H5课件

简介

上传入口

- 1.H5课件上传入口位于动态课件上传入口

注意事项

- 1.教室内同时仅支持打开一个H5课件
- 2.为避免造成服务器压力，尽量不要短时间内频繁操作课件
- 3.不符合压缩包格式和内容的H5课件将无法正常使用

压缩包格式

- 1.H5课件需以.zip压缩包格式上传
- 2.压缩包中必须包含index.html的静态页面文件，此文件将被作为H5课件的入口，需位于根目录
- 3.支持为H5课件提供预览图，预览图命名cover.png，格式必须为.png格式，需位于根目录（预览图大小建议范围160px×90px~320px×180px）
- 4.H5课件压缩包限制大小为50M

通信机制

桌面浏览器端

> H5课件作为`iframe`嵌入至直播页面，采用`postMessage`的方式进行相互通信。下文将称直播教室为父页面，H5课件为子页面。

我们为子页面提供了以下与直播服务器进行通信的机制：

- *发送消息*：子页面可通过父页面向服务器发送数据，此数据将被转发至教室内的所有人，并被缓存至服务器；
- *接受消息*：子页面可监听到由服务器转发的数据；
- *获取缓存*：子页面可主动调取缓存在服务器的数据。

在使用`postMessage`方法时，传输数据的格式必须满足以下形式。其中`name`和`id`为必需字段，`getCache`若为`true`，`value`将被忽略。

```
{
   // 必需字段 固定为bjy_h5_doc，用于标识此消息用于H5课件
   name: 'bjy_h5_doc',
   // 必需字段 由location.search中的docId字段获得，用于区分不同的iframe容器
   id: '0',
   // 是否为获取缓存数据
   getCache: true/false,
   // value字段为自定义字段，父页面将透传此字段
   value: {
      yourData: 'hello world'
   }
}
```

发送消息

向父页面发送消息的函数大致如下：

```
function sendMessage(value) {
    var message = {
        name: 'bjy_h5_doc',
        id: '0'
    };

    if (value) {
        message.value = value
    }
    // 如果message中的getCache字段为true，表示为从服务器拉取数据缓存
    else {
        message.getCache = true;
    }

    // 处于移动端webview中
    if (window.bridge) {
        // 安卓端发送stringify后的数据
        if (navigator.userAgent.toLowerCase().indexOf("android")) {
            message = JSON.stringify(message);
        }
        window.bridge.send(message);
    } else {
        window.parent && window.parent.postMessage(message, '*');
    }
};
```

WARNING: 发送消息的频率以及消息的长度存在限制，意味着消息可能被拒发。当消息被拒绝转发时，父页面会通`postMessage`将异常信息通知子页面，建议捕获异常并进行合适的处理。异常信息的格式如下：

```
{
    // 'BroadcastException'
    name: {string},
    // 2|3
    // 依次表示： 消息体积过大|发送频率过高
    code: {number},
    // 'Size exceeded'|'Frequency exceeded'
    message: {strinig}
}
```

接收消息

子页面通过监听`message`事件，收取来自服务器的数据:

```
window.onmessage = function (message) {
    if (message.name === 'bjy_h5_doc') {
        // 注意，此条消息有可能是父页面拒绝转发消息的异常通知
        if (message.exception) {
            console.warn(message.exception);
        }
        // value字段为sendMessage中传入的数据
        else if (message.value) {
            // 处理子页面内部逻辑
        }
    }
};
```

Android端

后续提供

iOS端

后续提供

入口文件

在压缩包中，必须包含一个名为`index.html`的静态页面文件，此文件将被作为H5课件的入口文件。以下是一个最简单的入口文件示例：

```
<body>
    <span class="icon prev"></span>
    <span class="step-info">0</span>
    <span class="icon next"></span>
</body>
```

```
// 约定与父页面通信的字段标识
var KEY_NAME = 'bjy_h5_doc';

var isInWebview = !!window.bridge;
var id = parseQuery(location.search).id;
var stepInfo = document.querySelector('.step-info');

function parseQuery(queryStr) {
    var result = {};

    if (typeof queryStr === 'string' && queryStr.indexOf('=') >= 0) {
        var firstChar = queryStr.charAt(0);
        var startIndex = (firstChar === '?' || firstChar === '#') ? 1 : 0;
        if (startIndex > 0) {
            queryStr = queryStr.substr(startIndex);
        }

        queryStr.split('&').forEach(
            function(item, index) {
                var terms = item.split('=');
                if (terms.length === 2) {
                    result[terms[0]] = decodeURIComponent(terms[1]);
                }
            }
```

```
        );
    }

    return result;
}

function sendMessage(value) {
    var message = {
        name: KEY_NAME
    };

    if (value instanceof Object) {
        message.id = id;
        message.value = value;
    }
    else {
        message.getCache = true;
    }

    // 处于移动端webview中
    if (isInWebview) {
        // 安卓端发送stringify后的数据
        if (navigator.userAgent.toLowerCase().indexOf("android")) {
            message = JSON.stringify(message);
        }
        window.bridge.send(message);
    } else {
        window.parent && window.parent.postMessage(message, '*');
    }
}

function messageReceiveHandler(event) {
    // window.bridge.receive的参数直接就是stringify的数据
    var message = event.data || event;
    if (message.name === KEY_NAME) {
        if (message.exception) {
            console.warn(message.exception);
        }
        else if (message.value) {
            stepInfo.innerText = message.value.step;
        }
    }
}

if (isInWebview) {
    window.bridge.receive = messageReceiveHandler;
    window.bridge.onReady = function () {
        sendMessage();
    };
}
else {
    window.onmessage = messageReceiveHandler;
}
```

```
77. }
78.
79. window.onload = function () {
80.     if (!isInWebview) {
81.         sendMessage();
82.     }
83.
84.     document.querySelector('.icon.prev').addEventListener(
85.         'click',
86.         function () {
87.             sendMessage({
88.                 step: +stepInfo.innerText - 1
89.             })
90.         }
91.     );
92.
93.     document.querySelector('.icon.next').addEventListener(
94.         'click',
95.         function () {
96.             sendMessage({
97.                 step: +stepInfo.innerText + 1
98.             });
99.         }
100.    );
101. };
```