel.frame) + 10);

\_paperAnswerView.origin = CGPointMake(0, CGRectGetMaxY(\_stuAnswerBtn.frame));

self.remarksView.origin = CGPointMake(0, CGRectGetMaxY(\_paperAnswerView.frame) + 30);

//刷新滚动区域

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, CGRectGetMaxY(self.remarksView.frame) + 64);

} else {

//刷新选项

\_chooseBgView.origin = CGPointMake(0, CGRectGetMaxY(\_trunkView.frame));

//刷新解析

\_paperAnswerView.origin = CGPointMake(0, CGRectGetMaxY(\_chooseBgView.frame));

//刷新答案

\_correctView.origin = CGPointMake(0, CGRectGetMaxY(\_paperAnswerView.frame));

//刷新滚动区域

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, CGRectGetMaxY(\_correctView.frame) + 64);

}

};

if (trunkHeight > 175) {

//设置300高度 然后显示更多按钮

NSLog(@"%f",trunkView.clickMoreBtn.height);

trunkView.height = 175 + 35 + trunkView.clickMoreBtn.height;

trunkView.clickMoreBtn.hidden = NO;

} else {

//设置真正的题干高度

trunkView.trunkLabelHeight.constant = trunkHeight;

trunkView.height = trunkHeight + 37;

trunkView.clickMoreBtn.hidden = YES;

}

}

- (void)refreshAnwserAndAnalysisWith:(id)model andRect:(CGRect)topUIFrame isChoose:(BOOL)isChoose isPaperCorrect:(BOOL)isPaperCorrect

{

NSString \*showAnalysis;

NSString \*showAnswer;

NSNumber \*ProId;

NSString \*OldProScore;

if ([model isKindOfClass:[PaperBrowseModel class]]) {

//直接显示的话

PaperBrowseModel \*tempModel = model;

showAnalysis = [tempModel.outProList[0] Analysis];

if (isPaperCorrect) {

showAnswer = [tempModel.Pros[0] Answer];

OldProScore = [tempModel.Pros[0] OldProScore];

} else {

showAnswer = [tempModel.outProList[0] Answer];

}

ProId = [tempModel.Pros[0] ProId];

} else if ([model isKindOfClass:[PaperBrowseOptionModel class]]) {

//显示小问的话

PaperBrowseOptionModel \*tempModel = model;

showAnalysis = [tempModel Analysis];

showAnswer = [tempModel Answer];

ProId = tempModel.ProId;

OldProScore = tempModel.OldProScore;

}

//3.解析 加分割线

PaperAnswerAnalysisView \*paperAnswerView = [PaperAnswerAnalysisView viewFromNIB];

NSString \*tipDetailStr = showAnalysis;//@"我是解析我是解析我是解析我是解析我是解析我是解析我是解析我是解析我是解析我是解析我是解析";

NSAttributedString \* tipAttrStr = [[NSAttributedString alloc] initWithData:[tipDetailStr dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

CGFloat height = [tipAttrStr boundingRectWithSize:CGSizeMake(SCREEN\_WIDTH - 60, CGFLOAT\_MAX) options:NSStringDrawingUsesLineFragmentOrigin context:nil].size.height;//针对富文本有专门的计算动态高度的方法，与nsstring 大同小异；

// Generate a text layout.

// YYTextLayout \*tipLayout = [YYTextLayout layoutWithContainer:tipContainer text:tipAttrStr];

// dispatch\_async(dispatch\_get\_main\_queue(), ^{

// paperAnswerView.tip.size = tipLayout.textBoundingSize;

// paperAnswerView.tip.textLayout = tipLayout;

\_\_weak \_\_typeof(&\*paperAnswerView)weakSelf\_PaperAnswerView = paperAnswerView;

paperAnswerView.clickLookTipBlock = ^(UIButton \*sender) {

sender.selected = !sender.selected;

if (sender.selected) {

CGFloat tipDetailHeight = height;//[tipDetailStr sizeWithFont:FONT(15) Size:CGSizeMake(SCREEN\_WIDTH - 60, MAXFLOAT)].height;

weakSelf\_PaperAnswerView.tip.attributedText = tipAttrStr;

weakSelf\_PaperAnswerView.height = 8 + 25 + 8 + tipDetailHeight + 9;

weakSelf\_PaperAnswerView.tipHeight.constant = tipDetailHeight;

} else {

weakSelf\_PaperAnswerView.tipHeight.constant = 0;

weakSelf\_PaperAnswerView.tip.attributedText = nil;

weakSelf\_PaperAnswerView.height = 8 + 25 + 9;

}

if (isChoose) {

//单选刷新

//刷新出题干之外的所有originy

[self refreshAllViewsOriginY];

} else {

//刷新填空题

//刷新解析

\_paperAnswerView.origin = CGPointMake(0, CGRectGetMaxY(topUIFrame));

if (isPaperCorrect) {

//试卷批改的话

\_paperAnswerView.origin = CGPointMake(0, CGRectGetMaxY(\_stuAnswerBtn.frame));

//刷新答案

self.remarksView.origin = CGPointMake(0, CGRectGetMaxY(\_paperAnswerView.frame) + 30);

self.remarksView.height = 203;

\_correctView.origin = CGPointMake(0, CGRectGetMaxY(self.remarksView.frame));

//刷新滚动区域

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, CGRectGetMaxY(\_correctView?\_correctView.frame:self.remarksView.frame) );

} else {

//刷新答案

\_correctView.origin = CGPointMake(0, CGRectGetMaxY(\_paperAnswerView.frame));

//刷新滚动区域

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, CGRectGetMaxY(\_correctView.frame) + 64 );

}

}

};

// });

paperAnswerView.backgroundColor = [UIColor clearColor];

paperAnswerView.frame = CGRectMake(0, CGRectGetMaxY(topUIFrame), SCREEN\_WIDTH, height);

paperAnswerView.tip.numberOfLines = 0;

paperAnswerView.tip.attributedText = nil;

NSLog(@"%f",paperAnswerView.height);

\_paperAnswerView = paperAnswerView;

[self.scrollView addSubview:paperAnswerView];

//解析的真实高度

paperAnswerView.height = 8 + 25 + 9;

// paperAnswerView.clookTipBtn.hidden = YES;

WEAKSELF

if (isPaperCorrect) {

//试卷批改的话

self.remarksView = [PaperAnswerAndRemarksView viewFromNIB];

self.remarksView.score.text = OldProScore;

\_\_weak \_\_typeof(&\*self.remarksView)weakSelfself\_remarksView = self.remarksView;

self.remarksView.clickBottomBlock = ^{

//点击批改

NSArray \*dict = @[@{@"ProId":ProId?:EMPTY\_STRING,@"Score":weakSelfself\_remarksView.topTextField.text?:EMPTY\_STRING,@"Memo":weakSelfself\_remarksView.bottomTextField.text?:EMPTY\_STRING}];

!weakSelf\_SC.clickCorrectBlock ?: weakSelf\_SC.clickCorrectBlock(dict.mj\_JSONString);

};

if (\_currentSwitchPage == \_allNumPage) {

if (\_isLastNum) {

//最后一页

[self.remarksView.bottomBtn setTitle:@"完成" forState:UIControlStateNormal];

} else {

[self.remarksView.bottomBtn setTitle:@"下一页" forState:UIControlStateNormal];

}

//最后一页时候

self.remarksView.bottomBtn.hidden = NO;

} else {

self.remarksView.bottomBtn.hidden = YES;

}

self.remarksView.size = CGSizeMake(SCREEN\_WIDTH, 203);

[self.remarksView setOrigin:CGPointMake(0, CGRectGetMaxY(paperAnswerView.frame) + 30)];

[self.scrollView addSubview:self.remarksView];

//计算所有高度，然后设置scrollview的滚动区域

// CGFloat scrollViewSizeHeight = CGRectGetMaxY(self.remarksView.frame);

// self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, scrollViewSizeHeight + 64 + 30);

}

NSAttributedString \* attrStr = [[NSAttributedString alloc] initWithData:[answerDetail dataUsingEncoding:NSUnicodeStringEncoding] options:@{ NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];

// CGFloat answerDetailHeight = [answerDetail sizeWithFont:FONT(14) Size:CGSizeMake(SCREEN\_WIDTH - 60, MAXFLOAT)].height;

PaperCorrectAnswerView \*correctView = [PaperCorrectAnswerView viewFromNIB];

correctView.frame = CGRectMake(0, CGRectGetMaxY(self.remarksView?self.remarksView.frame:paperAnswerView.frame), SCREEN\_WIDTH, 300);

correctView.backgroundColor = [UIColor clearColor];

correctView.answer.text = [NSString stringWithFormat:@"正确答案:%@",answer];

correctView.answerDetail.attributedText = attrStr;

\_correctView = correctView;

[self.scrollView addSubview:correctView];

CGFloat correntHeight = [attrStr boundingRectWithSize:CGSizeMake(SCREEN\_WIDTH - 60, CGFLOAT\_MAX) options:NSStringDrawingUsesLineFragmentOrigin context:nil].size.height;//针对富文本有专门的计算动态高度的方法，与nsstring 大同小异；

correctView.answerDetail.size = CGSizeMake(SCREEN\_WIDTH - 60, correntHeight);

correctView.anwserDetailHeight.constant = correctView.answerDetail.size.height + 100;

correctView.height = 41 + correctView.anwserDetailHeight.constant + 100;

//计算所有高度，然后设置scrollview的滚动区域

CGFloat scrollViewSizeHeight = CGRectGetMaxY(correctView.frame);

self.scrollView.contentSize = CGSizeMake(SCREEN\_WIDTH, scrollViewSizeHeight + 64 );

//答案真实高度

// correctView.height = 41 + answerDetailHeight + 15;

}

@end

@end

//

// PaperStructureCell.m

// MaestroEdu

//

// Created by jesse on 2017/7/11.

// Copyright © 2017年 jesse. All rights reserved.

//

#import "PaperStructureCell.h"

#import "PaperStructureOneProCell.h"

#import "PaperStructureModel.h"

#import "PaperStructureSonModel.h"

#import "PaperStructureSonSonModel.h"

@interface PaperStructureCell ()<UITextFieldDelegate>

{

PaperStructureModel \*\_currentModel;

}

@end

@implementation PaperStructureCell

- (void)awakeFromNib {

[super awakeFromNib];

// Initialization code

self.backgroundColor = [UIColor clearColor];

}

- (void)addDataWith:(PaperStructureModel \*)model

{

//每题

\_currentModel = model;

PaperStructureOneProCell \*proCell = [PaperStructureOneProCell viewFromNIB];

proCell.textField.tag = 20170831;

proCell.textField.delegate = self;

proCell.frame = CGRectMake(0, 0, SCREEN\_WIDTH, 30);

proCell.leftTitle.text = @"每题";

proCell.textField.text = \_currentModel.everyPaper;

proCell.leftTitle.textColor = [UIColor blackColor];

proCell.rightLabel.textColor = [UIColor blackColor];

proCell.bottomLine.backgroundColor = [UIColor blackColor];

[self.bgView addSubview:proCell];

int whichOne = 0;

for (int i = 0; i < model.Tests.count ; i ++) {

PaperStructureSonModel \*sonModel = model.Tests[i];

[sonModel addObserverBlockForKeyPath:@"Score" block:^(id \_Nonnull obj, id \_Nonnull oldVal, id \_Nonnull newVal) {

for (UIView \*view in self.bgView.subviews) {

if ([view isKindOfClass:[PaperStructureOneProCell class]]) {

PaperStructureOneProCell \*cell = (PaperStructureOneProCell \*)view;

if (cell.textField.tag == 420170901 + i) {

cell.textField.text = newVal;

}

}

}

//刷新总分数

[self refreshAllScore];

}];

PaperStructureOneProCell \*proCell = [PaperStructureOneProCell viewFromNIB];

proCell.textField.tag = 420170901 + i;

proCell.textField.delegate = self;

proCell.frame = CGRectMake(0, (whichOne + 1) \* 30, SCREEN\_WIDTH, 30);

proCell.textField.text = [sonModel.Score isEqualToString:@"0"]?EMPTY\_STRING:sonModel.Score;;

proCell.leftTitle.text = [NSString stringWithFormat:@"第%d小题",i + 1];

proCell.leftTitle.textColor = [UIColor lightGrayColor];

proCell.rightLabel.textColor = [UIColor lightGrayColor];

proCell.bottomLine.backgroundColor = [UIColor lightGrayColor];

proCell.leftTitleLeftGap.constant = 15;

[self.bgView addSubview:proCell];

whichOne ++;

if ([[model.Tests[i] Pros] count] != 0) {

//小题 每问

PaperStructureOneProCell \*proSonProCell = [PaperStructureOneProCell viewFromNIB];

proSonProCell.textField.tag = 120170902 + i;

proSonProCell.textField.delegate = self;

proSonProCell.frame = CGRectMake(0, (whichOne + 1) \* 30, SCREEN\_WIDTH, 30);

proSonProCell.leftTitleLeftGap.constant = 30;

proSonProCell.leftTitle.text = @"每问";

proSonProCell.textField.text = sonModel.everyQuestion;

proSonProCell.leftTitle.textColor = [UIColor blackColor];

proSonProCell.rightLabel.textColor = [UIColor blackColor];

proSonProCell.bottomLine.backgroundColor = [UIColor blackColor];

[self.bgView addSubview:proSonProCell];

whichOne ++;

for (int j = 0; j < [[model.Tests[i] Pros] count]; j ++) {

PaperStructureSonSonModel \*sonSoneModel = [model.Tests[i] Pros][j];

[sonSoneModel addObserverBlockForKeyPath:@"Score" block:^(id \_Nonnull obj, id \_Nonnull oldVal, id \_Nonnull newVal) {

for (UIView \*view in self.bgView.subviews) {

if ([view isKindOfClass:[PaperStructureOneProCell class]]) {

PaperStructureOneProCell \*cell = (PaperStructureOneProCell \*)view;

if (cell.textField.tag == 220170903 + i + (1000 \* j)) {

cell.textField.text = newVal;

}

}

}

//刷新总分数

[self refreshAllScore];

}];

PaperStructureOneProCell \*proSonCell = [PaperStructureOneProCell viewFromNIB];

proSonCell.textField.tag = 220170903 + i + (1000 \* j);

proSonCell.textField.delegate = self;

proSonCell.frame = CGRectMake(0, (whichOne + 1) \* 30, SCREEN\_WIDTH, 30);

proSonCell.textField.text = [sonSoneModel.Score isEqualToString:@"0"]?EMPTY\_STRING:sonSoneModel.Score;

proSonCell.leftTitle.text = [NSString stringWithFormat:@"第%d小问",j + 1];

proSonCell.leftTitle.textColor = [UIColor lightGrayColor];

proSonCell.rightLabel.textColor = [UIColor lightGrayColor];

proSonCell.bottomLine.backgroundColor = [UIColor lightGrayColor];

proSonCell.leftTitleLeftGap.constant = 30;

[self.bgView addSubview:proSonCell];

whichOne ++;

}

}

}

self.bgViewHeight.constant = (whichOne + 1) \* 30;

[self.sortBtn setTitle:[NSString stringWithFormat:@"%@、",model.Sort] forState:UIControlStateNormal];

self.showType.text = model.TitleType;

self.allProNum.text = [NSString stringWithFormat:@"总计%lu题",(unsigned long)model.Tests.count?:0];

self.allScoreNum.text = [NSString stringWithFormat:@"共%@分",model.ScoreSum?:EMPTY\_STRING];

}

- (IBAction)clickTypeBtn:(UIButton \*)sender {

!self.clickSortBtn ?: self.clickSortBtn();

}

- (void)refreshAllScore

{

float score = 0;

for (UIView \*view in self.bgView.subviews) {

if ([view isKindOfClass:[PaperStructureOneProCell class]]) {

PaperStructureOneProCell \*cell = (PaperStructureOneProCell \*)view;

for (int i = 0 ; i < \_currentModel.Tests.count ; i ++) {

if (cell.textField.tag == 420170901 + i) {

NSString \*temp = cell.textField.text;

if ([temp isEqualToString:EMPTY\_STRING]) {

temp = @"0";

}

score += [temp floatValue];

}

}

}

}

\_currentModel.ScoreSum = [NSString stringWithFormat:@"%.1f",score];

self.allScoreNum.text = [NSString stringWithFormat:@"共%@分",\_currentModel.ScoreSum];

}

-(BOOL)textField:(UITextField \*)textField shouldChangeCharactersInRange:(NSRange)range replacementString:(NSString \*)string

{

for (int i = 0 ; i < \_currentModel.Tests.count ; i ++) {

for (int j = 0; j < [\_currentModel.Tests[i] Pros].count; j ++) {

if (textField.tag == 220170903 + i + (1000 \* j)) {

//小问

for (UIView \*view in self.bgView.subviews) {

if ([view isKindOfClass:[PaperStructureOneProCell class]]) {

PaperStructureOneProCell \*cell = (PaperStructureOneProCell \*)view;

if (cell.textField.tag == 120170902 + i) {

cell.textField.text = EMPTY\_STRING;

PaperStructureSonModel \*sonModel = ((PaperStructureSonModel \*)\_currentModel.Tests[i]);

sonModel.everyQuestion = EMPTY\_STRING;

//设置当前小问的分数

NSString \*str = textField.text;

if ([string isEqualToString:EMPTY\_STRING]) {

if (str.length == 0) {

return NO;

}

str = [str substringToIndex:str.length - 1];

}

dispatch\_after(dispatch\_time(DISPATCH\_TIME\_NOW, (int64\_t)(0.01 \* NSEC\_PER\_SEC)), dispatch\_get\_main\_queue(), ^{

((PaperStructureSonSonModel \*)sonModel.Pros[j]).Score = [NSString stringWithFormat:@"%@%@",str,string];

});

}

}

}

}

}

if (textField.tag == 120170902 + i) {

//每问的

for (PaperStructureSonSonModel \*sonModel in [\_currentModel.Tests[i] Pros]) {

NSString \*str = textField.text;

if ([string isEqualToString:EMPTY\_STRING]) {

if (str.length == 0) {

return NO;

}

str = [str substringToIndex:str.length - 1];

}

sonModel.Score = [NSString stringWithFormat:@"%@%@",str,string];

((PaperStructureSonModel \*)\_currentModel.Tests[i]).everyQuestion = sonModel.Score;

}

} else if (textField.tag == 420170901 + i) {

//小题

for (UIView \*view in self.bgView.subviews) {

if ([view isKindOfClass:[PaperStructureOneProCell class]]) {

PaperStructureOneProCell \*cell = (PaperStructureOneProCell \*)view;

if (cell.textField.tag == 20170831) {

cell.textField.text = EMPTY\_STRING;

\_currentModel.everyPaper = EMPTY\_STRING;

//设置当前小题的分数

NSString \*str = textField.text;

if ([string isEqualToString:EMPTY\_STRING]) {

if (str.length == 0) {

return NO;

}

str = [str substringToIndex:str.length - 1];

}

dispatch\_after(dispatch\_time(DISPATCH\_TIME\_NOW, (int64\_t)(0.01 \* NSEC\_PER\_SEC)), dispatch\_get\_main\_queue(), ^{

((PaperStructureSonModel \*)\_currentModel.Tests[i]).Score = [NSString stringWithFormat:@"%@%@",str,string];

});

}

}

}

}

}

if (textField.tag == 20170831) {

//每题的

for (PaperStructureSonModel \*model in \_currentModel.Tests) {

NSString \*str = textField.text;

if ([string isEqualToString:EMPTY\_STRING]) {

if (str.length == 0) {

return NO;

}

str = [str substringToIndex:str.length - 1];

}

model.Score = [NSString stringWithFormat:@"%@%@",str,string];

\_currentModel.everyPaper = model.Score;

}

}

dispatch\_after(dispatch\_time(DISPATCH\_TIME\_NOW, (int64\_t)(0.01 \* NSEC\_PER\_SEC)), dispatch\_get\_main\_queue(), ^{

//刷新总分数

[self refreshAllScore];

});

return YES;

}

@end