观看条件模块

`观看条件模块(auth)` 主要负责观看条件授权，观众进入直播观看页前需要进行观看条件授权，开发者可通过 `watchCore.auth.isAuthorized` 方法获取观众的授权情况：

- 未授权：显示引导/授权页进行授权，授权后重新 `watchCore.setup` 观看页核心实例并调用 `watchCore.connect` 连接聊天室，连接成功后显示直播观看页。
- 已授权：直接调用 `watchCore.connect` 连接聊天室，连接成功后显示直播观看页。

示例如下：

```
// 观众未进行观看条件授权
if (!watchCore.auth.isAuthorized()) {
  // TODO: 显示引导/授权页

  // 观众点击观看页入口或执行观看条件授权，如：无条件授权
  const result = await watchCore.auth.verifyNoneAuth();
  if (!result.success) {
    console.error('授权失败了！！', result.failReason);
    return;
  }

  // 当观众执行观看条件授权成功，需要重新安装 watchCore
  await watchCore.setup();
}

// 当完成观看条件授权后，即可连接聊天室进入直播观看页
await watchCore.connect();

// TODO: 连接成功，显示直播观看页
```

> 关于各观看条件的说明请见本模块的开发文档。

一、观看条件类型

Enum 枚举： `AuthType`

常量	枚举成员	说明	设置信息
`'none'`	`AuthType.None`	无条件观看	AuthSettingItemNone

常量	枚举成员	说明	设置信息
`'pay'`	`AuthType.Pay`	付费观看	AuthSettingItemPay
`'phone'`	`AuthType.Phone`	白名单观看	AuthSettingItemPhone
`'info'`	`AuthType.Info`	登记观看	AuthSettingItemInfo
`'code'`	`AuthType.Code`	验证码观看	AuthSettingItemCode
`'custom'`	`AuthType.Custom`	自定义授权	AuthSettingItemCustom
`'external'`	`AuthType.External`	外部授权	AuthSettingItemExternal
`'direct'`	`AuthType.Direct`	独立授权	AuthSettingItemDirect

二、使用方式

2.1 获取观看条件设置列表

各观看条件的设置说明可见对应的文档。

- 当后台未开启观看限制，设置列表返回 [无条件授权]
- 当后台开启观看限制，设置列表返回 [主要观看条件, 次要观看条件]

Api 方法： `getAuthSettings(): AuthSettingItem[]`

返回值说明： 观看条件设置列表，`AuthSettingItem[]` 类型

示例：

```
const authSettings = watchCore.auth.getAuthSettings();
authSettings.forEach(settingItem => {
  console.log('设置类型信息', settingItem);
  console.log('设置类型', settingItem.authType);
});
```

2.2 判断当前用户是否已进行观看条件授权

Api 方法： `isAuthorized(): boolean`

返回值说明： 是否完成授权

示例：

```
const isAuthorized = watchCore.auth.isAuthorized();
if (isAuthorized) {
  console.log('观众已进行观看条件授权，进入观看页');
} else {
  console.log('观众未进行观看条件授权，进入引导页进行授权');
}
```

三、验证观看条件

观看页 SDK 提供每个观看条件的验证 API，开发者可根据相应的 API 进行观看条件的验证，当验证成功后即可显示直播观看页。

3.1 执行一次观看条件验证

所有观看条件的验证方法无论验证成功是否通过，Promise 均会返回验证结果 `result`，其类型为 `AuthVerifyResult`，当 `result.success` 为 true 时表示观看条件验证成功，false 时表示验证失败，具体失败可见 观看条件验证失败处理

代码示例如下：

```
/**
 * 验证白名单观看
 * @param phone 白名单
 */
async function verifyPhoneAuth(phone) {
  const result = await watchCore.auth.verifyPhoneAuth({
    phone,
  });

  if (result.success) {
    // 验证成功
    handleAuthVerifySuccess(result);
  } else {
    // 验证失败
    handleAuthVerifyFail(result);
  }
}

/**
 * 统一处理验证观看条件成功
 */
async function handleAuthVerifySuccess(successResult) {
  if (!successResult.success) {
    return;
  }

  // 重新安装观看页
  await watchCore.setup();
  console.log(watchCore.auth.isAuthorized()); // 此时返回 true
  console.log('验证成功，进入观看页');
}
```

`AuthVerifyResult` 类型

```
/** 观看条件验证结果 */
type AuthVerifyResult<T extends AuthType> = AuthVerifyResultSuccess<T> | AuthVerifyResultFail<T>;

/** 观看条件验证结果（成功） */
interface AuthVerifyResultSuccess<T extends AuthType = AuthType> {
  /** 观看条件类型 */
  authType: T;
  /** 成功状态 */
  success: true;
}

/** 观看条件验证结果（失败） */
interface AuthVerifyResultFail<T extends AuthType = AuthType> {
  /** 观看条件类型 */
  authType: T;
  /** 成功状态 */
  success: false;
  /** 失败原因 */
  failReason: AuthVerifyError;
  /** 失败信息 */
  failMessage?: string;
  /** 获取重定向跳转地址 */
  getRedirectUrl?: () => string;
}
```

3.2 验证失败处理

当观看条件验证失败时（即 `result.success` 为 false），可通过 `result.failReason` 获取验证失败原因，然后在页面进行错误类提示等其他处理，该字段类型为 `AuthVerifyError` 枚举，示例代码如下：

```
/**
 * 处理观看条件失败
 * @param failResult 失败结果
 */
function handleAuthVerifyFail(failResult) {
  switch (failResult.failReason) {
    // 未知原因
    case AuthVerifyError.Unknow:
      toast.error('验证失败：未知原因');
      break;

    // 白名单不存在
    case AuthVerifyError.PhoneNotExist:
      toast.error('验证失败：白名单不存在');
      break;

    // 需要重定向
    case AuthVerifyError.RedirectUrl:
      if (failResult.getRedirectUrl) {
        const redirectUrl = failResult.getRedirectUrl();
        toast.error('验证失败，需要重定向', redirectUrl);
      }
      break;
    // ...其他错误处理
  }
}
```

3.3 观看条件验证失败原因

Enum 枚举： `AuthVerifyError`

常量	枚举成员	说明	场景
`'Unknown'`	`AuthVerifyError.Unknown`	未知错误	通用
`'RedirectUrl'`	`AuthVerifyError.RedirectUrl`	需要重定向	通用
`'PhoneEmpty'`	`AuthVerifyError.PhoneEmpty`	白名单会员码为空	白名单观看
`'PhoneNotExist'`	`AuthVerifyError.PhoneNotExist`	白名单会员码不存在	白名单观看
`'SmsCodeError'`	`AuthVerifyError.SmsCodeError`	短信验证码错误	登记观看

常量	枚举成员	说明	场景
`'PhoneNotRegister'`	`AuthVerifyError.PhoneNotRegister`	手机号未登记	登记观看
`'CodeEmpty'`	`AuthVerifyError.CodeEmpty`	观看验证码为空	验证码观看
`'CodeError'`	`AuthVerifyError.CodeError`	观看验证码错误	验证码观看
`'CustomSignParamsMiss'`	`AuthVerifyError.CustomSignParamsMiss`	缺少自定义授权参数	自定义授权
`'ExternalError'`	`AuthVerifyError.ExternalError`	外部授权失败	外部授权

四、Api 方法概览

Api 方法	说明
isAuthorized	判断当前用户是否已进行观看条件授权
getAuthSettings	获取观看条件设置列表
verifyNoneAuth	验证无条件观看
verifyPhoneAuth	验证白名单观看
verifyCodeAuth	验证验证码观看
getAuthInfoFields	获取登记观看表单设置列表
verifyInfoAuth	验证登记观看
loginInfoAuth	登录登记观看