H5课件

简介

上传入口

- 1.【教室内上传压缩包】上传入口位于动态课件上传入口（目前仅支持zip格式且压缩包根目录中必须包含index.html的入口文件）；
- 2.【通过API接口创建教室绑定H5课件地址】详情请联系百家云技术支持人员；

注意事项

- 1.为避免造成服务器压力，尽量不要短时间内频繁操作课件
- 2.不符合压缩包格式和内容的H5课件将无法正常使用

通信机制

`BJYBridge`提供了iframe和外层交互的桥梁，你可以通过`BJYBridge`实例来实现自己的动态PPT。`BJYBridge`是单例模式，一个页面中只能存在一个`bridge`实例。
请引入以下地址的桥接文件:

```
1. https://live-cdn.baijiayun.com/js-sdk/tool/BJYBridge.js
```

BJYBridge

为了保证教室所有人的处于相同状态，操作课件需要先向服务器发送操作的请求，在得到服务器响应之后才进行实际操作。`BJYBridge`的使用过程大致如下: * 使用`getInstance`方法获取单例，在传入的参数中需要指定`onPageChange`和`onRecordChange`两个回调函数，分别用于在服务器响应了翻页请求和操作请求之后进行具体的业务操作； * 在课件准备就绪之后，调用`getReady`方法，并传入当前的页码、步骤、总页数、总步数信息； * 在需要翻页的时候，调用`requestChangePage`方法。注意：这里只是通知服务器将要翻页，具体翻页动作请在`onPageChange`的回调之中执行； * 在需要进行一些特殊操作如播放动画时，调用`pushRecord`方法。注意：同上，具体的业务处理请在`onRecordChange`回调之中执行

Variables

- `page` [number] 课件的页码
- `step` [number] 课件的步骤
- `pageCount` [number] 课件的总页码
- `stepCount` [number] 课件当前页的总步数

Functions

`static getInstance(options: BridgeOptions):Bridge`

静态函数，返回BJYBridge的唯一实例

```
// 获取Bridge单例
var bridge = BJYBridge.getInstance({
    // 响应由bridge通知而来的翻页指令
    onPageChange: function (page, step) {
        // 需要在此回调函数中返回一个promise对象。
        return new Promise(resolve => {
            // 执行具体的翻页逻辑，并在完成后将当前页的步数传递给Promise链
            // YOUR CODE HERE
            resolve(stepCount);
        });
    },
    // 响应由bridge通知而来的本页操作记录变化
    onRecordChange: function (record, prevRecord) {
        // 根据具体业务响应操作变化
        // 数组的末尾为最新的一条操作记录，你可以根据两个数组的长度得知是否为撤销操作
    }
});
```

getReady(data: InitData):void

初始化，当课件准备就绪时应该调用此函数

```
// 传入初始的页码、步骤、总页数、总步数信息
// 如果你的课件总页数（如同链表）无法得知，请pageCount传入0
bridge.getReady({
    page: 0,
    step: 0,
    pageCount: 1,
    stepCount: 1
});
```

requestChangePage(page: targetPage, step: targetStep):void

向服务器发送翻页的请求

```
// 注意：这里只是发送请求，具体执行操作请在onPageChange的回调之中
bridge.requestChangePage(0, 0);
```

pushRecord(record: anyData):void

向服务器添加一条当前页的操作记录

```
// 注意：这里只是发送请求，具体执行操作请在onRecordChange的回调之中
// 此方法可接受任何类型的参数
bridge.pushRecord({
    name: 'click',
    x: '0.123',
    y：'0.123'
});
```

popRecord():recordRemoved

请求服务器移除栈顶的一条操作记录

```
bridge.popRecord();
```

clearRecordStack():void

请求服务器移清空当前页的操作记录，每次页码改变时`BJYBridge`内部已经将操作记录清空，通常情况下你不会用到这个方法。

```
bridge.clearRecordStack();
```

replaceRecordStack():void

全量替换当前的操作记录栈，如果你不需要操作历史记录，请使用`replaceRecordStack`而非`pushRecord`。

```
bridge.replaceRecordStack();
```

完整示例

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>PPT Demo</title>
    <style>
        html,
        body {
            margin: 0;
            padding: 0;
            width: 100%;
            height: 100%;
            background: #ddd;
        }

        body {
            display: flex;
            flex-direction: column;
            align-items: center;
            justify-content: center;
            width: 100%;
            height: 100%;
        }

        .icon {
            display: inline-block;
            width: 30px;
            height: 30px;
            margin: 0 30px;
            border-radius: 50%;
            text-align: center;
            line-height: 30px;
            background-color: rgba(0, 0, 0, .5);
            color: #fff;
            cursor: pointer;
        }

        .icon + .icon {
            margin-left: 30px;
        }
```

```
41.
42.     .main {
43.         display: flex;
44.         flex-direction: column;
45.         align-items: center;
46.         flex: 4;
47.         width: 100%;
48.     }
49.
50.     .content {
51.         flex: 1;
52.         display: flex;
53.         align-items: center;
54.         justify-content: center;
55.         width: 80%;
56.         font-size: 30px;
57.         color: #666;
58.     }
59.
60.     .record {
61.         flex: 2;
62.         box-sizing: border-box;
63.         width: 80%;
64.         padding: 5px 12px;
65.         border: 1px solid #999;
66.         border-radius: 4px;
67.         overflow: scroll;
68.         font-size: 12px;
69.         line-height: 1.25;
70.         color: #999;
71.     }
72.
73.     .footer {
74.         flex: 1;
75.         display: flex;
76.         align-items: center;
77.         justify-content: center;
78.         width: 100%;
79.     }
80.   </style>
81. </head>
82. <body>
83.
84.   <div class='main'>
85.     <span class="content"></span>
86.     <span class='record'></span>
87.   </div>
88.   <div class='footer'>
89.     <span class="icon prev"><</span>
90.     <span class="icon next">></span>
91.   </div>
92.
```

```
    <script src='//live-cdn.baijiayun.com/js-sdk/tool/BJYBridge.js'></script>
    <script>
        // PPT每页内容假数据
        var MOCK_DATA = [
            [
                'Page0-Step0',
                'Page0-Step1',
                'Page0-Step2',
                'Page0-Step3',
                'Page0-Step4'
            ],
            [
                'Page1-Step0',
                'Page1-Step1',
                'Page1-Step2',
            ],
            [
                'Page2-Step0',
                'Page2-Step1',
                'Page2-Step2'
            ],
            [
                'Page3-Step0'
            ],
            [
                'Page4-Step0'
            ],
        ];

        window.onload = function () {
            // PPT内容区域
            var contentElement = document.querySelector('.content');
            // 操作记录区域
            var recordElement = document.querySelector('.record');

            // 获取Bridge单例
            var bridge = BJYBridge.getInstance({
                // 响应由bridge通知而来的翻页指令
                onPageChange: function (page, step) {
                    return new Promise(
                        function (resolve) {
                            var pageData = MOCK_DATA[page];
                            if (pageData) {
                                // 根据具体业务调用翻页的方法
                                contentElement.innerText = pageData[step];
                                // 翻页完成之后，将当前页的步数传递给Promise链
                                resolve(pageData.length);
                            }
                            else {
                                contentElement.innerText = 'Error: Illegal Page';
                            }
```

```
                }
            }
        );
    },
    // 响应由bridge通知而来的本页操作记录变化
    onRecordChange: function (record, prevRecord) {
        // 根据具体业务响应操作变化
        recordElement.innerHTML = record.reverse().join('<br>');
    }
});

// 在课件加载完成之后，调用bridge的getReady函数传入初始的页码、步骤、总页数、总步数信息
// 在实际场景中，此调用可能写在一个PPT加载完成的回调之中
new Promise(
    function (resolve) {
        var initData = {
            page: 0,
            step: 0
        };
        contentElement.innerText = MOCK_DATA[initData.page][initData.step];
        resolve(initData);
    }
).then(
    function (initData) {
        var page = initData.page;
        var step = initData.step;
        bridge.getReady({
            page: page,
            step: step,
            pageCount: MOCK_DATA.length,
            stepCount: MOCK_DATA[page].length
        });
    }
);

// 向前翻页
document.querySelector('.icon.prev').addEventListener(
    'click',
    function () {
        var page = bridge.page;
        var step = bridge.step;

        if (--step < 0) {
            page--;
            step = 0;
        }

        if (page >= 0) {
            bridge.requestChangePage(page, step);
        }
    }
```

```
        );

        // 向后翻页
        document.querySelector('.icon.next').addEventListener(
            'click',
            function () {
                var page = bridge.page;
                var step = bridge.step;

                if (++step >= bridge.stepCount) {
                    page++;
                    step = 0;
                }

                if (page < bridge.pageCount) {
                    bridge.requestChangePage(page, step);
                }
            }
        );

        // 操作PPT
        recordElement.addEventListener(
            'click',
            function () {
                bridge.pushRecord(new Date().toString());
            }
        );
    };
  </script>
</body>
</html>

```