互动白板

高级功能

腾讯云

文档目录

高级功能
播放视频文件

最近更新时间：2024-03-01 15:27:31

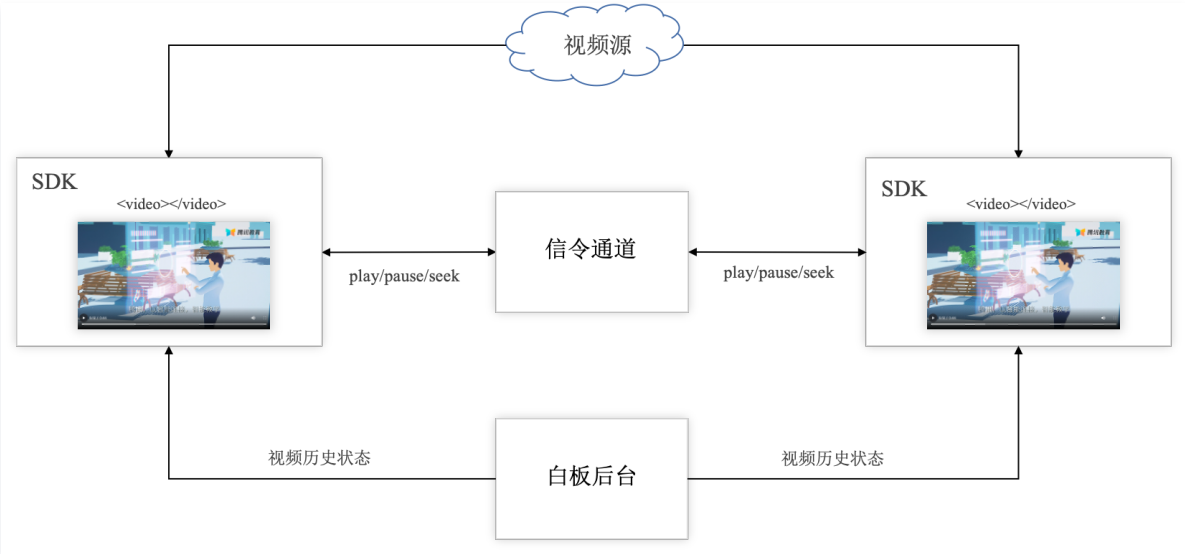
功能简介

本文主要介绍如何在互动白板中播放视频文件。

平台支持

iOS	Android	Windows	Mac OS	Web	小程序
✓	✓	✓	✓	✓	×

基本原理



基本接口

添加文件

addVideoFile

参数	类型	默认值
url	String	空

说明

- 支持 mp4/m3u8
- 触发 onTEBVideoStatusChanged() 回调

onTEBVideoStatusChanged **参数说明**

参数	类型	说明
fileId	String	文件 ID
status	TEduBoardVideoStatus	文件状态
progress	float	当前进度（秒），仅 mp4 格式支持

duration	float	总时长（秒），仅 mp4 格式支持

TEduBoardVideoStatus 状态说明

状态名	状态值	说明
TEDU_BOARD_VIDEO_STATUS_ERROR	1	播放出错
TEDU_BOARD_VIDEO_STATUS_LOADING	2	视频加载中
TEDU_BOARD_VIDEO_STATUS_LOADED	3	视频加载完成
TEDU_BOARD_VIDEO_STATUS_PLAYED	4	视频开始播放
TEDU_BOARD_VIDEO_STATUS_TIMEUPDATE	5	播放进度更新
TEDU_BOARD_VIDEO_STATUS_PAUSED	6	视频已暂停
TEDU_BOARD_VIDEO_STATUS_SEEKED	7	视频进度跳转
TEDU_BOARD_VIDEO_STATUS_ENDED	8	视频播放结束

隐藏控制栏

showVideoControl

参数	类型	默认值
show	bool	false

说明

- 全局控制项，对所有视频文件有效。
- 视频播放器默认显示系统自带的 video 控制栏，不同平台界面样式不同。
- 如果您需要自定义控制栏界面，请设置为 false，并调用 playVideo 、pauseVideo 、seekVideo 接口。
- 一般情况下，老师设置为 true，学生设置为 false。

管理同步权限

setSyncVideoStatusEnable

参数	类型	默认值
enable	bool	true

说明

- 全局控制项，对所有视频文件有效。
- 如果为 true，playVideo、pauseVideo、seekVideo 接口以及控制栏事件会影响远端。
- 一般情况下，老师设置为 true，学生设置为 false。

- 全局控制项，对所有视频文件有效。
- 如果为 true，playVideo、pauseVideo、seekVideo 接口以及控制栏事件会影响远端。
- 一般情况下，老师设置为 true，学生设置为 false。

视频状态监听

常见问题

1. 视频播放出现卡顿?
   互动白板使用 html 的 video 标签播放视频文件，视频的加载和缓冲依赖于当前的网络环境、机器负载、视频原站的出口带宽，请依次检查以上因素。
2. Android 播放进度不准确?

在 TBS 环境下，受限于 X5 内核和视频资源 I 帧间隔，在 Android 平台下无法精准同步。例如：10秒的视频，I 帧间隔5秒，seek 到4秒位置，在 TBS 上从0秒开始播放。

3. 部分 mp4 文件无法在 Chrome 正常播放
由于编码的专利问题，Chrome 仅支持 h264 编码的 mp4 文件，其他编码格式的 mp4 文件会出现黑屏有声音的情况。

> **说明**
> 可以通过使用 腾讯云点播转码服务 将视频文件进行转码，保证 mp4 文件的编码格式为 h264。

白板快照

最近更新时间：2024-02-20 15:48:41

内容介绍

本文主要介绍如何使用白板快照功能。

平台支持

iOS	Android	Windows	Mac OS	Web	小程序
✓	✓	✓	✓	✓	×

如何使用

平台	接口	回调
iOS	snapshot:	onTEBSnapshot:errorCode:errorMsg:
Android	snapshot(TEduBoardSnapshotInfo info)	onTEBSnapshot(int code, String msg)
Windows	snapshot(TEduBoardSnapshotInfo *info)	onTEBSnapshot(int code, String msg)
Mac OS	snapshot:	onTEBSnapshot:errorCode:errorMsg:
Web	snapshot({userData:'透传字段'})	TEB_SNAPSHOT({image:'base64**', userData:'透传字段'})

前端配置

如果您在互动白板中使用了自研 H5 页面，由于 iframe 禁止跨域截屏，请在 H5 页面中集成 h5webctrl.js 即可，没有集成 h5webctrl.js 的第三方页面因跨域限制则不支持白板快照功能。

```
<script src="https://res.qcloudtiw.com/board/third/h5webctrl/h5webctrl.min.js"></script>
```

后台配置

白板快照的原理是将 DOM 节点转为图片， 由于跨域资源共享（CORS）机制，浏览器或 Webview 默认禁止对非同域资源截屏，如果您的白板包含视频文件、图片元素、**已转存**的静态 PPT、**已转存**的缩略图、**已转存**的动画 PPT，请配置 **HTTP Header** 的 **Access-Control-Allow-Origin** 参数。如果您使用的是 腾讯云 CDN 服务，请在对应域名的【高级配置】中【添加 HTTP Header】即可。

参数	值
Access-Control-Allow-Origin	*

如果您需要对跨域访问做精确的控制，请按以下选项配置：

- 允许所有域名访问，配置 Access-Control-Allow-Origin 为 *
- 只允许特定域名访问
  - 只有移动端或桌面端，Access-Control-Allow-Origin 添加域名 https://res.qcloudtiw.com。
  - 只有 Web 端，Access-Control-Allow-Origin 添加 Web 所在域名。

课件预加载

最近更新时间：2023-08-11 11:45:21

内容介绍

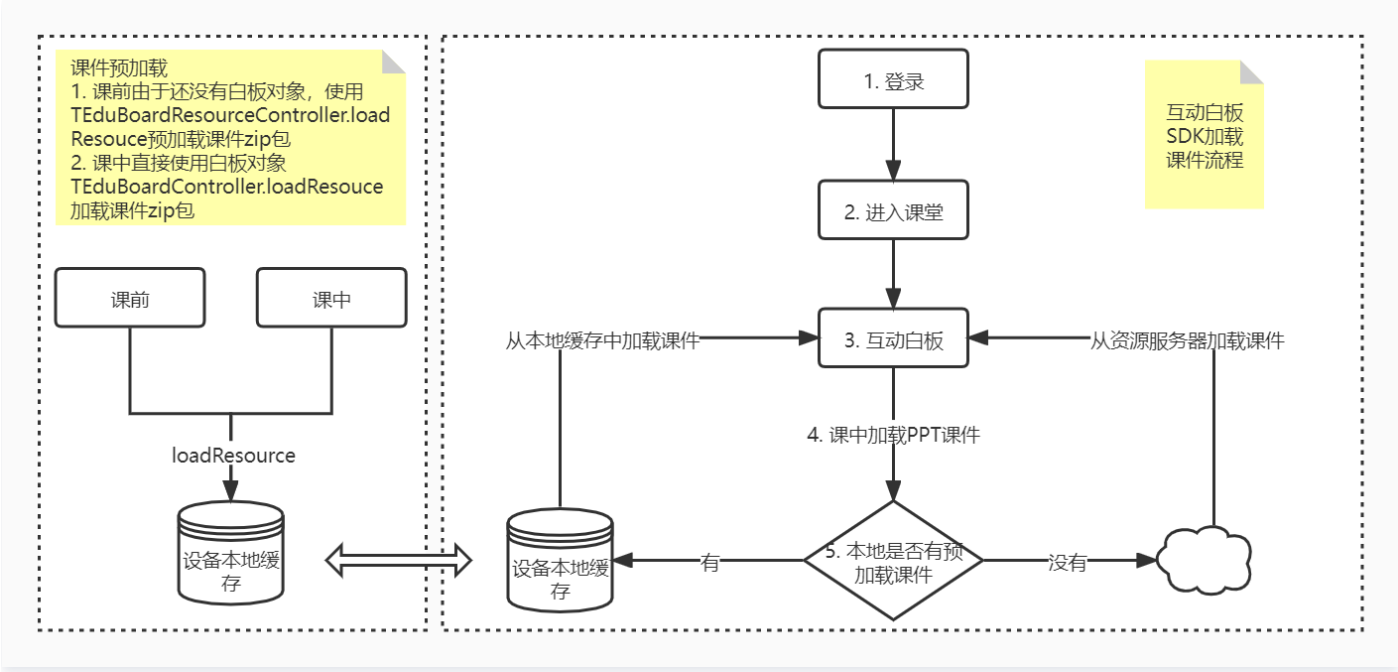
本文主要介绍如何使用白板课件预加载功能。

平台和版本支持

平台类型	Android	Windows	iOS	Mac OS	Web
平台版本	✓	✓	11.4+	11.0+	✖
TEduBoardSDK 版本	2.6.9.145+	2.6.9.240+	2.6.9.92+	2.6.9.92+	✖
TIWCache 版本	1.0.0.78+	✓	1.0.0.72+	1.0.0.72+	✖

说明
对于不支持的系统版本，系统平台和互动白板版本，课件加载则正常从公网加载课件。

工作原理



如何使用

1. 文件转码申请打压缩包

在 发起课件转码 的时候需要将 CompressFileType 设置为 zip，在 转码结果 中 CompressFileUrl 字段会生成一个 zip 包 url。
发起转码示例（Python）：

```
req = models.CreateTranscodeRequest()
params = {
    "SdkAppId": 1400127140,
    "CompressFileType": "zip" // 申请转码，并对转码结果进行打zip包
    ... // 其他参数
}
req.from_json_string(json.dumps(params))
```

```
resp = client.CreateTranscode(req)
```

2. 获取课件转码结果

课件转码结果可以通过 主动查询 或者 回调方式 获取。
结果示例：

```
{
  "Response": {
    "TaskId": "0poq8tn4ts23317ta77c",
    "ResultUrl": "https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/h5/index.html",
    "CompressFileUrl": "https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip"
    ... // 其他结果参数
  }
}
```

3. 课件本地预加载

3.1 课前使用

课前由于还没有互动白板对象，互动白板 SDK 提供了 TEduBoardResourceController 类来支持课件预加载。
Android

```
// 设置sdkAppId和userId
TEduBoardResourceController.setConfig(sdkAppId, userId);
// 加载课件zip包
TEduBoardResourceController.loadResource(context, "https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip");
```

iOS / Mac

```
TEduBoardResourceConfig *config = [[TEduBoardResourceConfig alloc] init];
config.sdkAppId = self.sdkAppId;
config.userId = self.userId;
[TEduBoardResourceController setConfig:config];
[TEduBoardResourceController loadResource:@"https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip"];
```

Windows

```
std::string strUserID = strUserID;
unsigned int sdkAppid = sdkAppid;
SetTeduBoardResourceConfig(sdkAppid, strUserID.c_str());
std::string strUrl = "https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip";
LoadTeduBoardResource(strUrl.c_str());
```

3.2 课中使用

直接使用白板对象上的 loadResource 接口进行课件预加载。
Android

```
// 加载课件zip包
teduBoardController.loadResource("https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip");
```

iOS / Mac

```
[self.boardController loadResource:@"https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip"];
```

Windows

```
std::string strUrl = "https://whiteboard-cam-test-1257307760.cos.ap-nanjing.myqcloud.com/doc/0poq8tn4ts23317ta77c_tiw/0poq8tn4ts23317ta77c.zip";
LoadTeduBoardResource(strUrl.c_str());
```

3.3 课件预加载状态回调

Android

```
TIWCacheManager.getInstance(Context).setListener(new TIWCacheListener() {
  @Override
  public void onTIWCacheCoursewareDownloaded(String zipUrl, int errcode, String message) {

  }
});
```

iOS / Mac

```
[[TIWCacheManager shareInstance] setDelegate:delegate];
// override 回调方法
- (void)onTIWCacheCoursewareDownloaded:(NSString *)zipUrl errorCode:(uint32_t)errcode message:(NSString *)message;
```

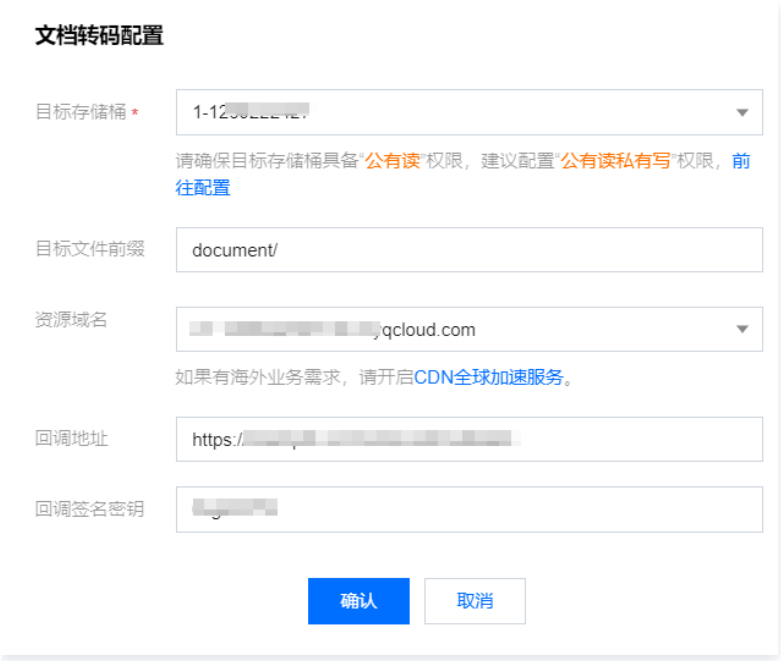
4. 互动白板中添加转码文件

完成以上3步后，只需要调用 addTranscodeFile 接口将转码后的结果添加到白板中，即可使用课件从本地加载的能力。

```
// 1. 具体的参数请以第2步查询的结果为准。
// 2. addTranscodeFile 接口使用方式请参考各个平台提供的接口为准。
teduBoard.addTranscodeFile({
    pages: Pages,
    resolution: Resolution,
    title: Title,
    url: ResultUrl
})
```

注意事项

1. 在互动白板转码控制台上进行文档转码的存储桶配置，不能直接使用互动白板默认的公共存储桶。



2. 课件预加载的暂只支持以 zip 格式的课件，所以在发起转码的时候需要将 CompressFileType 设置为 zip。
3. 课件预加载的 zip 包课件只在本地设备上保留7天，7天过后互动白板将在本地缓存中清除 zip 包课件。
4. iOS/Android 对同一个课件进行预加载，只会预加载一次。例如：本地设备缓存中已经存在 CompressFileUrl 参数对应的课件A，当使用相同的 CompressFileUrl 参数再次调用loadResource 去加载课件A时，会直接跳过。
5. Windows 对同一个课件进行预加载，会覆盖缓存中相同的课件。例如：本地设备缓存中已经存在由 CompressFileUrl 参数指定的课件B，当使用相同的 CompressFileUrl 参数再次调用 LoadTeduBoardResource 去加载课件B时，会覆盖缓存中的课件B。

H5 课件
接入指南

最近更新时间：2023-03-27 11:37:05

本文主要讲述第三方 H5 接入互动白板如何实现状态同步。

接入流程

引入 SDK

```
<script src="https://res.qcloudtiw.com/board/third/h5webctrl/h5webctrl.min.js"></script>
```

同步数据

互动白板 SDK 通过信令通道透传 data 协议数据，并更新 data 至互动白板后台作为历史数据。目前 data 数据大小限制为6K。

```
window.TIWH5WebCtrl.syncData(data);
```

参数	类型	含义
data	object	协议数据

> **注意**
> 互动白板后台只记录最新的 data，为保证中途进入课堂的用户能恢复H5页面状态，请确保 data 为全量的状态数据。

接收数据

互动白板在以下两种情况回调 data 协议数据。
1. 互动白板初始化并拉取历史数据成功回调最新的 data 协议数据。
2. 互动白板收到实时传输的 data 协议数据。

```
window.TIWH5WebCtrl.on('TIW_H5WEB_DATA', function(data) {

});
```

参数	类型	含义
data	object	协议数据

权限变化

互动白板调用 setDrawEnable 接口后触发权限更新。

```
window.TIWH5WebCtrl.on('TIW_H5WEB_PERMISSION', function(data) {

});
```

参数	类型	含义
enable	bool	是否有操作权限

事件移除

```
window.TIWH5WebCtrl.off(name, function);
```

测试流程

1. 打开 互动白板 DEMO（为演示同步效果，打开多个页面，以不同账号进入同一个房间）。
2. 进入房间后，在左侧工具中打开添加白板元素的面板，并在 H5元素tab中输入测试页面 https://tic-res-1259648581.cos.ap-shanghai.myqcloud.com/board/h5webctrl/h5web.html ，最后将白板工具切换为鼠标，即可体验 H5页面内容同步。

3. 测试页面源码，可在浏览器打开测试页面，通过鼠标右键"查看网页源代码"来查看。

常见问题

最近更新时间：2023-05-24 11:09:58

H5 课件有 iframe 的嵌套

1. 同步页面的 name 信息

将第一层的页面的 name 属性同步到 iframe 中

```
var iframeEl = document.querySelector('#test');
iframeEl.contentWindow.name = window.name;
```

2. 信令中转

```
window.addEventListener('message', function (event) {
  if (!event.data) {
    return;
  }
  var module = event.data.module;
  var cmd = event.data.cmd;
  var data = event.data.data;
  if (module === 'TIW_H5WEB') {
    // 如果信令来自 iframe 页面，则将信令往白板传递
    if (event.origin === iframeEl.contentWindow.location.origin) {
      window.parent.postMessage(event.data, '*');
    } else {
      // 其他信令则往 iframe 传递
      iframeEl.contentWindow.postMessage(event.data, '*');
    }
  }
});
```

完整代码

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <style>
    * {
      margin :0;
      padding: 0;
    }
    html, body {
      width: 100%;
      height: 100%;
    }
    iframe {
      width: 100%;
      height: 100%;
    }
  </style>
</head>

<body>
  <iframe id="test" src="https://tic-res-1259648581.cos.ap-shanghai.myqcloud.com/board/h5webctrl/h5web.html"
frameborder="0"></iframe>
  <script>
```

```
    var iframeEl = document.querySelector('#test');
    iframeEl.contentWindow.name = window.name;

    window.addEventListener('message', function (event) {
      if (!event.data) {
        return;
      }
      var module = event.data.module;
      var cmd = event.data.cmd;
      var data = event.data.data;
      if (module === 'TIW_H5WEB') {
        // 如果信令来至 iframe 页面，则将信令往白板传递
        if (event.origin === iframeEl.contentWindow.location.origin) {
          window.parent.postMessage(event.data, '*');
        } else {
          // 其他信令则往 iframe 传递
          iframeEl.contentWindow.postMessage(event.data, '*');
        }
      }
    });
  </script>
</body>
</html>
```

信令同步通道
信令同步概述

最近更新时间：2023-03-24 17:05:01

同一个白板房间里的白板操作(如：画笔轨迹、文档翻页、音视频播放等）同步到远端时，需要信令通道，此处以 IM 为信令通道来做说明：

1. IM 作为同步白板数据的信令通道，需要在白板init前创建好群组。
2. 白板初始化完成后，白板 userid 作为一个IM用户加入IM群组，如需用到音视频通话还可加入 TRTC 房间。
3. 上课时，用户操作白板（如：画笔轨迹、文档翻页、音视频播放等）会生成IM信令消息，本地触发白板的 onTEBSyncData 事件（事件名以各端实际为准），回调中返回的信令数据通过IM创建自定义消息发送到 IM 群组内，群组内（白板房间）其他成员收到白板信令消息后，调用互动白板 addSyncData 接口（接口名以各端实际为准）把信令数据添加到白板来进行操作实时同步。

> **注意：**
>
> 1. IM聊天消息和白板信令消息可以通过消息的 extension 字段来区分，白板信令消息的 extension 必须为 TXWhiteBoardExt，仅需将白板信令消息添加到白板。
> 2. 白板信令数据的格式是固定的，所以添加到白板的数据格式必须跟回调中返回的数据格式一致，否则会影响同步。
> 3. 通过添加收到的远端白板消息到白板时，白板操作端即消息发送者无需添加。
> 4. 当教室内除了老师外无其他学生，但有使用白板推流和白板实时录制时，也需要做白板信令数据同步。

示例：

- 具体示例代码参考文档：TIM 同步信令通道 。
- 此处消息收发用的是IM接口，使用时可直接参考 IM的各端API文档 。

自定义信令通道

最近更新时间：2023-05-23 10:34:47

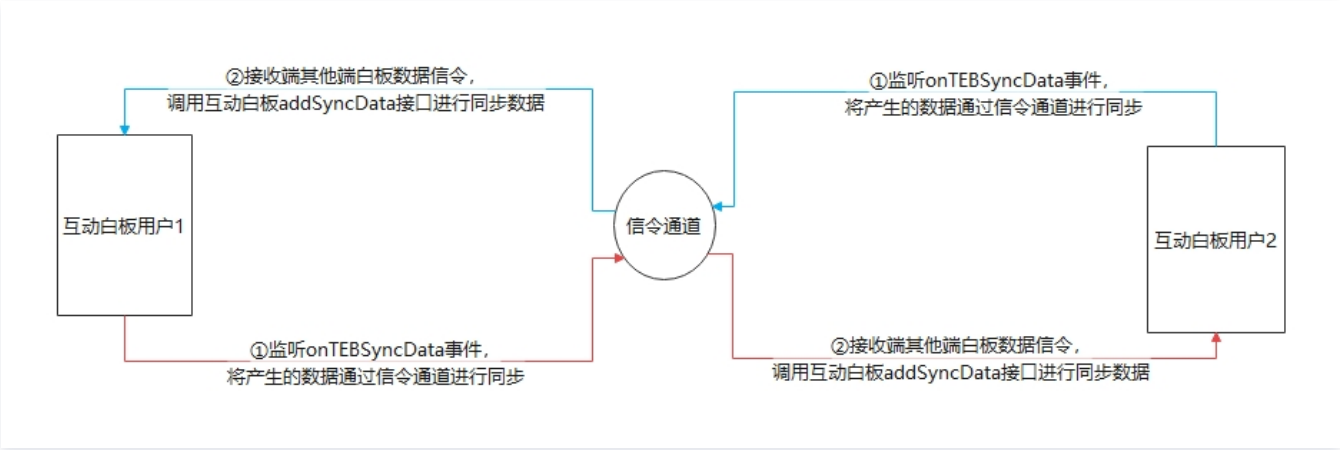
内容介绍

本文主要介绍如何使用自定义信令通道传输白板信令。

平台支持

iOS	Android	Windows	Mac OS	Web	小程序
✓	✓	✓	✓	✓	✓

同步原理



- 监听 onTEBSyncData 事件（事件名以各端实际为准），将事件中回调的数据通过信令通道进行广播。
- 收到白板信令后，调用互动白板 addSyncData 接口（接口名以各端实际为准）进行同步。

注意事项

- 使用自定义信令通道不支持实时录制，白板推流服务。
- 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到onTEBSyncData回调）。
- 互动白板信令通道默认使用腾讯云 即时通信 IM ，使用自定义信令通道则无需集成 IMSDK。

代码集成

Mac/iOS

```
// 1. 将 TEduBoardInitParam 的 timSync 参数初始为 NO
TEduBoardInitParam *initParam = [[TEduBoardInitParam alloc] init];
initParam.timSync = NO; // 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到 onTEBSyncData 回调）
_boardController = [[TEduBoardController alloc] initWithAuthParam:authParam roomId:_classId initParam:initParam];

// 2. 监听白板信令数据回调 onTEBSyncData，将数据发送给其他白板用户
- (void)onTEBSyncData:(NSString *)data {
    //使用自定义信令通道，发送 data 给其他白板用户
    //信令发送成功后调用 addAckData(data)，确认数据发送状态
}

// 3. 在收到其他用户的白板信令时，将消息传递给白板.
[_boardController addSyncData:data];
```

Android

```
// 1. 将 TEduBoardInitParam 的 timSync 参数初始为 NO
TEduBoardController.TEduBoardInitParam initParam = new TEduBoardController.TEduBoardInitParam();
initParam.timSync = false; // 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到 onTEBSyncData 回调）

// 2. 监听白板信令回调 onTEBSyncData，将信令发送给其他白板用户
 @Override
 public void onTEBSyncData(String data) {
    //使用自定义信令通道，发送 data 给其他白板用户
    //信令发送成功后调用 addAckData(data)，确认数据发送状态
 }

// 3. 在收到其他用户的白板信令时，将信令透传给白板
mBoard.addSyncData(data);
```

Windows

```
// 1. 将 TEduBoardInitParam 的 timSync 参数初始为 NO
TEduBoardInitParam initParam;
initParam.timSync = false; // 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到 onTEBSyncData 回调）
boardCtrl->Init(authParam, ROOM_ID, initParam); // 使用上面构造的初始化参数

// 2. 监听白板信令回调 onTEBSyncData，将信令发送给其他白板用户
virtual void onTEBSyncData(const char * data) override {
    //使用自定义信令通道，发送 data 给其他白板用户
    //信令发送成功后调用 addAckData(data)，确认数据发送状态
}

// 3. 在收到其他用户的白板信令时，将信令透传给白板
boardCtrl->AddSyncData(data);
```

Web

```
// 1. 在 onTEBSyncData 回调里，将数据发送给其他白板用户
teduBoard.on(TEduBoard.EVENT.TEB_SYNCDATA, data => {
    //使用自定义信令通道，发送 data 给其他白板用户
    //信令发送成功后调用 addAckData(data)，确认数据发送状态
});

// 2. 在收到其他用户的白板信令时，将信令透传给白板
teduBoard.addSyncData(data);
```

TIM 同步信令通道

最近更新时间：2023-05-24 11:09:58

本文主要介绍使用 TIM 作为同步通道。

平台支持

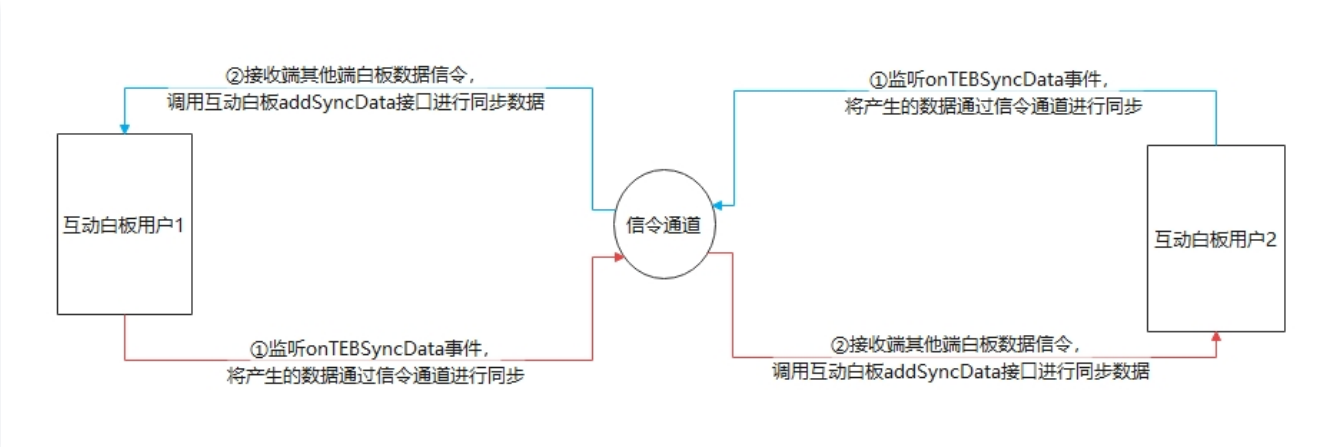
iOS	Android	Windows	Mac OS	Web	小程序
✓	✓	✓	✓	✓	✓

注意事项

因为实时录制会以 TIMSDK 作为信令通道，同时是以自定义消息的 extension:'TXWhiteBoardExt' 为标识，所以在同步信令的时候需要注意以下四点：

1. 使用 TIMSDK 作为信令通道。
2. 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到 onTEBSyncData 回调）。
3. 同步信令的时候发送的消息类型为自定义消息类型。
4. 自定义消息的 extension 字段必须为 TXWhiteBoardExt。

同步原理



- 监听 onTEBSyncData 事件（事件名以各端实际为准），将事件中回调的数据通过信令通道进行广播。
- 收到白板信令后，调用互动白板 addSyncData 接口（接口名以各端实际为准）进行同步。

代码集成

Mac/iOS

> **说明**
> 以下示例为 TIM V2 版本代码，IM 的版本请尽量用新版本，具体请查阅 IM 的 更新日志 。

```
// 1. 将 TEduBoardInitParam 的 timSync 参数初始为 NO（关闭互动白板内置的信令通道）
TEduBoardInitParam *initParam = [[TEduBoardInitParam alloc] init];
initParam.timSync = NO;  // 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到onTEBSyncData回调）
_boardController = [[TEduBoardController alloc] initWithAuthParam:authParam roomId:_classId initParam:initParam];

// 2. 监听白板信令数据回调 onTEBSyncData，将数据发送给其他白板用户
- (void)onTEBSyncData:(NSString *)data {
    V2TIMMessage *message = [[V2TIMManager sharedInstance] createCustomMessage:data desc:nil
extension:@"TXWhiteBoardExt"];
    if (message.customElem) {
        message.customElem.extension = @"TXWhiteBoardExt";
    }
    [[V2TIMManager sharedInstance] getConversation:@"groupid" succ:^(V2TIMConversation *conv) {
```

```
    if ([conv type] == V2TIM_GROUP) {
        BOOL onlineUserOnly = ![[conv.groupType lowercaseString] isEqualToString:@"avchatroom"];
        [[V2TIMManager sharedInstance] sendMessage: message
                                          receiver: nil
                                           groupID: conv.groupID
                                          priority: V2TIM_PRIORITY_HIGH
                                    onlineUserOnly:onlineUserOnly
                                   offlinePushInfo:nil
                                          progress:nil succ:^{
            // 发送 IM 消息成功
        } fail:^(int code, NSString *desc) {
            // 发送 IM 消息失败，建议进行重试
        }];
    }

    } fail:^(int code, NSString *desc) {
    // 获取回话失败
    }];

}

// 3. 监听IM的消息回调，在收到其他用户的白板信令时，将消息传递给白板
// 注意: 自己操作触发的信令，不需要再同步给本人
[_boardController addSyncData:data];
```

Android

> **说明**
> 以下示例为 TIM V2 版本代码，IM 的版本请尽量用新版本，具体请查阅 IM 的 更新日志 。

```
// 1. 将 TEduBoardInitParam 的 timSync 参数初始为 NO（关闭互动白板内置的信令通道）
TEduBoardController.TEduBoardInitParam initParam = new TEduBoardController.TEduBoardInitParam();
initParam.timSync = false; // 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到onTEBSyncData回调）

// 2. 监听白板信令回调 onTEBSyncData，将信令发送给其他白板用户
 @Override
  public void onTEBSyncData(String data) {
    final V2TIMMessage message = V2TIMManager.getMessageManager().createCustomMessage(data.getBytes(), "",
"TXWhiteBoardExt".getBytes());
    if (message.getCustomElem() != null) {
        message.getCustomElem().setExtension("TXWhiteBoardExt".getBytes());
    }
    V2TIMManager.getConversationManager().getConversation("groupid", new V2TIMValueCallback<V2TIMConversation>() {
        @Override
        public void onSuccess(V2TIMConversation v2TIMConversation) {
            if (v2TIMConversation.getType() == V2TIMConversation.V2TIM_GROUP) {
                boolean onlineUserOnly = !(v2TIMConversation.getGroupType().toLowerCase().equals("avchatroom"));
                V2TIMManager.getMessageManager().sendMessage(message, null, "groupid", 1, onlineUserOnly, null,  new
V2TIMSendCallback<V2TIMMessage>() {
                    @Override
                    public void onSuccess(V2TIMMessage v2TIMMessage) {
                        // 发送 IM 消息成功
                    }

                    @Override
                    public void onError(int i, String s) {
                        // 发送 IM 消息失败，建议进行重试
                    }

                    @Override
                    public void onProgress(int i) {
```

```
                }
            });
        }
    }

    @Override
    public void onError(int i, String s) {
        // 获取回话失败
    }
  });
}

// 3. 在收到其他用户的白板信令时，将信令透传给白板
// 注意: 自己操作触发的信令，不需要再同步给本人
mBoard.addSyncData(data);
```

Windows

以下代码为演示代码，最新 TIM 相关接口请参考 TIM V2 创建自定义消息， TIM V2 消息发送， TIM V2 消息回调，按以下步骤进行接入。

```
// 引入IM SDK头文件
#include "TIMCloud.h"

// 这里为了演示方便，使用ostringstream来构造JSON串，生产环境建议使用第三方JSON库来生成JSON串
#include <iostream>
#include <sstream>
#include <string>


// 1. 将 TEduBoardInitParam 的 timSync 参数初始为 NO（关闭互动白板内置的信令通道）
TEduBoardInitParam initParam;
initParam.timSync = false; // 关闭互动白板内置的信令通道（一定需要关闭，否则会收不到onTEBSyncData回调）
boardCtrl->Init(authParam, ROOM_ID, initParam); // 使用上面构造的初始化参数

// 2. 监听白板信令回调 onTEBSyncData，将信令发送给其他白板用户
virtual void onTEBSyncData(const char * data) override {
   //使用自定义信令通道，发送 data 给其他白板用户
  std::string message = data;
  std::ostringstream json;
  json << "{";
  json << "\"" << kTIMMsgElemArray << "\":";
  json << "[{";
  json << "\"" << kTIMMsgPriority << "\": " << kTIMMsgPriority_High << ",";  // 设置消息优先级为高
  json << "\"" << kTIMElemType << "\": " << kTIMElem_Custom << ",";  // 消息类型为自定义消息
  json << "\"" << kTIMCustomElemExt << "\": \"TXWhiteBoardExt\",";  // 扩展字段信息
  json << "\"" << kTIMCustomElemData << "\": \"" << message << "\",";  // 消息内容为白板数据
  json << "}]";
  json << "}";
  int ret = TIMMsgSendNewMsg("课堂id", kTIMConv_Group, json.str().c_str(), [](int32_t code, const char *desc, const char
*json_param, const void *user_data) {
     if (ERR_SUCC == code) {  // 消息发送成功
       //信令发送成功后调用 addAckData(data)，确认数据发送状态
     } else {  // 消息发送失败，建议进行重试
     }
  }, nullptr);
  if (ERR_SUCC != ret) {  // 消息发送失败，建议进行重试

  }
}

// 3. 在收到其他用户的白板信令时，将信令透传给白板
// 注意: 自己操作触发的信令，不需要再同步给本人
boardCtrl->AddSyncData(data);
```

Web

以下代码为演示代码，最新 TIM 相关接口请参考 TIM 创建自定义消息 ， TIM 消息发送 ， TIM 消息回调 ，按以下步骤进行接入。

```
// Web没有内置TIM通道，不需要额外关闭内置TIM通道。
// 1. 在 onTEBSyncData 回调里，将数据发送给其他白板用户
teduBoard.on(TEduBoard.EVENT.TEB_SYNCDATA, data => {
  const message = tim.createCustomMessage({
    to: '课堂ID',
    conversationType: TIM.TYPES.CONV_GROUP,
    priority: TIM.TYPES.MSG_PRIORITY_HIGH,  // 因为im消息有限频，白板消息的优先级调整为最高
    payload: {
        data: JSON.stringify(data),
        description: '',
        extension: 'TXWhiteBoardExt',
    },
  });
  // 发送消息
  tim.sendMessage(message).then(() => {
    // 发送成功
    //信令发送成功后调用 addAckData(data)，确认数据发送状态
  }, (error) => {
    // 发送失败，建议进行重试
  });
});
// 2. 监听im的消息接收事件，在收到其他用户的白板信令时，将信令透传给白板（addSyncData）
// 注意: 自己操作触发的信令，不需要再同步给本人；以下代码this.im，this.userId，teduBoard请以实际的业务变量为准
this.tim.on(TIM.EVENT.MESSAGE_RECEIVED, (event) => {
  const messages = event.data;
  const groupId = String(this.classInfo.classId);
  messages.forEach((message) => {
    // 群组消息
    if (message.conversationType === TIM.TYPES.CONV_GROUP) {
      if (message.to === groupId) { // 如果是当前群组
        const elements = message.getElements();
        if (elements.length) {
          elements.forEach(async (element) => {
            if (element.type === 'TIMCustomElem') { // 自定义消息
              if (element.content.extension === 'TXWhiteBoardExt') { // 是白板的自定义消息
                if (message.from != this.userId) { // 并且发消息的人不是自己
                  // 将白板信令设置给白板
                  teduBoard.addSyncData(data);
                }
              }
            }
          });
        }
      } else {
        // 其他群组消息自行处理，在互动白板的场景中可以忽略其他群组的消息
      }
    } else if (message.conversationType === TIM.TYPES.CONV_C2C) { // C2C消息
      // c2c消息在互动白板的场景中可以直接忽略
    }
  });
});
```

备份域名使用

最近更新时间：2023-02-03 15:02:25

内容介绍

本文主要介绍如何使用资源备份域名来提高资源加载成功率。

平台和版本支持

平台类型	Android	Windows	iOS	Mac OS	Web
平台版本	✓	✓	11.4+	11.0+	✓
TEduBoardSDK版本	2.7.1.153+	2.7.1.242+	2.7.1.100+	2.7.1.100+	2.7.1+
TIWCache版本	1.0.0.83+	✓	1.0.0.72+	1.0.0.72+	✖

备份域名的使用目的

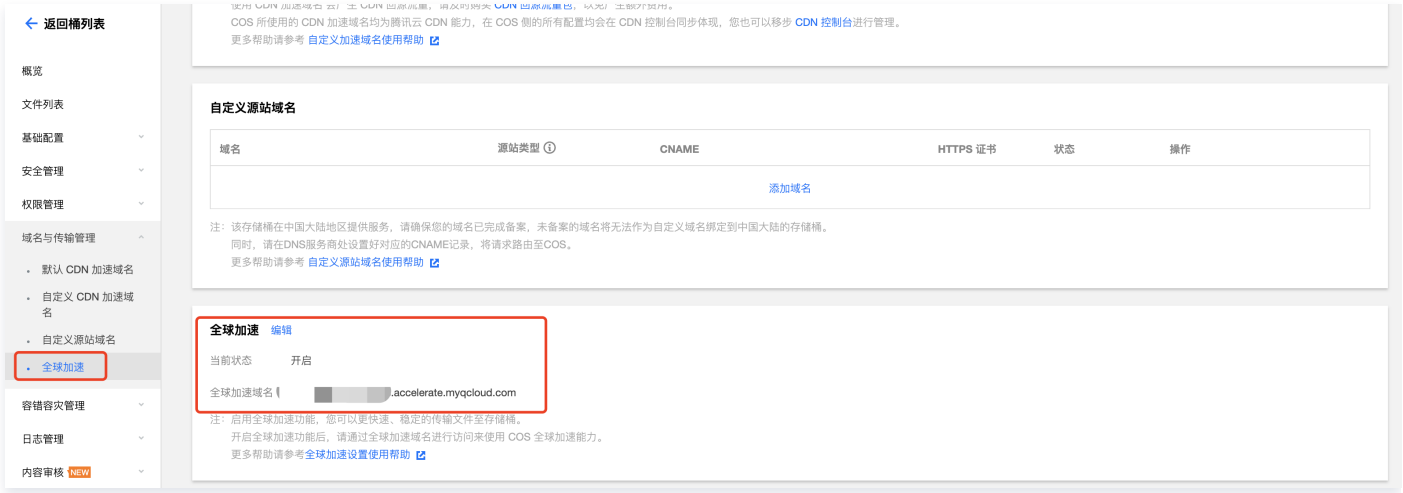
一些用户因为网络原因，导致无法加载资源或者资源加载超时、加载失败；在主域名加载资源失败的情况下；通过备份域名加载资源，从而提高资源加载成功率。

准备资源备份域名

> **说明**
> 本文以课件存储在腾讯云 Cos 桶为例。

1. 开通 Cos 桶默认的 CDN 加速域名，也可以自定义 CDN 加速域名。

2. 开通 Cos 桶全球加速域名。



3. 获取 Cos 源站域名。

调用 addBackupDomain 接口设置备份域名

> **说明**
> addBackupDomain接口详情，此接口建议在new TEduBoard()后就调用。

一般情况下我们会使用资源的 CDN 为主域名，此时就可以将 Cos 的全球加速和源站作为备份域名使用。
推荐域名优先级: CDN加速域名 > Cos全球加速 > Cos源站域名

```
const teduBoard = new TEduBoard({});
teduBoard.addBackupDomain('https://test-1259648581.file.myqcloud.com', 'https://test-1259648581.cos.accelerate.myqcloud.com');
teduBoard.addBackupDomain('https://test-1259648581.file.myqcloud.com', 'https://test-1259648581.cos.ap-shanghai.myqcloud.com');
```

白板操作权限控制

最近更新时间：2023-04-06 11:28:37

功能简介

互动白板默认所有权限校验都是不启用，即所有用户都是具有操作互动白板的能力；如业务场景中需要对用户权限进行限制，则可以参考本文档进行用户操作权限控制。

版本支持

iOS	Android	Windows	Mac OS	Web	小程序
2.8.0.170+	2.8.0.216+	2.8.0.317+	2.8.0.154+	2.8.0+	2.8.0+

使用指南

互动白板提供了两种权限控制方式，包含基础权限控制和高级权限控制，具体请看下面说明。

基础权限控制

基础权限是互动白板内置的一套权限控制，通过初始化参数 drawEnable 和 setDrawEnable 接口实现，主要包含以下权限：

权限名	描述
Element::Add::*	新增元素权限，如画笔工具，几何图形工具，以及调用addElement接口
Element::Delete::*	删除元素权限
Element::Select::*	选中元素权限
Element::Move::*	移动元素权限
Element::Update::*	更新元素权限，如改变元素颜色，大小等
Element::Scale::*	缩放元素权限
Element::Rotate::*	旋转元素权限
Background::Update::*	更新背景图权限
Board::Switch::*	翻页白板权限
Board::Clear::*	清空白板元素权限，对应clear接口
File::Clear::*	清空文件内的元素权限，对应clearFileDraws接口

如不允许用户操作白板，则可以在初始化参数将 drawEnable 设置 false，也可以调用 setDrawEnable(false) 来实现，当在使用互动白板过程中，则需要授予用户A操作白板的权限，则可以让用户 A 调用 setDrawEnable(true) 来实现。
如您想了解 drawEnable 的实现原理，请在阅读完高级权限控制后，再查看 附录1：drawEnable的实现原理 章节。

高级权限控制

如果基础权限控制不能满足业务场景，需要更精准的权限控制，则可以使用高级权限控制接口来实现。
默认场景下，用户是具有所有白板的操作权限，如需要限制用户的某一个操作，则启用对应的接口进行权限校验来实现。

启用操作白板权限校验

1. 该接口只对本地设置有效。
2. 调用该接口，设置项对应的权限将启用校验。只有校验通过，设置项对应的白板操作才允许执行。

enablePermissionChecker(permissions, filters)

参数名	类型	描述
permissions	Array.<string>	权限列表,可支持通配符 *，查阅目前支持的权限

filters	Array.<string>	条件列表,可支持通配符 *，查阅目前支持的条件

示例1：房间里有教师T、学生A。赋予 T 文件，白板，元素操作权限，禁用 A 的全部权限。
teduBoard.enablePermissionChecker(['File::*::*', 'Board::*::*', 'Element::*::*'], ['operator/T']); // T和A都调用
解读：
1.当老师T添加课件的时候('File::*::*')，则会根据权限配置进行校验，对应的操作权限只有用户T('operator/T)，此时添加课件的用户是T,权限配置也是允许T添加，则此时权限校验通过，T能正常添加课件。
2.当学生A添加课件的时候('File::*::*')，则会根据权限配置进行校验，对应的操作权限只有用户T('operator/T)，此时添加课件的用户是A,但权限配置只允许A添加，此时权限校验不通过，A不能添加课件，并触发 TEB_BOARD_PERMISSION_DENIED 无操作权限的事件。

禁用操作白板的权限校验

1. 该接口只对本地设置有效。
2. 调用该接口，设置项对应的权限将不做校验，权限将恢复成默认开启，调用者将获得对应的操作权限。

disablePermissionChecker(permissions)

参数名	类型	描述
permissions	Array.<string>	权限列表,可支持通配符 *，查阅目前支持的权限

示例2：接着上面的示例1：学生 A 此时是没有权限添加课件，添加白板，添加元素（画笔，直线等）的权限的。
1.如果此时需要允许学生 A 可以使用画笔功能，则学生 A 可以调用 teduBoard.disablePermissionChecker(['Element::*::*']) 接口来关闭对元素操作的校验。
解读：对元素操作的校验权限禁用，则对元素操作的时候不校验权限，不校验权限就是默认有操作元素的权限，这样就实现了学生A可以实现操作画笔的需求。
2.如果此时需要允许学生 A 可以添加课件，则学生 A 可以调用 teduBoard.disablePermissionChecker(['File::*::*']) 接口来关闭对文件操作的校验。
解读：对文件操作的校验权限禁用，则对文件操作的时候不校验权限，不校验权限就是默认有操作文件的权限，这样就实现了学生A可以实现操作文件的需求。

权限变更事件

调用 enablePermissionChecker 和 disablePermissionChecker 接口会触发该事件。

```
// web端示例
teduBoard.on(TEduBoard.EVENT.TEB_BOARD_PERMISSION_CHANGED, (permissions, filters) => {
});
```

无操作权限事件

当用户操作白板没有对应的操作权限时，则会触发该事件。

```
// web端示例
teduBoard.on(TEduBoard.EVENT.TEB_BOARD_PERMISSION_DENIED, (permission) => {});
```

权限列表

接口名	权限名	描述
setGlobalBackgroundColor	Background::Update::Color	设置全局背景色
setBackgroundColor	Background::Update::Color	设置单页白板背景色
setGlobalBackgroundPic	Background::Update::Image	设置全局背景图
clear	Board::Clear	清空白板
setBackgroundImage	Background::Update::Image	设置单页白板背景图

setBackgroundH5	Background::Update::Frame	设置单页白板背景h5
undo	Board::Undo::*	撤销
redo	Board::Redo::*	恢复
addBoard	Board::Add	新增白板
deleteBoard	Board::Delete	删除白板
prevStep	Board::Switch::Step	上一步
nextStep	Board::Switch::Step	下一步
prevBoard	Board::Switch::Page	上一页
nextBoard	Board::Switch::Page	下一页
gotoBoard	Board::Switch::Page	翻页
setBoardRatio	Board::Update::Ratio	设置白板比例
setBoardScale	Board::Scale::*	设置白板缩放
gotoStep	Board::Switch::Step	跳转动画步数
setBoardContentFitMode	Board::Update::ContentFitMode	设置白板内容填充方式
addTranscodeFile	File::Add	新增转码课件
addImagesFile	File::Add	新增图片组课件
deleteFile	File::Delete	删除课件
switchFile	File::Switch	切换课件
clearFileDraws	File::Clear	清空文件上的涂鸦
addVideoFile	File::Add	新增视频课件
playVideo	File::Update::Video	播放视频课件
pauseVideo	File::Update::Video	暂停视频课件
seekVideo	File::Update::Video	调整视频课件进度
muteVideo	File::Update::Video	静音视频课件
resetVideoProgress	File::Update::Video	重置视频课件进度
playAudio	File::Update::Audio	播放音频元素
pauseAudio	File::Update::Audio	暂停音频元素
seekAudio	File::Update::Audio	调整音频元素进度
muteAudio	File::Update::Audio	静音音频元素
setAudioVolume	File::Update::Audio	设置音频元素音量
setFileScale	File::Update::Scale	对文件进行缩放
addH5File	File::Add	新增h5文件
removeElement	Element::Delete	删除元素
addElement	Element::Add	新增元素，包含使用画笔工具，几何工具，调用addElement接口等
setTextValue	Element::Update	设置文本元素内容

useMathTool	Element::Add::MathTool	使用数学教具
updateElementById	Element::Update	更新元素位置，大小，颜色等

条件列表

条件名	类型	描述
operator	string	操作人
creator	string	创建人

```
// 示例一：允许操作人userId是T，A，B三个用户
['operator/T,A,B'] // 配置具名的userId

// 示例二：允许所有操作人
['operator/*'] // 配置通配符

// 示例三：禁止所有人操作
['operator/'] // 不配置userId，则表示所有人都匹配不上

// 示例四：允许操作创建人是A,B两个用户添加的文件或者添加的元素
['creator/A,B']

// 示例五：允许操作所有创建人添加的文件或者添加的元素
['creator/*']
```

使用示例

```
### 场景1
房间里有教师T、学生A。赋予 T 文件，白板，元素操作权限，禁用 A 的全部权限
/** T/A 调用 */
enablePermissionChecker(['File::*::*', 'Board::*::*', 'Element::*::*'], ['operator/T']);

### 场景2
房间里有教师T、学生A、学生B。TAB 均赋予元素的操作权限
/** 方式一 */
// 三人均设置
enablePermissionChecker(['Element::*::*'], ['operator/A,B,T']);
/** 方式二 */
// T 调用
enablePermissionChecker(['Element::*::*'], ['operator/T']);
// A 调用
enablePermissionChecker(['Element::*::*'], ['operator/A']);
// B 调用
enablePermissionChecker(['Element::*::*'], ['operator/B']);

### 场景3
房间里有教师T、学生A、学生B。T能删除(包括调用接口removeElement，橡皮擦，键盘del键)所有人创建的元素（T自己、A、B），A只能删
除A自己创建的元素，不能删除其他人的创建的元素，B不能删除任何人的元素；
// T 调用
enablePermissionChecker(['Element::Delete::*'], ['creator/*']);
// A 调用
enablePermissionChecker(['Element::Delete::*'], ['creator/A']);
// B 调用
enablePermissionChecker(['Element::Delete::*'], ['creator/']);


### 场景4
房间里有学生A、学生B、学生C。只允许 A 删除 A 和 B 创建的元素，不能删除 C 创建元素；
```

```
// A B C用户均按下面方式设置
enablePermissionChecker(['Element::Delete::*'], ['operator/A', 'creator/A,B']);


### 场景5
房间里有教师T、学生A、学生B。T 只能操作 B 创建的元素，
// T 调用
enablePermissionChecker(['Element::*::*'], ['creator/B']);

### 场景6
禁用当前用户的全部操作
// 所有用户调用
enablePermissionChecker(['*::*::*'], ['operator/']);
// 'operator/' 表示对于操作者（operator）这个条件，符合的值是空的，意味着没有操作者能够被允许通过这个权限校验

### 场景7
启用当前用户的全部操作
// 所有用户调用
enablePermissionChecker(['*::*::*'], ['operator/*']);
// 'operator/*' 表示对于操作者（operator）这个条件，符合的值是任意的，意味着所有操作者能够被允许通过这个权限校验

### 场景8
房间里有学生A。学生A允许跳转动画步数，但不能翻页
// A 调用
enablePermissionChecker(['Board::Switch::Step'], ['operator/A']);
// A 调用
enablePermissionChecker(['Board::Switch::Page'], ['operator/']);

### 场景8
房间内有学生A。不允许学生A操作尺规工具，只允许观看
// A 调用
enablePermissionChecker(['Element::Update::MathTool'], ['operator/']);
```

附录1：drawEnable的实现原理

初始化参数drawEnable和接口setDrawEnable包含以下权限：

```
'Element::Add::*','Element::Delete::*','Element::Move::*','Element::Select::*','Element::Update::*',
'Element::Scale::*','Element::Rotate::*','Background::Update::*','Board::Switch::*',
'Board::Clear::*','File::Clear::*'
```

当用户A drawEnable设置为 true 时，则等价于执行以下权限配置：

```
const permissions = ['Element::Add::*','Element::Delete::*','Element::Move::*','Element::Select::*',
'Element::Update::*','Element::Scale::*','Element::Rotate::*','Background::Update::*',
'Board::Switch::*', 'Board::Clear::*','File::Clear::*'];

teduBoard.enablePermissionChecker(permissions, ['operator/A', 'creator/A']); // A 用户具有以上操作权限
```

当用户A drawEnable设置为 false 时，则等价于执行以下权限配置：

```
const permissions = ['Element::Add::*','Element::Delete::*','Element::Move::*','Element::Select::*',
'Element::Update::*','Element::Scale::*','Element::Rotate::*','Background::Update::*',
'Board::Switch::*', 'Board::Clear::*','File::Clear::*'];

teduBoard.enablePermissionChecker(permissions, ['operator/', 'creator/']); // 没有用户拥有以上权限
```