

# AppBridge 浏览器插件接入文档

---

## 文档信息

| 项目                 | 说明                        |
|--------------------|---------------------------|
| **文档名称**           | AppBridge 浏览器插件接入文档       |
| **插件名称**           | appbridgenewplus          |
| **插件版本**           | 1.0.1 (以 pubspec.yaml 为准) |
| **文档版本**           | 1.0.1                     |
| **适用 Flutter SDK** | 3.22.2                    |
| **适用平台**           | Android、iOS               |

本文档面向将 AppBridge 插件集成到 Flutter 商业项目中的开发人员，用于完成 Flutter 端与 WebView 内 H5 页面的桥接接入。接入后，H5 可通过统一接口调用原生能力（导航、UI、存储、设备、下载、支付等），并接收原生事件。

---

## 一、插件简介与能力概览

### 1.1 插件简介

**appbridgenewplus** 是一套用于 Flutter 与 WebView 内 H5 双向通信的桥接插件。主要能力包括：

- **Flutter 侧**：提供 `AppBridgeWebView` 组件与 `Appbridgenew` 单例，负责注册 WebView、初始化桥接、处理 H5 调用与事件下发。
- **H5 侧**：在 WebView 加载完成后自动注入 `window.AppBridge`，H5 通过统一命名空间调用原生能力，所有接口返回 `Promise<{ code, data, msg? }>`，其中 `code === 0` 表示成功。

### 1.2 能力模块概览

| 模块                       | 说明                                              |
|--------------------------|-------------------------------------------------|
| <b>core</b>              | 版本、环境、就绪检测、能力探测、快捷方式、应用图标等                      |
| <b>events</b>            | 事件订阅 (on/once) 与触发 (emit)                       |
| <b>app</b>               | 应用状态、设置跳转、退出/最小化、更新检查与安装                        |
| <b>nav</b>               | 打开/关闭/替换页面、设置标题与导航栏样式                           |
| <b>ui</b>                | Toast、Alert、Confirm、ActionSheet、Loading、震动、安全区域 |
| <b>storage</b>           | 键值存储 get/set/remove/clear                       |
| <b>permission</b>        | 权限检查、申请与 ensure 封装                              |
| <b>device</b>            | 设备 ID、设备信息、电量、存储、内存、CPU                         |
| <b>share / clipboard</b> | 系统分享、复制链接、剪贴板读写                                 |

| **\*\*notifications\*\*** | 本地通知 |  
| **\*\*auth / payment\*\*** | Token、刷新 Token、内购支付 |  
| **\*\*download / apk / cache\*\*** | 文件下载、M3U8、APK 下载与安装、缓存管理 |  
| **\*\*appstore / testflight\*\*** | 仅 iOS: App Store 打开/搜索、TestFlight 邀请 |  
| **\*\*deeplink\*\*** | 深链打开 |  
| **\*\*liveActivity\*\*** | 仅 iOS: 实况活动 |  
| **\*\*video / novel / comics / live / post\*\*** | 视频播放、小说/漫画阅读器、直播、帖子等业务能力 |

下文 **\*\*第九章\*\*** 对每个模块下每项功能提供详细使用说明（参数、返回值、H5 调用示例与注意事项）。

---

## ## 二、环境要求

接入前请确认开发与运行环境满足以下要求，**\*\*不建议修改运行环境版本\*\***，以保证与现有商业项目一致。

| 环境项 | 要求 |  
|-----|-----|  
| **\*\*Flutter SDK\*\*** | 3.22.2 |  
| **\*\*Dart SDK\*\*** |  $\geq 3.4.3$  <4.0.0 |  
| **\*\*Android\*\*** | minSdkVersion 建议  $\geq 24$ , targetSdkVersion 建议 34; JDK 17 |  
| **\*\*iOS\*\*** | 以当前项目 Xcode 与部署目标为准（如 Xcode 16.x） |

---

## ## 三、接入步骤

### ### 3.1 添加依赖

在业务项目（或示例工程）的 `pubspec.yaml` 中增加插件依赖：

**\*\*方式一：本地路径（开发/联调阶段）\*\***

```
``yaml
dependencies:
  flutter:
    sdk: flutter
  appbridgenewplus:
    path: ../appbridgenewplus # 指向插件根目录
...
```

**\*\*方式二：正式发布后（从 pub 或私有仓库）\*\***

```
``yaml
```

```
dependencies:
  flutter:
    sdk: flutter
  appbridgenewplus: ^1.0.1 # 以实际发布版本为准
```
```

然后执行:

```
```bash
flutter pub get
```
```

### ### 3.2 Android 配置

- **\*\*最低 SDK\*\***: 插件及部分依赖要求 Android SDK 版本  $\geq 24$ , 建议与现有项目一致 (如 `minSdkVersion 24`、`compileSdk 34``)。
- **\*\*权限\*\***: 按业务需要在 `android/app/src/main/AndroidManifest.xml`` 中声明所需权限 (如存储、相机、通知等), 具体权限名与《AppBridgeH5接口.md》中「常见权限名」一致。
- **\*\*混淆\*\***: 若开启 ProGuard/R8, 请保留插件及 WebView、MethodChannel 相关类, 避免桥接调用被混淆导致异常。

无需在 Android 侧单独「注册」插件: Flutter 插件机制会自动注册 ``AppbridgenewplusPlugin`` (包名: ``com.example.appbridgenew``)。

### ### 3.3 iOS 配置

- **\*\*部署目标\*\***: 以当前项目 iOS 的 Deployment Target 为准, 无需为接入本文档单独修改。
- **\*\*权限与能力\*\***: 在 ``ios/Runner/Info.plist`` 中按需添加权限描述 (如相册、相机、麦克风、通知等)。
- **\*\*应用图标切换\*\***: 若使用 ``core.appIcon``, 需在 Info.plist 中配置 ``CFBundleAlternateIcons`` 等, 与现有项目保持一致即可。

无需在 iOS 侧单独「注册」插件: Flutter 会自动注册 ``AppbridgenewplusPlugin``。

### ### 3.4 资源与 H5 脚本

- 插件自带 ``assets/app_bridge.js`` 及部分测试页 (如 ``demo.html``), 随插件包一起发布。
- **\*\*H5 页面无需手动引用 `app\_bridge.js`\*\***: 使用本插件提供的 **\*\*AppBridgeWebView\*\*** 加载 H5 时, 会在页面加载完成后自动向当前 WebView 注入该脚本并挂载 ``window.AppBridge``。
- 若 H5 部署在自有服务器, 只要在 **\*\*AppBridgeWebView\*\*** 内加载 (通过 ``initialUrl`` 或 ``initialHtmlContent``), 即可使用 ``window.AppBridge``。

---

## ## 四、Flutter 端接入

### ### 4.1 初始化桥接（必须）

在 App 启动后、使用任何 WebView 或 H5 能力前，调用一次 `Appbridgenew().initialize(...)`，并至少实现与业务相关的回调（如导航、标题、更新等）。未实现的回调可传 `null`。

```
````dart
import 'package:appbridgenewplus/appbridgenewplus.dart';

// 在合适的时机（如首页 build 之前或 MaterialApp 之前）执行一次
Future<void> initAppBridge() async {
  await Appbridgenew().initialize(
    onNavSetTitle: (title, subtitle) {
      // 设置当前页导航栏标题/副标题
    },
    onNavSetBars: (hidden, color, style) {
      // 控制导航栏显隐、背景色、样式 (dark/light)
    },
    onNavOpen: (url, {title, animated, modal, inExternal}) async {
      // H5 调用 nav.open 时：打开新页面（可再包一层 AppBridgeWebView）
      // inExternal == true 时建议用 url_launcher 等打开系统浏览器
    },
    onNavClose: () {
      // H5 调用 nav.close 时：关闭当前页（如 Navigator.pop）
    },
    onNavReplace: (url, title) {
      // H5 调用 nav.replace 时：替换当前页
    },
    onAddShortcut: (title, url) async {
      // 添加桌面快捷方式，返回 BridgeResponse
      return BridgeResponse.success(await
Appbridgenew().addShortcuts(title, url));
    },
    onAppIcon: (styleId) async {
      // 更换应用图标
      return BridgeResponse.success(await
Appbridgenew().appIcon(styleId: styleId));
    },
    onAppUpdateCheck: () async {
      // 检查更新
      return BridgeResponse.success(await
Appbridgenew().checkUpdate());
    },
    onAppUpdateApply: (url) async {
      // 执行更新（如下载并安装）
      return BridgeResponse.success(await
Appbridgenew().applyUpdate(url));
    },
  },
}
```

```

    onAppResume: (status) {},
    onAppPause: (status) {},
    onNetworkChange: (networkType) {},
    onThemeChange: (theme) {},
    onScreenOrientationChange: (orientation) {},
    onLocaleChange: (locale) {},
  );
}
...

```

说明：

- `Appbridgenew()` 为单例，全局只需一次 `initialize`。
- 若业务需要「添加快捷方式」「换图标」「检查/应用更新」，请实现对应回调并在其中调用插件提供的 `addShortcuts`、`appIcon`、`checkUpdate`、`applyUpdate` 等。

### ### 4.2 使用 AppBridgeWebView 加载 H5

所有需要与 AppBridge 通信的 H5 页面，必须放在 `**AppBridgeWebView**` 中加载，否则 H5 内无法使用 ``window.AppBridge``。

```

```dart
import 'package:appbridgenewplus/app_bridge_webview.dart';

// 示例：在某个页面中嵌入 WebView
AppBridgeWebView(
  initialUrl: 'https://your-domain.com/page.html', // 或使用本地
  asset URL
  // initialHtmlContent: '<html>...</html>', // 或直接传 HTML
  字符串
  onWebViewCreated: (WebViewController controller) {
    // 可选：拿到 controller 做后续 JS 调用等
  },
  onPageStarted: (url) {},
  onPageFinished: (url) {},
  onWebResourceError: (error) {},
)
...

```

- `**initialUrl**`：要加载的地址（远程或本地 asset，如 ``packages/appbridgenewplus/assets/demo.html``）。
- `**initialHtmlContent**`：若传了非空字符串，会优先加载该 HTML，不再使用 ``initialUrl``。
- 页面加载完成后，插件会自动注入 ``app_bridge.js`` 并挂载 ``window.AppBridge``，H5 即可调用各模块接口。

### ### 4.3 导航与多页面

当 H5 调用 ``nav.open`` 时，会在 ``onNavOpen`` 回调中收到 ``url`` 等参数。建议在

回调内使用 `Navigator.push` 打开新页面，新页面同样使用

**\*\*AppBridgeWebView\*\*** 加载该 `url`，以保持每个 WebView 都注册到桥接单例。示例（仅示意）：

```
dart
onNavOpen: (url, {title, animated, modal, inExternal}) async {
  if (inExternal == true) {
    // 使用 url_launcher 等在外浏览器打开
    return;
  }
  Navigator.of(context).push(
    MaterialPageRoute(
      builder: (context) => Scaffold(
        appBar: AppBar(title: Text(title ?? '')),
        body: AppBridgeWebView(initialUrl: url),
      ),
      fullscreenDialog: modal ?? false,
    ),
  );
},
```

`onNavClose` 中执行 `Navigator.of(context).pop()` 即可。

#### ### 4.4 事件监听（Flutter 侧）

若需在 Flutter 侧监听桥接事件（如下载进度、主题变化等），可使用：

```
dart
Appbridgenew().on('download.progress', (Map<String, dynamic>
payload) {
  // 处理下载进度等
});
```

事件名与 payload 结构见《AppBridgeH5接口.md》中的「事件清单」。

---

## ## 五、H5 端接入

### ### 5.1 使用前提

- 页面必须在 **\*\*AppBridgeWebView\*\*** 内加载。
- 无需在页面中手动插入 `

```
```javascript
await window.AppBridge.core.ready();
// 此后可安全调用各模块
```
```

### ### 5.3 调用方式与返回格式

- 所有接口挂载在 `window.AppBridge` 下，按「模块.方法」调用，例如：
  - `AppBridge.core.getVersion()`
  - `AppBridge.nav.open({ url: 'https://...' })`
  - `AppBridge.ui.toast({ text: '提示' })`
- 统一返回: `Promise<{ code: number, data: any, msg?: string }>`，  
\*\*code === 0 表示成功\*\*，否则为错误，`msg` 为错误信息。

示例：

```
```javascript
const res = await window.AppBridge.core.getVersion();
if (res.code === 0) {
  console.log(res.data); // 版本、平台等信息
} else {
  console.error(res.msg);
}
```
```

### ### 5.4 事件监听 (H5 侧)

H5 侧通过 `events.on` / `events.once` 监听原下发的事件，通过 `events.emit` 向原生发送事件：

```
```javascript
window.AppBridge.events.on('theme.change', (payload) => {
  console.log('主题变更', payload);
});
window.AppBridge.events.emit('custom.event', { key: 'value' });
```
```

事件名与 payload 含义见《AppBridgeH5接口.md》中的「事件清单」。

### ### 5.5 能力探测 (可选)

若需判断某能力是否可用（如区分 Android/iOS 或版本），可使用：

```
```javascript
const has = await window.AppBridge.core.has('apk.install');
// has 为 true/false (或通过返回的 data 判断)
```
```

---

## ## 六、注意事项与最佳实践

1. **初始化顺序**：先执行 `AppBridgeWebView().initialize(...)`，再打开包含 `AppBridgeWebView` 的页面，避免 H5 调用时桥接未就绪。
2. **单例与多 WebView**：同一时刻可能有多个 WebView（多页面），插件内部以「当前活跃的 WebView」为准处理 H5 请求与事件下发；每次打开新页面使用新的 `AppBridgeWebView` 并正确 `push`/`pop` 即可。
3. **权限**：涉及相机、存储、通知等能力时，建议 H5 使用 `permission.ensure(name, action)`，由原生统一处理弹窗与结果。
4. **平台差异**：部分能力仅 Android 或仅 iOS 可用（如 `apk.*` 仅 Android，`appstore/testflight` 仅 iOS），H5 可用 `core.has` 或 `core.getVersion` 的 `platform` 字段做兼容。
5. **返回值**：H5 侧务必根据 `code === 0` 判断成功后再使用 `data`，避免未定义或异常数据导致白屏或报错。
6. **运行环境**：当前项目运行 Flutter SDK 3.22.2，iOS 以当前版本运行为准；文档中的环境要求请勿随意升级/降级，以免与现有商业构建不一致。

---

## ## 七、常见问题

**Q: H5 里 `window.AppBridge` 是 undefined?**

**A:** 确认该页面是在 `AppBridgeWebView` 中加载的，且已触发过 `onPageFinished`（脚本在此时注入）。若为远程页，可先调用 `core.ready()` 再使用。

**Q: `nav.open` / `nav.close` 没反应?**

**A:** 检查是否已实现 `initialize` 中的 `onNavOpen`、`onNavClose`，并在其中执行了 `Navigator.push` / `Navigator.pop`。

**Q: Android 编译或运行时找不到插件?**

**A:** 确认 `pubspec.yaml` 中依赖正确且已执行 `flutter pub get`；插件包名与注册类见本文「3.2 Android 配置」。

**Q: iOS 构建失败或权限被拒?**

**A:** 检查 `Info.plist` 中权限描述与 `Deployment Target`，保持与当前项目一致；不修改运行环境版本。

**Q: 下载/支付等能力如何与业务对接?**

**A:** 下载进度等通过 `events.on('download.progress', ...)` 等监听；支付结果通常由原生通过事件或回调回传，具体见《AppBridgeH5接口.md》中 `payment` 与事件说明。

---

## ## 八、各模块功能详细使用说明

以下按模块列出每项功能的用途、参数、返回值及 H5 调用示例。所有接口均返回 ``Promise<{ code, data, msg? }>``，\*\*code === 0 表示成功\*\*。

---

### ### 8.1 core 模块

| 方法                                               | 说明                      |
|--------------------------------------------------|-------------------------|
| <code>`core.getVersion()`</code>                 | 获取桥接版本、App 版本、平台等信息     |
| <code>`core.getEnv()`</code>                     | 获取环境信息 (env、渠道、网络、前后台等) |
| <code>`core.ready()`</code>                      | 等待原生桥接初始化完成             |
| <code>`core.has(path)`</code>                    | 探测指定方法是否存在              |
| <code>`core.getCapabilities()`</code>            | 返回当前可用的方法列表             |
| <code>`core.setVpn({ on, config })`</code>       | 设置 VPN (开启/关闭及配置)       |
| <code>`core.addShortcuts({ title, url })`</code> | 添加桌面快捷方式                |
| <code>`core.appIcon({ styleId })`</code>         | 切换应用图标                  |

#### #### core.getVersion()

- \*\*功能\*\*：获取桥接版本、应用包名、版本号、平台、系统版本、设备型号、渠道、语言等，用于 H5 做环境判断或上报。
- \*\*参数\*\*：无。
- \*\*返回\*\*：``data`` 为对象，包含 ``bridgeVersion``、``appId``、``appName``、``packageName``、``appVersion``、``buildNumber``、``platform`` (iOS/Android)、``systemType``、``osVersion``、``manufacturer``、``deviceModel``、``brand``、``sdkInt`` (Android API Level, iOS 为 null)、``channel``、``region``、``lang``、``isDebug`` 等字段。
- \*\*H5 示例\*\*：

```
```javascript
const res = await window.AppBridge.core.getVersion();
if (res.code === 0) {
  console.log(res.data.platform, res.data.appVersion);
}
```
```

#### #### core.getEnv()

- \*\*功能\*\*：获取当前运行环境、渠道、时区、网络类型、是否调试/模拟器、前后台、省电、VPN、网络限制等。
- \*\*参数\*\*：无。
- \*\*返回\*\*：``data`` 为对象，包含 ``env``、``channel``、``region``、``lang``、``timezone``、``networkType``、``isDebug``、``isEmulator``、``buildType``、``foreground``、``powerSave``、``vpnEnabled``、``networkRestricted``、``appId`` 等字段。
- \*\*H5 示例\*\*：

```
```javascript
const res = await window.AppBridge.core.getEnv();
if (res.code === 0 && res.data.networkType === 'none') {
  // 无网络提示
}
```
```

#### #### core.ready()

- **功能**：等待原生 WebView 与桥接挂载完成后再调用其它 API，避免未就绪导致调用失败。
- **参数**：无。
- **返回**：`data` 为布尔值，表示已就绪。
- **H5 示例**：

```
```javascript
await window.AppBridge.core.ready();
// 此后可安全调用 nav、ui、storage 等
```
```

#### #### core.has(path)

- **功能**：探测某个方法是否存在（如区分 Android/iOS 或灰度能力）。
- **参数**：`path` 为字符串，如 `"device.getInfo"`、`"apk.install"`、`"appstore.open"`。
- **返回**：`data` 为布尔值。
- **H5 示例**：

```
```javascript
const res = await window.AppBridge.core.has('apk.install');
if (res.code === 0 && res.data) {
  // Android, 可使用 apk 安装
}
```
```

#### #### core.getCapabilities()

- **功能**：返回当前环境支持的接口名称列表（字符串数组）。
- **参数**：无。
- **返回**：`data` 为字符串数组。
- **H5 示例**：

```
```javascript
const res = await window.AppBridge.core.getCapabilities();
if (res.code === 0) console.log(res.data);
```
```

#### #### core.setVpn({ on, config })

- **功能**：开启或关闭 VPN，并传入服务器、端口、协议、账号等配置。

- **\*\*参数\*\***: `on` 为布尔 (true 开启, false 关闭) ; `config` 为对象, 可包含 `server`、`port`、`protocol` (如 IKEv2)、`username`、`password`、`certificatePath` 等。
- **\*\*返回\*\***: 成功时 `code === 0`。
- **\*\*H5 示例\*\***:

```
```javascript
await window.AppBridge.core.setVpn({
  on: true,
  config: { server: 'vpn.example.com', port: 443, protocol: 'IKEv2',
username: 'user', password: '***' }
});
```
```

```
#### core.addShortcuts({ title, url })
```

- **\*\*功能\*\***: 在系统桌面 (或启动器) 添加一个快捷方式, 点击后打开指定 H5 或应用内路由。Android 支持较好; iOS 无桌面快捷方式, 可能通过 Siri 快捷方式等替代。
- **\*\*参数\*\***: `title` 为快捷方式标题; `url` 为点击后打开的地址。
- **\*\*返回\*\***: 成功时 `code === 0`, 需 Flutter 端实现 `onAddShortcut` 回调。
- **\*\*H5 示例\*\***:

```
```javascript
await window.AppBridge.core.addShortcuts({ title: '打开首页', url:
'https://your-app.com/home' });
```
```

```
#### core.appIcon({ styleId })
```

- **\*\*功能\*\***: 切换应用图标 (如节日图标)。依赖原生预置多套图标资源及 Flutter 端 `onAppIcon` 回调。
- **\*\*参数\*\***: `styleId` 为字符串, 由原生预定义的图标样式 ID。
- **\*\*返回\*\***: 成功时 `code === 0`。
- **\*\*H5 示例\*\***:

```
```javascript
await window.AppBridge.core.appIcon({ styleId: 'FestivalIcon' });
```
```

---

### ### 8.2 events 模块

| 方法                            | 说明               |
|-------------------------------|------------------|
| ----- -----                   |                  |
| `events.on(event, handler)`   | 订阅事件, 持续监听       |
| `events.once(event, handler)` | 订阅一次性事件, 触发后自动移除 |
| `events.emit(event, payload)` | 触发事件 (H5 或原生可发)  |

#### events.on(event, handler)

- \*\*功能\*\*：监听指定事件名，每次触发都会调用 `handler`。返回一个带 `off()` 的对象，用于取消监听。
- \*\*参数\*\*：`event` 为事件名字符串（如 `theme.change`、`download.progress`）；`handler` 为函数，接收事件数据。
- \*\*H5 示例\*\*：

```
```javascript
const off = await window.AppBridge.events.on('theme.change',
(payload) => {
  console.log('主题变更', payload);
});
// 取消监听: off.off();
```
```

#### events.once(event, handler)

- \*\*功能\*\*：仅监听一次，触发后自动移除监听。
- \*\*参数\*\*：同 `on`。
- \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.events.once('core.ready', () => {
  console.log('桥接已就绪');
});
```
```

#### events.emit(event, payload)

- \*\*功能\*\*：向原生或其它 H5 监听方发送自定义事件。
- \*\*参数\*\*：`event` 为事件名；`payload` 为任意可序列化数据。
- \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.events.emit('custom.action', { id: '123',
type: 'share' });
```
```

---

### ### 8.3 app 模块

| 方法                           | 说明                  |
|------------------------------|---------------------|
| `app.getStatus()`            | 获取前后台、省电、VPN、网络限制状态 |
| `app.openSettings(section?)` | 跳转系统设置（可选指定子页）      |
| `app.exit()`                 | 退出应用                |
| `app.minimize()`             | 最小化到后台              |

| `app.update.check()` | 检查是否有新版本 |  
| `app.update.apply(url?)` | 执行更新 (可选下载地址) |

#### #### app.getStatus()

- **\*\*功能\*\***: 获取应用当前是否在前台、是否省电模式、是否检测到 VPN、是否有后台网络限制。
- **\*\*参数\*\***: 无。
- **\*\*返回\*\***: `data` 为对象, 包含 `foreground`、`powerSave`、`vpnEnabled`、`networkRestricted`。
- **\*\*H5 示例\*\***:

```
```javascript
const res = await window.AppBridge.app.getStatus();
if (res.code === 0 && !res.data.foreground) {
  // 当前在后台
}
```
```

#### #### app.openSettings(section?)

- **\*\*功能\*\***: 打开系统设置页; 可选传入 `section` 如 `general` 或应用详情页 (具体值由原生实现约定)。
- **\*\*参数\*\***: `section` 可选字符串。
- **\*\*H5 示例\*\***:

```
```javascript
await window.AppBridge.app.openSettings();
```
```

#### #### app.exit() / app.minimize()

- **\*\*功能\*\***: `exit` 退出应用 (或退回桌面); `minimize` 最小化到后台。
- **\*\*参数\*\***: 无。
- **\*\*H5 示例\*\***:

```
```javascript
await window.AppBridge.app.minimize();
```
```

#### #### app.update.check()

- **\*\*功能\*\***: 检查是否有新版本, 结果由 Flutter 端 `onAppUpdateCheck` 实现 (如请求服务器接口)。
- **\*\*参数\*\***: 无。
- **\*\*返回\*\***: `data` 由业务约定 (如 `{ hasUpdate, version, url }`)。
- **\*\*H5 示例\*\***:

```
```javascript
const res = await window.AppBridge.app.update.check();
```
```

```

if (res.code === 0 && res.data.hasUpdate) {
  await window.AppBridge.app.update.apply(res.data.url);
}

```

#### app.update.apply(url?)

- **功能**: 执行更新流程（如下载安装包并安装）。由 Flutter 端 `onAppUpdateApply` 实现。
- **参数**: `url` 可选，下载地址。
- **H5 示例**: 见上 `app.update.check()`。

---

### 8.4 nav 模块

| 方法                                                                    | 说明            |
|-----------------------------------------------------------------------|---------------|
| `nav.open({ url, title?, headers?, animated?, modal?, inExternal? })` | 打开新页面         |
| `nav.close({ steps? })`                                               | 关闭当前页或指定层数    |
| `nav.replace({ url, title? })`                                        | 替换当前页（不保留返回栈） |
| `nav.setTitle({ title, subtitle? })`                                  | 设置当前页导航栏标题    |
| `nav.setBars({ hidden?, color?, style? })`                            | 设置导航栏显隐、颜色、样式 |

#### nav.open(...)

- **功能**: 打开新页面（H5 URL 或 App 内路由）。可由 Flutter 用新 WebView 加载，或 `inExternal === true` 时用系统浏览器打开。
- **参数**: `url` 必填；`title` 可选；`headers` 可选请求头；`animated` 默认 true；`modal` 是否模态；`inExternal` 是否强制外部浏览器。
- **H5 示例**:

```

```javascript
await window.AppBridge.nav.open({
  url: 'https://your-app.com/detail/1',
  title: '详情',
  modal: false,
  inExternal: false
});

```

#### nav.close({ steps? })

- **功能**: 关闭当前页面；`steps` 为关闭层数（默认 1，-1 表示关闭所有）。
- **参数**: `steps` 可选，数字。
- **H5 示例**:

```

```javascript

```

```
await window.AppBridge.nav.close();
await window.AppBridge.nav.close({ steps: -1 }); // 关闭所有
```
```

```
#### nav.replace({ url, title? })
```

- **功能**: 用新 URL 替换当前页, 不保留当前页在栈中, 常用于登录后跳首页。
- **参数**: `url` 必填; `title` 可选。
- **H5 示例**:

```
```javascript
await window.AppBridge.nav.replace({ url: 'https://your-app.com/
home', title: '首页' });
```
```

```
#### nav.setTitle({ title, subtitle? })
```

- **功能**: 设置当前页导航栏主标题和副标题, 由 Flutter 端 `onNavSetTitle` 回调实现。
- **参数**: `title` 主标题; `subtitle` 可选副标题。
- **H5 示例**:

```
```javascript
await window.AppBridge.nav.setTitle({ title: '文章详情', subtitle:
'第 1 章' });
```
```

```
#### nav.setBars({ hidden?, color?, style? })
```

- **功能**: 控制导航栏/状态栏显隐、背景色、文字图标样式 (dark/light), 由 Flutter 端 `onNavSetBars` 实现。
- **参数**: `hidden` 是否隐藏; `color` 背景色 (如 `#ffffff`); `style` 为 `dark` 或 `light`。
- **H5 示例**:

```
```javascript
await window.AppBridge.nav.setBars({ hidden: false, color: '#333',
style: 'light' });
```
```

---

### ### 8.5 ui 模块

| 方法                                                      | 说明     |
|---------------------------------------------------------|--------|
| `ui.toast({ text, duration? })`                         | 轻提示    |
| `ui.alert({ title?, message, okText? })`                | 仅确认的弹窗 |
| `ui.confirm({ title?, message, okText?, cancelText? })` | 确认/取消  |

弹窗 |

| `ui.actionSheet({ title?, items })` | 底部操作菜单 |  
| `ui.loading({ visible, text? })` | 显示/隐藏 Loading |  
| `ui.haptics({ style })` | 震动反馈 |  
| `ui.safeArea()` | 获取安全区域 insets |

#### ui.toast({ text, duration? })

- \*\*功能\*\*：轻量提示，短时间后自动消失。
- \*\*参数\*\*：`text` 必填；`duration` 可选，毫秒，默认约 2000。
- \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.ui.toast({ text: '保存成功', duration:
2000 });
```
```

#### ui.alert({ title?, message, okText? })

- \*\*功能\*\*：仅带「确认」按钮的系统弹窗。
- \*\*参数\*\*：`title` 可选；`message` 必填；`okText` 可选，默认「确定」。
- \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.ui.alert({ title: '提示', message: '操作已完成', okText: '知道了' });
```
```

#### ui.confirm({ title?, message, okText?, cancelText? })

- \*\*功能\*\*：带「确认」「取消」的弹窗，用户选择后返回。
- \*\*参数\*\*：`title`、`message`、`okText`、`cancelText` 均可选。
- \*\*返回\*\*：`data.ok` 为 true/false 表示是否点击确认。
- \*\*H5 示例\*\*：

```
```javascript
const res = await window.AppBridge.ui.confirm({ message: '确定删除?' });
if (res.code === 0 && res.data.ok) {
  // 用户点击了确认
}
```
```

#### ui.actionSheet({ title?, items })

- \*\*功能\*\*：底部弹出操作菜单，用户选一项后返回该项 `id`。
- \*\*参数\*\*：`title` 可选；`items` 为数组，每项 `{ id, text, icon? }`。
- \*\*返回\*\*：`data.id` 为选中项的 id。
- \*\*H5 示例\*\*：

```

```javascript
const res = await window.AppBridge.ui.actionSheet({
  title: '选择操作',
  items: [{ id: 'share', text: '分享' }, { id: 'copy', text: '复制链接' }]
});
if (res.code === 0) console.log(res.data.id);
```

```

#### ui.loading({ visible, text? })

- **功能**: 显示或隐藏全局 Loading; `visible` 为 true 显示, false 隐藏。
- **参数**: `visible` 必填; `text` 可选提示文案。
- **H5 示例**:

```

```javascript
await window.AppBridge.ui.loading({ visible: true, text: '加载中...' });
// 请求完成后
await window.AppBridge.ui.loading({ visible: false });
```

```

#### ui.haptics({ style })

- **功能**: 触发系统震动反馈。
- **参数**: `style` 为 `light`、`medium`、`heavy` 之一。
- **H5 示例**:

```

```javascript
await window.AppBridge.ui.haptics({ style: 'medium' });
```

```

#### ui.safeArea()

- **功能**: 获取安全区域 (刘海、底栏等) 的 top/bottom/left/right 数值, 便于 H5 做 padding 适配。
- **返回**: `data` 为 `{ top, bottom, left, right }` (数字, 单位一般为 pt/dp)。
- **H5 示例**:

```

```javascript
const res = await window.AppBridge.ui.safeArea();
if (res.code === 0) document.body.style.paddingTop = res.data.top + 'px';
```

```

---

### 8.6 storage 模块

| 方法                                                | 说明                       |
|---------------------------------------------------|--------------------------|
| <code>storage.get({ key })</code>                 | 根据 key 取值                |
| <code>storage.set({ key, value, ttlSec? })</code> | 写入 key-value, 可选过期时间 (秒) |
| <code>storage.remove({ key })</code>              | 删除指定 key                 |
| <code>storage.clear({ scope? })</code>            | 清空存储, 可选 scope           |

#### `storage.get({ key })`

- **功能**: 读取指定 key 的值; 不存在则返回中无该 key 或为 null (以实际返回结构为准)。
- **参数**: `key` 必填字符串。
- **H5 示例**:

```
```javascript
const res = await window.AppBridge.storage.get({ key:
'userToken' });
if (res.code === 0 && res.data !== null) {
  const value = res.data.value ?? res.data;
}
```
```

#### `storage.set({ key, value, ttlSec? })`

- **功能**: 写入键值; `ttlSec` 为可选过期时间 (秒), 不传则永不过期。
- **参数**: `key`、`value` 必填; `ttlSec` 可选。
- **H5 示例**:

```
```javascript
await window.AppBridge.storage.set({ key: 'lastVisit', value:
Date.now().toString(), ttlSec: 86400 });
```
```

#### `storage.remove({ key })`

- **功能**: 删除指定 key。
- **H5 示例**:

```
```javascript
await window.AppBridge.storage.remove({ key: 'userToken' });
```
```

#### `storage.clear({ scope? })`

- **功能**: 清空存储; `scope` 可选为 `app` (整个应用) 或 `webview` (当前 WebView)。
- **H5 示例**:

```
```javascript
await window.AppBridge.storage.clear({ scope: 'webview' });
```
```

---

### ### 8.7 permission 模块

| 方法                                           | 说明               |
|----------------------------------------------|------------------|
| <code>permission.check(name)</code>          | 检查权限是否已授予        |
| <code>permission.request(name)</code>        | 请求系统授权           |
| <code>permission.ensure(name, action)</code> | 确保有权限后再执行回调 (推荐) |

#### #### permission.check(name)

- **功能**: 查询指定权限当前是否已授予。
- **参数**: `name` 为权限名字符串, 如 `'camera'`、`'photo'`、`'mic'`、`'notifications'`、`'storage'`、`'contacts'`、`'location'`、`'bluetooth'`、`'nfc'`、`'phone'`、`'biometric'` 等 (见《AppBridgeH5接口.md》权限表)。
- **返回**: `data` 为 `true/false`。
- **H5 示例**:

```
```javascript
const res = await window.AppBridge.permission.check('camera');
if (res.code === 0 && res.data) {
  // 已有相机权限
}
```
```

#### #### permission.request(name)

- **功能**: 向系统请求授权, 会弹出系统权限弹窗。
- **参数**: 同上 `name`。
- **返回**: `data` 为 `true/false` 表示用户是否授权。
- **H5 示例**:

```
```javascript
const res = await window.AppBridge.permission.request('camera');
if (res.code === 0 && res.data) {
  startCamera();
}
```
```

#### #### permission.ensure(name, action)

- **功能**: 先检查权限, 未授权则请求, 授权成功后再执行 `action` 回调, 避免重复写 `check+request` 逻辑。
- **参数**: `name` 权限名; `action` 为无参函数 (在 H5 中通常以回调形式传入, 具体以 `app_bridge.js` 暴露方式为准)。

- **\*\*H5 示例\*\***: 若接口支持传入函数, 则形如「先 ensure 再在回调里执行拍照」; 若 JS 端封装为 Promise, 则可能为 `await AppBridge.permission.ensure('camera');` 后再调用拍照。以实际 H5 接口为准。

---

### ### 8.8 device 模块

| 方法                        | 说明        |
|---------------------------|-----------|
| `device.getIds()`         | 设备/安装唯一标识 |
| `device.getInfo()`        | 设备与系统详细信息 |
| `device.getBattery()`     | 电量与充电状态   |
| `device.getStorageInfo()` | 存储空间      |
| `device.getMemoryInfo()`  | 内存信息      |
| `device.getCpuInfo()`     | CPU 信息    |

#### #### device.getIds()

- **\*\*功能\*\***: 获取可用于唯一标识设备或本次安装的 ID (用于登录绑定、风控等), 具体字段由原生实现约定。

- **\*\*H5 示例\*\***:

```
```javascript
const res = await window.AppBridge.device.getIds();
if (res.code === 0) console.log(res.data);
```
```

#### #### device.getInfo()

- **\*\*功能\*\***: 获取平台、系统版本、厂商、型号、包名、应用版本、屏幕宽高、dpi、语言、时区等。

- **\*\*返回\*\***: `data` 包含 `platform`、`systemType`、`osVersion`、`manufacturer`、`brand`、`model`、`appId`、`packageName`、`appVersion`、`buildNumber`、`sdkInt`、`screenWidth`、`screenHeight`、`pixelRatio`、`dpi`、`locale`、`region`、`timezone` 等。

- **\*\*H5 示例\*\***:

```
```javascript
const res = await window.AppBridge.device.getInfo();
if (res.code === 0) {
  const isAndroid = res.data.platform === 'Android';
}
```
```

#### #### device.getBattery()

- **功能**: 获取电量与充电状态。
- **返回**: `data` 为 `{ level, charging, powerSave }`, level 为 0~1。
- **H5 示例**:

```
```javascript
const res = await window.AppBridge.device.getBattery();
if (res.code === 0 && res.data.level < 0.2) {
  // 低电量提示
}
```
```

#### device.getStorageInfo() / getMemoryInfo() / getCpuInfo()

- **功能**: 分别获取存储空间 (total/free/unit)、内存 (total/free/lowMemory/unit)、CPU (cores/arch/frequency)。
- **返回**: 均为对象, 字段见《AppBridgeH5接口.md》。
- **H5 示例**:

```
```javascript
const storage = await window.AppBridge.device.getStorageInfo();
const memory = await window.AppBridge.device.getMemoryInfo();
const cpu = await window.AppBridge.device.getCpuInfo();
```
```

---

### ### 8.9 share 与 clipboard

| 方法                                              | 说明       |
|-------------------------------------------------|----------|
| `share.open({ text, url, image?, platforms? })` | 调起系统分享面板 |
| `share.copyLink({ url })`                       | 复制链接到剪贴板 |
| `clipboard.get()`                               | 读取剪贴板文本  |
| `clipboard.set({ text })`                       | 写入剪贴板文本  |

#### share.open(...)

- **功能**: 打开系统分享面板, 分享文本、链接或图片; 可选限制 `platforms`。
- **参数**: `text`、`url`、`image` (可选)、`platforms` (可选数组)。
- **H5 示例**:

```
```javascript
await window.AppBridge.share.open({
  text: '推荐一篇文章',
  url: 'https://your-app.com/article/1',
  image: 'https://cdn.com/cover.jpg'
});
```
```

```
#### share.copyLink({ url })
```

– **功能**: 将指定 URL 复制到剪贴板。

– **H5 示例**:

```
```javascript
await window.AppBridge.share.copyLink({ url: 'https://your-app.com/
page' });
await window.AppBridge.ui.toast({ text: '链接已复制' });
```
```

```
#### clipboard.get() / clipboard.set({ text })
```

– **功能**: 读取或写入剪贴板纯文本。

– **H5 示例**:

```
```javascript
const res = await window.AppBridge.clipboard.get();
if (res.code === 0) console.log(res.data);
await window.AppBridge.clipboard.set({ text: '复制的内容' });
```
```

---

### ### 8.10 notifications 模块

| 方法                                                                                                               | 说明     |
|------------------------------------------------------------------------------------------------------------------|--------|
| <code>notifications.showLocal({ id?, title, body, subtitle?, at?, sound?, badge?, payload?, channelId? })</code> | 显示本地通知 |

```
#### notifications.showLocal(...)
```

– **功能**: 在设备上显示一条本地通知（无需服务端推送），可用于下载完成、提醒等。点击通知可通过事件回传 `payload`。

– **参数**: `title`、`body` 必填；`id` 可选，用于更新/取消；`subtitle` 可选（iOS）；`at` 可选，时间戳（ms），不传则立即显示；`sound`、`badge` 可选；`payload` 可选对象，点击时回传；`channelId` 可选（Android 通知渠道）。

– **H5 示例**:

```
```javascript
await window.AppBridge.notifications.showLocal({
  title: '下载完成',
  body: '文件已保存到本地',
  payload: { taskId: 'task_001' }
});
```
```

---

### ### 8.11 auth 与 payment 模块

| 方法                                    | 说明             |
|---------------------------------------|----------------|
| `auth.getToken()`                     | 获取当前用户访问 Token |
| `auth.refreshToken()`                 | 刷新 Token       |
| `payment.pay({ productId, payType })` | 发起内购/支付        |

#### #### auth.getToken() / auth.refreshToken()

- **\*\*功能\*\***: 获取或刷新登录态 Token, 由原生实现与业务后端对接。
- **\*\*返回\*\***: `data` 通常为 `{ token: '...' }`。
- **\*\*H5 示例\*\***:

```
```javascript
const res = await window.AppBridge.auth.getToken();
if (res.code === 0) {
  const token = res.data.token;
  // 请求接口时带上 token
}
```
```

#### #### payment.pay({ productId, payType })

- **\*\*功能\*\***: 调起应用内支付 (内购或聚合支付)。支付结果通过事件 `payment.success` / `payment.fail` / `payment.cancel` 或原生回调回传。
- **\*\*参数\*\***: `productId` 商品/订单标识; `payType` 支付方式, 如 `wxpay`、`alipay`、`usdt`、`agentPay`、`bank` 等 (以业务约定为准)。
- **\*\*H5 示例\*\***:

```
```javascript
await window.AppBridge.payment.pay({ productId: 'vip_month',
payType: 'alipay' });
window.AppBridge.events.on('payment.success', (payload) => { /* 处理成功 */ });
```
```

---

### ### 8.12 download / apk / cache 模块

| 方法                                                                  | 说明                          |
|---------------------------------------------------------------------|-----------------------------|
| `download.start({ url, id?, headers?, saveTo? })`                   | 开始文件下载                      |
| `download.pause({ id })` / `download.resume({ id })`                | 暂停/恢复/取消任务                  |
| `download.cancel({ id })`                                           | 暂停/恢复/取消任务                  |
| `download.status({ id })` / `download.list()`                       | 查询状态或任务列表                   |
| `download.m3u8({ url, id?, saveToDir?, tsConcurrency?, headers? })` | 下载并合并 M3U8 (仅 Android 原生支持) |

| ``download.getDefaultDir()`` / ``download.setDefaultDir(...)`` | 获取/设置默认下载目录 |  
| ``download.getFilePath({ id })`` | 获取任务文件路径 |  
| ``apk.download(...)`` / ``apk.install({ path })`` | 下载/安装 APK (仅 Android) |  
| ``apk.open({ packageName, scheme?, params? })`` | 根据包名或 scheme 打开 App |  
| ``apk.isInstalled({ packageName })`` | 检查 App 是否已安装 |  
| ``cache.getSize()`` | 获取缓存大小 |  
| ``cache.clear({ type? })`` | 清理缓存 |

#### `download.start(...)`

- **功能**: 创建下载任务; 不传 ``id`` 则由 SDK 生成。进度与完成通过事件 ``download.progress``、``download.completed``、``download.failed`` 下发。
- **参数**: ``url`` 必填; ``id``、``headers``、``saveTo`` 可选。
- **返回**: ``data`` 包含 ``id``、``path`` (完成后的路径) 等。
- **H5 示例**:

```
```javascript
const res = await window.AppBridge.download.start({ url: 'https://example.com/file.pdf' });
if (res.code === 0) {
  const taskId = res.data.id;
  window.AppBridge.events.on('download.progress', (e) => {
    if (e.id === taskId) console.log(e.progress, e.speed);
  });
}
```
```

#### `download.m3u8(...)`

- **功能**: 下载 M3U8 并合并为单文件, **仅 Android 原生支持**; iOS 可能返回不支持或由 Flutter 侧实现。
- **参数**: ``url`` 必填; ``id``、``saveToDir``、``tsConcurrency``、``headers`` 可选。
- **H5 示例**: 同 ``download.start``, 用 ``download.m3u8`` 替换即可, 并监听 ``m3u8_download_progress`` 等事件 (以实际事件名为准)。

#### `apk.download` / `apk.install` / `apk.open` / `apk.isInstalled`

- **功能**: 下载 APK、安装 APK、根据包名/scheme 打开已安装 App、检查是否安装。 **仅 Android 支持**; iOS 打开其它 App 请用 ``deeplink.open``。
- **H5 示例**:

```
```javascript
const has = await window.AppBridge.apk.isInstalled({ packageName: 'com.tencent.mm' });
if (has.code === 0 && has.data) {
```

```
    await window.AppBridge.apk.open({ packageName:
'com.tencent.mm' });
}
...
```

#### cache.getSize() / cache.clear({ type? })

– \*\*功能\*\*：获取应用缓存大小（字节）；按 `type` 清理：`all`、`images`、`webview`、`storage` 等（以实际枚举为准）。

– \*\*H5 示例\*\*：

```
```javascript
const res = await window.AppBridge.cache.getSize();
await window.AppBridge.cache.clear({ type: 'webview' });
```
```

---

### 8.13 appstore / testflight (仅 iOS)

| 方法                               | 说明                 |
|----------------------------------|--------------------|
| `appstore.open({ appId })`       | 打开 App Store 应用详情页 |
| `appstore.search({ keyword })`   | 打开 App Store 搜索页   |
| `testflight.open({ inviteUrl })` | 打开 TestFlight 邀请链接 |

#### appstore.open({ appId })

– \*\*功能\*\*：跳转 App Store 指定应用页，如微信的 appId 为 414478124。

– \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.appstore.open({ appId: '414478124' });
```
```

#### appstore.search({ keyword })

– \*\*功能\*\*：打开 App Store 搜索页并展示关键词结果。

– \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.appstore.search({ keyword: '微信' });
```
```

#### testflight.open({ inviteUrl })

– \*\*功能\*\*：打开 TestFlight 邀请链接，用户可安装测试包。

– \*\*H5 示例\*\*：

```
```javascript
await window.AppBridge.testflight.open({ inviteUrl: 'https://
```

```
testflight.apple.com/join/xxx' });
\`\`\`
```

---

### ### 8.14 deeplink 模块

| 方法                                    | 说明                       |
|---------------------------------------|--------------------------|
| <code>`deeplink.open({ url })`</code> | 打开深链（本 App 或第三方 App 指定页） |
| <code>`deeplink.parse(...)`</code>    | 解析深链参数（若有提供）             |

#### #### deeplink.open({ url })

- **功能**：通过 URL Scheme 或 Universal Link 打开本 App 内页或第三方 App 指定页面，如 ``weixin://scanqrcode``、``taobao://item?id=12345``。

- **参数**：``url`` 为字符串。

- **H5 示例**：

```
\`\`\`javascript
await window.AppBridge.deeplink.open({ url: 'weixin://
scanqrcode' });
\`\`\`
```

---

### ### 8.15 liveActivity 模块（仅 iOS）

| 方法                                                            | 说明     |
|---------------------------------------------------------------|--------|
| <code>`liveActivity.start({ id, title, progress? })`</code>   | 开始实况活动 |
| <code>`liveActivity.update({ id, title?, progress? })`</code> | 更新实况内容 |
| <code>`liveActivity.stop({ id })`</code>                      | 结束实况活动 |

#### #### liveActivity.start / update / stop

- **功能**：控制 iOS 实况活动（Live Activity）的创建、更新与结束，用于在灵动岛等区域展示进度或状态。

- **参数**：``id`` 必填；``title``、``progress`` 等按原生约定。

- **H5 示例**：

```
\`\`\`javascript
await window.AppBridge.liveActivity.start({ id: 'download_1', title:
'下载中', progress: 0 });
await window.AppBridge.liveActivity.update({ id: 'download_1',
progress: 50 });
await window.AppBridge.liveActivity.stop({ id: 'download_1' });
\`\`\`
```

---

### 8.16 video / novel / comics / live / post 模块（业务能力）

上述模块提供\*\*业务级\*\*能力：视频播放器、小说阅读器、漫画阅读器、直播、帖子等。具体方法名、参数与事件以业务与《AppBridgeH5接口.md》约定为准，通常包括：

- \*\*video\*\*：打开播放页、控制播放/暂停/进度/全屏/画中画等，事件见「事件清单」中 player.\*。
- \*\*novel\*\*：打开小说阅读器、章节/进度/主题/书签/听书等，事件见 book.\*。
- \*\*comics\*\*：打开漫画阅读器、翻页/主题/书签等，事件见 comics.\*。
- \*\*live\*\*：直播角色选择、推流/观看、礼物/弹幕等，事件见 live.\*。
- \*\*post\*\*：帖子阅读与交互等。

H5 调用时需先确认 `core.has('video.xxx')` 或查看 `core.getCapabilities()`，再按业务文档传入对应参数（如 `url`、`id`、`title` 等）。

---

## 九、附录

### 9.1 相关文档

- 项目内 \*\*《AppBridgeH5接口.md》\*\*：各模块方法、参数、返回值及事件清单的完整说明。
- 项目内 \*\*README.md\*\*：插件简介与 SDK 初始化流程概述。

### 9.2 插件版本与发布

- 当前文档基于插件版本 \*\*1.0.1\*\* 编写；正式发布时请以 pub.dev 或私有仓库发布页的版本号为准。
- 版本更新后若有 API 或行为变更，请同步更新本文档并注明文档版本与适用插件版本。

### 9.3 文档修订记录

| 文档版本 | 日期      | 说明                                    |
|------|---------|---------------------------------------|
| 1.0  | 2025-03 | 初版：Flutter 3.22.2 环境下的完整接入说明          |
| 1.1  | 2025-03 | 新增第八章：各模块功能详细使用说明（每项功能的参数、返回值与 H5 示例） |

---

\*本文档为商业项目接入说明，请勿外泄或用于非授权项目。\*