

RTCPlayer

以 es module为例 。xrtc为例

1. 导入: `import {RTCPlayer} from "...../2.0.0/rtcplayer/2.0.0/rtcplayer.esm.js"`
2. 创建实例 `const player = new RTCPlayer()`
3. 绑定player 按需事件

```
export enum PlayerEvents {  
  play = 'play',  
  waiting = 'waiting',  
  playing = 'playing',  
  stop = 'stop',  
  playNotAllowed = 'not-allowed',  
  error = 'error',  
}
```

Plain Text

```
1  player  
2      ?.on(PlayerEvents.play, () => {  
3          console.log('sdk event: player play')  
4      })  
5      .on(PlayerEvents.waiting, () => {  
6          console.log('sdk event: player waiting')  
7      })  
8      .on(PlayerEvents.playing, () => {  
9          console.log('sdk event: player playing')  
10     })  
11     .on(PlayerEvents.playNotAllowed, () => {  
12         console.log('sdk event: play not allowed, muted play')  
13     })
```

4. webapi 拿到流描述信息, `initResp`
5. 设置播放器模式 `player.playerType = 12` (xrtc 是 12)
6. 设置播放流信息
`player.stream = streamInfo` (由`initResp` 解析构造出来)

数据结构根据`pushMode` 不同 而不同 参考

```

export type IXRTCStreamInfo = {
  sid?: string
  server: string
  auth: string
  appid: string
  timeStr: string
  userId: string
  roomId: string
}
export type IWebRTCStreamInfo = { sid?: string; streamUrl: string }

```

7. 设置 webapi请求虚拟人流信息大小 player.videoSize = {width, height}
8. 设置播放器 父节点位置player.container = document.querySelector("#playerWrapper") 注意:
playerWrapper 要具有大小, player 会自动填充到该区域内。
9. player.play() 开始播放
10. player.stop() 停止播放 & 销毁dom
11. player.resume() //当由于浏览器限制自动播放导致 播放默认静音时 引导用户点击屏幕 并主动调用
该API (第三步 playNotAllowed 事件 在用户点击屏幕或交互页面时, 需要调用恢复播放)
12. player.muted = true / false 静音/解除
13. player.volume = 0 ~ 1 设置音量