模块事件

一、播放器信息修改事件

播放器模块会保存播放器的状态信息，通过该事件监听播放器状态改变。

Event 事件： `PlayerEvents.PlayerInfoChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`playerInfo`	播放器信息	`PlayerStoreInfo`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.PlayerInfoChange, (data) => {
  const playerInfo = data.playerInfo;
  console.log('播放状态', playerInfo.playStatus);
});
```

二、播放器暖场信息更新

暖场信息初始化有一定的延迟，开发者可通过该事件监听暖场信息初始化完成并获取暖场信息。

Event 事件： `PlayerEvents.PlayerWarmUpSettingChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`warmUpSetting`	暖场信息	`PlayerWarmUpSetting`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.PlayerWarmUpSettingChange, () => {
  const setting = watchCore.player.getPlayerWarmUpSetting();
  console.log('暖场类型', setting.warmUpType);
  console.log('暖场图片地址', setting.warmUpImg);
  console.log('暖场视频地址', setting.warmUpVideo);
});
```

三、播放器初始化完毕

调用播放器模块的 `setupPlayer` 方法创建播放器时，通过 `PlayerInited` 事件监听播放器初始化完成事件，触发该事件后即可调用播放器的 Api。

Event 事件： `PlayerEvents.PlayerInited`

示例：

```
// 创建播放器
watchCore.player.setupPlayer({
  container: 'NodeElement',
});

watchCore.player.eventEmitter.on(PlayerEvents.PlayerInited, () => {
  console.log('播放器初始化完成');
  const playerInfo = watchCore.player.getPlayerInfo();
  console.log(playerInfo.playerInited); // true
});
```

四、播放器播放事件

用户点击播放或通过 Api 播放后会回调播放事件，开发者也可通过 PlayerEvents.PlayerInfoChange 事件同步播放状态。

Event 事件： `PlayerEvents.PlayerPlaying`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.PlayerPlaying, () => {
  console.log('播放了');
});
```

五、播放器暂停事件

用户点击暂停或通过 Api 暂停后会回调暂停事件，开发者也可通过 PlayerEvents.PlayerInfoChange 事件同步播放状态。

Event 事件： `PlayerEvents.PlayerPause`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.PlayerPause, () => {
  console.log('暂停了');
});
```

六、音量改变

音量改变时触发该事件。

Event 事件： `PlayerEvents.VolumeChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`currentVolume`	当前音量	`number`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.VolumeChange, () => {
  const currentVolume = watchCore.player.getCurrentVolume();
  console.log('当前音量', currentVolume);
});
```

七、线路切换

线路切换时触发该事件。

Event 事件： `PlayerEvents.LineChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`currentLine`	当前线路	`number`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.LineChange, () => {
  const currentLine = watchCore.player.getCurrentLine();
  console.log('当前线路', currentLine);
});
```

八、清晰度级别切换

清晰度切换时触发该事件。

Event 事件： `PlayerEvents.QualityLevelChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`currentQualityLevel`	当前清晰度级别	`number`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.QualityLevelChange, () => {
  const currentLevel = watchCore.player.getCurrentQualityLevel();
  console.log('当前清晰度', currentLevel);
});
```

九、倍速修改

倍速切换时触发该事件。

Event 事件： `PlayerEvents.RateChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`currentRate`	当前播放倍速	`number`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.RateChange, () => {
  const currentRate = watchCore.player.getCurrentRate();
  console.log('当前倍速', currentRate);
});
```

十、播放器弹幕改变

播放器弹幕显示状态切换时触发该事件。

Event 事件： `PlayerEvents.BarrageStatusChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`barrageShow`	是否显示弹幕	`boolean`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.BarrageStatusChange, () => {
  const playerInfo = watchCore.player.getPlayerInfo();
  console.log('弹幕显示状态', playerInfo.barrageShow);
});
```

十一、调起弹幕设置弹窗

移动端下的弹幕设置需要由页面实现，该事件用于监听观众点击播放器的弹幕设置按钮，触发该事件后显示弹幕设置蒙层。

Event 事件： `PlayerEvents.BarrageSettingClick`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.BarrageSettingClick, () => {
  console.log('显示弹幕设置面板');
});
```

十二、播放时间改变

Event 事件： `PlayerEvents.TimeUpdate`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`currentTime`	当前播放时间，单位：秒	`number`
`durationTime`	播放总时长，单位：秒	`number`

十三、回放播放结束

Event 事件： `PlayerEvents.PlaybackOver`

十四、回放续播事件

Event 事件： `PlayerEvents.HistoryPlay`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`historyTime`	续播的时间，单位：秒	`number`

十五、无延迟播放异常

通过 `LowLatencyError` 事件监听无延迟播放异常，异常后切到正常延迟模式。

Event 事件： `PlayerEvents.LowLatencyError`

示例：

```
watchCore.player.eventEmitter.on(PlayerEvents.LowLatencyError, () => {
  toast.error('播放无延迟异常，已帮您切换到正常延迟模式');
  // TODO 重建播放器并使用正常延迟
});
```

十六、切换播放器全屏

Event 事件： `PlayerEvents.ToggleFullScreen`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`isPlayerFullscreen`	是否播放器全屏	`boolean`

十七、视频分辨率变化

Event 事件： `PlayerEvents.PlayerResolutionChange`

回调参数： `Object` 对象，详细类型说明如下

属性名	说明	类型
`width`	视频宽	`number`
`height`	视频高	`number`