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

蓝牙连接手写板

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。

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

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

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

```objc
/**
 设置静态课件尺寸
 加载课件图片时对图片做等比缩放，长边小于/等于 `imageSize`
 单位为像素，默认初始加载 1080，取值在 `BJLAliIMGMinSize` 到 `BJLAliIMGMaxSize` 之间 (1 ~ 4096)
 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
 只对静态课件生效，参考 `BJLRoom` 的 `disablePPTAnimation`
 */
self.room.slideshowViewController.imageSize = 720;
```

```objc
// 显示课件视图, 将 BJLRoom 的课件视图添加到当前 viewController 的对应视图
[self addChildViewController:self.room.slideshowViewCo
[self.slideshowView addSubview:self.room.slideshowViewController.viev
[self.room.slideshowViewController didMoveToParentViewController:self];

self.room.slideshowViewController.shouldSwitchNat = ^(NSString *_Nullable documentID, void (^_Nonnull callback)(BOOL)) {
        bjl_strongify(self);
```

```
        UIAlertController *alertViewController =
[UIAlertController alertControllerWithTitle:nil

message:@"PPT动画加载失败！\n网络较差建议跳过动
画"

preferredStyle:UIAlertControllerStyleAlert];
        [alertViewController
bjl_addActionWithTitle:BJLLocalizedString(@"重新
加载")

style:UIAlertActionStyleCancel

handler:^(UIAlertAction *_Nonnull action) {
                                        callback(NO);
                                    }];
        [alertViewController
bjl_addActionWithTitle:BJLLocalizedString(@"跳过
动画")

style:UIAlertActionStyleDefault

handler:^(UIAlertAction *_Nonnull action) {
                                        callback(YES);
                                    }];
        bjl_weakify(alertViewController);
        [alertViewController
bjl_kvo:BJLMakeProperty(self.room.slideshowViewC
 webPPTLoadSuccess)

observer:^BJLControlObserving(id _Nullable
value, id _Nullable oldValue, BJLPropertyChange
*_Nullable change) {

bjl_strongify(alertViewController);
```

```
                if (self.room.slideshowViewController.webPPTLoadSuc {
                    [alertViewController bjl_dismissAnimated:YES completion:nil];
                    return NO;
                }
                return YES;
            }];
        if (self.presentedViewController) {
            [self.presentedViewController bjl_dismissAnimated:YES completion:nil];
        }
        [self presentViewController:alertViewController animated:YES completion:nil];
    };
    self.room.slideshowViewController.imageSize = 1080;

self.room.slideshowViewController.view.backgrounc = BJLTheme.roomBackgroundColor; // 课件背景色和直播间整个底色一致

self.room.slideshowViewController.placeholderImag = [UIImage bjl_imageWithColor:BJLTheme.blackboardColor];

self.room.slideshowViewController.whiteboardBack = [UIImage bjl_imageWithColor:BJLTheme.blackboardColor];

self.room.slideshowViewController.prevPageIndicat = [UIImage bjl_imageNamed:@"bjl_sc_ppt_prev"];
```

```
41. 
self.room.slideshowViewController.nextPageIndicato
 = [UIImage
bjl_imageNamed:@"bjl_sc_ppt_next"];
42. 
self.room.slideshowViewController.pageControlButt
 = ({
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.backgroundCo
48.     [button setBackgroundImage:[UIImage
bjl_imageWithColor:[UIColor bjlsc_dimColor]]
49.                       forState:UIControlStateNormal];
50.     [button setBackgroundImage:[UIImage
bjl_imageWithColor:[UIColor
bjl_colorWithHexString:@"#89899C" alpha:0.5]]
51.                       forState:UIControlStateDisabled];
52.     button.layer.cornerRadius = buttonHeight /
2;
53.     button.layer.masksToBounds = YES;
54.     [button addTarget:self
action:@selector(showQuickSlideViewController)
forControlEvents:UIControlEventTouchUpInside];
55.     [self.room.slideshowViewController.view
addSubview:button];
56.     [button
bjl_makeConstraints:^(BJLConstraintMaker
*make) {
57. 
make.centerX.equalTo(self.room.slideshowViewCon
```

```
make.bottom.equalTo(self.room.slideshowViewCont
make.size.equal.sizeOffset(CGSizeMake(buttonWidt
 buttonHeight));
        }];
        button;
    });
    [self.room.slideshowViewController.view
bjl_makeConstraints:^(BJLConstraintMaker
*_Nonnull make) {
        make.edges.equalTo(self.slideshowView);
    }];
```

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

```
/** 黑板视图控制器 */
@property (nonatomic, readonly)
UIViewController<BJLBlackboardUI>
*blackboardViewController;
/** 黑板页数 */
@property (nonatomic, readonly) NSInteger
blackboardContentPages;
/** 设置黑板视图（blackboardViewController）的默
认背景图 */
@property (nonatomic) UIImage
*blackboardImage;
// 获取当前黑板页码
CGFloat localPageIndex =
self.room.documentVM.blackboardViewController.p

/**
 指定文档 ID 创建对应的视图控制器
```

```
 #param documentID 文档 ID，通过 BJLDocument 的 documentID 获得
 #return 对应文档的视图控制器
 */
- (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：

```
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_open; // 打开
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_updateRect; // 移动 & 缩放
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_stick; // 置顶
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_fullScreen; // 全屏
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_maximize; // 最大化
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_restore; // 还原
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_close; // 关闭
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_rename; // 视图标题更新
FOUNDATION_EXPORT NSString *const BJLWindowsUpdateAction_min; // 视图最小化
```

```
/**
 更新文档显示信息
 #param documentID 文档 ID
 #param displayInfo 文档显示信息
 */
[self.context.room.documentVM
updateDocumentWithID:BJLBlackboardID
displayInfo:displayInfo];

/**
 更新文档显示信息的通知
 #param document 更新后的文档信息
 */
[self
bjl_observe:BJLMakeMethod(self.context.room.docu
 didUpdateDocument:)
          filter:^BOOL(BJLDocument *document){
    // bjl_strongify(self);
    return [document.documentID
isEqualToString:BJLBlackboardID];
}
        observer:^BOOL(BJLDocument *document)
{
    bjl_strongify(self);
    // 黑板滑动
    [self
scrollToTopOffset:document.displayInfo.topOffset];
    return YES;
}];

/**
 更新文档窗口
 #param documentID 文档 ID
 #param action 更新类型，参考
BJLWindowsUpdateModel 的
```

```
BJLWindowsUpdateAction
 #param displayInfos 教室内所有文档窗口显示信息
 #return 调用错误, 参考 BJLErrorCode
 */
[self.room.documentVM
updateDocumentWindowWithID:documentID
                                action:action

displayInfos:self.documentWindowDisplayInfos];

/**
 文档窗口更新通知
 #param updateModel 更新信息
 #param shouldReset 是否重置
 */
[self
bjl_observe:BJLMakeMethod(self.room.documentVM

didUpdateDocumentWindowWithModel:shouldReset
        observer:
(BJLMethodObserver)^BOOL(BJLWindowUpdateMod
 *updateModel, BOOL shouldReset) {
    bjl_strongify(self);
    if (shouldReset) {
        [self
resetDocumentWindowsWithModel:updateModel];
    }
    else {
        [self
updateDocumentWindowWithModel:updateModel];
    }
    return YES;
}];
```

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：

```
#pragma mark - view type

/// 当前课件视图类型，不固定，根据 BJLDocument 的数据内容会发生改变
@property (nonatomic, readonly) BJLPPTViewType viewType;

/// 禁用课件的动效，默认为 NO，动效开启后 `viewType` 只会是 `BJLPPTViewType_H5`
@property (nonatomic) BOOL disablePPTAnimation;

#pragma mark - page control

/// 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
/// #param slidePage BJLSlidePage
- (void)updateMaxSlidePage:(BJLSlidePage *)slidePage;

/// 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
@property (nonatomic, readonly) BOOL syncPageChange;
- (void)updateSyncPageChange:(BOOL)syncPageChange;

/// 是否禁止超过翻页超过最大页码，默认 YES
```

```
@property (nonatomic, readonly) BOOL disableOverMaxPage;
- (void)updateDisableOverMaxPage:(BOOL)disableOverMaxPage;

/// 当前课件是否支持滑动翻页，默认可滑动
/// #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
@property (nonatomic, readonly) BOOL scrollEnabled;
- (void)updateScrollEnabled:(BOOL)scrollEnabled;

/// 当前页码
/// #discussion 不能滑动翻页时，参考 `scrollEnabled`，设置 pageIndex 无效
@property (nonatomic, readonly) NSInteger pageIndex;
- (nullable BJLError *)updatePageIndex:(NSInteger)pageIndex;

/// 能否向前翻页，或者存在前一步
@property (nonatomic, readonly) BOOL canStepForward;

/// 能否向后翻页，或者存在后一步
@property (nonatomic, readonly) BOOL canStepBackward;

/// 向前翻页
- (nullable BJLError *)pageStepForward;

/// 向后翻页
- (nullable BJLError *)pageStepBackward;

/// 课件视图处理多个 BJLDocument 数据时: 是否禁用跨越课件翻页
```

```
/// #discussion 默认允许跨课件，不同课件之间的 pageStepForward 和 pageStepBackward 是可用的
/// #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
@property (nonatomic) BOOL disableCrossDoc;

#pragma mark - customize

/// 自定义白板样式
@property (nonatomic) BJLWhiteboard *whiteboard;

/// 自定义页面控制按钮
@property (nonatomic) UIButton *pageControlButton;

#pragma mark - remark

/// 是否显示课件备注，默认显示
@property (nonatomic, readonly) BOOL showPPTRemarkInfo;
- (void)updateShowPPTRemarkInfo:(BOOL)show;

#pragma mark - draw

/// 画笔开关状态，默认为 NO，不可使用画笔
@property (nonatomic) BOOL drawingEnabled;

/// 课件当前页图像在视图中的布局信息
@property (nonatomic, readonly) CGRect imageFrameInPPTView;

/// 清除当前画布上所有画笔
- (void)clearDrawing;
```

- 静态课件 API。

```objc
/// 静态课件尺寸
/// #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
/// #discussion 单位为像素，默认初始加载 720、放大加载 1080，取值在 `BJLAliIMGMinSize` 到 `BJLAliIMGMaxSize` 之间 (1 ~ 4096)
/// #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
@property (nonatomic) NSInteger imageSize;

/// 静态课件占位图
@property (nonatomic, nullable) UIImage *placeholderImage;

/// 静态课件白板图片
@property (nonatomic, nullable) UIImage *whiteboardBackgroundImage;

/// 回放设置本地图片路径
@property (nonatomic, nullable) NSArray *localPPTImagePaths;

/// 是否加载课件原图，默认不加载原图，设置后改变 imageSize 无效
@property (nonatomic, readonly) BOOL useOriginalImage;
- (void)updateUseOriginalImage:(BOOL)useOriginalImage;

/// 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
```

```
@property (nonatomic, readonly) BJLContentMode contentMode;
- (void)updateContentMode:(BJLContentMode)contentMode;

/// 静态课件是否支持缩放
/// #param scaleEnabled 是否支持缩放
/// #discussion 默认支持缩放
@property (nonatomic, readonly) BOOL scaleEnabled;
- (void)updateScaleEnabled:(BOOL)scaleEnabled;
```

- 动态课件 API。

```
/// 动态课件加载成功
@property (nonatomic, readonly) BOOL webPPTLoadSuccess;

/// 动态课件加载失败时,是否要切静态课件
@property (nonatomic, copy, nullable) void (^shouldSwitchNativePPTBlock)(NSString *_Nullable documentID, void (^callback)(BOOL shouldSwitch));

/// 动态课件翻页指示图标
@property (nonatomic) UIImage *prevPageIndicatorImage, *nextPageIndicatorImage;

/// 动态课件手势触发的翻页是否能够向前翻页，默认能翻页
/// 不能滑动翻页时，参考 `scrollEnabled`，也不能手势翻页
@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。

```
bjl_weakify(self);
[self.room.documentVM uploadImageFile:fileURL
                             progress:^(CGFloat progress){
    bjl_strongify(self);
    // 显示进度
    self.progressView.progress = progress;
}
                               finish:^(BJLDocument *
_Nullable document, BJLError * _Nullable error) {
    bjl_strongify(self)
    if(document){
        // 添加课件到教室中
        [self.room.documentVM
addDocument:document];
    }
    else{
        NSLog(@"error:%@", error);
    }
}];
```

- 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。

```
// 上传
[self.room.documentVM uploadFile:url
                        mimeType:mimeType
                        fileName:name
                     isAnimated:isAnimated
                        progress:^(CGFloat progress) {
    //bjl_strongify(self);
    NSLog(@"progress", progress);
}
```

```
                    finish:^(BJLDocument * _Nullable
document, BJLError * _Nullable error) {
    bjl_strongify(self);
    if (error) {
        return;
    }
    if (image) {
        // 图片不需要转码，直接添加到教室
        [self.room.documentVM
addDocument:document];
    }
    else {
        // 开始轮询转码进度
        [self startPollTimer];
    }
}];

// 转码
[self.room.documentVM
requestTranscodingProgressWithFileIDList:array

completion:^(NSArray<BJLDocumentTranscodeMod
 *> * _Nullable transcodeModelArray, BJLError *
_Nullable error) {
    bjl_strongify(self);
    if (error) {
        return;
    }
    for (BJLDocumentTranscodeModel *model in
transcodeModelArray) {
        if (model.progress >= 100) {
            // 转码完成，请求转码完成之后的新文档
            [self requestDocumentList];
        }
        else {
            // 更新转码进度
```

```
            NSLog(@"progress", progress);
        }
    }
}];
// 添加文档
[self.room.documentVM
requestDocumentListWithFileIDList:@[fileID]

completion:^(NSArray<BJLDocument *> *
_Nullable documentArray, BJLError * _Nullable
error) {
    if (error) {
        return;
    }
    for (BJLDocument *document in
documentArray) {
        // 添加文档
        [self.room.documentVM
addDocument:document];
    }
}];
```

- 删除课件。

```
/**
 删除课件
 #discussion 删除成功将调用 `BJLDocumentVM` 的
`didDeleteDocument:`
 #param documentID 课件 ID
 #return BJLError:
 BJLErrorCode_invalidArguments  错误参数
 */
[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。

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

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

```
[self
bjl_observe:BJLMakeMethod(self.room.documentVM
 didDeleteDocument:)
        observer:^BOOL(BJLDocument *document)
{
    // bjl_strongify(self);
    if(document){
        NSLog(@"document: %@ delete",
document);
    }
    return YES;
}];
```

- 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。

```
[self
bjl_kvo:BJLMakeProperty(self.room.slideshowViewC
 pageIndex)
     observer:^BOOL(id  _Nullable value, id
_Nullable oldValue, BJLPropertyChange *
_Nullable change) {
    NSLog(@"localPage：%td", [value
integerValue]);
    return YES;
```

```
}];
```

- 课件授权。

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

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

- 课件备注

```
/**
 获取文档备注信息
 #param documentID 文档ID, 当请求的文档为白板时, 返回空
 #param completion 返回 remarkInfo 备注信息
 #return task
 */
- (nullable NSURLSessionDataTask *)requestCompleteRemarkInfoWithDocumentID:
```

```
(NSString *)documentID
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。

```
/**
 添加白板
 #discussion 最多同时存在 10 页白板
 #return BJLError
 */
- (nullable BJLError *)addWhiteboardPage;

/**
 添加白板通知
 #discussion 同时更新 `allDocuments`
 #param pageIndex 白板页码
```

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

```
/**
 删除白板
 #param pageIndex 白板页码，使用 BJLSlidePage 的 `slidePageIndex`
 #return BJLError
 */
- (nullable BJLError *)deleteWhiteboardPageWithIndex:(NSInteger)pageIndex;

/**
 删除白板通知
 #discussion 同时更新 `allDocuments`
 #param pageIndex 白板页码
 */
- (BJLObservable)didDeleteWhiteboardPageWithIndex:(NSInteger)pageIndex;
```

- 专业小班课课件加载使用原图

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

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

- 大班课授权 H5 课件操作权限。

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

5. 小黑板管理

- 收到小黑板发布。

```
/**
 发布小黑板的通知
 #discussion 直播间内所有用户不区分角色都会收到此信令
 #param writingBoard 小黑板信息, 此时 writingBoard 的 `submitedUsers` 和 `participatedUsers` 为空,此处不做统计
 */
[self bjl_observe:BJLMakeMethod(self.room.documentVM didPublishWritingBoard:)
          filter:^BOOL(BJLWritingBoard *writingBoard) {
    return (writingBoard
            && writingBoard.boardID.length);
}
```

```
        observer:^BOOL(BJLWritingBoard *writingBoard) {
    bjl_strongify(self);
    // 判断是否为学生/助教
    if (writingBoard.operate == BJLWritingBoardPublishOperate_begin) {
    }
    else if (writingBoard.operate == BJLWritingBoardPublishOperate_revoke
            && self.writingBoardViewController) {
    }
    else if (writingBoard.operate == BJLWritingBoardPublishOperate_end
            && self.writingBoardViewController) {
    }
    return YES;
}];
```

- 根据发布的小黑板信息创建小黑板。

```
/**
 根据 writingBoard 创建对应的小黑板控制器
 #param writingBoard 小黑板信息
 #return 对应小黑板的视图控制器
 */
 self.writingBoradViewController = [room.documentVM writingBoardViewControllerWithWritingBoard:writing
```

- 参与小黑板作答。

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

```
    [self.room.drawingVM updateWritingBoardEnabled:status == BJLIcWriteBoardStatus_studentEdit];
    // 提交小黑板内容
    [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。

```

/**
 加载当前直播间媒体文件
 #discussion 直播间内的媒体文件增加、删除都会触发
 #param mediaFiles 媒体文件
 */
- (BJLObservable)didLoadMediaFiles:(NSArray<BJLMediaFile *> *)mediaFiles;

/**
 删除媒体课件
 #param fileID 课件 ID
 */
- (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

要后台配置开启，参考 `enableCloudStorage` 。

- 获取云盘文件列表。

```
/**
 请求云盘文件列表
 #param targetFinderID 请求目录的finderID，为空表示请求根目录列表
 #param page 从第一页开始
 */
- (nullable NSURLSessionDataTask *)requestCloudListWithTargetFinderID:(nullable NSString *)targetFinderID
                                                                 page:(NSUInteger)page
                                                             pagesize:(NSUInteger)pagesize
                                                           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。

```
/**
 上传云盘文档
 #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
 #discussion 转码服务器主动进行，不需要调用其他接口
 #param fileURL 文档路径
 #param mimeType 文档类型
 #param fileName 文件名
 #param isAnimated 动态文件 or 静态文件
```

```
 #param progress 进度 0.0 ~ 1.0
 #param finish 结束回调
 - homework        非 nil 即为成功
 - 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
 - error           错误
 #return task
 */
- (nullable NSURLSessionUploadTask *)uploadCloudFile:(NSURL *)fileURL
                                            mimeType:(NSString *)mimeType
                                            fileName:(NSString *)fileName
                                          isAnimated:(BOOL)isAnimated
                                            progress:(nullable void (^)(CGFloat progress))progress
                                              finish:(void (^)(BJLCloudFile * _Nullable cloudFile, BJLError * _Nullable error))finish;

/** 请求删除云盘文件，仅支持删除文件，不支持删除文件夹 */
- (nullable NSURLSessionDataTask *)requestDeleteCloudFileWithFileID:(NSString *)fileID
                                                         completion:(nullable void (^)(BOOL success, BJLError * _Nullable error))completion;
```

- 搜索云盘文件。

```
/**
```

```
请求搜索云盘文件
 #param keyword 搜索关键词
 #param targetFinderID 搜索目录的finderID，为空表示在根目录搜索
 #param page 从第一页开始
 */
- (nullable NSURLSessionDataTask *)requestSearchCloudFileListWithKeyword:(NSString *)keyword
                                                                 targetFinderID:(nullable NSString *)targetFinderID
                                                                           page:(NSUInteger)page
                                                                       pagesize:(NSUInteger)pagesize
                                                                     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。

```
/** 教室作业列表 */
@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:BJLMakeMethod(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:`
```

```
#return BJLError:
 BJLErrorCode_invalidUserRole 仅支持老师/助教身份操作
 */
- (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。

```
/**
 上传作业文档
 #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
 #discussion 转码服务器主动进行，不需要调用其他接口
 #param fileURL 本地文档路径
 #param mimeType 文档类型
 #param fileName 文件名
 #param progress 进度 0.0 ~ 1.0
 #param finish 结束回调
 - homework        非 nil 即为成功，用于 `addHomework:`
 - 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
 - error           错误
 #return task
 */
- (NSURLSessionUploadTask *)uploadHomeworkFile:(NSURL *)fileURL
                                      mimeType:(NSString *)mimeType
```

```
                                fileName:(NSString
*)fileName
                                progress:(nullable void
(^)(CGFloat progress))progress
                                 finish:(void (^)
(BJLHomework * _Nullable homework,
BJLDocument * _Nullable document, BJLError *
_Nullable error))finish;

/**
 添加作业
 #discussion 添加成功将调用 `BJLHomeworkVM` 的
`didAddHomeworks:`
 #param homework 作业
 #return BJLError:
 BJLErrorCode_invalidArguments  错误参数
 */
- (nullable BJLError *)addHomework:
(BJLHomework *)homework;
```

```
// example：下载作业
bjl_weakify(self);
[self.room.homeworkVM
requestDownloadURLWithHomeworkID:file.remoteH
                                completion:^(BOOL
success, BJLError * _Nullable error, NSString *
_Nonnull downloadUrl) {
    bjl_strongify(self);
    if (!success) {

self.showErrorMessageCallback(error.localizedFailur
 ?: error.localizedDescription);
    }
    else {
```

```
        NSString *itemIdentifier = [self itemIdentifierWithHomeworkID:file.remoteHomewor
        [self.manager addDownloadItemWithIdentifier:itemIdentifier itemClass:[BJLHomeworkDownloadItem class] setting:^(__kindof BJLHomeworkDownloadItem * _Nonnull item) {
            bjl_strongify(self);
            item.sourceURL = [NSURL URLWithString:downloadUrl];
            item.homework = file.remoteHomework;
            item.downloadTimeInterval = [NSDate timeIntervalSinceReferenceDate];
            item.roomName = self.room.roomInfo.title;
            item.roomID = self.room.roomInfo.ID;
        }];
    }
}];
```

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

- 搜索作业。

```
/**
 根据关键字`keywork`搜索
 #discussion 搜索范围: 作业名称和作业上传者用户名
 #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
 */
- (nullable BJLError *)searchHomeworksWithKeyword:(NSString *)keyword
                                     lastHomework:(nullable BJLHomework *)lastHomework
                                            count:(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。

```
/**
 开启、关闭画笔
 #param drawingEnabled YES：开启，NO：关闭
 #return BJLError:
 #discussion BJLErrorCode_invalidCalling　错误调用，当前用户是学生、`drawingGranted` 是 NO
 #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
 #discussion `drawingGranted` 是 YES 时才可以开启，`drawingGranted` 是 NO 时会被自动关闭
 */
 BJLError *error = [self.room.drawingVM updateDrawingEnabled:YES];
```

```
/** 小黑板画笔开关状态 */
@property (nonatomic, readonly) BOOL writingBoardEnabled;

/**
 开启/关闭小黑板画笔
 #param writingBoardEnabled 是否开启小黑板画笔
 */
- (void)updateWritingBoardEnabled:(BOOL)writingBoardEnabled;
```

- 画笔授权。

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

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

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

- 画笔设置。

```
/**
 更新画笔操作模式

 #param operateMode 操作模式，参考 `BJLBrushOperateMode`
 #return BJLError:
 #discussion BJLErrorCode_invalidCalling drawingEnabled 是 NO
 */
BJLError *error = [self.room.drawingVM updateBrushOperateMode:operateMode];
```

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

```objc
/** 画笔边框颜色，默认 #1795FF */
@property (nonatomic, nonnull) NSString *strokeColor;
/** 画笔边框颜色透明度，取值范围 0~1，默认 1.0*/
@property (nonatomic) CGFloat strokeAlpha;
/** 画笔填充颜色，默认和画笔颜色一致 */
@property (nonatomic, nullable) NSString *fillColor;
/** 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 */
@property (nonatomic) CGFloat fillAlpha;
/** doodle 画笔线宽，默认 4.0 */
@property (nonatomic) CGFloat doodleStrokeWidth;
/** 图形画笔边框线宽，默认 2.0 */
@property (nonatomic) CGFloat shapeStrokeWidth;
/** 文字画笔字体大小，默认 20.0 */
@property (nonatomic) CGFloat textFontSize;
/** 文字画笔是否加粗，默认 NO */
@property (nonatomic) BOOL textBold;
/** 文字画笔是否为斜体，默认 NO */
@property (nonatomic) BOOL textItalic;
/** 画笔开关状态，参考 `drawingGranted`、`updateDrawingEnabled:` */
@property (nonatomic, readonly) BOOL drawingEnabled;
/** doodle 画笔是否虚线，默认 NO */
@property (nonatomic) BOOL isDottedLine;
/** 选中画笔时是否显示归属信息，默认 NO */
@property (nonatomic) BOOL showBrushOwnerNameWhenSelected;
```

- 画笔颜色分配。

```
/** 画笔分配颜色记录 <hex color, user.number> */
@property (nonatomic, readonly, copy) NSDictionary <NSString *, NSString *> *drawingGrantedColors;

/** 是否不使用分配的画笔颜色 */
@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。

```
/**
 添加图片画笔
 #param imageURL 图片 url
 #param relativeFrame 图片相对于画布的 frame, 各项数值取值范围为 [0.0, 1.0]
 #param documentID 目标文档 ID
 #param pageIndex  目标页
 #param isWritingBoard  是否为小黑板
 #return BJLErrorCode_invalidCalling drawingEnabled 是 NO
 */
- (nullable BJLError *)addImageShapeWithURL:(NSString *)imageURL
                              relativeFrame:(CGRect)relativeFrame
                                 toDocumentID:(NSString *)documentID
                                    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。

```
/**
 大班课文档区域是否绘制激光笔
 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；
 如果设置为不绘制，需要自行实现激光笔效果
 专业小班课不支持设置，默认不自动绘制
 */
@property (nonatomic) BOOL drawsLaserPointer;
```

```
/**
 激光笔位置移动请求
 #param location          激光笔目标位置
 #param documentID        激光笔所在文档的 ID
 #param pageIndex         激光笔所在文档页码
 */
- (nullable BJLError *)moveLaserPointToLocation:(CGPoint)location
                                     documentID:(nonnull NSString *)documentID
                                      pageIndex:(NSUInteger)pageIndex;
```

```
/**
 激光笔位置移动监听
 #param location          激光笔位置
 #param documentID        激光笔所在文档的 ID
 #param pageIndex         激光笔所在文档页码
 */
```

```
- (BJLObservable)didLaserPointMoveToLocation:(CGPoint)location
                                  documentID:(NSString *)documentID
                                   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。

```
/**
 BJLDrawingShapeType_doodle 类型的画笔移动位置回调
 #param location        画笔位置，[0, 1] 区间内的坐标
 #param documentID      画笔所在文档的 ID
 #param pageIndex       画笔所在文档页码
 #param color           画笔颜色
 #param fromCurrentUser 是否是当前用户使用画笔
 */
- (BJLObservable)didPaintPointMoveToLocation:(CGPoint)location
                                  documentID:(NSString *)documentID
                                   pageIndex:(NSUInteger)pageIndex
                                       color:(nullable UIColor *)color
                                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。

```
/**
 落笔位置点同步
 #param location        画笔位置，[0, 1] 区间内的坐标
 #param documentID      画笔所在文档的 ID
 #param pageIndex       画笔所在文档页码
 #param color           画笔颜色
 */
- (BJLObservable)didMousePointMoveToLocation:(CGPoint)location
                                  documentID:(NSString *)documentID
                                   pageIndex:(NSUInteger)pageIndex
                                       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。

```
/** 搜索手写板 */
- (void)scanHandWritingBoard;

/**
 连接手写板
 #param handWritingBoard 手写板
```

```
 */
- (void)connectHandWritingBoard:(CBPeripheral *)handWritingBoard;

/**
 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`
 #param 顶部视图，作为手写板的数据的基准视图
 */
- (void)updateTopDrawingController:(UIViewController *)topDrawingController;

/**
 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
 #discussion point 在设置的 topDrawingController 的位置
 #discussion 返回的控制器必须是满足 BJLSlideshowUI 或者 BJLBlackboardUI 协议的文档控制器
 */
@property (nonatomic, nullable) UIViewController *(^requestFirstRespondDocumentCallback)(CGPoint point);
```