```objc
//
//  XTPDFContainerViewController.m
//  PDFViewer
//
//  Created by Pavel Dolgov on 25/01/16.
//  Copyright © 2016 Pavel Dolgov. All rights reserved.
//

#import "XTPDFContainerViewController.h"
#import "XTPDFTiledView.h"
#import "XTPDFMiniatureCustomCell.h"
#import "XTPDFPageNumberBackgroundView.h"

typedef enum
{
    XTPDFNavigationPrevious = 0,
    XTPDFNavigationNext,
    XTPDFNavigationShowPreviews,
    XTPDFNavigationUndefined
} XTPDFNavigation;

NSString * XTPDFMiniatureCustomCellIdentifier =
@"XTPDFMiniatureCustomCell";

@interface XTPDFContainerViewController ()
{
    NSString * identifier;
    NSTimer * timerToHidePageNumber;
    UILabel * pageNumberLabel;
    UIView * pageNumberBackgroundView;
    NSInteger _pageNumber;
    CGSize oldSize;
}

@end

@implementation XTPDFContainerViewController

- (instancetype)initWithFileData:(NSData *)data
{
    self = [super initWithNibName:nil bundle:nil];
    if (self)
    {
        self.pageNumber = -1;
        identifier = XTPDFMiniatureCustomCellIdentifier;
        self.data = [XTPDFFileModel pagedPdfByData:data];

        [self showFile];
    }
    return self;
}

- (instancetype)initWithFilePath:(NSURL *)path
{
    self = [super initWithNibName:nil bundle:nil];
    if (self)
    {
        self.pageNumber = -1;
        identifier = XTPDFMiniatureCustomCellIdentifier;
        self.data = [XTPDFFileModel pagedPdfByData:[NSData 
dataWithContentsOfURL:path]];

        [self showFile];
    }
    return self;
}

- (void)showFile
{
    if (!self.scrollView)
    {
        UITapGestureRecognizer * tap = [[UITapGestureRecognizer 
alloc] initWithTarget:self action:@selector(tap:)];
        [tap setDelaysTouchesBegan:YES];
        [tap setDelegate:self];

        UISwipeGestureRecognizer * right = 
[[UISwipeGestureRecognizer alloc] initWithTarget:self 
action:@selector(swipe:)];
        [right setDirection:UISwipeGestureRecognizerDirectionRight];
        [right setDelegate:self];

        UISwipeGestureRecognizer * left = [[UISwipeGestureRecognizer 
alloc] initWithTarget:self action:@selector(swipe:)];
        [left setDirection:UISwipeGestureRecognizerDirectionLeft];
        [left setDelegate:self];

        UISwipeGestureRecognizer * up = [[UISwipeGestureRecognizer 
alloc] initWithTarget:self action:@selector(swipe:)];
        [up setDirection:UISwipeGestureRecognizerDirectionUp];
        [up setDelegate:self];

        UISwipeGestureRecognizer * down = [[UISwipeGestureRecognizer 
alloc] initWithTarget:self action:@selector(swipe:)];
        [down setDirection:UISwipeGestureRecognizerDirectionDown];
        [down setDelegate:self];

        self.scrollView = [[XTPDFScrollView alloc] 
initWithFrame:CGRectZero];
        [self.scrollView setBackgroundColor:[UIColor whiteColor]];
        [self.scrollView addGestureRecognizer:tap];
        [self.scrollView addGestureRecognizer:right];
        [self.scrollView addGestureRecognizer:left];
        [self.scrollView addGestureRecognizer:up];
        [self.scrollView addGestureRecognizer:down];
        [self.view addSubview:self.scrollView];

        if (!pageNumberLabel)
        {
            pageNumberLabel = [[UILabel alloc]
initWithFrame:CGRectMake(20.0f, 60.0f, 100.0f, 40.0f)];
            [pageNumberLabel
setTextAlignment:NSTextAlignmentCenter];
            [pageNumberLabel.layer
setCornerRadius:pageNumberLabel.frame.size.height / 2];
            pageNumberBackgroundView =
[[XTPDFPageNumberBackgroundView alloc]
initWithFrame:pageNumberLabel.frame];

            [self.view addSubview:pageNumberBackgroundView];
            [self.view addSubview:pageNumberLabel];

            [self changePageNumberViewState];
        }
    }

    [self showCurrentPage];
}

- (void)showCurrentPage
{
    self.currentPage = CGPDFDocumentGetPage(self.data.pdf,
self.pageNumber);
    if(self.currentPage != NULL) CGPDFPageRetain(self.currentPage);

    [self setRightFrame];
    [self changePageNumberViewState];
}

- (void)viewDidLayoutSubviews
{
    [super viewDidLayoutSubviews];
    [self setRightFrame];
}

- (void)viewDidAppear:(BOOL)animated
{
    [super viewDidAppear:animated];
    NSIndexPath * path = [NSIndexPath indexPathForItem:_pageNumber -
1 inSection:0];
    if (path.row < self.data.numberOfPages)
    {
        [self.miniaturesView selectItemAtIndexPath:path
                                          animated:YES

scrollPosition:UICollectionViewScrollPositionCenteredHorizontally];
    }
}

- (void)setPageNumber:(NSInteger)pageNumber
{
    _pageNumber = pageNumber;
    NSIndexPath * path = [NSIndexPath indexPathForItem:pageNumber -
1 inSection:0];
    if (path.row < self.data.numberOfPages)
    {
        [self.miniaturesView selectItemAtIndexPath:path
                                          animated:YES

scrollPosition:UICollectionViewScrollPositionCenteredHorizontally];
    }
}

- (NSInteger)pageNumber
{
    return _pageNumber;
}

- (void)setRightFrame
{
    CGFloat yCoordinate = 0.0f;
    oldSize = self.scrollView.frame.size;
    [self.scrollView setFrame:CGRectMake(0.0f, yCoordinate,
self.view.frame.size.width, self.view.frame.size.height -
yCoordinate)];
    if (self.scrollView.frame.size.width != oldSize.width ||
self.scrollView.frame.size.height != oldSize.height)
    {
        [self.scrollView setHasChangedSize:YES];
    }
    if (!self.miniaturesView && self.view.bounds.size.width != 0)
    {
        [self setupMiniaturesView];
    }
    [self.scrollView setPage:self.currentPage];
}

- (void)redraw
{
    [self.scrollView setPage:self.currentPage];
}

- (void)setupMiniaturesView
{
    CGRect miniaturesViewRect = self.view.bounds;
    miniaturesViewRect.size.height = miniaturesViewRect.size.height
* 2 / 9;
    miniaturesViewRect.origin.y = self.view.bounds.size.height -
miniaturesViewRect.size.height;
    self.miniaturesView = [[UICollectionView alloc]
initWithFrame:miniaturesViewRect
                                             collectionViewLayout:
[self collectionViewLayoutForFrame:miniaturesViewRect]];
    [self.miniaturesView setBackgroundColor:[UIColor grayColor]];
    [self.miniaturesView registerClass:[XTPDFMiniatureCustomCell
class] forCellWithReuseIdentifier:identifier];
```

```
    [self.miniaturesView setDelegate:self];
    [self.miniaturesView setDataSource:self];
    [self.miniaturesView
setAutoresizingMask:UIViewAutoresizingFlexibleTopMargin |
UIViewAutoresizingFlexibleWidth];
    [self.miniaturesView setHidden:YES];
    [self.view addSubview:self.miniaturesView];

    UISwipeGestureRecognizer * down = [[UISwipeGestureRecognizer
alloc] initWithTarget:self action:@selector(swipeMiniatures:)];
    [down setDirection:UISwipeGestureRecognizerDirectionDown];
    [down setDelegate:self];
    [self.miniaturesView addGestureRecognizer:down];
}

- (UICollectionViewLayout *)collectionViewLayoutForFrame:
(CGRect)frame
{
    UICollectionViewFlowLayout * layout =
[[UICollectionViewFlowLayout alloc] init];
    [layout
setScrollDirection:UICollectionViewScrollDirectionHorizontal];
    [layout setItemSize:CGSizeMake(frame.size.height - 4.0f,
frame.size.height - 4.0f)];
    [layout setMinimumInteritemSpacing:2.0f];
    [layout setMinimumLineSpacing:2.0f];
    return layout;
}

- (void)changeMiniaturesViewState
{
    BOOL hidden = self.miniaturesView.isHidden;
    if (hidden) [self.miniaturesView setHidden:NO];
    [UIView animateWithDuration:0.25 animations:
     ^{
         if (!hidden)
         {
             self.miniaturesView.alpha = 0.0;
         }
         else
         {
             self.miniaturesView.alpha = 1.0;
         }
     }
                     completion:^(BOOL finished)
     {
         [self.miniaturesView setHidden:!hidden];
     }];
}

- (void)changePageNumberViewState
{
    [timerToHidePageNumber invalidate];
    [pageNumberLabel setText:[NSString stringWithFormat:@"%li /
%lu", (long)_pageNumber, (unsigned long)self.data.numberOfPages]];
    [pageNumberLabel setHidden:NO];
    [pageNumberBackgroundView setHidden:NO];
    timerToHidePageNumber = [NSTimer scheduledTimerWithTimeInterval:
1.0f target:self selector:@selector(onTimer) userInfo:nil
repeats:NO];
}

- (void)onTimer
{
    [pageNumberLabel setHidden:YES];
    [pageNumberBackgroundView setHidden:YES];
}

#pragma mark - navigation handlers and helpers

- (void)tap:(UIGestureRecognizer *)recognizer
{
    BOOL handleOnlyBarZone = NO;
    BOOL handleOnlyPreviewZone = NO;

    if (!self.miniaturesView.isHidden)
    {
        [self changeMiniaturesViewState];
        handleOnlyBarZone = YES;
    }

    CGPoint location = [recognizer locationInView:self.view];
    NSUInteger squareNumber = [self squareNumberByPoint:location];

    if (squareNumber < 9)
    {
        return;
    }

    if (squareNumber >= 63)
    {
        if (!handleOnlyBarZone)
            [self handle:XTPDFNavigationShowPreviews];
        return;
    }


    if (handleOnlyBarZone || handleOnlyPreviewZone)
    {
        return;
    }

    if (squareNumber % 9 == 0 || squareNumber % 9 == 1 ||
squareNumber % 9 == 2)
    {
        [self handle:XTPDFNavigationPrevious];
        return;
    }
    if (squareNumber % 9 == 6 || squareNumber % 9 == 7 ||
squareNumber % 9 == 8)
    {
        [self handle:XTPDFNavigationNext];
        return;
    }

    return;
}

- (void)swipe:(UISwipeGestureRecognizer *)recognizer
{
    // Suddenly sometimes contensize is equal to zero
    if (self.scrollView.contentSize.width *
self.scrollView.contentSize.height == 0) return;

    // It means: "PDF can be slightly zoomed and still answer to
swipe".
    // Zero will cause wrong behaviour.
    if (self.scrollView.contentSize.width >
self.scrollView.frame.size.width + 50.0f) return;
    if (self.scrollView.contentSize.height >
self.scrollView.frame.size.height + 50.0f) return;
    switch (recognizer.direction)
    {
        case UISwipeGestureRecognizerDirectionDown:
        case UISwipeGestureRecognizerDirectionRight:
            [self handle:XTPDFNavigationPrevious];
            break;

        case UISwipeGestureRecognizerDirectionUp:
        case UISwipeGestureRecognizerDirectionLeft:
            [self handle:XTPDFNavigationNext];
            break;
    }
}

- (void)swipeMiniatures:(UISwipeGestureRecognizer *)recognizer
{
    [self changeMiniaturesViewState];
}

- (void)handle:(XTPDFNavigation)navigation
{
    switch (navigation)
    {
        case XTPDFNavigationNext:
            if (self.pageNumber == self.data.numberOfPages) return;
            self.pageNumber++;
            [self showCurrentPage];
            break;

        case XTPDFNavigationPrevious:
            if (self.pageNumber == 1) return;
            self.pageNumber--;
            [self showCurrentPage];
            break;

        case XTPDFNavigationShowPreviews:
            [self changeMiniaturesViewState];
            break;

        case XTPDFNavigationUndefined:
            break;
    }
}

/*

 as following:
 0  1  2  3  4  5  6  7  8
 ...

 ...
 63 64 65 66 67 68 69 70 71
 72 73 74 75 76 77 78 79 80

 -1 if something went wrong (in fact cannot occure)
 */

- (NSUInteger)squareNumberByPoint:(CGPoint)point
{
    NSUInteger piecesNumber = 9; // the screen is splitted into 9 *
9 pieces;
    CGFloat squareWidth = self.view.frame.size.width / piecesNumber;
    CGFloat squareHeight = self.view.frame.size.height /
piecesNumber;
    for (NSUInteger i = 0; i < piecesNumber; ++i)
    {
        for (NSUInteger j = 0; j < piecesNumber; ++j)
        {
            CGRect rect = CGRectMake(squareWidth * i, squareHeight *
j, squareWidth, squareHeight);
            if (rect.origin.x < point.x && rect.origin.y < point.y
&&
                rect.origin.x + squareWidth > point.x &&
rect.origin.y + squareHeight > point.y)
            {
                return i + j * piecesNumber;
            }
        }
    }
    return -1;
}

- (UIView *)viewForScreenshot
{
    return self.scrollView;
}

#pragma mark - UICollectionViewDataSource

- (UICollectionViewCell *)collectionView:(UICollectionView
*)collectionView cellForItemAtIndexPath:(NSIndexPath *)indexPath
{
    XTPDFMiniatureCustomCell * cell = [collectionView
dequeueReusableCellWithReuseIdentifier:identifier
forIndexPath:indexPath];
    NSUInteger pageNumber = indexPath.row + 1;
    [cell setPage:CGPDFDocumentGetPage(self.data.pdf, pageNumber)
pageNumber:pageNumber];
    return cell;
}

- (NSInteger)numberOfSectionsInCollectionView:(UICollectionView
*)collectionView
{
    return 1;
}

- (NSInteger)collectionView:(UICollectionView *)collectionView
numberOfItemsInSection:(NSInteger)section
{
    return self.data.numberOfPages;
}

#pragma mark - UICollectionViewDelegate

- (void)collectionView:(UICollectionView *)collectionView
didSelectItemAtIndexPath:(NSIndexPath *)indexPath
{
    self.pageNumber = indexPath.row + 1;
    [self showCurrentPage];
}

#pragma mark - gesture delegate

- (BOOL)gestureRecognizer:(UIGestureRecognizer *)gestureRecognizer
shouldRecognizeSimultaneouslyWithGestureRecognizer:
(UIGestureRecognizer *)otherGestureRecognizer
{
    return YES;
}

@end
```