

文档

文档模块主要包括 **课件** 和 **画笔**，课件由 **BJLDocumentVM** 管理，画笔由 **BJLDrawingVM** 管理。

课件

上传、添加课件，删除课件，课件翻页、同步

课件视图控制器、加载课件

课件控制器翻页、同步、画笔控制、页码信息

课件窗口位置同步

课件操作授权

小黑板发布、学生参与作答、提交，位置同步

获取、上传、删除、搜索云盘文件

获取、上传、删除、搜索作业

画笔

设置画笔类型，画笔操作模式，颜色，线宽等

文档画笔，小黑板画笔授权、开启、关闭

激光笔显示、移动

添加图片画笔

画笔轨迹、画笔落笔前位置

1. 课件

SDK 支持 PDF、Word、动效 PPT 等文档的显示，教室里面默认至少有一个白板课件，上传课件支持图片，PDF 文件，Word 文件，PPT 文件。

1. 课件控制器显示

SDK 支持为每一个课件获取一个视图，也支持所有课件使用同一个视图，目前大班课场景一般是使用同一个视图，专业小班课场景下每个课件单独一个视图管理。课件视图支持设置的参数参考 `BJLslideshowUI`。

- 设置课件类型：SDK 提供 native 和 H5 两种类型的课件。
native 课件加载快、支持缩放手势，但不支持 PPT 动画；
H5 课件加载略慢，不支持缩放手势，支持 PPT 动画。**默认使用 H5 课件**。目前 SDK 支持课件的动态切换，在使用 H5 课件的情况下，教室里存在动态课件，将会使用 H5 课件，教室里只有静态课件的时候，将会使用 native 课件，native 课件是使用 `CollectionView` 加载静态图片的方式实现的，因此课件的尺寸和位置必须设置为整数的 `pt` 值，避免视图无法渲染的情况。SDK 支持动态开关 PPT 动效，设置 `BJLRoom` 的 `disablePPTAnimation` 可即时切换课件类型。

```
1. // 设置课件类型，不设置则默认使用 H5 课件  
2. self.room.disablePPTAnimation = NO; // YES:  
    native, NO: H5
```

- 大班课直接通过 `BJLRoom` 的 `slideshowViewController` 获取课件视图。

```
1. @property (nonatomic, readonly, nullable)
    UIViewController<BJLSlideshowUI>
    *slideshowViewController;
```

```
1. /**
2. 设置静态课件尺寸
3. 加载课件图片时对图片做等比缩放，长边小于/等于
`imageSize`
4. 单位为像素，默认初始加载 1080，取值在
`BJLAiiIMGMinSize` 到 `BJLAiiIMGMaxSize` 之间 (1
~ 4096)
5. 不建议进教室成功后设置此参数，因为会导致已经加载
过的图片缓存失效
6. 只对静态课件生效，参考 `BJLRoom` 的
`disablePPTAnimation`
7. */
8. self.room.slideshowViewController.imageSize =
    720;
```

```
1. // 显示课件视图，将 BJLRoom 的课件视图添加到当前
viewController 的对应视图
2. [self
    addChildViewController:self.room.slideshowViewController]
3. [self.slideshowView
    addSubview:self.room.slideshowViewController.view]
4. [self.room.slideshowViewController
    didMoveToParentViewController:self];
5.
6.
    self.room.slideshowViewController.shouldSwitchNative
    = ^(NSString *_Nullable documentID, void
        (^_Nonnull callback)(BOOL)) {
7.     bjl_strongify(self);
```

```
8.     UIAlertController *alertViewController =  
9.         [UIAlertController alertControllerWithTitle:nil  
10.            message:@"PPT动画加载失败！\n网络较差建议跳过动  
11.            preferredStyle:UIAlertControllerStyleAlert];  
12.            [alertViewController  
13.                bjl.addActionWithTitle:BJLLocalizedString(@"重新  
14.                style:UIAlertActionStyleCancel  
15.                handler:^(UIAlertAction *_Nonnull action) {  
16.                    callback(NO);  
17.                }];  
18.            [alertViewController  
19.                bjl.addActionWithTitle:BJLLocalizedString(@"跳过  
20.                style:UIAlertActionStyleDefault  
21.                handler:^(UIAlertAction *_Nonnull action) {  
22.                    callback(YES);  
23.                }];  
24.            bjl_weakify(alertViewController);  
25.            [alertViewController  
26.                bjl_kvo:BJLMakeProperty(self.room.slideshowViewCo  
27.                webPPTLoadSuccess)  
28.                observer:^(BJLControlObserving(id _Nullable  
29.                    value, id _Nullable oldValue, BJLPropertyChange  
30.                    *_Nullable change) {  
31.                bjl_strongify(alertViewController);
```

```
25.          if
    (.room.slideshowViewController.webPPTLoadSuc
    {
26.              [alertViewController
        bjl_dismissAnimated:YES completion:nil];
27.              return NO;
28.          }
29.          return YES;
30.      }];
31.      if (.presentedViewController) {
32.          [.presentedViewController
            bjl_dismissAnimated:YES completion:nil];
33.      }
34.      [
            presentViewController:alertViewController
            animated:YES completion:nil];
35.  };
36.  .room.slideshowViewController.imageSize
    = 1080;
37.
    .room.slideshowViewController.view.backgroundColor
    = BJLTheme.roomBackgroundColor; // 课件背景色
    和直播间整个底色一致
38.
    .room.slideshowViewController.placeholderImage
    = [UIImage
        bjl_imageWithColor:BJLTheme.blackboardColor];
39.
    .room.slideshowViewController.whiteboardBackgroun
    = [UIImage
        bjl_imageWithColor:BJLTheme.blackboardColor];
40.
    .room.slideshowViewController.prevPageIndicatorImage
    = [UIImage
        bjl_imageNamed:@"bjl_sc_ppt_prev"];
```

```
41.         self.room.slideshowViewController.nextPageIndicatorImage
        = [UIImage
        bjl_imageNamed:@"bjl_sc_ppt_next"];
42.         self.room.slideshowViewController.pageControlButton
        = ({
43.             const CGFloat buttonWidth = 72.0,
buttonHeight = 32.0;
44.             UIButton *button = [UIButton new];
45.             [button setTitleColor:[UIColor whiteColor]
 forState:UIControlStateNormal];
46.             button.titleLabel.font = [UIFont
systemFontOfSize:14.0];
47.             // !!!: should be same to
`BJLContentView.clearDrawingButton.backgroundColor
48.             [button setBackgroundImage:[UIImage
bjl_imageWithColor:[UIColor bjlsc_dimColor]]
 forState:UIControlStateNormal];
49.             [button setBackgroundImage:[UIImage
bjl_imageWithColor:[UIColor
bjl_colorWithHexString:@"#89899C" alpha:0.5]]
 forState:UIControlStateDisabled];
50.             [button.layer.cornerRadius = buttonHeight /
2;
51.             button.layer.masksToBounds = YES;
52.             [button addTarget:self
action:@selector(showQuickSlideViewController)
forControlEvents:UIControlEventTouchUpInside];
53.             [self.room.slideshowViewController.view
addSubview:button];
54.             [button
bjl_makeConstraints:^(BJLConstraintMaker
*make) {
55.                 make.centerX.equalTo(self.room.slideshowViewCon
```

```
58.     make.bottom.equalTo(self.room.slideshowViewCont
59.         make.size.equalTo.sizeOffset(CGSizeMake(buttonWidt
        buttonHeight));
60.     }];
61.     button;
62.   );
63.   [self.room.slideshowViewController.view
      bjL_makeConstraints:^(BJLConstraintMaker
      *_Nonnull make) {
64.     make.edges.equalTo(self.slideshowView);
65.   }];

```

- 专业小班课通过 `BJLDocumentVM` 获取黑板视图和课件视图。

```
1. /** 黑板视图控制器 */
2. @property (nonatomic, readonly)
    UIViewController<BJLBlackboardUI>
    *blackboardViewController;
3. /** 黑板页数 */
4. @property (nonatomic, readonly) NSInteger
    blackboardContentPages;
5. /** 设置黑板视图（blackboardViewController）的默
    认背景图 */
6. @property (nonatomic) UIImage
    *blackboardImage;
7. // 获取当前黑板页码
8. CGFloat localPageIndex =
    self.room.documentVM.blackboardViewController.p
9.
10. /**
11. 指定文档 ID 创建对应的视图控制器
```

```
12. #param documentID 文档 ID, 通过 BJLDocument  
    的 documentID 获得  
13. #return 对应文档的视图控制器  
14. */  
15. - (UIViewController<BJLSlideshowUI>  
*)documentViewControllerWithID:(NSString  
*)documentID;
```

2. 同步以及更新课件视图位置

专业小班课引入了课件窗口，各端可以在教室内相同比例的桌面上同步课件窗口的尺寸和位置。

更新窗口行为：

```
1. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_open; // 打开  
2. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_updateRect; // 移动 & 缩放  
3. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_stick; // 置顶  
4. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateActionFullScreen; // 全屏  
5. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_maximize; // 最大化  
6. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_restore; // 还原  
7. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_close; // 关闭  
8. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_rename; // 视图标题更新  
9. FOUNDATION_EXPORT NSString *const  
    BJLWindowsUpdateAction_min; // 视图最小化
```

```
1. /**
2.  更新文档显示信息
3.  #param documentID 文档 ID
4.  #param displayInfo 文档显示信息
5. */
6. [self.context.room.documentVM
    updateDocumentWithID:BJLBlackboardID
    displayInfo:displayInfo];
7.
8. /**
9.  更新文档显示信息的通知
10. #param document 更新后的文档信息
11. */
12. [self
    bjl_observe:BJLMakeMethod(self.context.room.docu
        didUpdateDocument:)
    filter:^BOOL(BJLDocument *document){
14.     // bjl_strongify(self);
15.     return [document.documentID
        isEqualToString:BJLBlackboardID];
16. }
17.     observer:^BOOL(BJLDocument *document)
{
18.     bjl_strongify(self);
19.     // 黑板滑动
20.     [self
        scrollToTopOffset:document.displayInfo.topOffset];
21.     return YES;
22. }];
23.
24. /**
25.  更新文档窗口
26. #param documentID 文档 ID
27. #param action 更新类型, 参考
    BJLWindowsUpdateModel 的
```

```
BJLWindowsUpdateAction
28. #param displayInfos 教室内所有文档窗口显示信息
29. #return 调用错误, 参考 BJLErrorCode
30. */
31. [self.room.documentVM
    updateDocumentWindowWithID:documentID
32.             action:action
33.
    displayInfos:self.documentWindowDisplayInfos];
34.
35. /**
36. 文档窗口更新通知
37. #param updateModel 更新信息
38. #param shouldReset 是否重置
39. */
40. [self
    bjl_observe:BJLMakeMethod(self.room.documentVM
        didUpdateDocumentWindowWithModel:shouldReset
41.         observer:
        (BJLMethodObserver)^BOOL(BJLWindowUpdateMod
            *updateModel, BOOL shouldReset) {
42.     bjl_strongify(self);
43.     if (shouldReset) {
44.         [self
            resetDocumentWindowsWithModel:updateModel];
45.     }
46.     else {
47.         [self
            updateDocumentWindowWithModel:updateModel];
48.     }
49.     return YES;
50. }];
```

3. 课件控制器管理

课件控制器实现了 `BJLSlideshowUI` 协议，提供了通用的 API，加载图片尺寸，翻页，备注，画笔状态，能否翻页状态等，也有仅静态课件可用的接口和仅动态课件可用的接口，可以根据具体需求查询使用。

- 共同 API:

```
1. #pragma mark - view type
2.
3. /// 当前课件视图类型，不固定，根据 BJLDocument 的
   数据内容会发生改变
4. @property (nonatomic, readonly) BJLPPTViewType
   viewType;
5.
6. /// 禁用课件的动效，默认为 NO，动效开启后
   `viewType` 只会是 `BJLPPTViewType_H5`
7. @property (nonatomic) BOOL
   disablePPTAnimation;
8.
9. #pragma mark - page control
10.
11. /// 设置当前允许翻到的最大页码，在未主动翻页前作为
    当前页码，在 `disableOverMaxPage` 为 YES 时，否
    则仅作为当前的页码使用
12. /// #param slidePage BJLSlidePage
13. - (void)updateMaxSlidePage:(BJLSlidePage
   *)slidePage;
14.
15. /// 是否需要同步控制教室内页码，默认为 NO，设置为
    YES 时，本地的翻页等操作都会同步给教室内其他用户
16. @property (nonatomic, readonly) BOOL
   syncPageChange;
17. - (void)updateSyncPageChange:
   (BOOL)syncPageChange;
18.
19. /// 是否禁止超过翻页超过最大页码，默认 YES
```

```
20. @property (nonatomic, readonly) BOOL
    disableOverMaxPage;
21. - (void)updateDisableOverMaxPage:
    (BOOL)disableOverMaxPage;
22.
23. /// 当前课件是否支持滑动翻页, 默认可滑动
24. /// #discussion 开启画笔或者动态课件可交互的情况下, 仍然不能滑动翻页
25. @property (nonatomic, readonly) BOOL
    scrollEnabled;
26. - (void)updateScrollEnabled:(BOOL)scrollEnabled;
27.
28. /// 当前页码
29. /// #discussion 不能滑动翻页时, 参考
    `scrollEnabled`, 设置 pageIndex 无效
30. @property (nonatomic, readonly) NSInteger
    pageIndex;
31. - (nullable BJLError *)updatePageIndex:
    (NSInteger)pageIndex;
32.
33. /// 能否向前翻页, 或者存在前一步
34. @property (nonatomic, readonly) BOOL
    canStepForward;
35.
36. /// 能否向后翻页, 或者存在后一步
37. @property (nonatomic, readonly) BOOL
    canStepBackward;
38.
39. /// 向前翻页
40. - (nullable BJLError *)pageStepForward;
41.
42. /// 向后翻页
43. - (nullable BJLError *)pageStepBackward;
44.
45. /// 课件视图处理多个 BJLDocument 数据时: 是否禁用
    跨越课件翻页
```

```
46. /// #discussion 默认允许跨课件，不同课件之间的  
    pageStepForward 和 pageStepBackward 是可用的  
47. /// #discussion 如果禁用，在多个 pageStepForward  
    和 pageStepBackward 不可用，只能通过设置页码  
    index 进行翻页  
48. @property (nonatomic) BOOL disableCrossDoc;  
49.  
50. #pragma mark - customize  
51.  
52. /// 自定义白板样式  
53. @property (nonatomic) BJLWhiteboard  
    *whiteboard;  
54.  
55. /// 自定义页面控制按钮  
56. @property (nonatomic) UIButton  
    *pageControlButton;  
57.  
58. #pragma mark - remark  
59.  
60. /// 是否显示课件备注，默认显示  
61. @property (nonatomic, readonly) BOOL  
    showPPTRemarkInfo;  
62. - (void)updateShowPPTRemarkInfo:(BOOL)show;  
63.  
64. #pragma mark - draw  
65.  
66. /// 画笔开关状态，默认为 NO，不可使用画笔  
67. @property (nonatomic) BOOL drawingEnabled;  
68.  
69. /// 课件当前页图像在视图中的布局信息  
70. @property (nonatomic, readonly) CGRect  
    imageFrameInPPTView;  
71.  
72. /// 清除当前画布上所有画笔  
73. - (void)clearDrawing;
```

- 静态课件 API。

```
1. /// 静态课件尺寸
2. /// #discussion 加载课件图片时对图片做等比缩放，长
   边小于/等于 `imageSize`，放大时加载 1.5 倍尺寸的图
   片
3. /// #discussion 单位为像素，默认初始加载 720、放大
   加载 1080，取值在 `BJLAlilIMGMinSize` 到
   `BJLAlilIMGMaxSize` 之间 (1 ~ 4096)
4. /// #discussion 不建议进教室成功后设置此参数，因为
   会导致已经加载过的图片缓存失效
5. @property (nonatomic) NSInteger imageSize;
6.
7. /// 静态课件占位图
8. @property (nonatomic, nullable) UIImage
   *placeholderImage;
9.
10. /// 静态课件白板图片
11. @property (nonatomic, nullable) UIImage
   *whiteboardBackgroundImage;
12.
13. /// 回放设置本地图片路径
14. @property (nonatomic, nullable) NSArray
   *localPPTImagePaths;
15.
16. /// 是否加载课件原图，默认不加载原图，设置后改变
   imageSize 无效
17. @property (nonatomic, readonly) BOOL
   useOriginalImage;
18. - (void)updateUseOriginalImage:
   (BOOL)useOriginalImage;
19.
20. /// 静态课件显示模式，每次动态和静态课件切换时都会
   重置成 BJLContentMode_scaleAspectFit 完整显示
```

```
21. @property (nonatomic, readonly)  
    BJLContentMode contentMode;  
22. - (void)updateContentMode:  
    (BJLContentMode)contentMode;  
23.  
24. /// 静态课件是否支持缩放  
25. /// #param scaleEnabled 是否支持缩放  
26. /// #discussion 默认支持缩放  
27. @property (nonatomic, readonly) BOOL  
    scaleEnabled;  
28. - (void)updateScaleEnabled:(BOOL)scaleEnabled;
```

- 动态课件 API。

```
1. /// 动态课件加载成功  
2. @property (nonatomic, readonly) BOOL  
    webPPTLoadSuccess;  
3.  
4. /// 动态课件加载失败时,是否要切静态课件  
5. @property (nonatomic, copy, nullable) void  
    (^shouldSwitchNativePPTBlock)(NSString  
    *_Nullable documentID, void (^callback)(BOOL  
    shouldSwitch));  
6.  
7. /// 动态课件翻页指示图标  
8. @property (nonatomic) UIImage  
    *prevPageIndicatorImage,  
    *nextPageIndicatorImage;  
9.  
10. /// 动态课件手势触发的翻页是否能够向前翻页, 默认能  
    翻页  
11. /// 不能滑动翻页时, 参考 `scrollEnabled`, 也不能手  
    势翻页  
12. @property (nonatomic) BOOL  
    pageGestureCanStepForward;
```

```
13.  
14. /// 动态课件手势触发的翻页是否能够向后翻页, 默认能  
    翻页  
15. /// 不能滑动翻页时, 参考 `scrollEnabled`, 也不能手  
    势翻页  
16. @property (nonatomic) BOOL  
    pageGestureCanStepBackward;  
17.  
18. /// 动态课件交互状态, 默认不可交互  
19. /// #discussion 不能滑动翻页时, 参考  
    `scrollEnabled`, 也无法交互  
20. @property (nonatomic, readonly) BOOL  
    webPPTInteractable;  
21. - (void)updateWebPPTInteractable:  
    (BOOL)interactable;  
22.  
23. /// 是否允许在可以滑动翻页时翻页动态课件, 默认可  
    以,  
24. /// #discussion 设置为 NO 时, 只能操作课件中动效,  
    不能滑动翻页  
25. /// #discussion 即使设置为 NO, 如果课件中的动效会  
    触发翻页, 也将会成功翻页  
26. @property (nonatomic, readonly) BOOL  
    enableWebPPTChangePage;  
27. - (void)updateEnableWebPPTChangePage:  
    (BOOL)enableWebPPTChangePage;
```

4. 上传、添加课件, 课件管理

- 上传图片格式课件。

可以通过相册、拍照以及文件管理中的文件 URL 上传图片到教室
中。使用 `uploadImageFile:progress:finish:` 上传完成
后, 然后通过 `addDocument` 添加到教室文档中。

```
1. bjl_weakify(self);
2. [self.room.documentVM uploadImageFile:fileURL
3.           progress:^(CGFloat progress){
4.     bjl_strongify(self);
5.     // 显示进度
6.     self.progressView.progress = progress;
7. }
8.           finish:^(BJLDocument *
9. _Nullable document, BJLError * _Nullable error) {
10.    bjl_strongify(self)
11.    if(document){
12.        // 添加课件到教室中
13.        [self.room.documentVM
14.         addDocument:document];
15.    }
16. }
17. }];
```

- 上传更多文件格式课件。

上传 PPT、PDF 等文档不同于上传图片作为课件，需要在上传完成后请求文档服务器的转码处理，在等待转码完成后才能添加到教室中。

```
1. // 上传
2. [self.room.documentVM uploadFile:url
3.           mimeType:mimeType
4.           fileName:name
5.           isAnimated:isAnimated
6.           progress:^(CGFloat progress) {
7.     //bjl_strongify(self);
8.     NSLog(@"%@", progress);
9. }
```

```
10.         finish:^(BJLDocument *_Nullable
11.             document, BJLError *_Nullable error) {
12.             bjl_strongify(self);
13.             if (error) {
14.                 return;
15.             }
16.             if (image) {
17.                 // 图片不需要转码，直接添加到教室
18.                 [self.room.documentVM
19.                  addDocument:document];
20.             }
21.             else {
22.                 // 开始轮询转码进度
23.                 [self startPollTimer];
24.             }
25.         }];
26.         [self.room.documentVM
27.             requestTranscodingProgressWithFileIDList:array
28.             completion:^(NSArray<BJLDocumentTranscodeMod
29.                         *> *_Nullable transcodeModelArray, BJLError *
30.                         _Nullable error) {
31.                 bjl_strongify(self);
32.                 if (error) {
33.                     return;
34.                 }
35.                 for (BJLDocumentTranscodeModel *model in
36.                     transcodeModelArray) {
37.                     if (model.progress >= 100) {
38.                         // 转码完成，请求转码完成之后的新文档
39.                         [self requestDocumentList];
40.                     }
41.                 }
42.             }
43.         ];
44.     }
45. }
```

```
39.         NSLog(@"progress", progress);
40.     }
41. }
42. }];
43. // 添加文档
44. [self.room.documentVM
    requestDocumentListWithFileIDList:@[fileID]
45.
    completion:^(NSArray<BJLDocument *> * _Nullable documentArray, BJLError * _Nullable error) {
46.     if (error) {
47.         return;
48.     }
49.     for (BJLDocument *document in
        documentArray) {
50.         // 添加文档
51.         [self.room.documentVM
            addDocument:document];
52.     }
53. }];

```

- 删除课件。

```
1. /**
2.  删除课件
3. #discussion 删除成功将调用 `BJLDocumentVM` 的
   `didDeleteDocument:`
4. #param documentID 课件 ID
5. #return BJLError:
6. BJLErrorCode_invalidArguments 错误参数
7. */
8. [self.room.documentVM
    deleteDocumentWithID:documentID];
```

- 监听课件变化。

通过监听 `self.room.documentVM` 的属性变化及方法调用来实现。

```
1. // 以监听所有课件 allDocuments 的变化为例
2. bjl_weakify(self);
3. [self
   bjl_kvo:BJLMakeProperty(self.room.documentVM,
   allDocuments)
4.   observer:^BOOL(NSArray<BJLDocument *> * _Nullable value, id _Nullable oldValue,
   BJLPropertyChange * _Nullable change) {
5.     bjl_strongify(self);
6.     // 更新数据源及相关界面控件
7.     self.allDocuments = [value mutableCopy];
8.     [self.tableView reloadData];
9.     [self updateViewsForDataCount];
10.    return YES;
11.  }];
12. }
```

```
1. // 监听添加课件的通知
2. [self
   bjl_observe:BJLMakeMethod(self.room.documentVM
   didAddDocument:)
3.   observer:^(BJLDocument *document) {
4.     // tableView的数据源及相关界面已经通过监听
   allDocuments的变化进行更新
5.     if(document){
6.       NSLog(@"document: %@", added",
   document);
7.     }
8.     return YES;
9.  }];
10. // 删除课件
```

```

11. [self
    bjL_observe:BJLMakeMethod(self.room.documentVM
        didDeleteDocument:)
12.     observer:^BOOL(BJLDocument *document)
    {
13.     // bjL_strongify(self);
14.     if(document){
15.         NSLog(@"%@", @"document: %@", delete,
            document);
16.     }
17.     return YES;
18. }];

```

- 监听本地课件页码。

学生翻页不会影响到远程课件翻页，如果要禁止学生本地翻页，需要在上层限制。对于单实例的文档，`BJLDocumentVM` 的 `currentSlidePage` 表示整个教室的当前页，随老师 / 助教翻动课件而改变。因为学生可以回顾之前的课件，所以它不一定是本地的当前页，不能用于显示本地课件页码，SDK 限制了学生不能翻页超过远程的课件的当前页。本地课件页码通过监听 `BJLRoom` 的 `slideshowViewController` 的 `pageIndex` 获得。对于多实例的文档，只能获得本地文档的当前页 `self.documentViewController.pageIndex`，学生翻页能够超过远程课件的当前页。

```

1. [self
    bjL_kvo:BJLMakeProperty(self.room.slideshowViewController
        pageIndex)
2.     observer:^BOOL(id _Nullable value, id
        _Nullable oldValue, BJLPropertyChange *
        _Nullable change) {
3.         NSLog(@"%@", @"localPage: %d", [value
            integerValue]);
4.     return YES;
}

```

5. }];

- 课件授权。

通过监听 `self.room.documentVM` 的
`authorizedPPTUserNumbers` 获取课件授权用户。

```
1. // 监听授权用户列表
2. [self
   bjl_kvo:BJLMakeProperty(self.room.documentVM,
   authorizedPPTUserNumbers)
3.   observer:^BOOL(id _Nullable value, id
   _Nullable oldValue, BJLPropertyChange *
   _Nullable change) {
4.     bjl_strongify(self);
5.     NSLog(@"%@", authorizedPPTUserNumbers);
6.     return YES;
7.   }];
8. // 给某个用户授权
9. BJLError *error = [self.room.documentVM
   updateStudentPPTAuthorized:authorized
   userNumber:user.number];
```

- 课件备注

```
1. /**
2. 获取文档备注信息
3. #param documentID 文档ID, 当请求的文档为白板
   时, 返回空
4. #param completion 返回 remarkInfo 备注信息
5. #return task
6. */
7. - (nullable NSURLSessionDataTask
   *)requestCompleteRemarkInfoWithDocumentID:
```

```
(NSString *)documentID  
8.  
completion:(void (^)(NSDictionary *_Nullable  
remarkInfo, BJLError *_Nullable error))completion;
```

- 是否禁止学生本地翻页。

通过设置 `BJLOnlineUserVM` 的 `forbidStudentChangePPT` 属性决定是否允许教室内学生本地翻页，仅限大班课类型。

- 多白板。
 - 大班课支持多白板功能，老师或助教可动态添加、删除白板；
 - 单页白板实例为 `BJLSlidePage` 类型，`documentID` 均为 `0`，`slidePageIndex` 表示序号。
 - 通过 `BJLDocumentVM` 的 `addWhiteboardPage` 添加一页白板，`deleteWhiteboardPageWithIndex:` 方法删除对应 `slidePageIndex` 的白板页；
 - 通过监听 `BJLDocument` 的 `didAddWhiteboardPage:` 方法获取白板页添加通知，监听 `didDeleteWhiteboardPageWithIndex:` 方法获取白板页删除通知。

```
1. /**  
2. 添加白板  
3. #discussion 最多同时存在 10 页白板  
4. #return BJLError  
5. */  
6. - (nullable BJLError *)addWhiteboardPage;  
7.  
8. /**  
9. 添加白板通知  
10. #discussion 同时更新 `allDocuments`  
11. #param pageIndex 白板页码
```

```
12. */
13. - (BJLObservable)didAddWhiteboardPage:
    (NSInteger)pageIndex;
```

```
1. /**
2. 删除白板
3. #param pageIndex 白板页码, 使用 BJLSlidePage
   的 `slidePageIndex`
4. #return BJLError
5. */
6. - (nullable BJLError
   *)deleteWhiteboardPageWithIndex:
    (NSInteger)pageIndex;
7.
8. /**
9. 删除白板通知
10. #discussion 同时更新 `allDocuments`
11. #param pageIndex 白板页码
12. */
13. -
    (BJLObservable)didDeleteWhiteboardPageWithIndex
    (NSInteger)pageIndex;
```

- 专业小班课课件加载使用原图

```
1. /** 当前ppt课件品质 YES: 原图, NO: 流畅, 默认为
   NO */
2. @property (nonatomic, assign, readonly) BOOL
   pptQualityIsOriginal;
3. /**
4. 切换课件品质
5. #discussion 仅对专业小班课静态PPT有效
6. #param isOriginal YES: 原图, NO: 流畅
7. */
```

```
8. - (nullable NSError *)pptQualityChange:  
    (BOOL)isOriginal;
```

- 大班课授权 H5 课件操作权限。

```
1. /**  
2. 给大班课所有学生授权操作h5课件权限  
3. #param authorized 是否可以操作课件  
4. */  
5. - (nullable BJLError  
*)updateAllStudentH5PPTAuthorized:  
    (BOOL)authorized;
```

5. 小黑板管理

- 收到小黑板发布。

```
1. /**  
2. 发布小黑板的通知  
3. #discussion 直播间内所有用户不区分角色都会收到此  
信令  
4. #param writingBoard 小黑板信息, 此时  
writingBoard 的 `submittedUsers` 和  
`participatedUsers` 为空, 此处不做统计  
5. */  
6. [self  
bjl_observe:BJLRequestMethod(self.room.documentVM  
didPublishWritingBoard:)  
7.     filter:^ BOOL(BJLWritingBoard  
*writingBoard) {  
8.         return (writingBoard  
9.             && writingBoard.boardID.length);  
10.    }  
11. }  
12. }  
13. }  
14. }  
15. }  
16. }  
17. }  
18. }  
19. }  
20. }  
21. }  
22. }  
23. }  
24. }  
25. }  
26. }  
27. }  
28. }  
29. }  
30. }  
31. }  
32. }  
33. }  
34. }  
35. }  
36. }  
37. }  
38. }  
39. }  
40. }  
41. }  
42. }  
43. }  
44. }  
45. }  
46. }  
47. }  
48. }  
49. }  
50. }  
51. }  
52. }  
53. }  
54. }  
55. }  
56. }  
57. }  
58. }  
59. }  
60. }  
61. }  
62. }  
63. }  
64. }  
65. }  
66. }  
67. }  
68. }  
69. }  
70. }  
71. }  
72. }  
73. }  
74. }  
75. }  
76. }  
77. }  
78. }  
79. }  
80. }  
81. }  
82. }  
83. }  
84. }  
85. }  
86. }  
87. }  
88. }  
89. }  
90. }  
91. }  
92. }  
93. }  
94. }  
95. }  
96. }  
97. }  
98. }  
99. }  
100. }  
101. }  
102. }  
103. }  
104. }  
105. }  
106. }  
107. }  
108. }  
109. }  
110. }  
111. }  
112. }  
113. }  
114. }  
115. }  
116. }  
117. }  
118. }  
119. }  
120. }  
121. }  
122. }  
123. }  
124. }  
125. }  
126. }  
127. }  
128. }  
129. }  
130. }  
131. }  
132. }  
133. }  
134. }  
135. }  
136. }  
137. }  
138. }  
139. }  
140. }  
141. }  
142. }  
143. }  
144. }  
145. }  
146. }  
147. }  
148. }  
149. }  
150. }  
151. }  
152. }  
153. }  
154. }  
155. }  
156. }  
157. }  
158. }  
159. }  
160. }  
161. }  
162. }  
163. }  
164. }  
165. }  
166. }  
167. }  
168. }  
169. }  
170. }  
171. }  
172. }  
173. }  
174. }  
175. }  
176. }  
177. }  
178. }  
179. }  
180. }  
181. }  
182. }  
183. }  
184. }  
185. }  
186. }  
187. }  
188. }  
189. }  
190. }  
191. }  
192. }  
193. }  
194. }  
195. }  
196. }  
197. }  
198. }  
199. }  
200. }  
201. }  
202. }  
203. }  
204. }  
205. }  
206. }  
207. }  
208. }  
209. }  
210. }  
211. }  
212. }  
213. }  
214. }  
215. }  
216. }  
217. }  
218. }  
219. }  
220. }  
221. }  
222. }  
223. }  
224. }  
225. }  
226. }  
227. }  
228. }  
229. }  
230. }  
231. }  
232. }  
233. }  
234. }  
235. }  
236. }  
237. }  
238. }  
239. }  
240. }  
241. }  
242. }  
243. }  
244. }  
245. }  
246. }  
247. }  
248. }  
249. }  
250. }  
251. }  
252. }  
253. }  
254. }  
255. }  
256. }  
257. }  
258. }  
259. }  
260. }  
261. }  
262. }  
263. }  
264. }  
265. }  
266. }  
267. }  
268. }  
269. }  
270. }  
271. }  
272. }  
273. }  
274. }  
275. }  
276. }  
277. }  
278. }  
279. }  
280. }  
281. }  
282. }  
283. }  
284. }  
285. }  
286. }  
287. }  
288. }  
289. }  
290. }  
291. }  
292. }  
293. }  
294. }  
295. }  
296. }  
297. }  
298. }  
299. }  
300. }  
301. }  
302. }  
303. }  
304. }  
305. }  
306. }  
307. }  
308. }  
309. }  
310. }  
311. }  
312. }  
313. }  
314. }  
315. }  
316. }  
317. }  
318. }  
319. }  
320. }  
321. }  
322. }  
323. }  
324. }  
325. }  
326. }  
327. }  
328. }  
329. }  
330. }  
331. }  
332. }  
333. }  
334. }  
335. }  
336. }  
337. }  
338. }  
339. }  
340. }  
341. }  
342. }  
343. }  
344. }  
345. }  
346. }  
347. }  
348. }  
349. }  
350. }  
351. }  
352. }  
353. }  
354. }  
355. }  
356. }  
357. }  
358. }  
359. }  
360. }  
361. }  
362. }  
363. }  
364. }  
365. }  
366. }  
367. }  
368. }  
369. }  
370. }  
371. }  
372. }  
373. }  
374. }  
375. }  
376. }  
377. }  
378. }  
379. }  
380. }  
381. }  
382. }  
383. }  
384. }  
385. }  
386. }  
387. }  
388. }  
389. }  
390. }  
391. }  
392. }  
393. }  
394. }  
395. }  
396. }  
397. }  
398. }  
399. }  
400. }  
401. }  
402. }  
403. }  
404. }  
405. }  
406. }  
407. }  
408. }  
409. }  
410. }  
411. }  
412. }  
413. }  
414. }  
415. }  
416. }  
417. }  
418. }  
419. }  
420. }  
421. }  
422. }  
423. }  
424. }  
425. }  
426. }  
427. }  
428. }  
429. }  
430. }  
431. }  
432. }  
433. }  
434. }  
435. }  
436. }  
437. }  
438. }  
439. }  
440. }  
441. }  
442. }  
443. }  
444. }  
445. }  
446. }  
447. }  
448. }  
449. }  
450. }  
451. }  
452. }  
453. }  
454. }  
455. }  
456. }  
457. }  
458. }  
459. }  
460. }  
461. }  
462. }  
463. }  
464. }  
465. }  
466. }  
467. }  
468. }  
469. }  
470. }  
471. }  
472. }  
473. }  
474. }  
475. }  
476. }  
477. }  
478. }  
479. }  
480. }  
481. }  
482. }  
483. }  
484. }  
485. }  
486. }  
487. }  
488. }  
489. }  
490. }  
491. }  
492. }  
493. }  
494. }  
495. }  
496. }  
497. }  
498. }  
499. }  
500. }  
501. }  
502. }  
503. }  
504. }  
505. }  
506. }  
507. }  
508. }  
509. }  
510. }  
511. }  
512. }  
513. }  
514. }  
515. }  
516. }  
517. }  
518. }  
519. }  
520. }  
521. }  
522. }  
523. }  
524. }  
525. }  
526. }  
527. }  
528. }  
529. }  
530. }  
531. }  
532. }  
533. }  
534. }  
535. }  
536. }  
537. }  
538. }  
539. }  
540. }  
541. }  
542. }  
543. }  
544. }  
545. }  
546. }  
547. }  
548. }  
549. }  
550. }  
551. }  
552. }  
553. }  
554. }  
555. }  
556. }  
557. }  
558. }  
559. }  
560. }  
561. }  
562. }  
563. }  
564. }  
565. }  
566. }  
567. }  
568. }  
569. }  
570. }  
571. }  
572. }  
573. }  
574. }  
575. }  
576. }  
577. }  
578. }  
579. }  
580. }  
581. }  
582. }  
583. }  
584. }  
585. }  
586. }  
587. }  
588. }  
589. }  
590. }  
591. }  
592. }  
593. }  
594. }  
595. }  
596. }  
597. }  
598. }  
599. }  
600. }  
601. }  
602. }  
603. }  
604. }  
605. }  
606. }  
607. }  
608. }  
609. }  
610. }  
611. }  
612. }  
613. }  
614. }  
615. }  
616. }  
617. }  
618. }  
619. }  
620. }  
621. }  
622. }  
623. }  
624. }  
625. }  
626. }  
627. }  
628. }  
629. }  
630. }  
631. }  
632. }  
633. }  
634. }  
635. }  
636. }  
637. }  
638. }  
639. }  
640. }  
641. }  
642. }  
643. }  
644. }  
645. }  
646. }  
647. }  
648. }  
649. }  
650. }  
651. }  
652. }  
653. }  
654. }  
655. }  
656. }  
657. }  
658. }  
659. }  
660. }  
661. }  
662. }  
663. }  
664. }  
665. }  
666. }  
667. }  
668. }  
669. }  
670. }  
671. }  
672. }  
673. }  
674. }  
675. }  
676. }  
677. }  
678. }  
679. }  
680. }  
681. }  
682. }  
683. }  
684. }  
685. }  
686. }  
687. }  
688. }  
689. }  
690. }  
691. }  
692. }  
693. }  
694. }  
695. }  
696. }  
697. }  
698. }  
699. }  
700. }  
701. }  
702. }  
703. }  
704. }  
705. }  
706. }  
707. }  
708. }  
709. }  
710. }  
711. }  
712. }  
713. }  
714. }  
715. }  
716. }  
717. }  
718. }  
719. }  
720. }  
721. }  
722. }  
723. }  
724. }  
725. }  
726. }  
727. }  
728. }  
729. }  
730. }  
731. }  
732. }  
733. }  
734. }  
735. }  
736. }  
737. }  
738. }  
739. }  
740. }  
741. }  
742. }  
743. }  
744. }  
745. }  
746. }  
747. }  
748. }  
749. }  
750. }  
751. }  
752. }  
753. }  
754. }  
755. }  
756. }  
757. }  
758. }  
759. }  
760. }  
761. }  
762. }  
763. }  
764. }  
765. }  
766. }  
767. }  
768. }  
769. }  
770. }  
771. }  
772. }  
773. }  
774. }  
775. }  
776. }  
777. }  
778. }  
779. }  
780. }  
781. }  
782. }  
783. }  
784. }  
785. }  
786. }  
787. }  
788. }  
789. }  
790. }  
791. }  
792. }  
793. }  
794. }  
795. }  
796. }  
797. }  
798. }  
799. }  
800. }  
801. }  
802. }  
803. }  
804. }  
805. }  
806. }  
807. }  
808. }  
809. }  
810. }  
811. }  
812. }  
813. }  
814. }  
815. }  
816. }  
817. }  
818. }  
819. }  
820. }  
821. }  
822. }  
823. }  
824. }  
825. }  
826. }  
827. }  
828. }  
829. }  
830. }  
831. }  
832. }  
833. }  
834. }  
835. }  
836. }  
837. }  
838. }  
839. }  
840. }  
841. }  
842. }  
843. }  
844. }  
845. }  
846. }  
847. }  
848. }  
849. }  
850. }  
851. }  
852. }  
853. }  
854. }  
855. }  
856. }  
857. }  
858. }  
859. }  
860. }  
861. }  
862. }  
863. }  
864. }  
865. }  
866. }  
867. }  
868. }  
869. }  
870. }  
871. }  
872. }  
873. }  
874. }  
875. }  
876. }  
877. }  
878. }  
879. }  
880. }  
881. }  
882. }  
883. }  
884. }  
885. }  
886. }  
887. }  
888. }  
889. }  
890. }  
891. }  
892. }  
893. }  
894. }  
895. }  
896. }  
897. }  
898. }  
899. }  
900. }  
901. }  
902. }  
903. }  
904. }  
905. }  
906. }  
907. }  
908. }  
909. }  
910. }  
911. }  
912. }  
913. }  
914. }  
915. }  
916. }  
917. }  
918. }  
919. }  
920. }  
921. }  
922. }  
923. }  
924. }  
925. }  
926. }  
927. }  
928. }  
929. }  
930. }  
931. }  
932. }  
933. }  
934. }  
935. }  
936. }  
937. }  
938. }  
939. }  
940. }  
941. }  
942. }  
943. }  
944. }  
945. }  
946. }  
947. }  
948. }  
949. }  
950. }  
951. }  
952. }  
953. }  
954. }  
955. }  
956. }  
957. }  
958. }  
959. }  
960. }  
961. }  
962. }  
963. }  
964. }  
965. }  
966. }  
967. }  
968. }  
969. }  
970. }  
971. }  
972. }  
973. }  
974. }  
975. }  
976. }  
977. }  
978. }  
979. }  
980. }  
981. }  
982. }  
983. }  
984. }  
985. }  
986. }  
987. }  
988. }  
989. }  
990. }  
991. }  
992. }  
993. }  
994. }  
995. }  
996. }  
997. }  
998. }  
999. }  
1000. }
```

```
11.     observer:^BOOL(BJLWritingBoard  
12.         *writingBoard) {  
13.             bjl_strongify(self);  
14.             // 判断是否为学生/助教  
15.             if (writingBoard.operate ==  
16.                 BJLWritingBoardPublishOperate_begin) {  
17.                 }  
18.             else if (writingBoard.operate ==  
19.                 BJLWritingBoardPublishOperate_revoke  
20.                     && self.writingBoardViewController) {  
21.                 }  
22.             else if (writingBoard.operate ==  
23.                 BJLWritingBoardPublishOperate_end  
24.                     && self.writingBoardViewController) {  
25.                 }  
26.             return YES;  
27.         }];
```

- 根据发布的小黑板信息创建小黑板。

```
1. /**  
2.  根据 writingBoard 创建对应的小黑板控制器  
3.  #param writingBoard 小黑板信息  
4.  #return 对应小黑板的视图控制器  
5. */  
6. self.writingBoardViewController =  
    [room.documentVM  
    writingBoardViewControllerWithWritingBoard:writingBoard];
```

- 参与小黑板作答。

```
1. // 参与作答  
2. [self.room.documentVM  
    participateWritingBoard:self.writingBoard.boardID];  
3. // 开启小黑板画笔
```

```
4. [self.room.drawingVM  
    updateWritingBoardEnabled:status ==  
    BJLcWriteBoardStatus_studentEdit];  
5. // 提交小黑板内容  
6. [self.room.documentVM  
    submitWritingBoard:self.writingBoard.boardID];
```

- 课件内的媒体文件

进入教室后SDK内部会自动加载课件媒体文件，外界只需要监听 `didLoadMediaFiles`，处理返回数据。

```
1.  
2. /**  
3. 加载当前直播间媒体文件  
4. #discussion 直播间内的媒体文件增加、删除都会触发  
5. #param mediaFiles 媒体文件  
6. */  
7. - (BJLObservable)didLoadMediaFiles:  
    (NSArray<BJLMediaFile *> *)mediaFiles;  
8.  
9. /**  
10. 删除媒体课件  
11. #param fileID 课件 ID  
12. */  
13. - (void)requestDeleteMediaFileWithID:(NSString  
    *)fileID;
```

6. 云盘

云盘功能是基于机构、老师的账号体系下，便捷地管理课件、媒体等文件的系统。

基于云盘，可以快速关联以往上传过的课件，减少重复上传、关联课件的操作。云盘由 `BJLCloudDiskVM` 管理，云盘功能需

- 获取云盘文件列表。

```
1. /**
2.  请求云盘文件列表
3. #param targetFinderID 请求目录的finderID，为空
   表示请求根目录列表
4. #param page 从第一页开始
5. */
6. - (nullable NSURLSessionDataTask
   *)requestCloudListWithTargetFinderID:(nullable
   NSString *)targetFinderID
7.                               page:
   (NSUInteger)page
8.                               pagesize:
   (NSUInteger)pagesize
9.                               completion:
   (nullable void (^)(NSArray <BJLCloudFile *> *
   _Nullable documentList, BJLError * _Nullable
   error))completion;
```

- 上传、删除云盘文件。上传过程的示例代码可以参考 [课件上传](#) 部分。

```
1. /**
2.  上传云盘文档
3. #discussion 图片作为文档上传没有转码过程，其余的
   类型的文档需要等待转码过程
4. #discussion 转码服务器主动进行，不需要调用其他接
   口
5. #param fileURL 文档路径
6. #param mimeType 文档类型
7. #param fileName 文件名
8. #param isAnimated 动态文件 or 静态文件
```

```

9. #param progress 进度 0.0 ~ 1.0
10. #param finish 结束回调
11. - homework 非 nil 即为成功
12. - document 非 nil 即为成功, 可按需调用
`BJLDocumentVM` 的 `addDocument:` 同步到教室内
文件打开
13. - error 错误
14. #return task
15. */
16. - (nullable NSURLSessionUploadTask
*)uploadCloudFile:(NSURL *)fileURL
               mimeType:(NSString
*)mimeType
17.               fileName:(NSString
*)fileName
18.               isAnimated:
               (BOOL)isAnimated
19.               progress:(nullable
void (^)(CGFloat progress))progress
20.               finish:(void (^)
(BJLCloudFile * _Nullable cloudFile, BJLError *
_Nullable error))finish;
21.
22.
23. /** 请求删除云盘文件, 仅支持删除文件, 不支持删除文
件夹 */
24. - (nullable NSURLSessionDataTask
*)requestDeleteCloudFileWithFileID:(NSString
*)fileID
               completion:
               (nullable void (^)(BOOL success, BJLError *
_Nullable error))completion;

```

- 搜索云盘文件。

```
1. /**
```

```
2. 请求搜索云盘文件
3. #param keyword 搜索关键词
4. #param targetFinderID 搜索目录的finderID，为空
   表示在根目录搜索
5. #param page 从第一页开始
6. */
7. - (nullable NSURLSessionDataTask
   *)requestSearchCloudFileListWithKeyword:
   (NSString *)keyword
8.
   targetFinderID:(nullable NSString *)targetFinderID
9.           page:
   (NSUInteger)page
10.
  pagesize:(NSUInteger)pagesize
11.
  completion:(nullable void (^)(NSString *keyword,
   NSArray <BJLCloudFile *> *_Nullable
   documentList, BJLError *_Nullable
   error))completion;
```

7. 作业

作业系统是教室内老师发布作业，学生需要上传作业文件时使用
的功能。

不同于课堂中的测验、答题，一般用于需要较长时间才能完成的
课后作业的提交。

教室内的作业由 `BJLHomeworkVM` 管理。

- 获取教室作业列表。

```
1. /** 教室作业列表 */
2. @property (nonatomic, readonly, copy, nullable)
   NSArray<BJLHomework *> *allHomeworks;
```

```
3.  
4. // example: 监听到作业列表更新，加载作业列表  
5. [self  
    bjl_observe:BJLRequestMethod(self.room.homeworkVM  
        allHomeworksDidOverwrite:  
6.     observer:^BOOL(NSArray<BJLHomework  
        *> *homeworks) {  
7.         bjl_strongify(self);  
8.         [self  
            loadAllRemoteHomeworks:self.room.homeworkVM.a  
9.         return YES;  
10.    }];
```

```
1. /** 分页加载更多作业 */  
2. - (nullable BJLError  
    *)loadMoreHomeworksWithCount:  
    (NSInteger)count;  
3. /*  
4. 学生是否被允许上传作业通知  
5. */  
6. -  
    (BJLObservable)didReceiveAllowStudentUploadHom  
    (BOOL)allow;
```

- 上传作业权限控制。可以根据
`allStudentsSupportHomework` 判断教室内是否所有学
生都支持作业功能，通过
`allowStudentUploadHomework` 获取教室内是否支持学
生上传作业，默认支持。

```
1. /**  
2. 学生能否上传作业  
3. #discussion 设置成功将回调 `BJLHomeworkVM` 的  
    `didReceiveAllowStudentUploadHomework:`
```

```
4. #return BJLError:  
5. BJLErrorCode_invalidUserRole 仅支持老师/助教身份  
操作  
6. */  
7. - (nullable BJLError  
*)requestAllowStudentUploadHomework:  
(BOOL)allow;
```

- 上传、下载、删除作业。上传过程的示例代码可以参考[课件上传](#)部分，通过接口上传后，等待转码，完成后添加到作业区。

```
1. /**  
2. 上传作业文档  
3. #discussion 图片作为文档上传没有转码过程，其余的  
作业等文档需要等待转码过程  
4. #discussion 转码服务器主动进行，不需要调用其他接  
口  
5. #param fileURL 本地文档路径  
6. #param mimeType 文档类型  
7. #param fileName 文件名  
8. #param progress 进度 0.0 ~ 1.0  
9. #param finish 结束回调  
10. - homework 非 nil 即为成功，用于  
`addHomework:`  
11. - document 非 nil 即为成功，可按需调用  
`BJLDocumentVM` 的 `addDocument:` 同步到教室内  
文件打开  
12. - error 错误  
13. #return task  
14. */  
15. - (NSURLSessionUploadTask  
*)uploadHomeworkFile:(NSURL *)fileURL  
16. mimeType:(NSString  
*)mimeType
```

```
17.         fileName:(NSString *)fileName
18.             progress:(nullable void (^)(CGFloat progress))progress
19.                 finish:(void (^)(BJLHomework *_Nullable homework,
BJLDocument *_Nullable document, BJLError *_Nullable error))finish;
20.
21. /**
22. 添加作业
23. #discussion 添加成功将调用 `BJLHomeworkVM` 的
`didAddHomeworks:`
24. #param homework 作业
25. #return BJLError:
26. BJLErrorCode_invalidArguments 错误参数
27. */
28. - (nullable BJLError *)addHomework:
(BJLHomework *)homework;
```

```
1. // example: 下载作业
2. bjl_weakify(self);
3. [self.room.homeworkVM
requestDownloadURLWithHomeworkID:file.remoteH
4.                                     completion:^(BOOL
success, BJLError *_Nullable error, NSString *
_Nonnull downloadUrl) {
5.     bjl_strongify(self);
6.     if (!success) {
7.
self.showErrorMessageCallback(error.localizedDescription
?: error.localizedDescription);
8. }
9. else {
```

```
10.     NSString *itemIdentifier = [self
    itemIdentifierWithHomeworkID:file.remoteHomewo
11.     [self.manager
        addDownloadItemWithIdentifier:itemIdentifier
        itemClass:[BJLHomeworkDownloadItem class]
        setting:^(__kindof BJLHomeworkDownloadItem *
        _Nonnull item) {
12.         bjl_strongify(self);
13.         item.sourceURL = [NSURL
            URLWithString:downloadUrl];
14.         item.homework = file.remoteHomework;
15.         item.downloadTimeInterval = [NSDate
            timeIntervalSinceReferenceDate];
16.         item.roomName =
            self.room.roomInfo.title;
17.         item.roomID = self.room.roomInfo.ID;
18.     }];
19. }
20. }];
```

```
1. /**
2.  删除作业
3. #discussion 添加成功将调用 `BJLHomeworkVM` 的
   `didDeleteHomework:`
4. #param homeworkID 作业ID
5. */
6. - (nullable NSURLSessionDataTask
*)requestDeleteHomeworkWithHomeworkID:
   (nullable NSString *)homeworkID
7.                                         completion:
   (nullable void (^)(BOOL success, BJLError *
   _Nullable error))completion;
```

- 搜索作业。

```
1. /**
2. 根据关键字`keyword`搜索
3. #discussion 搜索范围: 作业名称和作业上传者用户名
4. #param lastHomework 上次分页请求的最后一个作业数据, 传nil表示首次拉取分页数据
5. */
6. - (nullable BJLError
7. *)searchHomeworksWithKeyword:(NSString
8. *)keyword
9.           lastHomework:(nullable
10. BJLHomework *)lastHomework
11.           count:
12.       (NSInteger)count;
```

画笔

老师和处于发言状态的学生可以在白板和 PPT 上添加、清除画笔，对于单文档实例，**操作画笔时用户的当前课件页面必须与老师保持一致**，对于多文档实例，操作画笔的当前页面可以和远程页面不一致。画笔管理使用 `BJLDrawingVM`。2.x的画笔支持多种画笔模式，参考 `BJLBrushOperateMode`，多种画笔形状，参考 `BJLDrawingShapeType`。

1. 画笔控制

- 显示画笔视图。

目前画笔与课件共用同一个视图。

单文档: `self.room.slideshowViewController.view` ;

多文档: 通过 `BJLDocumentVM` 以及 `documentID` 获取
`- (UIViewController<BJLSlideshowUI>`
`*)documentViewControllerWithID:(NSString`

`*)documentID;`。因此所有能使用画笔的视图必须是文档视图。

- 开启、关闭画笔。

```
1. /**
2.  开启、关闭画笔
3.  #param drawingEnabled YES: 开启, NO: 关闭
4.  #return BJLError:
5.  #discussion BJLErrorCode_invalidCalling 错误调用, 当前用户是学生、`drawingGranted` 是 NO
6.  #discussion 开启画笔时, 单文档实例情况下如果本地页数与服务端页数不同步则无法绘制
7.  #discussion `drawingGranted` 是 YES 时才可以开启, `drawingGranted` 是 NO 时会被自动关闭
8. */
9. BJLError *error = [self.room.drawingVM
updateDrawingEnabled:YES];
```

```
1. /** 小黑板画笔开关状态 */
2. @property (nonatomic, readonly) BOOL
writingBoardEnabled;
3.
4. /**
5.  开启/关闭小黑板画笔
6.  #param writingBoardEnabled 是否开启小黑板画笔
7. */
8. - (void)updateWritingBoardEnabled:
(BOOL)writingBoardEnabled;
```

- 画笔授权。

```
1. /**
2.  老师、助教: 给学生授权/取消画笔
3.  #param granted 是否授权
```

```
3. #param user 授权操作的对象用户
4. #return BJLError:
5. BJLErrorCode_invalidUserRole 当前用户不是老师或
   者助教
6. BJLErrorCode_invalidArguments 参数错误
7. */
8. BJLError *error = [self.room.drawingVM
   updateDrawingGranted:grant
   userNumber:user.number color:nil];
```

```
1. // 所有被授权使用画笔的学生编号
2. @property (nonatomic, readonly, copy)
   NSArray<NSString *>
   *drawingGrantedUserNumbers;
```

- 画笔设置。

```
1. /**
2.  更新画笔操作模式
3.
4. #param operateMode 操作模式, 参考
   `BJLBrushOperateMode`
5. #return BJLError:
6. #discussion BJLErrorCode_invalidCalling
   drawingEnabled 是 NO
7. */
8. BJLError *error = [self.room.drawingVM
   updateBrushOperateMode:operateMode];
```

```
1. // 切换画笔图形, 参考 `BJLDrawingShapeType`、
2. self.room.drawingVM.drawingShapeType =
   BJLDrawingShapeType_rectangle; // 画矩形
```

```
1. /** 画笔边框颜色, 默认 #1795FF */
2. @property (nonatomic, nonnull) NSString
   *strokeColor;
3. /** 画笔边框颜色透明度, 取值范围 0~1, 默认 1.0*/
4. @property (nonatomic) CGFloat strokeAlpha;
5. /** 画笔填充颜色, 默认和画笔颜色一致 */
6. @property (nonatomic, nullable) NSString
   *fillColor;
7. /** 画笔填充颜色透明度, 取值范围 0~1, fillColor 不为
   空时有效, 默认 1.0 */
8. @property (nonatomic) CGFloat fillAlpha;
9. /** doodle 画笔线宽, 默认 4.0 */
10. @property (nonatomic) CGFloat
    doodleStrokeWidth;
11. /** 图形画笔边框线宽, 默认 2.0 */
12. @property (nonatomic) CGFloat
    shapeStrokeWidth;
13. /** 文字画笔字体大小, 默认 20.0 */
14. @property (nonatomic) CGFloat textFontSize;
15. /** 文字画笔是否加粗, 默认 NO */
16. @property (nonatomic) BOOL textBold;
17. /** 文字画笔是否为斜体, 默认 NO */
18. @property (nonatomic) BOOL textItalic;
19. /** 画笔开关状态, 参考 `drawingGranted`、
   `updateDrawingEnabled:` */
20. @property (nonatomic, readonly) BOOL
    drawingEnabled;
21. /** doodle 画笔是否虚线, 默认 NO */
22. @property (nonatomic) BOOL isDottedLine;
23. /** 选中画笔时是否显示归属信息, 默认 NO */
24. @property (nonatomic) BOOL
    showBrushOwnerNameWhenSelected;
```

- 画笔颜色分配。

```
1. /** 画笔分配颜色记录 <hex color, user.number> */
2. @property (nonatomic, readonly, copy)
NSDictionary <NSString *, NSString *>
*drawingGrantedColors;
3.
4. /** 是否不使用分配的画笔颜色 */
5. @property (nonatomic) BOOL
shouldRejectColorGranted;
```

2. 添加图片画笔

可以将图片作为画笔添加到文档的可绘制画笔的画布区域。

```
1. /**
2. 添加图片画笔
3. #param imageURL 图片 url
4. #param relativeFrame 图片相对于画布的 frame, 各
项数值取值范围为 [0.0, 1.0]
5. #param documentID 目标文档 ID
6. #param pageIndex 目标页
7. #param isWritingBoard 是否为小黑板
8. #return BJLErrorCode_invalidCalling
drawingEnabled 是 NO
9. */
10. - (nullable BJLError *)addImageShapeWithURL:
(NSString *)imageURL
11. relativeFrame:
(CGRect)relativeFrame
12. toDocumentID:(NSString
*)documentID
13. pageIndex:
(NSUInteger)pageIndex;
```

3. 激光笔

激光笔作为特殊的画笔，只会跟随使用激光笔的轨迹显示，不留
下画笔形状。

```
1. /**
2. 大班课文档区域是否绘制激光笔
3. 默认绘制，为圆点样式，将不回调激光笔位置等监听，  
    并且设置画笔模式为激光笔时不会主动触发激光笔；  
4. 如果设置为不绘制，需要自行实现激光笔效果
5. 专业小班课不支持设置，默认不自动绘制
6. */
7. @property (nonatomic) BOOL drawsLaserPointer;
```

```
1. /**
2. 激光笔位置移动请求
3. #param location      激光笔目标位置
4. #param documentID    激光笔所在文档的 ID
5. #param pageIndex       激光笔所在文档页码
6. */
7. - (nullable BJLError *)moveLaserPointToLocation:  
    (CGPoint)location
8.           documentID:(nonnull  
    NSString *)documentID
9.           pageIndex:  
    (NSUInteger)pageIndex;
```

```
1. /**
2. 激光笔位置移动监听
3. #param location      激光笔位置
4. #param documentID    激光笔所在文档的 ID
5. #param pageIndex       激光笔所在文档页码
6. */
```

```
7. - (BJLObservable)didLaserPointMoveToLocation:  
    (CGPoint)location  
8.                     documentID:(NSString  
    *)documentID  
9.                     pageIndex:  
    (NSUInteger)pageIndex;
```

4. 画笔位置

使用 `BJLDrawingShapeType_doodle` 类型的画笔涂鸦时，如果配置了画笔轨迹，参考 `BJLFeatureConfig` 的 `enablePaintPoint`，SDK 回调画笔的移动到的位置，可以根据回调显示画笔的标识等图案。

```
1. /**  
2. BJLDrawingShapeType_doodle 类型的画笔移动位置  
回调  
3. #param location      画笔位置, [0, 1] 区间内的坐  
标  
4. #param documentID    画笔所在文档的 ID  
5. #param pageIndex      画笔所在文档页码  
6. #param color          画笔颜色  
7. #param fromCurrentUser 是否是当前用户使用画笔  
8. */  
9. - (BJLObservable)didPaintPointMoveToLocation:  
    (CGPoint)location  
10.                    documentID:(NSString  
    *)documentID  
11.                    pageIndex:  
    (NSUInteger)pageIndex  
12.                    color:(nullable UIColor  
    *)color  
13.                    fromCurrentUser:  
    (BOOL)fromCurrentUser;
```

在 PC 端老师使用画笔时，如果配置了落笔前轨迹，参考 `BJLFeatureConfig` 的 `enablePaintTraceBeforeDraw`，SDK 会回调画笔落笔前的位置，可以根据回调显示画笔的标识图案。

```
1. /**
2. 落笔位置点同步
3. #param location      画笔位置, [0, 1] 区间内的坐
   标
4. #param documentID    画笔所在文档的 ID
5. #param pageIndex       画笔所在文档页码
6. #param color          画笔颜色
7. */
8. - (BJLObservable)didMousePointMoveToLocation:
   (CGPoint)location
9.           documentID:(NSString
   *)documentID
10.          pageIndex:
   (NSUInteger)pageIndex
11.         color:(nullable UIColor
   *)color;
```

5. 手写板

SDK 适配了智写云笔的手写板，具体可以咨询商务人员，使用手写板可以直接将绘制在手写板上的内容显示到画布上。大班课在手写板书写的画笔位置和文档的位置一致，不需要额外设置即可使用，小班课需要设置画笔落笔前回调位置时的第一响应文档。

```
1. /**
2. 搜索手写板 */
3. - (void)scanHandWritingBoard;
4. /**
5. 连接手写板
6. #param handWritingBoard 手写板
```

```
7. */
8. - (void)connectHandWritingBoard:(CBPeripheral
*)handWritingBoard;
9.
10. /**
11. 设置顶部的绘制视图, 仅小班课需要设置, 未设置时为
`BJLDocumentVM` 的 `blackboardViewController`
12. #param 顶部视图, 作为手写板的数据的基准视图
13. */
14. - (void)updateTopDrawingController:
(UIViewController *)topDrawingController;
15.
16. /**
17. 设置手写板画笔落笔前位置的第一响应文档, 仅小班课
需要处理, 大班课始终是教室当前文档和页码才能响应
18. #discussion point 在设置的 topDrawingController
的位置
19. #discussion 返回的控制器必须是满足
BJLSlideshowUI 或者 BJLBlackboardUI 协议的文档控
制器
20. */
21. @property (nonatomic, nullable) UIViewController
*(^requestFirstRespondDocumentCallback)
(CGPoint point);
```



下载为pdf格式