Ant Technology

HTML5 Container and Offline Package User Guide

Document Version: 20231226

蚂蚁集团
ANT GROUP

Legal disclaimer

Trademark statement

蚂蚁集团 ANT GROUP and other trademarks related to Ant Group are owned by Ant Group. The third-party registered trademarks involved in this document are owned by the right holder according to law.

Disclaimer

The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Ant Group reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through channels authorized by Ant Group. You must pay attention to the version changes of this document as they occur and download and obtain the latest version of this document from Ant Group's authorized channels. Ant Group does not assume any responsibility for direct or indirect losses caused by improper use of documents.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	**Danger:** Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	**Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.HTML5 Container and Offline Package

1.1. HTML5 Container overview

HTML5 Container is a mobile hybrid solution SDK (Nebula SDK) that provides good external extension capability and has the capabilities of enabling pluggable functions, event mechanism, JSAPI customization, and HTML5 App push update and management.

Functions

HTML5 Container provides the following functions:

- Loading HTML5 pages, and managing the HTML5 pages by Session.
- Providing rich built-in JSAPI to implement such functions as page push, pop, and title setting.
- Supporting you to customize JSAPI and plugins so as to extend business requirements.
- Connecting Mobile Delivery Service platform to facilitate management of offline packages.
- Android client uses UCWebView and has the capability of solving system-level WebView crash, featuring more reasonable memory management, higher network loading speed, and better compatibility. This has completely overcome the problem that HTML5 Container with Android operating system cannot be compatible with different WebView.

Features

HTML5 Container has outstanding and powerful features in the following aspects.

Outstanding stability

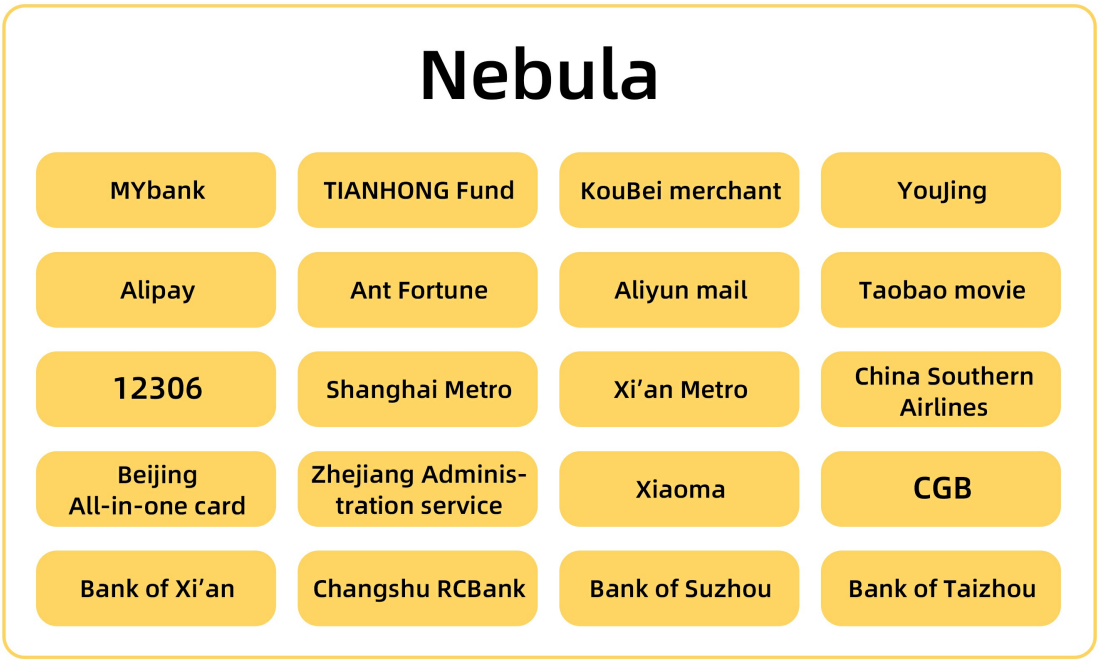
- HTML5 Container has withstood the test of hundreds of millions of users, with guaranteed crash rate, ANR (Application Not Responding) rate and other stability indicators.
- Android client is highly customized based on UCWebView, features lower crash rate and ANR rate than the system WebView, and has the capability of solving system-level WebView problems.

Powerful offline package capability

- **Strong offline package pushing platform**: With the MDS package pushing platform, the offline package can be quickly pushed to the client to ensure that the client data can be synchronized in a short time.
- **Preload offline package**: To meet the demands of special application scenarios, HTML5 container supports preloading offline package on the client, thus improving App opening speed.

Extensive ecosystem

HTML5 Container has accessed all Ant Financial Apps, providing a stable ecosystem.



1.2. Offline Package overview

Traditional HTML5 technology is subject to the network environment, thus degrading the performance of HTML5 pages. By using offline packages, you can solve such problem while preserving the benefits of HTML5.

Offline package refers to the package that includes HTML, JavaScript, CSS, and other in-page static resources. You can download the offline packages, open them through client, and then load them locally so as to get rid of the influence of network environment on HTML5 pages.

The advantages of using HTML5 offline packages are as follows:

- **Improve user experience**: Embedding the in-page static resources into application through offline package and release the application. When you start the application for the first time, you can download the resources without dependence on the network environment, and use the application immediately.
- **Realize dynamic update**: When you push a new version or make urgent release, you can put the modified resources into the offline package, and update configurations to enable the application automatically download updates. Therefore, you can make users receive updates as soon as possible and don't have to get your application approved by App Store.

How offline packages work

This section introduces how an offline package works from the following points:

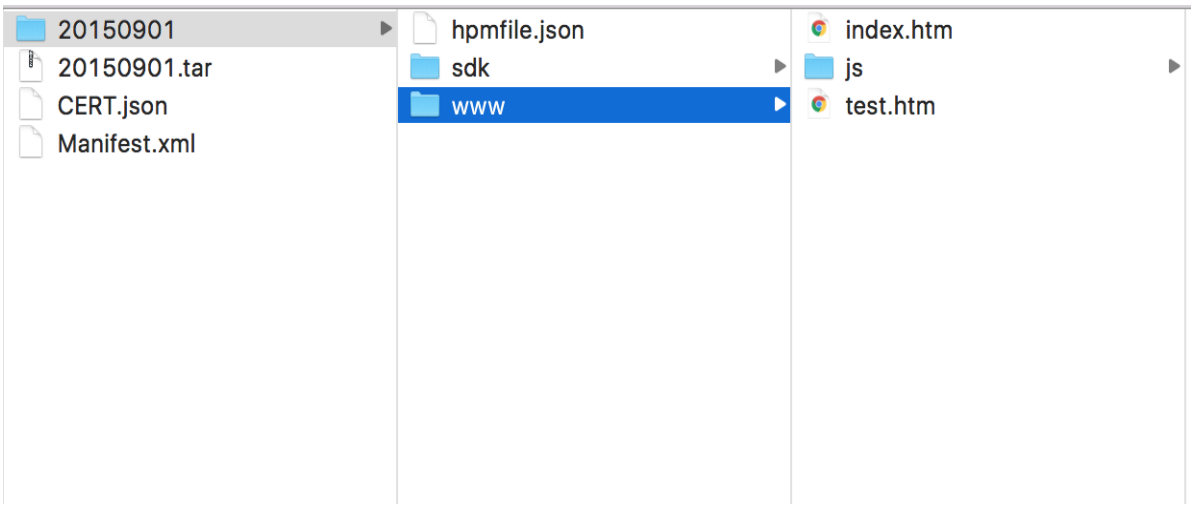
- Offline package structure
- Offline package type
- Rendering process

Offline package structure

The offline package is an `.amr` compressed file. You can see the included HTML resource and JavaScript codes by changing the suffix `amr` to `zip`. When the HTML5 container restarts, all the resource and codes can be rendered in WebView.

The following screenshots show the structure of general resource package:

- Level-1 directory: The ID of the general resource package, such as `20150901`.
- Level-2 and its sub-directories contain the custom resource files. It is best to save all frontend files in the same directory, such as `/www`, and set a main entry file for the current offline package, for example: `/www/index.html`.



Offline package type

Generally, some basic common libraries are used during HTML5 development, such as zepto and fastclick. As for the WebView in app, cache is sometimes unreliable. In some cases, the cache becomes invalid when the users exit from some device models.

To further promote the performance of HTML5 page, you can use global offline package to pack a series of common resources to a special App package, and then deliver the package to the client.

The offline package falls into the following types:

- **Global offline package**: Global offline package includes public resources which can be shared by multiple applications.
- **Private offline package**: Private offline package can only be exclusively used by one application.

After you use the global offline package, the system will attempt to read the package every time you visit the HTML5 pages. If the offline package has the corresponding resource, the system directly fetches the resource from the package, not relying on the network. However, the mechanism of the global offline package is designed to solve the use of common libraries.

To ensure the client coverage and universality, the offline package is generally updated at least every 1 month, and the package size is strictly restricted.

Rendering process

When the application initiates a resource request, the URL used to access local resource or online resource is the same.

The HTML5 Container intercepts the request first, and then:

- If there is local resource available to meet the request, HTML5 Container uses the local resource.
- If there is no proper local resource, HTML5 Container uses the online resource.

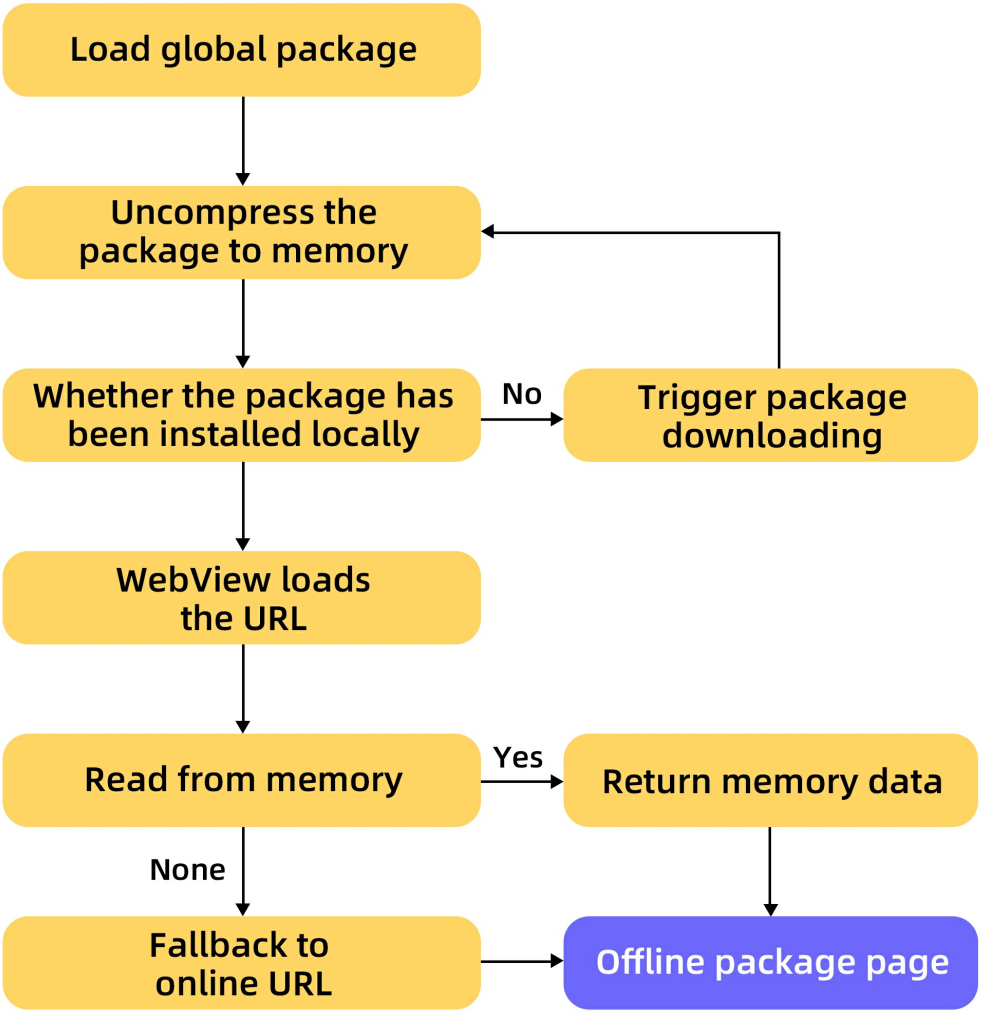
Therefore, no matter local or online resource, the WebView is insensitive.

Downloading the offline package depends on your settings when creating the offline package:

- If the **Download time** is set to **Download over WiFi only**, the offline package is automatically downloaded in the background only in WiFi network.

- If the **Download time** is set to **Download over all networks**, in non-WiFi network, the offline package is still automatically downloaded consuming user's mobile data. Thus, set it with caution.
- If the offline package is not completely downloaded when users use the App, it goes to the fallback address instead to display an online page.

  Fallback technology emerges when the offline package is not completely downloaded. The system synchronously releases a corresponding online version on CDN every time it releases an offline package. The online package has the same directory structure as the offline package. Fallback address will be delivered locally together with the offline package. When the offline package is not completely downloaded, the client will intercept the page request and redirect to the corresponding CDN address to realize the random switch between online page and offline page.



Offline package running mode

To open an offline package, complete the following steps:

1. Request package information: It involves the process from requesting offline package information from the server to saving the information to local database. The offline package information includes download address, offline package version number, etc.
2. Download offline package: Download the offline package from server to mobile phone.

3. Install offline package: Copy the offline package from the download directory to the mobile installation directory.

Virtual domain

Virtual domain is a unique mechanism of container, which only works on offline application. After the page is saved in the client, WebView loads and accesses the page through file Schema. In this way, the users can see the file path in the address bar. This leads the following problems:

- User experience: Users might feel insecure and uncomfortable about the exposed file address when they see the file address.
- Security: The file protocol contains local path, and every user can see the file path, so there is potential safety risk.

In view of above concerns, the virtual domain name mechanism rather than using pure file path is recommended. Virtual domain name is an HTTP address that meets URL schema specifications, for example, `https://xxxxxxx.h5app.example.com`. The parent domain name `example.com` must be your own registered domain.

> **Note**
>
> The domain can be the one that have been registered on Internet. Generally, it is not suggested to configure the virtual domain as the one consistent with the Internet domain, because it increases difficulty upon problem decision, tends to produce errors and is inconvenient to manage. You only have to ensure that the parent domain name `example.com` is your own registered domain. Currently, the standard virtual domain format is as follows:

```
https://xxxxxxx.h5app.example.com
```

Related links

- Generate offline packages

1.3. Integrate Android SDK

1.3.1. Upgrade instructions

Changes in 10.1.60

MPNebula

- The following API has been modified. Other than calling this API, you still need to enable the signature verification, see HTML5 container configuration for more information.

```
public static void enableAppVerification(final String publicKey)
```

- The following API has been deprecated, you cannot disable signature verification with this method any more. If you need to disable the signature verification, see HTML5 container configuration for more information.

```
public static void disableAppVerification();
```

HTML5ExtConfigProvider

Deprecated. If you need to configure HTML5 container switches, see HTML5 container configuration.

1.3.2. Quick Start

The HTML5 container and offline packages are supported in the **native AAR mode** and the **component mode**. The HTML5 container allows you to open an online web page in an app, call a native API operation on the frontend, call a custom JSAPI on the frontend, customize the title bar of an HTML5 page, and use the UC kernel. The HTML5 offline package service allows you to publish, preset, start, and update offline packages.

Before you begin

- If you connect to mPaaS based on native AAR, perform the prerequisite steps and subsequent steps. For more information, see Add mPaaS to your project.
- If you connect to mPaaS based on components, complete the access procedure first. For more information, see Access procedure under Connect to mPaaS mPaaS based on components.

Add the UC SDK

Native AAR mode

In your project, install the **HTML5 container** component on the **Component Management (AAR)** page. For more information, see Manage component dependencies in the native AAR mode.

Component mode

In your Portal and Bundle projects, install the **HTML5 container** component on the **Component Management** page.

For more information, see Manage component dependencies in Access procedure.

Initialize the mPaaS

In the **native AAR** mode, you must initialize the mPaaS.

Add the following code to the Application node:

```
public class MyApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();
        // mPaaS initialization
        MP.init(this);
    }
}
```

For more details, see Initialize mPaaS.

Configure the interval between requests for a Mini Program package

The mPaaS allows you to globally or individually configure the interval between requests for a Mini Program package.

- **Global configuration**: In the Android project, create a file named `custom_config.json` in `assets/config` and enter the following content in the file:

```
{
"value": "{\"config\":{\"al\":\"3\",\"pr\":
{\"4\":\"86400\",\"common\":\"864000\"},\"ur\":\"1800\",\"fpr\":
{\"common\":\"3888000\"}},\"switch\":\"yes\"}",
"key": "h5_nbmngconfig"\
}
```

`\"ur\":\"1800\"` specifies the interval of global updates. The value `1800` is the default value and is measured in seconds. You can modify the value to set the interval between requests for a global offline package. The value ranges from 0 to 86400 seconds, that is, 0 to 24 hours. The value 0 indicates that no intervals are specified.

Important: Do not modify other parameters unless required.

- **Individual configuration**: The configuration takes effect only for the current Mini Program package. In the console, click **Add** to add an offline package. In the **Extension Information** field, enter `{"asyncReqRate":"1800"}` to set the request interval. For more information, see the description of **Extension Information** in Create an HTML5 offline package.

Check whether the configured request interval takes effect. You can open a project connected to the HTML5 offline package service. Open the log in Logcat, and select the keyword `H5BaseAppProvider`. If the log contains the following information, the configuration has taken effect.

```
lastUpdateTime: xxx updateRate: xxx
```

Use the SDK

The mPaaS Nebula HTML5 container provides a unified API class named `MPNebula` to implement operations of the HTML5 container and offline packages.

The call process is as follows:

1. Start an HTML5 offline package.
   - Start an offline package:

     ```
     /**
     * Start an offline package.
     *
     * @param appId: offline package ID
     */
     public static void startApp(String appId);
     ```

   - Start an offline package with startup parameters:

     ```
     /**
     * Start an offline package.
     *
     * @param appId: offline package ID
     * @param params: startup parameters
     */
     public static void startApp(String appId, Bundle params);
     ```

2. Start an online page.
   - Start an online page:

```
/**
* Start an online URL.
*
* @param url: online URL
*/
public static void startUrl(String url)
```

- Start an online page with startup parameters:

```
/**
* Start an online URL.
*
* @param url: online URL
* @param param: sartup parameters
*/
public static void startUrl(String url, Bundle param);
```

3. Set a custom `UserAgent`.
   i. Implement a UA setter.

```
public class H5UaProviderImpl implements H5UaProvider {
@Override
public String getUa(String defaultUaStr) {
  // Do not modify the defaultUaStr parameter or return a result that does not contain
the defaultUaStr parameter.
  return defaultUaStr + " AlipayClient/mPaaS";
}
}
```

   ii. Call the UA setup API:

```
/**
*Set UA.
*
* @param uaProvider: UA setter. The getUa method needs to be implemented.
*/
public static void setUa(H5UaProvider uaProvider)
```

   iii. Perform setting as follows:

```
MPNebula.setUa(new H5UaProviderImpl());
```

4. Set a custom container view.

   The container provides methods for setting a custom title bar, navigation menu, the header of pull-to-refresh, and the WebView layout. For more information, see Sample code.

i. Implement a view setter.

```
public class H5ViewProviderImpl implements H5ViewProvider {
@Override
public H5WebContentView createWebContentView(Context context) {
  // The custom layout of WebView is returned here. If null is returned, the default v
iew takes effect.
  return null;
}

@Override
public H5TitleView createTitleView(Context context) {
  // The custom title bar is returned here. If null is returned, the default view take
s effect.
  return null;
}

@Override
public H5PullHeaderView createPullHeaderView(Context context, ViewGroup viewGroup) {
  // The pull-to-refresh head is returned here. If null is returned, the default view
takes effect.
  return null;
}

@Override
public H5NavMenuView createNavMenu() {
  // The custom navigation menu is returned here. If null is returned, the default vie
w takes effect.
  return null;
}
}
```

ii. Call the view setup API:

```
/**
*Set container related custom views, such as the title bar, menu bar, web layout, and
slide-the pull-down refresh head view.
* @param viewProvider: custom view provider
*/
public static void setCustomViewProvider(H5ViewProvider viewProvider);
```

iii. Set the custom views.

```
MPNebula.setCustomViewProvider(new H5ViewProviderImpl());
```

Note: To set a custom title bar, set the bundle name first. Otherwise, the resource cannot be found.

```
// Set the name of the bundle where the title bar resource resides. If it is not set,
the resource cannot be loaded and the title bar cannot take effect.
H5Utils.setProvider(H5ReplaceResourceProvider.class.getName(), new
H5ReplaceResourceProvider() {
@Override
public String getReplaceResourcesBundleName() {
  return BuildConfig.BUNDLE_NAME;
}
});
MPNebula.setCustomViewProvider(new H5ViewProviderImpl());
```

5. Embed the view of a single container into the page.

   Select one or more of the following methods as required to embed the HTML5 page into the view of a single container. The API provides synchronous and asynchronous methods.

   - **Synchronous method**

```
/**
*Obtain the view of the HTML5 container.
*
* @param activity: page context
* @param param: startup parameters, where the app ID or URL may be included
* @return: view of the HTML5 container
*/
public static View getH5View(Activity activity, Bundle param);
```

   - **Asynchronous method**

```
/**
*Obtain the view of the HTML5 container asynchronously.
*
* @param activity: page context
* @param param: startup parameters, where the app ID or URL may be included
* @param h5PageReadyListener: asynchronous callback
*/
public static void getH5ViewAsync(Activity activity, Bundle param, H5PageReadyListener
h5PageReadyListener);
```

   Note:

1.3.3. Manage offline packages

Offline package management operations include: Preload an HTML5 app, Use the global resource package, Update an HTML5 app, Download an HTML5 app, Install an HTML5 app, Obtain app information, Verify the security signature, and Delete local app information.

Prerequisite

- You have completed access configuration. For details, see Add the SDK.
- You have generated an offline package. For details, see Generate an offline package.

Preload an HTML5 app

Generally, the offline package may have not been downloaded when an HTML5 app is opened for the first time. In this case, you need to open the app by using the fallback URL.

Preloading an HTML5 app involves preinstalling an available HTML5 app in the installation package released by the client. When opening a preinstalled app for the first time, the user can directly use the offline package resources, which improves user experience.

> **Note**
>
> We recommend that you preinstall only core HTML5 apps and do not preinstall apps with a low usage rate.

To preload an HTML5 App, you need to complete the following steps:

1. Download the HTML5 app configuration file `.json` and required offline packages from the HTML5 app release backend.
2. Add the `.json` file and offline packages to the `asset` directory of the project.
3. When the app is started, call the preload code to install the app. The code sample is as follows:

```
MPNebula.loadOfflineNebula("h5_json.json", new
MPNebulaOfflineInfo("90000000_1.0.0.6.amr", "90000000", "1.0.0.6"));
```

> **Note**
>
> - This method is a block-type call method. Do not call the offline package method embedded on the main thread.
> - This method can be called only once. If it is called multiple times, only the first call is valid. Therefore, you need to input the information about all required offline packages at a time.
> - If multiple amr packages are embedded, ensure that the file already exists. If it does not exist, other embedded offline packages will fail.

Use the global resource package

The Nebula global resource package addresses redundancy caused by multiple HTML5 apps using the same resource. For example, the React app uses the ReactJS framework code. You can include public resources into the global resource package to downsize the HTML5 app.

Generally, a global resource package needs to be preset for the project, and subsequent updates can still be delivered through the HTML5 app backend.

The following code sample specifies the resource package with the app ID (`appId`) **66666692** as the global resource package and presets the `assets/nebulaPreset/66666692` offline package.

- `getCommonResourceAppList` : notifies the HTML5 container that an offline package with the specified ID will be used as the global resource package. If this ID is not configured, this offline package does not take effect even if it is embedded.
- `getH5PresetPkg` : specifies the path and version of the embedded global resource package. The HTML5 container loads the resource package from the specified asset resource directory. However, if a later version is detected on the server, this embedded package of an earlier version will not be loaded. In addition, you can use the aforementioned `loadOffLineNebula` method to preset a global resource package. In this case, you do not need to specify the path and version of the embedded offline package in this method.

> **Note**
>
> This method applies only to an HTML5 global resource package.

- `getTinyCommonApp`: returns the framework resource package ID of a mini app. If your global resource package is used only by the HTML5 container, do not write the ID of this public resource package in this method.

Reference code sample:

```
public class H5AppCenterPresetProviderImpl implements H5AppCenterPresetProvider {
    private static final String TAG = "H5AppCenterPresetProviderImpl";

    // The public resource package of the business, try to avoid the beginning of 666666
    private static final String COMMON_BIZ_APP = "xxxxxxxx";

    // Special resource package for mini programs, do not move for business
    private static final String TINY_COMMON_APP = "66666692";

    // The asset directory of the preset package
    private final static String NEBULA_APPS_PRE_INSTALL = "nebulaPreset" + File.separator;

    // Preset package collection
    private static final Map<String, H5PresetInfo> NEBULA_LOCAL_PACKAGE_APP_IDS = new Hash
Map();

    static {

        H5PresetInfo h5PresetInfo2 = new H5PresetInfo();
        // File name of the built-in directory
        h5PresetInfo2.appId = TINY_COMMON_APP;
        h5PresetInfo2.version = "1.0.0.0";
        h5PresetInfo2.downloadUrl = "";

        NEBULA_LOCAL_PACKAGE_APP_IDS.put(TINY_COMMON_APP, h5PresetInfo2);

    }

    @Override
    public Set<String> getCommonResourceAppList() {
        Set<String> appIdList = new HashSet<String>();
        appIdList.add(getTinyCommonApp());
        appIdList.add(COMMON_BIZ_APP);
        return appIdList;
    }

    @Override
    public H5PresetPkg getH5PresetPkg() {
        H5PresetPkg h5PresetPkg = new H5PresetPkg();
        h5PresetPkg.setPreSetInfo(NEBULA_LOCAL_PACKAGE_APP_IDS);
        h5PresetPkg.setPresetPath(NEBULA_APPS_PRE_INSTALL);
        return h5PresetPkg;
    }

    @Override
    public Set<String> getEnableDegradeApp() {
```

```
        return null;
    }

    @Override
    public String getTinyCommonApp() {
        return TINY_COMMON_APP;
    }

    @Override
    public InputStream getPresetAppInfo() {
        return null;
    }

    @Override
    public InputStream getPresetAppInfoObject() {
        return null;
    }
}
```

Call the following code when the App starts:

```
H5Utils.getH5ProviderManager().setProvider(H5AppCenterPresetProvider.class.getName(), new
H5AppCenterPresetProviderImpl());
```

Update an HTML5 app

By default, Nebula checks for a later version each time an HTML5 app is opened. Considering the pressure on the server, the check duration is restricted and is 30 minutes by default. To immediately check for the latest available version, call the following code to request for an update. Generally, the code can be called after the app starts or after the user logs on.

```
MPNebula.updateAllApp(new MpaasNebulaUpdateCallback(){
    @Override
    public void onResult(final boolean success, final boolean isLimit) {
        super.onResult(success, isLimit);
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                AUToast.makeToast(NebulaAppActivity.this,
                        success ? R.string.update_success : R.string.update_failure, 2000).
show();
            }
        });
    }
});
```

Download an HTML5 app

The MPNebula API allows you to manually download an HTML5 app.

```
/**
     *Download an offline package.
     *
     * @param appId: offline package ID
     * @param mpaasNebulaDownloadCallback: download callback
     */
    public static void downloadApp(final String appId, final MpaasNebulaDownloadCallback mpaasNebulaDownloadCallback)
```

Install an HTML5 app

The MPNebula API allows you to manually install an HTML5 app.

```
/**
     *Install an offline package.
     *
     * @param appId: offline package ID
     * @param mpaasNebulaInstallCallback: installation callback
     */
    public static void installApp(final String appId, final MpaasNebulaInstallCallback mpaasNebulaInstallCallback)
```

Obtain app information

Call the H5AppProvider method to obtain related information about HTML5 apps.

```
H5AppProvider provider = H5Utils.getProvider(H5AppProvider.class.getName());
AppInfo appInfo = provider.getAppInfo("1000000"); // Obtain app configuration.
boolean isInstalled = provider.isInstalled("100000", "1.0.0.0"); // Whether an app version is installed.
boolean isAvailable = provider.isAvailable("100000", "1.0.0.0");// Whether the offline package of an app version is downloaded successfully.
```

Verify the security signature

Nebula has an offline package signature verification mechanism to prevent malicious programs from tampering with offline packages downloaded to the device. Call MPNebula interface to use signature verification. In 10.1.60 and above, additional container configuration is required. See HTML5 container configuration for details.

> **Note**
>
> - Call this API before opening the offline package for the first time. Otherwise, public key initialization fails. About public and private keys, see Configure offline packages.
> - Signature verification is forcibly performed on a mobile phone judged as root, regardless of whether signature verification is enabled on the client.

```java
/**
     * @param publicKey: public key for signature verification
     */
    public static void enableAppVerification(final String publicKey)
```

Delete local app information

Nebula provides the interface for deleting local app information. After local app information is deleted, the client side will send request to the service side to download and update local app when the app is opened again.

```
public class MPNebula {
    // appId is the ID of the offline package or the mini program
    public static boolean deleteAppInfo(String appId);
}
```

> **Note**
>
> The lowest baseline versions supported by this API are 10.1.68.8 and 10.1.60.14 respectively.

1.3.4. Advanced Guide

1.3.4.1. Add UC SDK

Connecting the UC SDK to Android apps helps resolve the compatibility issue between browsers provided by different vendors. This reduces the crash rate and improves performance of the browsers as compared with the system browser. The UC SDK provides security support to mitigate security threats.

Before you begin

- If you connect to mPaaS based on native AAR, perform the prerequisite steps and subsequent steps. For more information, see Add mPaaS to your project.
- If you connect to mPaaS based on mPaaS Inside, complete the access procedure first. For more information, see Access procedure under Connect to mPaaS mPaaS based on mPaaS Inside.
- If you connect to mPaaS based on components, complete the access procedure first. For more information, see Access procedure under Connect to mPaaS mPaaS based on components.

Add the UC SDK

Native AAR mode

In your project, install the **UC kernel** component on the **Component Management (AAR)** page. For more information, see Manage component dependencies in the native AAR mode.

mPaaS Inside mode

In your project, install the **UC kernel** component on the **Component Management** page. For more information, see Manage component dependencies in the mPaaS Inside mode.

Component mode

In your Portal and Bundle projects, install the **UC kernel** component on the **Component Management** page. For more information, see Manage component dependencies in Access procedure.

Apply for a UC kernel

Use the tool for generating UC key signatures

Android Studio mPaaS plug-in V2.20062211 and later provide the tool for **generating UC key signatures**. This tool helps you apply for a UC SDK key. If the version of your mPaaS plug-in is V2.20062211 or later, you can use this tool to apply for a UC kernel. For more information, see Use the mPaaS plug-in.

Use the CLI

Apply for a key of the UC SDK as follows:

> **Note**
> - Make sure that the dependencies of the UC kernel are added to the project.
> - Make sure that the application ID of the app is provided.

Procedure

1. Obtain SHA1.
   i. Use the tool for generating UC key signatures

      Android Studio mPaaS plug-in V2.20062211 and later provide the tool for **generating UC key signatures**. This tool helps you apply for a UC SDK key. If the version of your mPaaS plug-in is V2.20062211 or later, you can use this tool to apply for a UC kernel. For more information, see Use the mPaaS plug-in.

   ii. Use the CLI

      Based on the development environment, execute the command to obtain the SHA1 value of the fingerprint on the application's signature certificate.

      > **Note**
      > - Make sure that the dependencies of the UC kernel are added to the project.
      > - Make sure that the application ID of the app is provided.

      - Windows

        ```
        Absolute path of keytool -v -list -keystore keystore
        ```

      - macOS

        ```
        Absolute path of keytool -list -v -keystore keystore
        ```

2. Fill and submit UC_key application form.
3. Enter the obtained key in the `AndroidManifest.xml` file in the project:

   ```
   <meta-data android:name="UCSDKAppKey" android:value="Obtained key"/>
   ```

   > **Note**
   > The authorization information of the UC SDK is bound to the **package name** and **signature** of the APK. Therefore, if UCWebView does not take effect, check whether the signature and package name are consistent with those you enter to apply for a key.

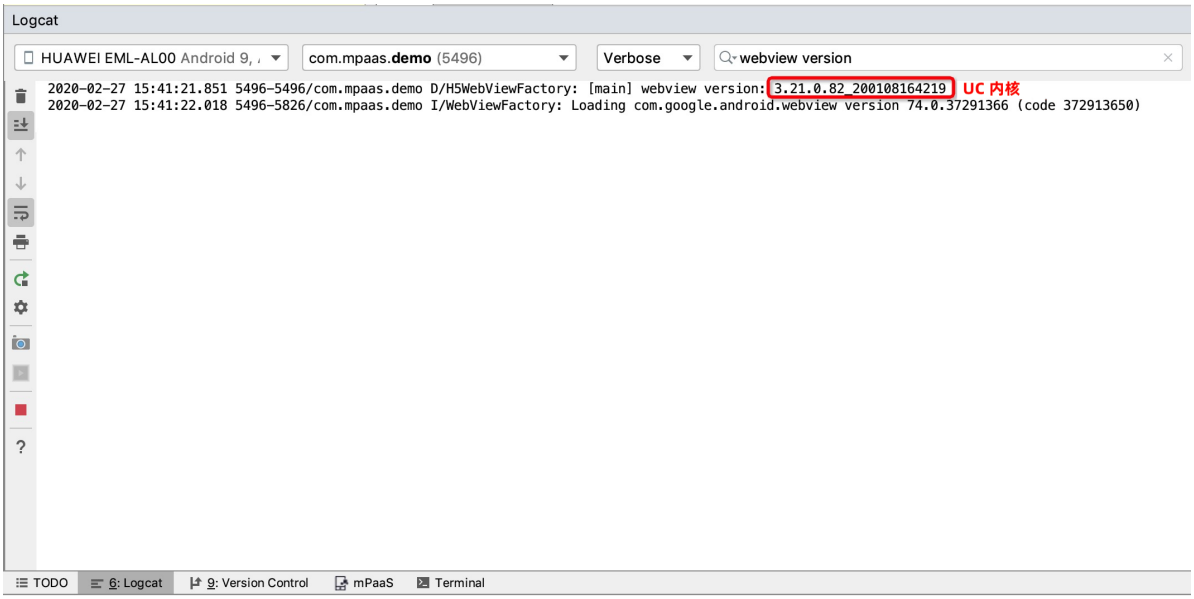
Check whether the UC core takes effect

After you generate a debugging package and install the debugging package, check whether the UC kernel takes effect.

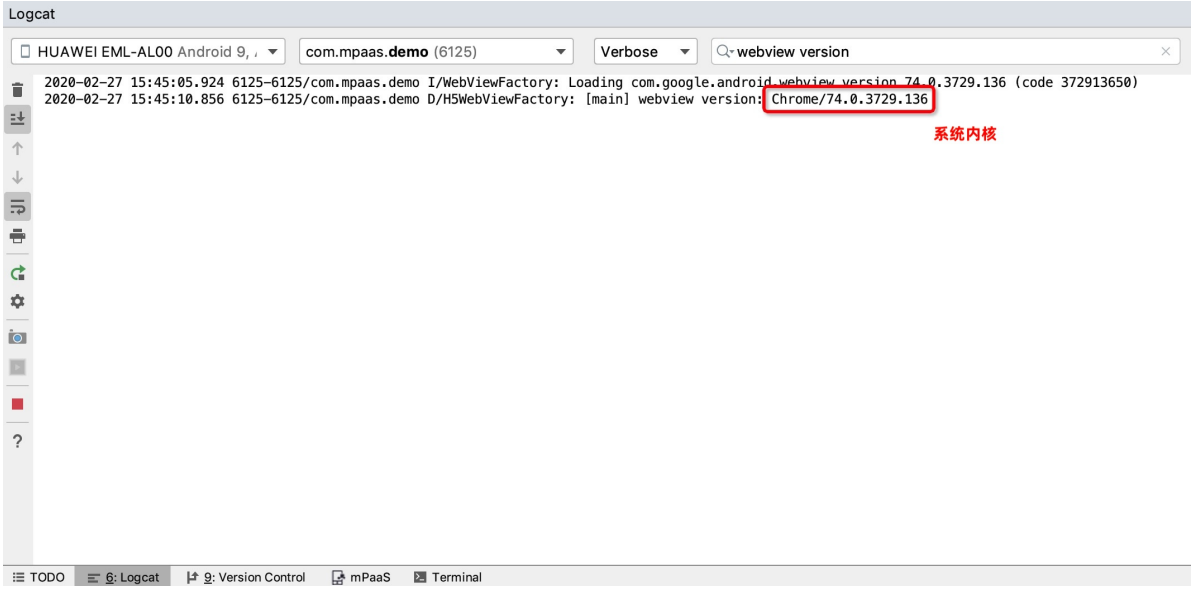
Open an HTML5 page. In the log in Logcat, select `webview version`.

The following figures show the sample logs.

- If the version starts with 2 or 3, the UC kernel takes effect.



- If the version starts with Chrome, the system kernel takes effect.



1.3.4.2. Extend UC SDK

> **Important**
>
> This document is only used for debugging and support for existing UC SDK users. Due to product policy changes, UC SDK is no longer open for applications.

If the UC kernel is used, you can select the system keypad(recommended) or the UC keypad.

Select an input keypad

If the UC kernel is used, you can select the system keypad(recommended) or the UC keypad.

> **Note**
>
> UC keypad does not support foldable phones, foldable phones can only use the system keypad.

Configure the `mp_h5_uc_number_input_use_system` switch for the HTML5 container to control whether to use the system keypad in the UC kernel.The value YES indicates that the system keypad is used, and NO indicates that the UC keypad is used. The default value is NO, which indicates that the UC keypad is used by default when the UC kernel is used. For more information about how to configure the HTML5 container, see Configure the HTML5 container.

1.3.4.3. Turn on inspect mode of UC SDK

The UC kernel in the mPaaS baseline 10.1.68.14 and later version no longer supports the inspect mode. If you need to debug in the inspect mode during development, you must add the debugging package for UC kernels.

Turn on the inspect mode

1. Download the corresponding .so file of UC kernel debugging package.
2. Copy the corresponding .so file of the UC kernel debugging package to `/sdcard/slm` directory, and the full path of the file is `/sdcard/slm/libWebViewCore_ri_7z_uc.so`.

> **Important**
>
> - The path and the name of the file must be consistent.
> - Make sure to copy the .so file of the corresponding CPU architecture.
> - The read permission of the SD card must be enabled.
> - This mode is only available for debugging packages.
> - After inspect mode is successfully turned on, the content in `/sdcard/slm` will be cut into the internal storage of the app, so when you wipe data and cache of the app or uninstall the app, you need to copy the .so file of the UC kernel debugging package to `/sdcard/slm` again.

UC kernel debugging package version

UC kernel debugging package version: 3.21.0.174.200825145737.

1.3.4.4. Get UC kernel crash log

> **Important**
>
> This document is only used for debugging and support for existing UC SDK users. Due to product policy changes, UC SDK is no longer open for applications.

The UC kernel is native C code, and UCCrashSDK is required for getting the UC kernel. The SDK will be installed automatically when the UC kernel component is added.

To report the kernel crash to Mobile Analysis Service backend, you need to add the corresponding receiver in the `Manifest` file.

```
<receiver
  android:name="com.alipay.mobile.common.logging.process.LogReceiverInToolsProcess"
  android:enabled="true"
  android:exported="false"
  android:process=":tools">
  <intent-filter>
    <action android:name="${applicationId}.monitor.command"/>
  </intent-filter>
</receiver>

<receiver
  android:name="com.alipay.mobile.logmonitor.ClientMonitorWakeupReceiver"
  android:enabled="true"
  android:exported="false"
  android:process=":push">
  <intent-filter>
    <action android:name="${applicationId}.push.action.CHECK" />
    <action android:name="${applicationId}.monitor.command" />
  </intent-filter>
</receiver>
```

Note

When the UCCrashSDK is used, an authentication request will be sent to UC.Refer to Use third-party service data for more information.

1.3.4.5. Specify the version of UC kernel

Important

This document is only used for debugging and support for existing UC SDK users. Due to product policy changes, UC SDK is no longer open for applications.

This topic introduces the method to use specific version of UC kernel .

If you need to specify the version of UC kernel when using it, you can refer to the following steps:

1. Remove current UC dependency
2. Access to specified UC version

Remove current UC dependency

If you select the Native AAR access mode to access your project, you do not need to remove the current UC dependency and directly Access to specified UC version.

Access to specified UC version

In the `dependencies` under the `build.gradle` of the Application Module, force a specific version of dependency by adding the following codes. Among the codes, `version` is replaced with the specified version, referring to the value of GAV in the version list.

> **Note**
>
> If the component-based (Portal&Bundle) access mode is selected, you should store the codes in the `dependencies` under the `build.gradle` of the Application Module of the Portal project.

```
dependencies {
    ...
    implementation ('com.alipay.android.phone.wallet:nebulaucsdk-build:version@aar'){
        force = true
    }
    ...
}
```

Version List

UC version	GAV	Remarks
3.22.2.66.230817192043	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.22.2.66.230817192043	Compatible with Android 14.
3.22.2.46.220614210535	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.22.2.46.220614210535	Compatible with Android 13.
3.22.2.30.211011154625	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.22.2.30.211011154625	Based on 3.22.2.18, the low-probability occasional JS channel cache abnormal problem is fixed.
3.22.2.18.210803145558	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.22.2.18.210803145558	The input problem of Rich Media is fixed based on version 3.22.2.17. (Beta version)
3.22.2.17.210719105414	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.22.2.17.210719105414	The latest version of Alipay. (Beta version)
3.21.0.184.210105191337	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.21.0.184.210105191337	Based on version 3.21.0.174, the problem of occasional JS channel cache exceptions is fixed.

3.21.0.174.200825145737	com.alipay.android.phone.wallet:nebulaucsdk-build:999.3.21.0.174.200825145737	The current default baseline of 10.1.68.

1.3.4.6. Custom JSAPI

JavaScript API (JSAPI) is an API that provides native capabilities to HTML5 apps. With these APIs, extensive native capabilities and control capabilities can be leveraged to improve the user experience of HTML5 apps.

The HTML5 container component has the following capabilities:

- Provides abundant embedded JSAPIs to implement functions such as push, pop, and title setup. For more information, see Embedded JSAPIs.
- Allows users to customize JSAPIs and plug-ins to satisfy their business expansion needs.

This topic describes how to customize JSAPIs and plug-ins.

About this task

Custom JSAPI plug-ins have the following features:

- To allow services to flexibly connect to the HTML5 container, the HTML5 container provides a mechanism for registering plug-in configurations for external services.
- In this way, the client can register custom external plug-ins and place the code in their bundles, without the intervention of the HTML5 container during the entire process. When plug-ins are registered based on plug-in configurations, the HTML5 container initializes objects only during page calling and does not generate objects immediately.
- Plug-ins may be implemented and registered in different bundles, achieving decoupled implementation and registration. You need to use `H5Service` to dynamically inject plug-ins into the container.

Note:

- Plug-ins must be injected into JS before a page calls JS. Generally, plug-ins are injected in pipelines of static links. If a plug-in is injected early, JS may be called before the bundle of the container is loaded (h5service==null).
- If a registered bundle is not a static link but a lazy-loaded bundle, JS may be called before metainfo is loaded.

A frontend page calls the native JSAPI

1. Register a JSAPI by using the registration JSAPI of MPNebula.

   Note: You can view the source code of `MyJSApiPlugin` in Code sample.

   - Register a JSAPI as follows:

```
/**
*Register a custom HTML5 plug-in (JSAPI).
*
* @param className: plug-in class name. The full path (package+class) is required.
* @param bundleName: bundle name (the bundle name can be viewed in
module/build/intermediates/bundle/META-INF/BUNDLE.MF. If the plug-in is written into t
he portal project, bundleName should be set to an empty string "".)
* @param scope: scope, which is usually page.
* @param events: a registered event.
*/
public static void registerH5Plugin(String className, String bundleName, String scope,
String[] events)
```

- Registration example:

```
MPNebula.registerH5Plugin(
  MyJSApiPlugin.class.getName(),
  BuildConfig.BUNDLE_NAME,
  "page",
  new String[]{"myapi1","myapi2",H5Plugin.CommonEvents.H5_PAGE_SHOULD_LOAD_URL}
);
```

2. Enable the frontend to call the custom JSAPI.

   Change event to the preceding plug-in registration event to enable the frontend to call the custom JSAPI. The plug-in obtains transferred JS values by using `event.getParam()` and parses data from the values.

```
AlipayJSBridge.call('myapi2', {
 param2: 'World'
},
function(result) {
 console.log(result);
});
```

The native calls a frontend page

The HTML5 container allows the native to actively call a page.

Take a network change JSAPI for example. The page listens to this event as follows:

1. The frontend registers listening.

```
document.addEventListener('h5NetworkChange',
function(e) {
 alert("For any network environment changes, call the getNetworkType API to obtain detai
ls.");
},
false);
```

2. The client listens to network changes and sends a call event to the page.

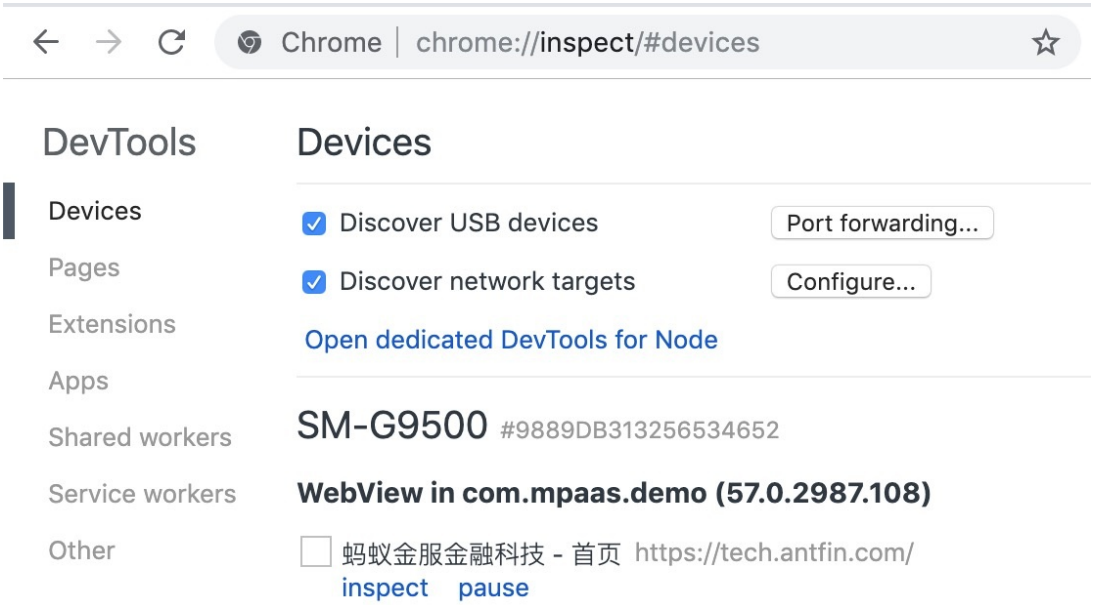
```
JSONObject param = new JSONObject();
// param indicates a custom parameter.
param.put("data", param);
H5Page h5Page = h5Service.getTopH5Page();
if (h5Page != null) {
 h5Page.getBridge().sendDataWarpToWeb("h5NetworkChange", param, null);
}
```

Use the inspect tool

Use the inspect tool of Chrome to check whether the JSAPI is called successfully:

1. Connect a mobile phone to a PC. Open the Chrome browser on the PC and enter `chrome://inspect` in the address bar to open the commissioning page.
2. Use mPaaS demo to open the homepage of Ant Financial. The inspect page of Chrome changes, as shown in the following figure.

   Note: On some PCs, a white screen appears after you input `chrome://inspect`. In this case, upgrade Chrome to the latest version.



3. Click **inspect**. The page shown in the following figure appears.

4. Click **Console** on the toolbar to enter the commissioning mode. Then you can call the custom API.

Related links

- Code sample
- FAQ

1.3.4.7. Custom title bar(10.1.68)

The Nebula container allows you to customize a navigation bar. You can define the style of the navigation bar, such as the position of the title and the style of the Back button. This topic describes how to customize a navigation bar in baseline 10.1.68.

Prerequisites

Before you read this guide, take note of the following key points:

- To develop a navigation bar to be shared by Mini Programs and HTML5 pages, you must take into account the use of navigation bars on HTML5 pages and that in Mini Programs. You can ignore this rule if you want to use the navigation bar only in Mini Programs or only on HTML5 pages.
- The custom navigation bar **must comply with the standard process of calling API operations in the container**. Please carefully read this document and develop the custom navigation bar as required.
- By default, Mini Programs use the built-in navigation bar. To use a custom navigation bar in Mini Programs, see Configure the HTML5 container.
- The color of the navigation bar can be dynamically set. To achieve the best experience, you must prepare **two sets of theme configurations** and switch between them based on scenarios.

Procedure

1. Inherit the abstract class **AbsTitleView** and implement a custom navigation bar.
2. Implement `H5ViewProvider`. In the `createTitleView` method, create and return an instance of the custom navigation bar.

```
public class H5ViewProviderImpl implements H5ViewProvider {
     @Override
     public H5TitleView createTitleView(Context context) {
         return new NewH5TitleViewImpl(context);
     }

     @Override
     public H5NavMenuView createNavMenu() {
         return null;
     }

     @Override
     public H5PullHeaderView createPullHeaderView(Context context, ViewGroup viewGroup)
{
         return null;
     }

     @Override
     public H5WebContentView createWebContentView(Context context) {
         return null;
     }
 }
```

3. Set `H5ViewProvider` in the container in specific scenarios, such as application startup.

```
MPNebula.setCustomViewProvider(new H5ViewProviderImpl());
```

4. If your project is based on the Portal and Bundle architecture, additional configurations are required.

```
H5Utils.setProvider(H5ReplaceResourceProvider.class.getName(), new
H5ReplaceResourceProvider() {
     @Override
     public String getReplaceResourcesBundleName() {
         return BuildConfig.BUNDLE_NAME;
     }
 });
```

More information

Background color

```
/**
     * Return the value of the background color of the navigation bar.
     * @return
     */
    public abstract int getBackgroundColor();

    /**
     * Set the transparency of the navigation bar.
     * @param alpha
     */
    public abstract void setBackgroundAlphaValue(int alpha);

    /**
     * Sets the background color of the navigation bar without using Alpha values.
     * @param color
     */
    public abstract void setBackgroundColor(int color);

    /**
     * Resets the navigation bar.
     */
    public abstract void resetTitle();
```

The resetTitle operation is triggered when the JSAPI named setTitleColor is called on a foreground page. In this case, we recommend that you reset the display elements to default elements. These display elements include the background of the navigation bar and the color values of other components mentioned later in this topic.

Title

Subtitles are supported in HTML5 page-only scenarios. If the application does not need a subtitle, you may ignore implementation of the subtitle.

To enable an HTML5 page to listen to events of tapping the title bar, you must enable the navigation bar to call the `invokeTitleClickEvent` method in specified scenarios. To enable an HTML5 page to listen to events of tapping the subtitle bar, you must enable the navigation bar to call the `invokeSubTitleClickEvent` method.

```
/**
     * Return the text of the main title.
     * @return
     */
    public abstract String getTitle();

    /**
     * Set the text of the main title.
     * @param title
     */
    public abstract void setTitle(String title);

    /**
     * Set the text of the subtitle.
     * @param subTitle
     */
    public abstract void setSubTitle(String subTitle);

    /**
     * Return the view of the main title.
     * @return
     */
    public abstract TextView getMainTitleView();

    /**
     * Return the view of the subtitle.
     * @return
     */
    public abstract TextView getSubTitleView();
```

Left-side control section

```
/**
     * Set whether to show the Close button.
     * @param visible
     */
    public abstract void showCloseButton(boolean visible);

    /**
     * Set whether to show the Back button.
     * @param visible
     */
    public abstract void showBackButton(boolean visible);

    /**
     * Set whether to show the Home button.
     * @param visible
     */
    public abstract void showBackHome(boolean visible);

    /**
     * Set whether to show the loading progress icon in the title bar.
     * @param visible
     */
    public abstract void showTitleLoading(boolean visible);
```

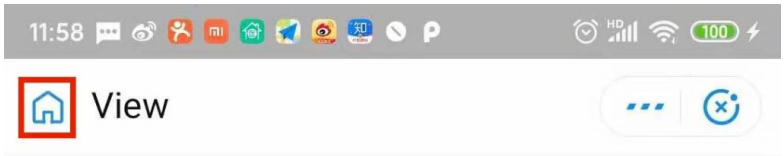
Close button

As shown in the red box in the preceding figure, the Close button only appears in HTML5 pages. When more than one historical online pages exist in a browser, the container calls the `showCloseButton` method to control whether to show the Close button. When the Close button is tapped, you must call the `invokePageCloseEvent` method to comply with the container behavior specification.

Back button

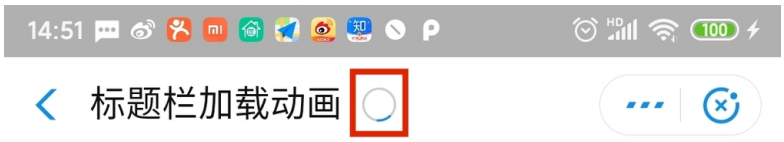
The Back button shown in the red box in the preceding figure is a control that **must** be implemented in a custom navigation bar. The container calls the `showBackButton` method to control whether to show the Back button. When the Back button is tapped, you must call the `invokePageBackEvent` method to comply with the container behavior specification.

Home button



The Home button is used only in Mini Programs. When you are redirected to a non-homepage of a Mini Program, the container calls the `showBackHome` method to control whether to show the Home button. When the Home button is tapped, you must call the `invokeHomeClickEvent` method to comply with the container behavior specification.

Loading progress icon

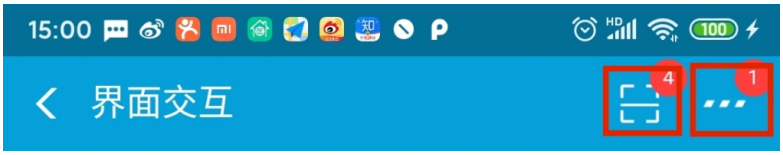


When an HTML5 page or a Mini Program calls the API for the loading animation in the navigation bar, the container calls the `showTitleLoading` method to control whether to show the loading progress icon.

Right-side control section

The right-side control section is also referred to as the `OptionMenu` section. It is mainly used to provide more operations for users.

- In the HTML5 container:



- In Mini Programs:

As shown in the preceding figures, you must reserve two view sections for the `OptionMenu` section. The container manages these two sections based on indexes. The indexes are sorted **from right to left and starts with 0** .

```
public abstract void showOptionMenu(boolean visible);

    public abstract View getOptionMenuContainer(int index);

    public abstract void setOptionMenu(boolean reset, boolean override, boolean isTinyApp,
List<MenuData> menus);
```

The container calls the `showOptionMenu` method to control whether to show the `OptionMenu` section. In some cases, the container must obtain the view in this section and perform operations based on the view. You must properly implement the `getOptionMenuContainer` method.

To implement the `setOptionMenu` method, see request parameters in Set upper-right buttons. `icontype` parameter is a built-in button style. It is available only on HTML5 pages. If HTML5 pages are not used in your access scenarios, you can ignore this parameter. Otherwise, you must configure buttons for different styles. Similar to the `icontype` parameter, the `redDot` parameter is optional.

To enable an HTML5 page or a Mini Program to listen to the events of tapping upper-right buttons, you must call the `invokeOptionClickEvent` method.

Take note of the following settings in Mini Programs:

- To distinguish between HTML5 pages and Mini Programs, you must set the `isTinyApp` parameter in the `setOptionMenu` method to `true`.
- When you set multiple buttons, you must start from **the second button from the right**.

Upper-right control section in Mini Programs

In Mini Programs, you must specially implement the right-side section by performing the following steps:

1. Use the legacy abstract class `AbsTinyOptionMenuView` to implement a custom control section.
2. Set `TinyOptionMenuViewProvider` in the container in specific scenarios, such as application startup.

```
H5Utils.setProvider(TinyOptionMenuViewProvider.class.getName(), new
TinyOptionMenuViewProvider() {
    @Override
    public AbsTinyOptionMenuView createView(Context context) {
        return new TinyOptionMenuView(context);
    }
});
```

You must implement and configure the views of the **More** and **Close** buttons based on the container specification.

```
public abstract void setOptionMenuOnClickListener(View.OnClickListener listener);

    public abstract void setCloseButtonOnClickListener(View.OnClickListener listener);

    public abstract void setCloseButtonOnLongClickListener(View.OnLongClickListener listen
er);

    public abstract void onStateChanged(TinyAppActionState currentState);

    public abstract View getView();
```

The container calls the first three methods in the preceding code to set a reasonable response callback. You must set the response callback in the specified view.

`The onStateChanged` method is called in LBS(location-based service) and Bluetooth scenarios. For example, when a Mini Program is using LBS(location-based service), the container calls this method, and you can respond to the callback. The following code shows the style.

The following sample code is for your reference.

```
public class TinyOptionMenuView extends AbsTinyOptionMenuView {

    private View container;

    private ImageView ivMore;

    private View ivClose;

    private Context context;
```

```
    private Context context;

    public TinyOptionMenuView(Context context) {
        this.context = context;
        ViewGroup parent = null;
        if (context instanceof Activity) {
            parent = (ViewGroup) ((Activity) context).findViewById(android.R.id.content);
        }
        container = LayoutInflater.from(context).inflate(R.layout.layout_tiny_right,
parent, false);
        ivClose = container.findViewById(R.id.close);
        ivMore = (ImageView) container.findViewById(R.id.more);

    }

    @Override
    public View getView() {
        return container;
    }

    @Override
    public void setOptionMenuOnClickListener(View.OnClickListener onClickListener) {
        ivMore.setOnClickListener(onClickListener);
    }

    @Override
    public void setCloseButtonOnClickListener(View.OnClickListener onClickListener) {
        ivClose.setOnClickListener(onClickListener);
    }

    @Override
    public void setCloseButtonOnLongClickListener(View.OnLongClickListener
onLongClickListener) {
        ivClose.setOnLongClickListener(onLongClickListener);
    }

    @Override
    public void onStateChanged(TinyAppActionState state) {
        if (state == null) {

ivMore.setImageDrawable(context.getResources().getDrawable(R.drawable.icon_more));
        } else if (state.getAction().equals(TinyAppActionState.ACTION_LOCATION)) {

ivMore.setImageDrawable(context.getResources().getDrawable(R.drawable.icon_miniprogram_locat
n));
        }
    }
}
```

Theme changes

Different Mini Programs or HTML5 applications may use different background colors in navigation bars. To improve user experience, you also need to adjust other elements in the navigation bar, such as the upper-right control section, in response to changes in the background colors in navigation bars.

In the extension to the navigation bar, the upper-right control section and the navigation bar are different components. Therefore, API operations are provided for you to enable the upper-right control section to respond to changes in the navigation bar.

- `The AbsTinyOptionMenuView` class provides the `onTitleChanged` method to allow you to use override to respond to changes in the navigation bar. When this method is called, the H5TitleView object can be passed in to obtain the information about the navigation bar, such as the background color. In addition, the `AbsTinyOptionMenuView` class provides the `getTitleBar` method to allow you to directly obtain objects of the navigation bar. Alternatively, you can convert H5TitleView to a navigation bar object that you can implement, because the `AbsTitleView` class implements the `H5TitleView` class. This allows you to obtain more information about the navigation bar.

```
protected void onTitleChange(H5TitleView title)
```

- You must proactively call the `notifyTitleBarChanged` method provided by the `AbsTitleView` class, to enable the `onTitleChange` operation to be called, for example, when the background color of the navigation bar is set. The following sample code is for your reference.

```
package com.mpaas.demo.nebula;

 public class NewH5TitleViewImpl extends AbsTitleView {

     @Override
     public void setBackgroundAlphaValue(int i) {
         content.getContentBgView().setAlpha(i);
         notifyTitleBarChanged();
     }

     @Override
     public void setBackgroundColor(int i) {
         content.getContentBgView().setColor(i);
         notifyTitleBarChanged();
     }

     @Override
     public void resetTitle() {

content.getContentBgView().setColor(context.getResources().getColor(R.color.h5_default_tit
bar_color));
         notifyTitleBarChanged();
     }
 }
```

In the preceding sample code, when the `notifyTitleBarChange` method is called, objects of the `AbsTinyOptionMenuView` subclass may be not 100% initialized. Therefore, we recommend that you override the `setH5Page` method to obtain the information about the navigation bar and determine the current theme. The following sample code is for your reference.

```
public class TinyOptionMenuView extends AbsTinyOptionMenuView {

    @Override
    public void setH5Page(H5Page h5Page) {
        super.setH5Page(h5Page);
        // title becomes available from here.
        if (getTitleBar().getBackgroundColor() == -1) {
            bgView.setBackgroundColor(Color.RED);
        }
    }
}
```

1.3.4.8. Custom title bar

If you need to customize HTML5 container title bar, refer to the following code samples:

10.1.60

H5TitleViewImpl.java

```
package com.mpaas.demo.nebula;

import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.TextView;

import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.alipay.mobile.h5container.api.H5Page;
import com.alipay.mobile.h5container.api.H5Param;
import com.alipay.mobile.h5container.api.H5Plugin;
import com.alipay.mobile.nebula.util.H5Log;
import com.alipay.mobile.nebula.util.H5StatusBarUtils;
import com.alipay.mobile.nebula.util.H5Utils;
import com.alipay.mobile.nebula.view.H5TitleBarFrameLayout;
import com.alipay.mobile.nebula.view.H5TitleView;
import com.alipay.mobile.nebula.view.IH5TinyPopMenu;

import java.util.ArrayList;
import java.util.List;

/**
 * Created by omg on 2018/7/23.
 */

public class H5TitleViewImpl implements H5TitleView, View.OnClickListener {
```

```
    private static final String TAG = "H5TitleViewImpl";

    private Context mContext;

    private H5TitleBarFrameLayout contentView;

    private String title;

    // Define the basic controls of the title bar
    private TextView mTitleView;

    private TextView mSubTitleView;

    private View mCloseButton;

    private View mBackButton;

    private View vDivider;

    private View hDivider;

    private View statusBarAdjustView;

    /**
     * ==== Each View of OptionMenu. Start ====
     */
    // Container of OptionMenu
    public View h5NavOptions;
    public View h5NavOptions1;
    public List<View> h5NavOptionsList = new ArrayList<View>();

    // ---- Three typs of OptionMenu. Start ---- //
    // 1. OptionType.MENU (default) - The default Option button.
    public TextView btMenu;
    public TextView btMenu1;
    public List<TextView> btMenuList = new ArrayList<TextView>();

    // 2. OptionType.ICON - The icon manually set via setOptionMenu.
    public ImageButton btIcon;
    public ImageButton btIcon1;
    public List<ImageButton> btIconList = new ArrayList<ImageButton>();

    // 3. OptionType.TEXT - test
    public TextView btText;
    public TextView btText1;
    public List<TextView> btTextList = new ArrayList<TextView>();
    // ---- Three typs of OptionMenu. Over ---- //

    // Instance interface class of Web page, used to control web action
    private H5Page h5Page;

    private int visibleOptionNum = 0;
    private IH5TinyPopMenu h5TinyPopMenu;

    /**
```

```
     * The method to construct HTML5 container title bar
     * Note: Title bar layout xml file must have H5TitleBarFrameLayout as the root node.
     * @param context
     */
    public H5TitleViewImpl(Context context) {
        mContext = context;
        ViewGroup parent = null;
        if (context instanceof Activity) {
            parent = (ViewGroup) ((Activity) mContext).findViewById(android.R.id.content);
        }
        contentView = (H5TitleBarFrameLayout)
LayoutInflater.from(context).inflate(R.layout.h5_navigation_bar, parent, false);

contentView.getContentBgView().setColor(context.getResources().getColor(R.color.h5_default_t
lebar_color));
        mTitleView = (TextView) contentView.findViewById(R.id.h5_tv_title);
        mTitleView.setOnClickListener(this);
        mSubTitleView = (TextView) contentView.findViewById(R.id.h5_tv_subtitle);
        mSubTitleView.setOnClickListener(this);
        mCloseButton = contentView.findViewById(R.id.h5_nav_close);
        mCloseButton.setOnClickListener(this);
        mBackButton = contentView.findViewById(R.id.h5_tv_nav_back);
        mBackButton.setOnClickListener(this);
        vDivider = contentView.findViewById(R.id.h5_v_divider);
        hDivider = contentView.findViewById(R.id.h5_h_divider_intitle);
        h5NavOptions = contentView.findViewById(R.id.h5_nav_options);
        h5NavOptions1 = contentView.findViewById(R.id.h5_nav_options1);
        statusBarAdjustView = contentView.findViewById(R.id.h5_status_bar_adjust_view);

        btIcon = (ImageButton) contentView.findViewById(R.id.h5_bt_image);
        btText = (TextView) contentView.findViewById(R.id.h5_bt_text);
        btMenu = (TextView) contentView.findViewById(R.id.h5_bt_options);

        btIcon1 = (ImageButton) contentView.findViewById(R.id.h5_bt_image1);
        btText1 = (TextView) contentView.findViewById(R.id.h5_bt_text1);
        btMenu1 = (TextView) contentView.findViewById(R.id.h5_bt_options1);

        //add view to list
        h5NavOptionsList.add(h5NavOptions);
        h5NavOptionsList.add(h5NavOptions1);

        btIconList.add(btIcon);
        btIconList.add(btIcon1);

        btTextList.add(btText);
        btTextList.add(btText1);

        btMenuList.add(btMenu);
        btMenuList.add(btMenu1);

        btText.setOnClickListener(this);
        btIcon.setOnClickListener(this);
        btText1.setOnClickListener(this);
        btIcon1.setOnClickListener(this);
        btMenu.setOnClickListener(this);
        btMenu1.setOnClickListener(this);
```

```
        btnMenu1.setOnClickListener(this);

    }

    /**
     * Container calls this method to obtain the content of the main title.
     */
    @Override
    public String getTitle() {
        return title;
    }

    /**
     * Container calls this method to set the content of the main title.
     */
    @Override
    public void setTitle(String s) {
        title = s;
        mTitleView.setText(s);
    }

    /**
     * Container calls this method to set the content of the sub title.
     */
    @Override
    public void setSubTitle(String s) {
        mSubTitleView.setVisibility(View.VISIBLE);
        mSubTitleView.setText(s);
    }

    /**
     * Ignore for now, not necessary.
     */
    @Override
    public void setImgTitle(Bitmap bitmap) {

    }

    /**
     * Ignore for now, not necessary.
     */
    @Override
    public void setImgTitle(Bitmap bitmap, String s) {

    }

    /**
     * Set whether to dispaly the close button for the container.
     */
    @Override
    public void showCloseButton(boolean b) {
        mCloseButton.setVisibility(b ? View.VISIBLE : View.GONE);
    }

    /**
     * Container obtains title bar View.
     */
```

```
     */
    @Override
    public View getContentView() {
        return contentView;
    }

    /**
     * Container obtains title bar background for setting background color.
     */
    @Override
    public ColorDrawable getContentBgView() {
        return contentView.getContentBgView();
    }

    /**
     * Container obtains main title View.
     * Cannot be empty
     */
    @Override
    public TextView getMainTitleView() {
        return mTitleView;
    }

    /**
     * Container obtains subtitle View.
     * Cannot be empty
     */
    @Override
    public TextView getSubTitleView() {
        return mSubTitleView;
    }

    /**
     * Set whether to display back button.
     */
    @Override
    public void showBackButton(boolean b) {
        mBackButton.setVisibility(b ? View.VISIBLE : View.GONE);
    }

    /**
     * Set whether to display the top right menu.
     */
    @Override
    public void showOptionMenu(boolean isShow) {
        if (isShow) {
            switch (visibleOptionNum) {
                case 1:
                    h5NavOptions.setVisibility(View.VISIBLE);
                    break;
                case 2:
                    h5NavOptions.setVisibility(View.VISIBLE);
                    h5NavOptions1.setVisibility(View.VISIBLE);
                    break;
            }
        } else {
```

```
            h5NavOptions.setVisibility(View.GONE);
            h5NavOptions1.setVisibility(View.GONE);
        }
    }

    /**
     * Seth the display type of the top right menu, which can be icon or text.
     */
    @Override
    public void setOptionType(H5Param.OptionType optionType) {
        setOptionType(optionType, 0, true);
    }

    /**
     * Seth the display type of the top right menu, which can be icon or text.
     * @param byIndex : whether to set the display type only for a certain menu.
     */
    @Override
    public void setOptionType(H5Param.OptionType type, int num, boolean byIndex) {
        boolean icon = false;
        boolean text = false;
        boolean menu = false;
        if (type == H5Param.OptionType.ICON) {
            icon = true;
        } else if (type == H5Param.OptionType.TEXT) {
            text = true;
        } else if (type == H5Param.OptionType.MENU) {
            menu = true;
        }
        ctrlbtText(num, text ? View.VISIBLE : View.GONE, byIndex);
        ctrlbtIcon(num, icon ? View.VISIBLE : View.INVISIBLE, byIndex);
        ctrlbtMenu(num, menu ? View.VISIBLE : View.INVISIBLE, byIndex);
    }

    //view visible control
    private boolean isOutOfBound(int num, int length) {
        if (length == 0) {
            return true;
        }
        return length < num;
    }

    private void ctrlbtText(int num, int visible, boolean byIndex) {
        if (isOutOfBound(num, btTextList.size())) {
            return;
        }
        if (byIndex) {
            btTextList.get(num).setVisibility(visible);
        } else {
            for (int i = 0; i < num; i++) {
                btTextList.get(i).setVisibility(visible);
            }
        }
    }

    private void ctrlbtIcon(int num, int visible, boolean byIndex) {
```

```
        if (isOutOfBound(num, btIconList.size())) {
            return;
        }
        if (byIndex) {
            btIconList.get(num).setVisibility(visible);
        } else {
            for (int i = 0; i < num; i++) {
                btIconList.get(i).setVisibility(visible);
            }
        }
    }

    private void ctrlbtMenu(int num, int visible, boolean byIndex) {
        if (isOutOfBound(num, btMenuList.size())) {
            return;
        }
        if (byIndex) {
            btMenuList.get(num).setVisibility(visible);
        } else {
            for (int i = 0; i < num; i++) {
                btMenuList.get(i).setVisibility(visible);
            }
        }
    }

    /**
     * Set whether to display the loading status on the title bar, you can choose the impl
etment method yourself.
     */
    @Override
    public void showTitleLoading(boolean b) {

    }

    /**
     * Ignore for now
     */
    @Override
    public void showTitleDisclaimer(boolean b) {

    }

    // Set the icon of the top right button.
    @Override
    public void setBtIcon(Bitmap btIcon, int index) {
        if (isOutOfBound(index, btIconList.size())) {
            return;
        }
        btIconList.get(index).setImageBitmap(btIcon);
    }

    @Override
    public void setH5Page(H5Page h5Page) {
        this.h5Page = h5Page;
    }
```

```
    /**
     * Set the top right menu according to the parameters passed from JS
     */
    @Override
    public void setOptionMenu(JSONObject params) {
        boolean reset = H5Utils.getBoolean(params, "reset", false);
        boolean override = H5Utils.getBoolean(params, "override", false);
        JSONArray menus = H5Utils.getJSONArray(params, "menus", null);
        if (reset) {
            h5NavOptions1.setVisibility(View.GONE);
            setOptionType(H5Param.OptionType.MENU, 0, true);
            visibleOptionNum = 1;
            return;
        }
        if (menus != null && !menus.isEmpty()) {
            visibleOptionNum = 0;
            if (override) {
                int menuSize = menus.size() > 2 ? 2 : menus.size();
                for (int i = 0; i < menuSize; i++) {
//                    h5NavOptionsList.get(i).setVisibility(View.VISIBLE);
                    JSONObject menusItem = menus.getJSONObject(i);
                    setOptionMenuInternal(menusItem, i);
                    visibleOptionNum++;
                }
            } else {
                visibleOptionNum = 2;
//                h5NavOptionsList.get(1).setVisibility(View.VISIBLE);
                JSONObject menusItem = menus.getJSONObject(0);
                setOptionMenuInternal(menusItem, 1);
            }
        } else {
            setOptionMenuInternal(params, 0);
            visibleOptionNum = 1;
        }
    }

    private void setOptionMenuInternal(JSONObject params, int index) {
        String title = H5Utils.getString(params, "title");
        String icon = H5Utils.getString(params, "icon");
        String icontype = H5Utils.getString(params, "icontype");
        String contentDesc = H5Utils.getString(params, "contentDesc");
        String colorText = H5Utils.getString(params, "color");

        // default white color
        int color = 0xFF108ee9;
        if (!TextUtils.isEmpty(colorText)) {
            try {
                color = Color.parseColor(colorText);
            } catch (Throwable ignore) {
                //can not find logutil
            }
            color = 0xFF000000 | color;
            btTextList.get(index).setTextColor(color);
        } else {
            int currentColor = mTitleView.getCurrentTextColor();
            currentColor = 0xFF000000 | currentColor;
```

```
            H5Log.d(TAG, "setOptionMenuInternal currentColor is " + currentColor);
            if (currentColor != 0xFF111111) {
                btText.setTextColor(0xFFFFFFFF);
                btText1.setTextColor(0xFFFFFFFF);
            } else {
                btText.setTextColor(0xFF108ee9);
                btText1.setTextColor(0xFF108ee9);
            }
        }

        if (!TextUtils.isEmpty(title)) {
            title = title.trim();
            btTextList.get(index).setText(title);
            setOptionType(H5Param.OptionType.TEXT, index, true);
            btTextList.get(index).setContentDescription(title);
        } else if (!TextUtils.isEmpty(icon) || !TextUtils.isEmpty(icontype)) {
            if (!TextUtils.isEmpty(contentDesc)) {
                btIconList.get(index).setContentDescription(contentDesc);
            }
        }
    }

    /**
     * Container obtais the split line between the back button and the title content.
     * The return value can be empty.
     */
    @Override
    public View getDivider() {
        return vDivider;
    }

    /**
     * Container obtains the split line between the title bar and Web page.
     * Cannot be empty
     */
    @Override
    public View getHdividerInTitle() {
        return hDivider;
    }

    /**
     * Container obtains the anchor view of pull-down menu pop-up position.
     */
    @Override
    public View getPopAnchor() {
        return btMenu;
    }

    /**
     * Container resets the background color of title bar.
     */
    @Override
    public void resetTitleColor(int color) {

    }
```

```
    /**
     * Ignore for now
     */
    @Override
    public void switchToWhiteTheme() {

    }

    /**
     * Ignore for now
     */
    @Override
    public void switchToBlueTheme() {

    }

    /**
     * Triggered when container page is distroyed. The referred View can be released here.
     */
    @Override
    public void releaseViewList() {
        if (h5NavOptionsList != null) {
            h5NavOptionsList.clear();
        }
        if (btIconList != null) {
            btIconList.clear();
        }
        if (btTextList != null) {
            btTextList.clear();
        }
        if (btMenuList != null) {
            btMenuList.clear();
        }
    }


    /**
     * Container sets the color of translucent title bar
     */
    @Override
    public void openTranslucentStatusBarSupport(int color) {
        if (H5StatusBarUtils.isSupport()) {
            int statusBarHeight = H5StatusBarUtils.getStatusBarHeight(mContext);

            if (statusBarHeight == 0) { //Protection. In case rom cannot get the height of
status bar, it doesn't take effect here.
                return;
            }
            LinearLayout.LayoutParams layoutParams =
                    (LinearLayout.LayoutParams) statusBarAdjustView.getLayoutParams();
            layoutParams.height = statusBarHeight;
            statusBarAdjustView.setLayoutParams(layoutParams);
            statusBarAdjustView.setVisibility(View.VISIBLE);

            try {
                H5StatusBarUtils.setTransparentColor((Activity) mContext, color);
```

```
            } catch (Exception e) {
                H5Log.e(TAG, e);
            }
        }
    }

    /**
     * Ignore for now
     */
    @Override
    public void switchToTitleBar() {

    }

    /**
     * Ignore for now
     */
    @Override
    public View setTitleBarSearch(Bundle bundle) {
        return null;
    }

    /**
     * Ignore for now
     */
    @Override
    public void setBackCloseBtnImage(String s) {

    }

    /**
     * Set the font color of title bar
     */
    @Override
    public void setTitleTxtColor(int i) {
        mTitleView.setTextColor(i);
        mSubTitleView.setTextColor(i);
    }

    /**
     * Container obtains the top right menu View, which must be ViewGroup and its sub-clas
s.
     */
    @Override
    public View getOptionMenuContainer() {
        return h5NavOptions;
    }

    /**
     * According to the position, container obtains the top right menu View, which must be
ViewGroup and its sub-class.
     */
    @Override
    public View getOptionMenuContainer(int index) {
        switch (index) {
            case 0:
```

```
                return h5NavOptions;
            case 1:
                return h5NavOptions1;
            default:
                return getOptionMenuContainer();
        }
    }

    /**
     * Ignore for now
     */
    @Override
    public void setIH5TinyPopMenu(IH5TinyPopMenu tinyPopMenu) {
        this.h5TinyPopMenu = tinyPopMenu;
    }

    /**
     * Ignore for now
     */
    @Override
    public IH5TinyPopMenu getH5TinyPopMenu() {
        return null;
    }

    /**
     * Ignore for now
     */
    @Override
    public void setTitleView(View view) {

    }

    /**
     * Ignore for now
     */
    @Override
    public void initTitleSegControl(JSONObject jsonObject) {

    }

    /**
     * Ignore for now
     */
    @Override
    public void enableTitleSegControl(boolean b) {

    }

    /**
     * Ignore for now
     */
    @Override
    public void enableBackButtonBackground(boolean b) {

    }
```

```
    /**
     * Handle the click events of different controls on the title bar
     * If JS needs to receive events, the events need to be sent to JS as shown in the cod
e.
     * For example, clicking the back button will send the
"H5Plugin.CommonEvents.H5_TOOLBAR_BACK" event.
     */
    @Override
    public void onClick(View view) {
        if (h5Page == null) {
            return ;
        }

        String eventName = null;
        JSONObject data = null;

        if (view == mBackButton) {
            eventName = H5Plugin.CommonEvents.H5_TOOLBAR_BACK; // Send back event
        } else if (view == mCloseButton) {
            eventName = H5Plugin.CommonEvents.H5_TOOLBAR_CLOSE; // Send closing page event
        }else if (view.equals(btIcon) || view.equals(btText)) {
            eventName = H5Plugin.CommonEvents.H5_TITLEBAR_OPTIONS;
            data = new JSONObject();
            data.put("index", 0);
        } else if (view.equals(btIcon1) || view.equals(btText1)) {
            eventName = H5Plugin.CommonEvents.H5_TITLEBAR_OPTIONS;
            data = new JSONObject();
            data.put("index", 1);
        } else if (view.equals(btMenu) || view.equals(btMenu1)) {
            eventName = H5Plugin.CommonEvents.H5_TITLEBAR_OPTIONS;
            data = new JSONObject();
            data.put("fromMenu", true);
            data.put("index", view.equals(btMenu) ? 0 : 1);
        } else if (view.equals(mTitleView)) {
            eventName = H5Plugin.CommonEvents.H5_TITLEBAR_TITLE;
        } else if (view.equals(mSubTitleView)) {
            eventName = H5Plugin.CommonEvents.H5_TITLEBAR_SUBTITLE;
        }

        if (!TextUtils.isEmpty(eventName)) {
            h5Page.sendEvent(eventName, data);
        }
    }
}
```

h5_navigation_bar.xml

```
<?xml version="1.0" encoding="utf-8"?>
<com.alipay.mobile.nebula.view.H5TitleBarFrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/h5_title_bar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@android:color/transparent">
```

```
<LinearLayout
    android:id="@+id/h5_rl_title_bar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@android:color/transparent"
    android:orientation="vertical">

    <View
        android:id="@+id/h5_status_bar_adjust_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:visibility="gone"/>

    <RelativeLayout
        android:id="@+id/h5_title_bar_layout"
        android:layout_width="match_parent"
        android:layout_height="48dp">

        <ImageButton
            android:id="@+id/h5_tv_nav_back"
            android:layout_width="48dp"
            android:layout_height="match_parent"
            android:layout_alignParentLeft="true"
            android:layout_centerVertical="true"
            android:background="@android:color/transparent"
            android:scaleType="centerInside"
            android:padding="12dp"
            android:src="@drawable/back"/>

        <ImageButton
            android:id="@+id/h5_nav_close"
            android:layout_width="48dp"
            android:layout_height="match_parent"
            android:layout_centerVertical="true"
            android:padding="12dp"
            android:layout_marginLeft="-6dp"
            android:layout_toRightOf="@+id/h5_tv_nav_back"
            android:background="@android:color/transparent"
            android:clickable="true"
            android:scaleType="centerInside"
            android:src="@drawable/close"/>

        <View
            android:id="@+id/h5_v_divider"
            android:layout_width="0.7dp"
            android:layout_height="24dp"
            android:layout_centerVertical="true"
            android:layout_gravity="center"
            android:layout_marginRight="12dp"
            android:layout_toRightOf="@+id/h5_nav_close"
            tools:ignore="ContentDescription"/>

        <RelativeLayout
            android:id="@+id/h5_rl_title"
            android:layout_width="match_parent"
```

```
            android:layout_height="wrap_content"
            android:layout_centerVertical="true"
            android:gravity="center">


            <LinearLayout
                android:id="@+id/h5_ll_title"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:gravity="center"
                android:orientation="vertical">

                <FrameLayout
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content">

                    <TextView
                        android:id="@+id/h5_tv_title"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:ellipsize="end"
                        android:singleLine="true"
                        android:textColor="@android:color/white"
                        android:textSize="16dp" />

                    <ImageView
                        android:id="@+id/h5_tv_title_img"
                        android:layout_width="wrap_content"
                        android:layout_height="36dp"
                        android:scaleType="centerInside"
                        android:visibility="gone"/>
                </FrameLayout>

                <TextView
                    android:id="@+id/h5_tv_subtitle"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:singleLine="true"
                    android:textColor="@android:color/white"
                    android:textSize="12dp"
                    android:visibility="gone"
                    tools:visibility="visible" />
            </LinearLayout>

        </RelativeLayout>

        <!-- optionmenu0-->
        <FrameLayout
            android:id="@+id/h5_nav_options"
            android:layout_width="wrap_content"
            android:layout_height="fill_parent"
            android:layout_alignParentRight="true"
            android:layout_centerVertical="true"
            android:layout_marginLeft="6dp">

            <ImageButton
```

```
            android:id="@+id/h5_bt_image"
            android:layout_width="32dp"
            android:layout_height="32dp"
            android:layout_gravity="center|right"
            android:layout_marginRight="12dp"
            android:background="@android:color/transparent"
            android:padding="4dp"
            android:scaleType="fitCenter"
            android:visibility="gone"/>

        <TextView
            android:id="@+id/h5_bt_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center|right"
            android:layout_marginRight="12dp"
            android:background="@null"
            android:ellipsize="end"
            android:maxLength="8"
            android:singleLine="true"
            android:textColor="#ffffff"
            android:textSize="16dp"/>

        <TextView
            android:id="@+id/h5_bt_options"
            android:layout_width="48dp"
            android:layout_height="match_parent"
            android:background="@null"
            android:gravity="center"
            android:textSize="23dp"
            android:visibility="gone"
            tools:visibility="gone" />
    </FrameLayout>

    <!-- optionmenu1-->
    <FrameLayout
        android:id="@+id/h5_nav_options1"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
        android:layout_centerVertical="true"
        android:layout_marginLeft="6dp"
        android:layout_toLeftOf="@id/h5_nav_options"
        android:visibility="gone"
        tools:visibility="visible">

        <ImageButton
            android:id="@+id/h5_bt_image1"
            android:layout_width="32dp"
            android:layout_height="32dp"
            android:layout_gravity="center|right"
            android:layout_marginRight="12dp"
            android:background="@android:color/transparent"
            android:padding="4dp"
            android:scaleType="fitCenter"
            android:visibility="gone"/>
```

```
                <TextView
                    android:id="@+id/h5_bt_text1"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_gravity="center|right"
                    android:layout_marginRight="12dp"
                    android:background="@null"
                    android:ellipsize="end"
                    android:maxLength="8"
                    android:singleLine="true"
                    android:textColor="#108ee9"
                    android:textSize="16dp"/>

            </FrameLayout>
        </RelativeLayout>

        <View
            android:id="@+id/h5_h_divider_intitle"
            android:layout_width="match_parent"
            android:layout_height="1px"
            android:background="@android:color/white"
            android:visibility="gone"/>

    </LinearLayout>

</com.alipay.mobile.nebula.view.H5TitleBarFrameLayout>
```

1.3.4.9. Manage HTML5 pages

After opening an HTML5 offline package, you can choose synchronous method or asynchronous method to embed the view of a single container into the page.

> **Note**
>
> The asynchronous method does not occupy the main thread and therefore does not compromise the performance.

The codes of choosing synchronous method to embed the view of a single container into the HTML5 page are as follows:

```
public static final void openH5(String url) {
      if (TextUtils.isEmpty(url)) {
          return;
      }
      H5Service h5Service =
LauncherApplicationAgent.getInstance().getMicroApplicationContext()
              .findServiceByInterface(H5Service.class.getName());
      H5Bundle bundle = new H5Bundle();
      Bundle param = new Bundle();
      // The app ID of the offline package to be opened.
      param.putString(H5Param.APP_ID, appId);
      // The URL of the offline package to be opened, /www/index.html, in which the slash
(/) must be added.
     // If no URL is transferred, the container opens the URL configured for the offline p
ackage by default.
      param.putString(H5Param.LONG_URL,url);
      bundle.setParams(param);
      if (h5Service != null) {
          H5Page h5Page=h5Service.createPage(activity,bundle);
        View view=h5Page.getContentView(),
       // The view is finally added to the page.
      }
  }
```

The codes of choosing asynchronous method to embed the view of a single container into the HTML5 page are as follows:

```
H5Service h5Service = LauncherApplicationAgent.getInstance().getMicroApplicationContext()
                    .findServiceByInterface(H5Service.class.getName());
H5Bundle bundle = new H5Bundle();
Bundle param = new Bundle();
       param.putString(H5Param.APP_ID, appId);
       param.putString(H5Param.LONG_URL, url);
       bundle.setParams(param);
       if (h5Service != null) {
               h5Service.createPageAsync(activity, bundle, h5PageReadyListener);
       }
```

1.3.4.10. HTML5 container configuration

HTML5 container has many switch configurations. By modifying the switch configuration, you can change the specific behavior of the container. For example, the offline package signature verification can be turned on or off by the verification configuration.

There are three ways to modify the switch configuration:

- Add the `custom_config.json` file to the config folder under assets directory of the portal project or the application's main project. This method is only available for **10.1.60** and above. The file format of `custom_config.json` is as follows:

```
[
{
  "value": "NO",
  "key": "h5_shouldverifyapp"
},
{
  "value": "0",
  "key": "TSBS"
}
]
```

- Use `H5ExtConfigProvider` to configure the switch in the codes. This method is only available for versions below **10.1.60**. `H5ExtConfigProvider` instructions for use are as follows

```
public class H5ExtConfigProviderImpl implements H5ExtConfigProvider {
  @Override
  public String getConfig(String key) {
      if ("h5_shouldverifyapp".equalsIgnoreCase(key)) {
          return "YES";
      } else if ("TSBS".equalsIgnoreCase(key)) {
          return "0";
      }
      return null;
  }
}
// It is recommended that you call at startup, only one instance of H5ExtConfigProvider
will be valid globally, subject to the instance in actual settings.
H5Utils.setProvider(H5ExtConfigProvider.class.getName(), new H5ExtConfigProviderImpl());
```

- Send the switch configuration through the MDS platform, see Switch configuration management.

Container switches list

You can customize whether or not to use the corresponding function by using the switches in the table below.

Switch name	Usage	Description	Default value
`h5_shouldverifyapp`	Turn the signature verification on or off. It is recommended that you turn on it. When the phone is considered to be the root phone, the signature verification will be forced to open. At this time, the switch configuration does not take effect.	YES means ON, NO means OFF.	YES
`TSBS`	Whether to use the translucent title bar, only for Android.	“1” means use, “0” means don’t use. **Notice**:”1” and “0” should be String type.	1

Switch name	Usage	Description	Default value
`h5_remote_debug_host`	Remote server address for real device debugging.	• If configured, the remote real-device debugging is enabled. The server address for debugging needs to be set in `h5_remote_debug_host`. • If not configured, it will not appear in the code, which means that the remote real-device is not used for debugging, and there is no default value.	-
`androidFallbackNetwork`	Whether to load the fallback resource using the mPaaS network library.	YES means load the fallback resource using the mPaaS network library, NO means load the fallback resource using the system network library.	YES
`mp_h5_push_window_use_activity`	Whether to force a new activity to start when calling pushWindow.	YES means enable, other value means disable.	NO
`mp_ta_showOptionMenu`	Whether to display the option menu in the upper right corner of the MINI program. **Note**: This configuration is only effective when selecting whether to display the upper right menu during the release of the MINI program.	YES means display, other value means don't display.	NO
`mp_ta_showShareMenuItem`	Whether to display the sharing option in the option menu in the upper right corner of the MINI program.	YES means display, other value means don't display.	NO
`mp_ta_use_orginal_mini_nagivationbar`	Whether to use the built-in navigation bar of the MINI program.	YES means use, NO means don't use.	YES

Switch name	Usage	Description	Default value
`h5_CORSWhiteList`	Domain name whitelist. The offline resources under the domain name can be accessed across domains. **Note**: For the resources requested online, a correct cross-domain setting need to be enabled on the resource server.	The content is JSON array, and special characters require escaping. Example: `{ "value": "[\"oss-cn-hangzhou.aliyuncs.com\"]", "key": "h5_CORSWhiteList" }`	Null
`mp_h5_allow_mix_content`	Allow MixContent mode. There may be potential security risk to enable this mode, please operate with caution. **Note**: This switch is only supported in version 10.1.60.	YES means allow, No means don't allow.	NO

1.3.4.11. HTML5 container extension

HTML5 container provides rich extension capabilities, and to make it easier for users to use HTML5 containers in more scenarios, this topic introduces the following HTML5 container extensions with examples.

- H5Plugin obtaining Activity result
- Customize HTML5 error page
- Enable translucent status bar
- Add third-party JavaScriptInterface
- Add cutscene in HTML5 container
- Configure the blacklist for the JSAPI of the H5 Container

H5Plugin obtaining Activity result

In many scenes, such as facial recognition and code recognition, you need to start a new Activity to get the result returned by the Activity. But in such a scenario, JSAPI cannot directly get the result by rewriting H5Activity. Thus, when using the HTML5 container, you need to get the result returned by the Activity in the following way:

1. In the custom H5Plugin, register `OnH5ActivityResult` callback. Code sample is as follows:

```
H5ActivityResultManager.getInstance().put(onH5ActivityResult);
```

> **Note**
>
> - `put` method doesn't check for duplicate registrations, developers need to prevent duplicate registrations themselves.
> - After regestration, you need to call the `remove` method to remove the callback. Normally, it is recommended that you remove the callback in the `onRelease` method of H5Plugin. Code sample is as follows:
>
> ```
> H5ActivityResultManager.getInstance().remove(onH5ActivityResult);
> ```

2. Start the target `Activity` using `startActivityForResult` method, for example, you can start the activity in the custom `H5Plugin` handleEvent method. Code sample is as follows:

```
public boolean handleEvent(H5Event event, H5BridgeContext context) {
 if ("CustomJSAPI".equals(event.getAction())) {
     if (event.getActivity()!=null){
         Intent intent = new Intent(event.getActivity(), yourDestinationActivity.class);
         event.getActivity().startActivityForResult(intent,requestCode,bundle);
     }
     return true;
 }
 return false;
}
```

> **Note**
>
> This method is only used to call back the result of H5Activity.

3. In the callback method of `OnH5ActivityResult`, pass the result to the front end via `H5BridgeContext` object.

```
public interface OnH5ActivityResult {
     void onGetResult(int requestCode, int resultCode, Intent intent);
}
```

Customize HTML5 error page

When you need to customize the HTML5 error page, the steps are as follows:

1. Create a custom error page in HTML format.

```
<!doctype html>
<html lang="zh-cn">

<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,maximum-scale=1.0,minimum-
scale=1.0,user-scalable=no" />
  <meta name="format-detection" content="telephone=no" />
  <title>Custom error</title>
</head>

<body>
    <p>This is a custom error page</p>
</body>

</html>
```

2. Implement `H5ErrorPageView`. Set the error page you just created in `APWebView`.

```
public class H5ErrorPageViewImpl implements H5ErrorPageView {
    @Override
    public boolean enableShowErrorPage() {
        // True indicates launching the custom error page.
        return true;
    }
    @Override
    public void errorPageCallback(H5Page h5Page, APWebView view, String errorUrl, int s
tatusCode, String errorMsg, String subErrorMsg) {
        // Obtain the html of the error page. In this demo, it is put into raw, but you
can also put it in another place.
        String html = H5ResourceManager.readRawFromResource(R.raw.custom_error,
LauncherApplicationAgent.getInstance().getApplicationContext().getResources());
        // Set the error page in webview
        view.loadDataWithBaseURL(errorUrl, html, "text/html", "utf-8", errorUrl);
    }
}
```

3. Register `H5ErrorPageView`. Before opening the HTML5 container, register the custom `H5ErrorPageView` in the container.

```
H5Utils.setProvider(H5ErrorPageView.class.getName(),new H5ErrorPageViewImpl());
```

> **Note**
>
> mPaaS baseline 10.1.68.7 and above supports the new `MPH5ErrorPageView` method, which is consistent with `H5ErrorPageView` in the aspects of method name and usage, but the parameters are extended.

```
/**
 * Interface for customizing network error page
 */
public interface MPH5ErrorPageView {
    /**
     * @param h5Page      page object
     * @param view        webview object
     * @param errorUrl    error URL
     * @param statusCode  error code
     * @param errorMsg    error description
     * @param subErrorMsg sub error description
     * @param extInfo     extended information,you need to check if it is empty
     * @param extObj      extended class,you need to check if it is empty
     * @return true indicates custom page is required,and errorPageCallback method below w
ill be used
     */
    boolean enableShowErrorPage(H5Page h5Page, APWebView view, String errorUrl, int status
Code, String errorMsg, String subErrorMsg, Bundle extInfo, Object extObj);
    /**
     * @param h5Page      page object
     * @param view        webview object
     * @param errorUrl    error URL
     * @param statusCode  error code
     * @param errorMsg    error description
     * @param subErrorMsg sub error description
     * @param extInfo     extended information,you need to check if it is empty
     * @param extObj      extended class,you need to check if it is empty
     */
    void errorPageCallback(H5Page h5Page, APWebView view, String errorUrl, int statusCode,
String errorMsg, String subErrorMsg, Bundle extInfo, Object extObj);
}
```

Enable translucent status bar

> **Note**
>
> - This function is only supported in mPaaS baseline 10.1.60 and above.
> - This method sets the status bar color of all HTML5 pages opened by the HTML5 container. If you have more requirements on status bar color, you can implement the HTML5 container Custom title bar.
> - You can choose to set the status bar color in `openTranslucentStatusBarSupport` method of container title bar interface, or handle it in other places.

1. Enable `TSBS` in HTML5 container configuration.
2. If you use built-in title bar, developers can implement `H5TransStatusBarColorProvider` interface, and set the instance in HTML5 container via `H5Utils.setProvider` method. Code sample is as follows:

```
package com.mpaas.demo.nebula;

import android.graphics.Color;

import com.alipay.mobile.nebula.provider.H5TransStatusBarColorProvider;

public class H5TransStatusBarColorProviderImpl implements H5TransStatusBarColorProvider
{
    @Override
    public int getColor() {
        return Color.argb(70, 255, 255, 255);
    }
}
```

Add third-party JavaScriptInterface

Many third-party pages require `JavaScriptInterface`. You can implement it in the following steps:

1. Implement plug-in to intercept the loading event of the third-party page.
2. Obtain WebView and inject JavaScript object.

Code sample:

```
package com.mpaas.demo.nebula;

import android.text.TextUtils;

import com.alibaba.fastjson.JSONObject;
import com.alipay.mobile.h5container.api.H5BridgeContext;
import com.alipay.mobile.h5container.api.H5Event;
import com.alipay.mobile.h5container.api.H5EventFilter;
import com.alipay.mobile.h5container.api.H5Param;
import com.alipay.mobile.h5container.api.H5SimplePlugin;

public class TechFinSitePlugin extends H5SimplePlugin {

    @Override
    public void onPrepare(H5EventFilter filter) {
        super.onPrepare(filter);
        filter.addAction(CommonEvents.H5_PAGE_SHOULD_LOAD_URL);
    }

    @Override
    public boolean interceptEvent(H5Event event, H5BridgeContext context) {
        String action = event.getAction();
        if (CommonEvents.H5_PAGE_SHOULD_LOAD_URL.equals(action)) {
            JSONObject params = event.getParam();
            String url = params.getString(H5Param.LONG_URL);
            if (!TextUtils.isEmpty(url) && url.contains("tech.antfin.com")) {
                event.getH5page().getWebView().addJavascriptInterface(new TechFinJavaScript
Interface(), "techFinBridge");
            }
        }

        return false;
    }
}
```

Note

Do not return `true` in the `interceptEvent` method, otherwise the loading page of the container may be compromised.

```
package com.mpaas.demo.nebula;

import android.webkit.JavascriptInterface;

public class TechFinJavaScriptInterface {

    @JavascriptInterface
    @com.uc.webview.export.JavascriptInterface
    public String whoAmI() {
        return "It is tech fin.";
    }
}
```

> **Note**
>
> The annotation classes used in System core and UC core are different, you must make it compatible with these two annotation classes.

Add cutscene in HTML5 container

To add a cutscene to an HTML5 container, you only need to add the animation resources to the project's `res/anim` folder. The steps are as follows:

1. Create an **anim** folder under the project's **res** folder. Skip this step if it already exists.
2. Add the resource files of the animation to the **anim** folder. The HTML5 container automatically recognizes resource files based on their file names, so the file names for resource files can only be h5_slide_out_right.xml, h5_slide_out_left.xml, h5_slide_in_right.xml, or h5_slide_in_left.xml. You can refer to the following example to create your own resource file.
   - h5_slide_out_right.xml

     ```
     <?xml version="1.0" encoding="utf-8"?>
     <set xmlns:android="http://schemas.android.com/apk/res/android">
     <translate
       android:fromXDelta="0%"
       android:toXDelta="100%"
       android:duration="300" />
     </set>
     ```

   - h5_slide_out_left.xml

     ```
     <?xml version="1.0" encoding="utf-8"?>
     <set xmlns:android="http://schemas.android.com/apk/res/android">
     <translate
       android:fromXDelta="0%"
       android:toXDelta="-100%"
       android:duration="300" />
     </set>
     ```

   - h5_slide_in_right.xml

     ```
     <?xml version="1.0" encoding="utf-8"?>
     <set xmlns:android="http://schemas.android.com/apk/res/android">
     <translate
       android:fromXDelta="100%"
       android:toXDelta="0"
       android:duration="300" />
     </set>
     ```

   - h5_slide_in_left.xml

     ```
     <?xml version="1.0" encoding="utf-8"?>
     <set xmlns:android="http://schemas.android.com/apk/res/android">
     <translate
       android:fromXDelta="-100%"
       android:toXDelta="0%"
       android:duration="300" />
     </set>
     ```

Configure the blacklist for the JSAPI of the H5 Container

Configure a blacklist for JSAPI to control the specified domain name's calling authority to JSAPI. The steps are as follows:

1. Inherit the `H5JSApiPermissionProvider` class and override the `hasDomainPermission` method. The two input parameters of the `hasDomainPermission` method are `action` and `url`. `action` represents the event name of the custom JSAPI, and `url` represents the sign of the current page. The return value is of type `boolean`. `true` means that the event can be processed, and `false` means that the event is not authorized to be processed. The demo is as follows, for reference only.

```
public class H5JSApiPermissionProviderImpl implements H5JSApiPermissionProvider {
     private static final List blackList = new ArrayList<String>();
     static {
     // URLs in the blacklist don't have authority to execute JSAPI or other related eve
nts.
     blackList.add("https://mcube-prod.cn-hangzhou.oss.aliyuncs.com/ONEX4B905F1032156-MU
AT/20210728/0.0.0.1_all/nebula/fallback/www/index.html");
     }

     @override
     public boolean hasDomainPermission(String action, String url) {
         // The accessor can judge whether he has the authority to execute the current a
ction according to the action name and url.
         // The action is the defined JSAPI event, return true means that it can handle
the event, return false means that it is not authorized to handle the event.
         if (blackList.contains(url)) {
             return false;
         }
         return true;
     }
     @override
     public boolean hasThisPermission(String permission, String url) {
         return true;
     }
}
```

2. Set the Provider after the initialization of the framework is completed.

```
H5Utils.setProvider(H5JSApiPermissionProvider.class.getName(), new
H5JSApiPermissionProviderImpl());
```

1.3.4.12. Interpret physical button by using H5 Container

To intercept physical button by using H5 Container, you need to upgrade the mPaaS baseline version to 10.1.68.33 or later. Set the Provider intercepted by the physical back button with the following codes.

```
public interface MPH5OnKeyDownProvider {
    boolean needIntercept(H5Page page, int keyCode, KeyEvent intent);
    boolean onKeyDown(H5Page page, int keyCode, KeyEvent intent);
}
```

If the interception operation is to be excuted, set the return value of `needIntercept` to `true`, automatically execute the `onKeyDown` method, and no longer execute the mPaaS physical back button logic. If `needIntercept` returns `false`, it means that the physical back button event is returned to the original mPaaS logic for processing, and the `onKeyDown` method is no longer executed.

> **Note**
>
> When executing the `needIntercept` method and the `onKeyDown` method, it is necessary to check whether the parameter value is null.

1.3.4.13. Implement resource interception of H5 Container

There are too many images displayed on the page of an App, which causing the slow loading speed. Thus, it is necessary to optimize the loading speed of the H5 Container, and strive to achieve the H5 page opening in seconds. The H5 Container can greatly improve the loading speed of H5 page by intercepting and replacing the resource files(replace the files that need to be loaded online with local cache files). The demo is as follows, for reference only.

1. Inherit the `H5ResProvider` class provided by the H5 Container, and override the `contains` method and `getResource` method.

```
//return true means intercepting (using local resources), while fasle means not intercep
ting (loading resources online)
 @Override
 public boolean contains(String sourceUrl) {
     if (isCache(sourceUrl)) {
         if (ResourceCache.contains(sourceUrl)) {
             LoggerFactory.getTraceLogger().debug(TAG, "contains: " + sourceUrl);
             //Do not intercept.
             return true;
         } else {
             ResourceCache.download(sourceUrl);
             return false;
         }
     }
     return false;
 }
```

```
@Override
 public InputStream getResource(String sourceUrl) {
     //Get resources from the local cache.
     if (isCache(sourceUrl)) {
         if (ResourceCache.contains(sourceUrl)) {
             try {
                 InputStream inputStream = ResourceCache.getResource(sourceUrl);
                 if (null == inputStream) {
                     LoggerFactory.getTraceLogger().debug(TAG, "File null: " + sourceUrl)
;
                     return new URL(sourceUrl).openStream();
                 }
                 LoggerFactory.getTraceLogger().debug(TAG, "getResource: " + sourceUrl);
                 return inputStream;
             } catch (Exception e) {
             }
         }
     } else {
         //Get resources from network links.
         try {
             return new URL(sourceUrl).openStream();
         } catch (IOException e) {
             e.printStackTrace();
         }
     }
     return null;
 }
```

2. Register `H5ResProvider`.

```
public static void register() {
     H5Utils.setProvider(H5ResProvider.class.getName(), new GapResProvider());
}
```

By using customizing `H5ResProvider`, users can decide whether to intercept loaded resources and select the resource acquisition methods (using local resources, loading resources online), and users can customize their own business scenarios.

1.4. Integrate iOS SDK

1.4.1. Quick start

You can use the HTML5 container and offline package to implement the following functions: container initialization and invocation, bi-directional communication between HTML5 and Native, loading and use of offline packages, integration of the automatic tracking ability of the Nebula container, and check of tracking data.

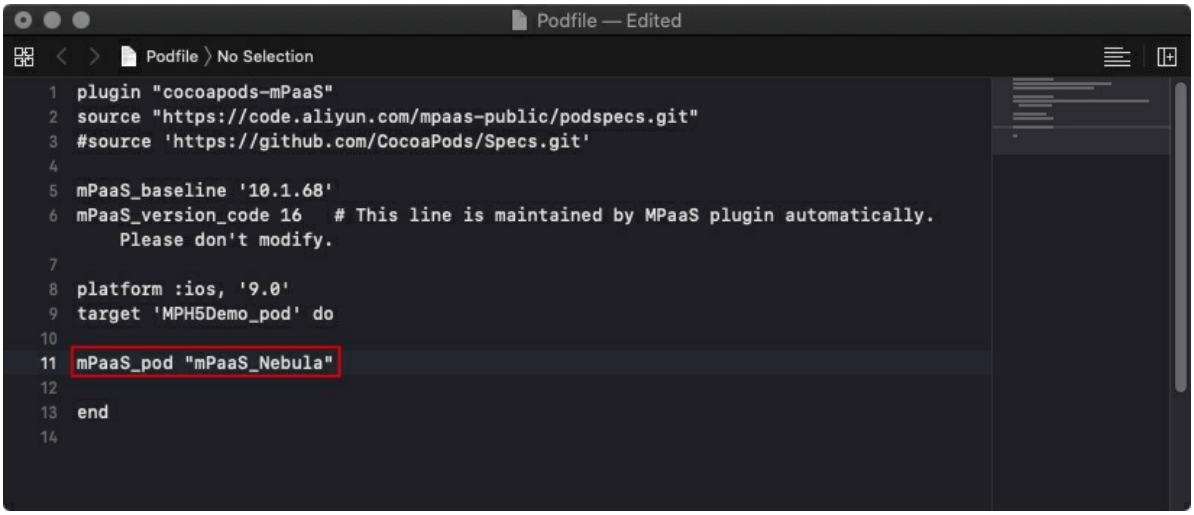
Before you begin

The project already gets access to mPaaS. For more information, see the following content: Access based on an existing project and use of CocoaPods

Add the SDK

Use the cocoapods-mPaaS plug-in. This method is applicable to the **access mode based on an existing project and use of CocoaPods**.

1. In the Podfile file, use `mPaaS_pod "mPaaS_Nebula"` to add the dependencies of the HTML5 container component.



2. Run `pod install` to complete the access of HTML5 container.

Use the SDK

This topic describes how to use the HTML5 container SDK in baseline version 10.1.60 or later versions with the official demo of the HTML5 container and offline package.

Initialize the container

Start the container

- To use the Nebula container, you need to call the SDK API to initialize the container after the program is started. Initiate the container in `- (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions` of `DTFrameworkInterface`. Override this method in the category `DTFrameworkInterface + (project project name)` provided by the mPaaS framework.

```
- (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
  {
      // Initialize the container.
      [MPNebulaAdapterInterface initNebula];
  }
```

- If you need to use the **preset offline package**, **customize JSAPI**, and **plug-in** functions, replace the `initNebula` API in the previous code with the `initNebulaWith` API in the following code. Then pass in the corresponding parameters to initialize the container.
  - `presetApplistPath`: custom path of the preset offline package information.
  - `appPackagePath`: custom path of the preset offline package.

- `pluginsJsapisPath`: storage path of custom JSAPI and custom plug-in files.

```
 - (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:
(NSDictionary *)launchOptions
{
// Initialize the container.
NSString *presetApplistPath = [[NSBundle mainBundle] pathForResource:[NSString stringW
ithFormat:@"MPCustomPresetApps.bundle/h5_json.json"] ofType:nil];
NSString *appPackagePath = [[NSBundle mainBundle] pathForResource:[NSString stringWith
Format:@"MPCustomPresetApps.bundle"] ofType:nil];
NSString *pluginsJsapisPath = [[NSBundle mainBundle] pathForResource:[NSString stringW
ithFormat:@"Poseidon-UserDefine-Extra-Config.plist"] ofType:nil];
[MPNebulaAdapterInterface initNebulaWithCustomPresetApplistPath:presetApplistPath cust
omPresetAppPackagePath:appPackagePath customPluginsJsapisPath:pluginsJsapisPath];
}
```

Note: `initNebula` and `initNebulaWithCustomPresetApplistPath` are two methods for container initialization. Do not call both of them at the same time.

- Configure the time interval of requests for Mini program packages. mPaaS supports both global and individual configuration of the time interval.
  - **Global configuration**: You can use the following code to set the update request interval of offline packages or Mini programs during container initialization.

    ```
    [MPNebulaAdapterInterface shareInstance].nebulaUpdateReqRate = 7200;
    ```

    Where, `7200` indicates the global update request interval. `7200` is the default value. The unit is second. You can change this value to set the global interval of requests for offline packages. The value range is 0 to 86400 seconds (that is, 0 to 24 hours). 0 indicates that there is no limit on request interval.
  - **Individual configuration**: The time interval of requests is only set for the current Mini program package. You can go to **Add Offline Package** > **Extended Information** in the console and then enter `{"asyncReqRate":"1800"}` to set the time interval of requests. For details, see **Extended information** in Create HTML5 offline package.

Customize the container

- You can set the attributes in `MPNebulaAdapterInterface` to customize the container configuration when necessary. The configuration must be implemented in `-(void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions` of `DTFrameworkInterface`. Otherwise, the configuration will be overridden by the default configuration of the container.

```
 - (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NS
Dictionary *)launchOptions
  {
      // Customize the container.
      [MPNebulaAdapterInterface shareInstance].nebulaVeiwControllerClass =
[MPH5WebViewController class];
      [MPNebulaAdapterInterface shareInstance].nebulaNeedVerify = NO;
      [MPNebulaAdapterInterface shareInstance].nebulaUserAgent = @"mPaaS/Portal";
      [MPNebulaAdapterInterface shareInstance].nebulaCommonResourceAppList =
@[@"77777777"];
  }
```

- The following table describes the attributes:

Attribute	Description	Remark
nebulaVeiwControllerClass	Base class of HTML5 page	The default is H5WebViewController. You can set this API directly if it is required to specify base classes for all HTML5 pages. Caution: All base classes must be derived from H5WebViewController by inheritance.
nebulaWebViewClass	Sets base class of WebView	> 10.1.60: The default is H5WKWebView. A custom WebView must be derived from H5WKWebView by inheritance. = 10.1.60: customization not supported.
nebulaUseWKArbitrary	Sets whether to use WKWebView to load the offline package page	> 10.1.60: The default is YES. = 10.1.60: The default is NO.
nebulaUserAgent	Sets UserAgent of the application	The UserAgent is attached to the default UA of the container as a suffix.
nebulaNeedVerify	Sets whether signature verification is required. The default is YES.	Set the attribute value to NO if no private key file is uploaded when you Configure offline package. Otherwise, the offline package loading will fail.
nebulaPublicKeyPath	Path of the public key that is used for signature verification of offline packages	It is the path of the public key corresponding to the private key uploaded when you configure offline package.
nebulaCommonResourceAppList	The app ID list of the common resource package	-
errorHtmlPath	Path of the HTML error page that is displayed when HTML5 page loading fails	Read from `MPNebulaAdapter.bundle/error.html` by default.
configDelegate	Sets the custom switch delegate.	Supports global modification of the default container switch.

Update an offline package

Make a request for full information of offline packages after startup and check whether update packages are available on the server. To prevent impact on the startup speed, we recommend that you request calling after `(void)application:(UIApplication \*)application afterDidFinishLaunchingWithOptions:(NSDictionary \*)launchOptions`.

```
- (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    // Customize the container.
    [MPNebulaAdapterInterface shareInstance].nebulaVeiwControllerClass = [MPH5WebViewController class];
    [MPNebulaAdapterInterface shareInstance].nebulaNeedVerify = NO;
    [MPNebulaAdapterInterface shareInstance].nebulaUserAgent = @"mPaaS/Portal";
    [MPNebulaAdapterInterface shareInstance].nebulaCommonResourceAppList = @[@"77777777"];

    // Full update of offline packages
    [[MPNebulaAdapterInterface shareInstance] requestAllNebulaApps:^(NSDictionary *data, NSError *error) {
        NSLog(@"");
    }];
}
```

The following figure shows the effect after the initialization is complete:

```
MPH5Demo_pod 〉 ... 〉 ... 〉 ... 〉 ... 〉 DTFrameworkInterface+MPH5Demo_pod.m 〉 -application:afterDidFinishLaunchingWithOptions:

- (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    // 初始化 rpc
    [MPRpcInterface initRpc];

    // 初始化容器
    //    [MPNebulaAdapterInterface initNebula];

    // 自定义jsapi路径和预置离线包信息
    NSString *presetApplistPath = [[NSBundle mainBundle] pathForResource:[NSString
        stringWithFormat:@"DemoCustomPresetApps.bundle/NebulaApplist.plist"] ofType:nil];
    NSString *appPackagePath = [[NSBundle mainBundle] pathForResource:[NSString stringWithFormat:@"DemoCustomPresetApps.bundle"]
        ofType:nil];
    NSString *pluginsJsapisPath = [[NSBundle mainBundle] pathForResource:[NSString
        stringWithFormat:@"DemoCustomPlugins.bundle/Poseidon-UserDefine-Extra-Config.plist"] ofType:nil];
    [MPNebulaAdapterInterface initNebulaWithCustomPresetApplistPath:presetApplistPath customPresetAppPackagePath:appPackagePath
        customPluginsJsapisPath:pluginsJsapisPath];
}

- (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    // 定制容器
    [MPNebulaAdapterInterface shareInstance].nebulaVeiwControllerClass = [MPH5WebViewController class]; //设置H5容器基类
    [MPNebulaAdapterInterface shareInstance].nebulaUserAgent = @"mPaaS/Portal";//设置H5容器UserAgent
    [MPNebulaAdapterInterface shareInstance].nebulaUseWKArbitrary = YES; //开启 WKWebview
    [MPNebulaAdapterInterface shareInstance].nebulaCommonResourceAppList = @[@"77777777"];// 设置全局资源包
    [MPNebulaAdapterInterface shareInstance].nebulaNeedVerify = NO; // 关闭离线包验签，正式版本请开启验签

    // 更新离线包
    [[MPNebulaAdapterInterface shareInstance] requestAllNebulaApps:^(NSDictionary *data, NSError *error) {
        NSLog(@"[mpaas] nebula rpc data :%@", data);
    }];
}

@end
```

Configuration not managed by the framework

If the lifecycle of your app is managed by a custom delegate instead of the mPaaS framework, additional configuration is required, as shown in the following figure. If your app is managed by the mPaaS framework, skip this step.

```
MPH5Demo_pod 〉 MPH5Demo_pod 〉 main.m 〉 No Selection
//
//  main.m
//  MPH5Demo_pod
//
//  Created by yangwei on 2019/3/26.
//  Copyright © 2019 yangwei. All rights reserved.
//

#import <UIKit/UIKit.h>
#import "AppDelegate.h"

int main(int argc, char * argv[]) {
    [MPAnalysisHelper enableCrashReporterService]; // USE MPAAS CRASH REPORTER
    @autoreleasepool {
        return UIApplicationMain(argc, argv, nil, NSStringFromClass([AppDelegate class]));
//        return UIApplicationMain(argc, argv, @"DFApplication", @"DFClientDelegate"); // NOW USE MPAAS FRAMEWORK
    }
}
```

Start the mPaaS framework

In the `didFinishLaunchingWithOptions` method of the current application, call `[[DTFrameworkInterface sharedInstance] manualInitMpaasFrameworkWithApplication:application launchOptions:launchOptions];` to start the mPaaS framework.

```
MPH5Demo_pod 〉 M...S 〉 T...ts 〉 M...d 〉 A...rk 〉 AppDelegate.m 〉 -application:didFinishLaunchingWithOptions:
@end

@implementation AppDelegate


+ (AppDelegate *)sharedInstance
{
    return [UIApplication sharedApplication].delegate;   Returning 'id<UIApplicationDelegate> _Nullable' from a function with incom...
}

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {

    self.window = [[UIWindow alloc]initWithFrame:[UIScreen mainScreen].bounds];
    self.window.rootViewController = [[DFNavigationController alloc]
        initWithRootViewController:[[TestDemoViewController alloc]init]];
    self.navigationController = self.window.rootViewController;   Incompatible pointer types assigning to 'DFNavigationController.
    [self.window makeKeyAndVisible];
    self.window.backgroundColor = [UIColor whiteColor];

    // navigationcontroller创建完成后，启动 mPaaS 框架
    [[DTFrameworkInterface sharedInstance] manualInitMpaasFrameworkWithApplication:application
        launchOptions:launchOptions];

    return YES;
}
```

Note: To ensure successful startup of the framework, implement the call after the initialization of `window` and `navigation` of the current application is complete.

Create a bootloader

Create a subclass derived from `DTBootLoader`, rewrite the `createWindow` and `createNavigationController` methods, and return the `window` and `navigationControlle` of the current application.

- Set `window`: keyWindow of the current application.
- Set `navigationController`: `rootviewcontroller` of keyWindow of the current application, which must be derived from `DFNavigationController`.

MPH5Demo_pod 〉 MPaaS 〉 Targets 〉 MPH5Demo_pod 〉 APMobil...mework 〉 MPBootLoaderImpl.h 〉 No Selection

```
//
//  MPBootLoaderImpl.h
//  Portal
//
//  Created by yemingyu on 2019/6/24.
//  Copyright © 2019 Alibaba. All rights reserved.
//

#import <APMobileFramework/APMobileFramework.h>

NS_ASSUME_NONNULL_BEGIN

@interface MPBootLoaderImpl : DTBootLoader

@end

NS_ASSUME_NONNULL_END
```

MPH5Demo_pod 〉 MPaaS 〉 Targets 〉 MPH5Demo_pod 〉 APMobileFramework 〉 MPBootLoaderImpl.m 〉 No...ction

```
//
//  MPBootLoaderImpl.m
//  Portal
//
//  Created by yemingyu on 2019/6/24.
//  Copyright © 2019 Alibaba. All rights reserved.
//

#import "MPBootLoaderImpl.h"
#import "AppDelegate.h"

@implementation MPBootLoaderImpl

- (UINavigationController *)createNavigationController
{
    return [AppDelegate sharedInstance].navigationController;
}

- (UIWindow *)createWindow
{
    return [AppDelegate sharedInstance].window;
}

@end
```

Specify a bootloader

Rewrite the method in `category` of `DTFrameworkInterface`, specify the `bootloader` of the current application, and hide the default `window` and `launcher` applications of the mPaaS framework.

MPH5Demo_pod 〉 MPaaS 〉 Ta...ets 〉 M...od 〉 AP...rk 〉 DTFrameworkInterface+MPH5Demo_pod.m 〉 No Selection

```
    }];
}

#pragma mark 非框架托管配置
- (DTBootLoader *)bootLoader {
    static MPBootLoaderImpl *_bootLoader;
    static dispatch_once_t onceToken;
    dispatch_once(&onceToken, ^{
        _bootLoader = [[MPBootLoaderImpl alloc] init];
    });
    return _bootLoader;
}

- (BOOL)shouldWindowMakeVisable {
    return NO;
}

- (BOOL)shouldShowLauncher {
    return NO;
}

@end
```

Invoke a container

After container initialization, you can evoke an HTML5 container. The following describes three invoking methods:

- Create an HTML5 container based on an online URL or local HTML file. Refer to the following code example:

```
// Open an online URL.
[[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"ht
tps://tech.antfin.com/products/MPAAS"}];
// Open a local HTML page.
NSString *path = [[NSBundle mainBundle].bundlePath stringByAppendingFormat:@"/%@/%@", @"
MPH5Demo.bundle", @"H52Native.html"];
if ([path length] > 0) {
[[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": path
}];
}
```

- Create an HTML5 container based on the received offline package information and use automatic push to open the container. Refer to the following code example:

```
 [[MPNebulaAdapterInterface shareInstance]
startH5ViewControllerWithNebulaApp:@{@"appId":@"90000000"}];
```

- Create an HTML5 container based on the received offline package information and return the created HTML5 container instance. This is usually used on a tab of homepage. Refer to the following code example:

```
 [[MPNebulaAdapterInterface shareInstance]
createH5ViewControllerWithNebulaApp:@{@"appId":@"90000000"}];
```

Implement bidirectional communication between HTML5 and native apps.

You can call JSAPIs and listen to certain events to implement bidirectional communication between HTML5 and native apps.

Call the native function on an HTML5 page

You can call JSAPIs to enable HTML5 to communicate with Native.

For details about JSAPI supported by the Nebula container and related parameters, see Built-in JSAPI.

Example

Call a JSAPI `pushWindow` to load a new page when a button is tapped on an HTML5 page.

```
AlipayJSBridge.call('pushWindow', {
    url: 'https://tech.antfin.com',
    param: {
    readTitle: true,
    defaultTitle: true,
    // ...
    }
}, function(data) {alert('Call result'+JSON.stringify(data)); });
```

AlipayJSBridge description

`AlipayJSBridge` AlipayJSBridge is a JSBridge automatically injected by the Nebula container. After the implementation of `Window.onload` , the container will generate a global variable `AlipayJSBridge` and trigger an `AlipayJSBridgeReady` event. `AlipayJSBridge` injection is an asynchronous process. Therefore, listen to the `AlipayJSBridgeReady` event before calling an API.

Refer to the following code example:

```
<h1>bridge API usage instruction</h1>

<script>
function ready(callback) {
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}

ready(function(){
  alert('bridge ready');
});
</script>
```

Call HTML5 on a native page

You can listen to certain events to enable communication between the native and HTML5. For information about events supported by the Nebula container, see Extended event list.

```
document.addEventListener('back', function (e) {
    if(confirm('back intercepted. Are you sure you want to return?')) {
        // do something;
    }
}, false);
```

In addition to default events supported by the Nebula container, you can use the following method on the Native end to define an event that the front end can listen.

```
// self: VC where the current HTML5 page is located
// data: parameter passed to the front end by native
// callBack: callback after the front end receives the event
[self callHandler:@"customEvent" data:@{@"key":@"value"} responseCallback:^(id responseDat
a) {
        NSLog(@"callback after the front end receives the event: %@", responseData);
    }];
```

Expand Nebula container capabilities

If the basic bidirectional communication between HTML5 pages provided by the Nebula container is insufficient, you can expand the capabilities of Nebula.

- **JSAPI**: To call the native function on a page, for example, to display an ActionSheet or contacts dialog box, you need to expand a JSAPI. JSAPI allows you to use the handler method to add a native feature for an HTML5 page to implement a specific function. For detailed instructions, see Customize a JSAPI.
- **Plugin**: To finish certain events (such as recording tracking and modifying returned data) at a certain time point (such as when entering a page or receiving a request), you need to develop a plug-in. After subscribing to corresponding events, the plug-in can process data carried in the events in the handler. For detailed instructions, see Customize a plug-in.

Load an offline package

The traditional online HTML5 technology is subject to actual network environment, which may compromise the performance of HTML5 pages. To minimize network impact on HTML5 page loading, you can encapsulate different services as an offline package and deliver it to the client through the release platform to update client resources. For more information, see Introduction to offline packages and Use offline packages.

HTML5 container tracking

When an HTML5 page is loaded, the Nebula container will automatically monitor the loading performance and capture related behavior data and exception data. For more information, see HTML5 container tracking.

1.4.2. Manage offline packages

The traditional online HTML5 technology is susceptible to the impact of the network environment, thereby compromising the performance of HTML5 pages. You can encapsulate different services as an offline package and deliver it through the release platform to update client resources.

This topic describes how to manage offline packages.

- Generate an offline package
- Load an offline package
- Use global resource package
- Update an offline package dynamically

Prerequisite

The client project has integrated `NebulamPaaSBiz.framework` after the SDK is added.

Generate an offline package

To generate an `.amr` offline package, you need to build a frontend `.zip` package and generate an `.amr` package online. For details, see Generate an offline package.

Load an offline package

There are two methods:

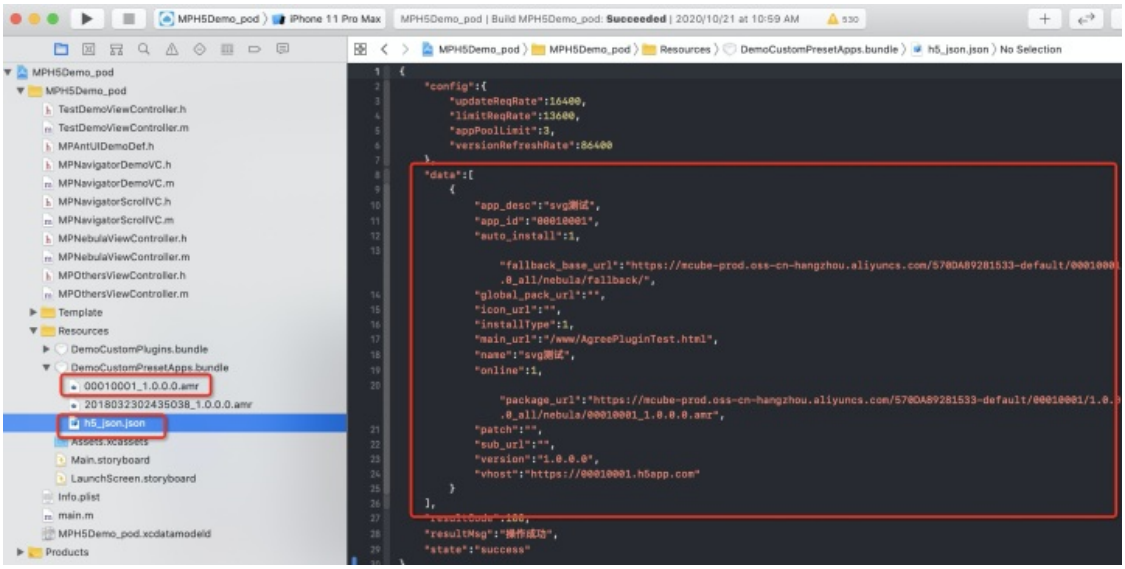
- Preset an offline package
- Load a remote offline package

Preset an offline package

The homepage and login page must be quickly loaded regardless of the network conditions. This type of resources can be encapsulated as an offline package and preset in the project so that resources can be quickly loaded offline.

Perform the following steps:

1. Create an independent bundle, for example, `DemoCustomPresetApps.bundle`. Add the offline package and `h5_json.json` file downloaded from the release platform to the bundle.



Note: Currently, the release platform allows you to download the `h5_json.json` configuration file of a single offline package. If multiple offline packages are to be preset, merge the `data` arrays in the JSON files.

2. When initializing the container, call the `initNebulaWithCustomPresetApplistPath` interface and set the preset offline package path as the bundle created in the Step 1.

```
 - (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(N
SDictionary *)launchOptions
 {
     // Initialize rpc
     [MPRpcInterface initRpc];

     // Initialize container
     //    [MPNebulaAdapterInterface initNebula];

     // Customize JSAPI path and preset offline package information
     NSString *presetApplistPath = [[NSBundle mainBundle] pathForResource:[NSString stri
ngWithFormat:@"DemoCustomPresetApps.bundle/h5_json.json"] ofType:nil];
     NSString *appPackagePath = [[NSBundle mainBundle] pathForResource:[NSString stringW
ithFormat:@"DemoCustomPresetApps.bundle"] ofType:nil];
     NSString *pluginsJsapisPath = [[NSBundle mainBundle] pathForResource:[NSString stri
ngWithFormat:@"DemoCustomPlugins.bundle/Poseidon-UserDefine-Extra-Config.plist"] ofType:
nil];
     [MPNebulaAdapterInterface initNebulaWithCustomPresetApplistPath:presetApplistPath c
ustomPresetAppPackagePath:appPackagePath customPluginsJsapisPath:pluginsJsapisPath];
 }
```

3. Similar to loading a non-preset offline package, when you go to the corresponding page, call the interface method provided by the Nebula container to load the offline package.

```
 - (void)openPresetPackage {
     [[MPNebulaAdapterInterface shareInstance]
startH5ViewControllerWithNebulaApp:@{@"appId":@"20180910"}];
 }
```

Load a remote offline package

In addition to preset an offline package on the client, you can dynamically release an offline package on the release platform. Then the client will directly load the remote offline package, preventing the package size on the client from becoming excessively large due to a large number of preset offline packages.

Perform the following steps:

1. After an app is started, you can preload package information and download the offline package to avoid a blank screen being displayed when the offline package is opened.
   - Code sample

```
  [[MPNebulaAdapterInterface shareInstance] requestAllNebulaApps:^(NSDictionary *data,
 NSError *error) {
    NSLog(@"[mpaas] nebula rpc data :%@", data);
 }];
```

   - Interface method

```
 @interface MPNebulaAdapterInterface : NSObject

/**
* Fully update local offline package information.
*
*  @param finish: Callback is finished.
*
*/
- (void)requestAllNebulaApps:(NAMRequestFinish)finish;

/**
* Single app request
*
*  @param params: request list, in the format of {appid:version}. Multiple app IDs can
be transferred. The version number consists of a maximum of four digits, for example,
1.0.0.1. If version is not set, the latest version takes effect by default. Fuzzy matc
h with version numbers is supported, for example, '*' matches the latest version, and
'1.*' matches the latest version number beginning with 1.
*  @param finish: Callback is finished.
*/
- (void)requestNebulaAppsWithParams:(NSDictionary *)params finish:
(NAMRequestFinish)finish;

@end
```

2. After client configuration is complete, you can download an offline package from the release platform. For details, see Delivery service > Manage offline packages > Release an offline package.
3. When you access a page, the interface method provided by the Nebula container is called to load the offline package, and you can see the offline package delivered on the release platform.
   - Code sample

```
 - (void)openPresetPackage {
[[MPNebulaAdapterInterface shareInstance]
startH5ViewControllerWithNebulaApp:@{@"appId":@"20180910"}];
}
```

   - Interface method

```
 @interface MPNebulaAdapterInterface : NSObject

/**
* Create an HTML5 container based on transferred offline package information and open
it through automatic push.
*
*  @param params: startup parameters of the HTML5 container. appId is required. For op
tional parameters, see the reference document available at
https://tech.antfin.com/docs/2/85001.
*
*/
- (void)startH5ViewControllerWithNebulaApp:(NSDictionary *)params;

@end
```

Use global resource package

Nebula global resource package can be used to solve redundancy problems when the same resource is used by multiple HTML5 apps, for example, ReactJS framework code of React apps. You can reduce the size of HTML5 apps by putting public resources into a global resource package. You can configure the global offline package in the `afterDidFinishLaunchingWithOptions` method, as shown in the following code sample, where `7777777` is the appId of the global resource package.

`nebulaCommonResourceAppList` is used to inform the HTML5 container that the offline package with the specified ID will be used as a global resource package. Without this ID, the offline package will not take effect even if it's built in the HTML5 app.

```
```
- (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    [MPNebulaAdapterInterface shareInstance].nebulaCommonResourceAppList = @[@"77777777"];//Set global resource package
}
```
```

In order to increase page loading speed, it is recommended to preset the global resource package, which can still be updated through the MDS platform .

Update an offline package dynamically

mPaaS provides powerful dynamic update functions. You can deliver an offline package of a later version on the release platform to update the corresponding page on the client. For details, see Delivery service > Manage offline packages > Release an offline package.

Related links

- Offline package overview
- Code sample

1.4.3. 10.1.60 upgrade guide

Initialize container

- Time of initialization: Before loading the framework, call it in `- (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions` of `DTFrameworkInterface` .

```
MPH...plugin 〉 DTFrameworkInterface+MPH5Demo_plugin.m 〉 -application:beforeDidFinishLaunchingWithOptions:
37  - (BOOL)shouldAutoactivateShareKit
38  {
39      return YES;
40  }
41
42  - (DTNavigationBarBackTextStyle)navigationBarBackTextStyle
43  {
44      return DTNavigationBarBackTextStyleAlipay;
45  }
46
47  - (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
48  {
49
50
51      [MPNebulaAdapterInterface initNebula];
52
53  //    NSString *presetApplistPath = [[NSBundle mainBundle] pathForResource:[NSString
        stringWithFormat:@"DemoCustomPresetApps.bundle/NebulaApplist.plist"] ofType:nil];
54  //    NSString *appPackagePath = [[NSBundle mainBundle] pathForResource:[NSString
        stringWithFormat:@"DemoCustomPresetApps.bundle"] ofType:nil];
55  //    NSString *pluginsJsapisPath = [[NSBundle mainBundle] pathForResource:[NSString
        stringWithFormat:@"DemoCustomPlugins.bundle/Poseidon-UserDefine-Extra-Config.plist"] ofType:nil];
56  //    [MPNebulaAdapterInterface initNebulaWithCustomPresetApplistPath:presetApplistPath
        customPresetAppPackagePath:appPackagePath customPluginsJsapisPath:pluginsJsapisPath];
57  }
58
```

- If the baseline of the existing project is 10.1.32:
  - Modify the path of custom JsApi and the path of preset offline package and its information:

    `initNebulaWithCustomPresetApplistPath` must be called it in `- (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions` of `DTFrameworkInterface`.

```
MPH5Demo_plugin 〉 iPhone XR   Running MPH5Demo_plugin on iPhone XR   397 1
MPH...lugin 〉 DTFrameworkInterface+MPH5Demo_plugin.m 〉 -application:beforeDidFinishLaunchingWithOptions:
MPH5Demo_plugin
  MPH5Demo_plugin
    MPH5Demo_plugin.entitlements
    Sources                                   M
    Resources
      DemoCustomPlugins.bundle
        Poseidon-UserDefine-Extra-Config.plist   R
      DemoCustomPresetApps.bundle
        00025000_1.0.0.1.amr
        201803230243503 8_1.0.0.0.amr
        NAMApplist.plist
    Assets.xcassets
    LaunchScreen.storyboard
  Supporting Files
  Products
  Frameworks
  MPaaS                                       M

37  - (BOOL)shouldAutoactivateShareKit
38  {
39      return YES;
40  }
41
42  - (DTNavigationBarBackTextStyle)navigationBarBackTextStyle
43  {
44      return DTNavigationBarBackTextStyleAlipay;
45  }
46
47  - (void)application:(UIApplication *)application beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
48  {
49
50
51  //    [MPNebulaAdapterInterface initNebula];
53      NSString *presetApplistPath = [[NSBundle mainBundle] pathForResource:[NSString
            stringWithFormat:@"DemoCustomPresetApps.bundle/NebulaApplist.plist"] ofType:nil];
54      NSString *appPackagePath = [[NSBundle mainBundle] pathForResource:[NSString
            stringWithFormat:@"DemoCustomPresetApps.bundle"] ofType:nil];
55      NSString *pluginsJsapisPath = [[NSBundle mainBundle] pathForResource:[NSString
            stringWithFormat:@"DemoCustomPlugins.bundle/Poseidon-UserDefine-Extra-Config.plist"] ofType:nil];
56      [MPNebulaAdapterInterface initNebulaWithCustomPresetApplistPath:presetApplistPath
            customPresetAppPackagePath:appPackagePath customPluginsJsapisPath:pluginsJsapisPath];
57  }
```

- Specify the configurations, including the basic class of all HTML5 pages / Global resource package / UA / Whether to verify the signature, and so on:

  After container initialization, set it in `- (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions` of `DTFrameworkInterface`, otherwise it will be overwritten by the default configuration of the container.

```
Portal 〉 MPaaS 〉 Targets 〉 Portal 〉 APMo...ework 〉 DTFrameworkInterface+Portal.m 〉 -application:afterDidFinishLaunchingWithOptions:

//    [[MASSAccess shareInstance] dispatchTokenSvrInfo:nil uploadInfo:nil downloadInfo:nil downloadInfoCrypto:nil
    configStorage:nil];

//    [MPTinyAppAdapterInterface initTinyApp];

    NSString *customPresetApplistPath = [[NSBundle mainBundle].bundlePath stringByAppendingFormat:@"/%@/%@",
        @"MPH5Demo.bundle", @"h5_json.json"];
    NSString *customPresetAppPackagePath = [[NSBundle mainBundle].bundlePath stringByAppendingFormat:@"/%@",
        @"MPH5Demo.bundle"];
    NSString *customPluginsJsapisPath = [[NSBundle mainBundle].bundlePath stringByAppendingFormat:@"/%@/%@",
        @"MPH5Demo.bundle", @"Poseidon-Custom-Config.plist"];
    [MPNebulaAdapterInterface initNebulaWithCustomPresetApplistPath:customPresetApplistPath
        customPresetAppPackagePath:customPresetAppPackagePath customPluginsJsapisPath:customPluginsJsapisPath];
}

- (void)application:(UIApplication *)application afterDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions  Categ...
{
    [MPNebulaAdapterInterface shareInstance].nebulaVeiwControllerClass = NSClassFromString(@"MPH5WebViewController");
    [MPNebulaAdapterInterface shareInstance].nebulaNeedVerify = NO;
    [MPNebulaAdapterInterface shareInstance].nebulaUserAgent = @"mPaaS/Portal";
    [MPNebulaAdapterInterface shareInstance].nebulaCommonResourceAppList = @[@"77777777"];

    [[MPNebulaAdapterInterface shareInstance] requestAllNebulaApps:^(NSDictionary *data, NSError *error) {

    }];
}
```

Container basic class

- Custom basic class of all HTML5 pages, it must be the subclass of `H5WebViewController`.

```
MPH5Demo_plugin 〉 iPhone XR   Running MPH5Demo_plugin on iPhone XR   397 1
MPH5Demo_plugin 〉 MPH5Demo_plugin 〉 Sources 〉 Template 〉 MPH5WebViewController.h 〉 No Selection

MPH5Demo_plugin
  MPH5Demo_plugin
    MPH5Demo_plugin.entitlements
    Sources
      MPAntUIDemoDef.h
      DemoViewController.h
      DemoViewController.m
      MPNavigatorDemoVC.h
      MPNavigatorDemoVC.m
      MPNavigatorScrollVC.h
      MPNavigatorScrollVC.m
      RPC
      Template
        MPH5WebViewController.h   M
        MPH5WebViewController.m   M
        H52Native.html
      JsApis
      Plugins   M
    Launcher

//
//  MPH5WebViewController.h
//  MPH5Demo
//
//  Created by shifei.wkp on 2019/2/3.
//  Copyright © 2019 alipay. All rights reserved.
//

#import <UIKit/UIKit.h>

NS_ASSUME_NONNULL_BEGIN

@interface MPH5WebViewController : H5WebViewController

@end

NS_ASSUME_NONNULL_END
```

- If `back` method is implemented in the original HTML5 basic class, it need to be deleted.

```
MPH5Demo_plugin 〉 MP...gin 〉 So...ces 〉 Te...late 〉 MPH5WebViewController.m 〉 @implementation MPH5WebViewController
107
108 }
109
110 - (void)viewDidAppear:(BOOL)animated
111 {
112     [super viewDidAppear:animated];
113 }
114
115 - (void)viewWillDisappear:(BOOL)animated
116 {
117     [super viewWillDisappear:animated];
118 }
119
120 //#pragma mark - back
121 //- (void)back
122 //{
123 //
124 //    id<PSDPluginProtocol>  navigationItemPlugin = [[[self psdScene] pluginManager]
        plugin:@"NBPlugin4NavigationItem"];
125 //    if ([navigationItemPlugin respondsToSelector:@selector(backItemClicked:)]) {
126 //        [navigationItemPlugin performSelector:@selector(backItemClicked:) withObject:nil];
127 //    }
128 //}
129
```

Custom navigation bar

- Return button: Monitor `kNBEvent_Scene_NavigationItem_Left_Back_Create_Before` event, modify the default style of the navigation bar.

```
MPH5Demo_plugin 〉 MPH5Demo_plugin 〉 Sources 〉 Template 〉 Plugins 〉 MPPlugin4TitleView.m 〉 -handleEvent:
MPH5Demo_plugin
MPH5Demo_plugin
MPH5Demo_plugin.entitlements
Sources
MPAntUIDemoDef.h
DemoViewController.h
DemoViewController.m
MPNavigatorDemoVC.h
MPNavigatorDemoVC.m
MPNavigatorScrollVC.h
MPNavigatorScrollVC.m
RPC
Template
MPH5WebViewController.h M
MPH5WebViewController.m M
H52Native.html
JsApis
Plugins
MPPlugin4TitleView.h
MPPlugin4TitleView.m M

63     [self.target addEventListener:kEvent_Proxy_Request_Start_Handler withListener:self useCapture:NO];
64
65     [super pluginDidLoad];
66 }
67
68 - (void)handleEvent:(PSDEvent *)event
69 {
70     [super handleEvent:event];
71
72     if ([kNBEvent_Scene_NavigationItem_Left_Back_Create_Before isEqualToString:event.eventType]) {
73
74
75         NSArray *leftBarButtonItems = event.context.currentViewController.navigationItem.leftBarButtonItems;
76         if ([leftBarButtonItems count] == 1) {
77             if (leftBarButtonItems[0] && [leftBarButtonItems[0] isKindOfClass:[AUBarButtonItem class]]) {
78
79                 AUBarButtonItem *backItem = leftBarButtonItems[0];
80                 backItem.backButtonColor = [UIColor greenColor];
81                 backItem.titleColor = [UIColor colorFromHexString:@"#00ff00"];  'colorFromHexString:' is deprecate
82             }
83         }
84         [event preventDefault];
85         [event stopPropagation];
```

1.4.4. Use container

1.4.5. Advanced Guide

1.4.5.1. Customize the navigation bar for HTML5 pages

Before customizing the navigation bar style for the HTML5 page, familiarize yourself with related knowledge about the framework navigation bar and HTML5 container, including:

Prerequisites

Before customizing the navigation bar style for the HTML5 page, familiarize yourself with related knowledge about the framework navigation bar and HTML5 container, including:

- iOS custom navigation bar
- Custom JSAPIs

- Custom plug-in

Set the default navigation bar style for all HTML5 pages

Given an app theme, to specify a uniform style for all HTML5 pages, you can customize a style by listening to the events provided by the Nebula container.

- For details about events supported by the container, see the header file `NebulaSDK/NBDefine.h`.

MPH5Demo_plugin 〉 MPaaS 〉 Frameworks 〉 NebulaSDK.framework 〉 Headers 〉 NBDefines.h 〉 No Selection

```
#define kNBEvent_Scene_TitleView_All                    @"scene.titleView.*"


// navitaion item btn
#define kNBEvent_Scene_NavigationItem_Left_Back_Create_Before   @"scene.navigationItem.left.back.create.before" // 监听该事件，可以调用preventDefault阻止默认行为（创建默认的返回按钮），若要自定义返回按钮，则设置其`customView`
#define kNBEvent_Scene_NavigationItem_Left_Back_Create_After    @"scene.navigationItem.left.back.create.after" // 监听该事件，可以获取返回按钮，并设置它的属性
#define kNBEvent_Scene_NavigationItem_Left_Back_Click           @"scene.navigationItem.left.back.click" // 监听该事件，可以调用preventDefault阻止默认行为（默认是返回行为），并自己实现返回逻辑
#define kNBEvent_Scene_NavigationItem_Left_Back_All             @"scene.navigationItem.left.back.*"


#define kNBEvent_Scene_NavigationItem_Left_Close_Create_Before  @"scene.navigationItem.left.close.create.before" // 监听该事件，可以调用preventDefault阻止默认行为（创建默认的关闭按钮），若要自定义关闭按钮，则设置其`customView`
#define kNBEvent_Scene_NavigationItem_Left_Close_Create_After   @"scene.navigationItem.left.close.create.after" // 监听该事件，可以获取关闭按钮，并设置其属性
#define kNBEvent_Scene_NavigationItem_Left_Close_Click          @"scene.navigationItem.left.close.click" // 监听该事件，可以调用preventDefault阻止默认行为（默认是关闭行为），并自己实现关闭行为
#define kNBEvent_Scene_NavigationItem_Left_Close_All            @"scene.navigationItem.left.close.*"


#define kNBEvent_Scene_NavigationItem_Right_Setting_Create_Before  @"scene.navigationItem.right.setting.create.before" // 监听该事件，可以调用preventDefault阻止默认行为（默认是创建设置按钮），若要自定义设置按钮，则设置其`customView`，自定义的customView要实现协议NBSettingButtonProtocol
#define kNBEvent_Scene_NavigationItem_Right_Setting_Create_After   @"scene.navigationItem.right.setting.create.after" // 监听该事件，可
```

- The listening event of Customize a plugin needs to be customized:

```
  @implementation MPPlugin4TitleView

  - (void)pluginDidLoad
  {
      self.scope = kPSDScope_Scene;
      // -- Return area
      [self.target
addEventListener:kNBEvent_Scene_NavigationItem_Left_Back_Create_Before withListener:self
useCapture:NO];
      [self.target addEventListener:kNBEvent_Scene_NavigationItem_Left_Back_Create_After
withListener:self useCapture:NO];
      [self.target
addEventListener:kNBEvent_Scene_NavigationItem_Left_Close_Create_Before
withListener:self useCapture:NO];
      [self.target
addEventListener:kNBEvent_Scene_NavigationItem_Left_Close_Create_After withListener:self
useCapture:NO];

      // -- Title area
      [self.target addEventListener:kNBEvent_Scene_TitleView_Create_Before
withListener:self useCapture:NO];
      [self.target addEventListener:kNBEvent_Scene_TitleView_Create_After
withListener:self useCapture:NO];

      // -- Control button area
      [self.target
addEventListener:kNBEvent_Scene_NavigationItem_Right_Setting_Create_Before
withListener:self useCapture:NO];
      [self.target
addEventListener:kNBEvent_Scene_NavigationItem_Right_Setting_Create_After
withListener:self useCapture:NO];
      [self.target
addEventListener:kNBEvent_Scene_NavigationItem_Right_SubSetting_Create_After withListene
r:self useCapture:NO];
      [self.target addEventListener:kNBEvent_Scene_NavigationItem_Right_Setting_Change w
ithListener:self useCapture:NO];

      // -- Progress bar
      [self.target addEventListener:kNBEvent_Scene_ProgressView_Create_Before
withListener:self useCapture:NO];
      [self.target addEventListener:kNBEvent_Scene_ProgressView_Create_After
withListener:self useCapture:NO];

      [super pluginDidLoad];
  }
```

- Set the styles of the Back button and Close button in the return area:

```
```
- (void)handleEvent:(PSDEvent *)event
{
    [super handleEvent:event];

    if ([kNBEvent_Scene_NavigationItem_Left_Back_Create_Before
isEqualToString:event.eventType]) {
        [event preventDefault];
```
```

```
        event.context.currentViewController.navigationItem.leftBarButtonItem =
[[UIBarButtonItem alloc] initWithTitle:@"Cancel" style:UIBarButtonItemStylePlain target:se
lf action:@selector(onClickBack)];
    }else if ([kNBEvent_Scene_NavigationItem_Left_Back_Create_After
isEqualToString:event.eventType]){
        //Modify the style of the Back button.
        NSArray *leftBarButtonItems =
event.context.currentViewController.navigationItem.leftBarButtonItems;
        if ([leftBarButtonItems count] == 1) {
            if (leftBarButtonItems[0] && [leftBarButtonItems[0] isKindOfClass:
[AUBarButtonItem class]]) {
                // Based on the default Back button, modify the Back arrow and text color.
                AUBarButtonItem *backItem = leftBarButtonItems[0];
                backItem.backButtonColor = [UIColor greenColor];
                backItem.titleColor = [UIColor colorFromHexString:@"#00ff00"];

                // Hide the Back arrow.
//                backItem.hideBackButtonImage = YES;

                // Hide the Back text: Set the text to transparent and retain the s click
area of the Back button.
//                backItem.titleColor = [UIColor clearColor];
        }
    }
}else if ([kNBEvent_Scene_NavigationItem_Left_Close_Create_Before
isEqualToString:event.eventType]){
//        // Hide the Close button.
//        [event preventDefault];
//        NBNavigationItemLeftCloseEvent *itemEvent = (NBNavigationItemLeftCloseEvent *)ev
ent;
//        UIButton *button = [UIButton buttonWithType:UIButtonTypeCustom];
//        button.frame = CGRectMake(0, 0, 44, 44);
//        button.backgroundColor = [UIColor greenColor];
//        [button setTitle:@"Close" forState:UIControlStateNormal];
//        itemEvent.customView = button;
    }else if ([kNBEvent_Scene_NavigationItem_Left_Close_Create_After
isEqualToString:event.eventType]){
//        // Modify the style of the Close button.
//        [event preventDefault];
//        NBNavigationItemLeftCloseEvent *itemEvent = (NBNavigationItemLeftCloseEvent *)ev
ent;
//        UIButton *closeButton = (UIButton *)itemEvent.customView;
//        [closeButton setTitle:@"Close" forState:UIControlStateNormal];
//        [closeButton setTitleColor:[UIColor greenColor] forState:UIControlStateNormal];
    }
}
```

- Set the title style of an HTML5 page:

```
if ([kNBEvent_Scene_TitleView_Create_Before isEqualToString:event.eventType]) {
        // Rewrite the style of TitleView.
        NBNavigationTitleViewEvent *e = (id)event;
        [e preventDefault];

    }else if ([kNBEvent_Scene_TitleView_Create_After isEqualToString:event.eventType]) {
        // Modify the created TitleView style.
        NBNavigationTitleViewEvent *e = (id)event;
        [[e.titleView mainTitleLabel] setFont:[UIFont systemFontOfSize:16]];
        [[e.titleView mainTitleLabel] setTextColor:[UIColor greenColor]];
        [e.titleView mainTitleLabel].lineBreakMode = NSLineBreakByTruncatingMiddle;
    }
```

- Set the style of the OptionMenu control button:

```
if ([kNBEvent_Scene_NavigationItem_Right_Setting_Create_After
isEqualToString:event.eventType] ||
[kNBEvent_Scene_NavigationItem_Right_SubSetting_Create_After
isEqualToString:event.eventType]) {
        // Modify the created RightBarItem style.
        NBNavigationItemRightSettingEvent *settingEvent = (id)event;
        settingEvent.adjustsWidthToFitText = YES;
        settingEvent.maxWidth = [UIScreen mainScreen].bounds.size.width / 3.0f;
        UIButton *button = settingEvent.customView;
        button.titleLabel.font = [UIFont systemFontOfSize:14.0f];
        CGRect frame = CGRectMake(0, 0, 22, 22);
        button.frame = frame;
        [button setTitleColor:[UIColor blackColor] forState:UIControlStateNormal];
        if (!CGSizeEqualToSize(button.bounds.size, frame.size)) {
            button.frame = frame;
        }
    }
```

- Set the style of the progress bar displayed when an HTML5 page is loaded:

```
if([kNBEvent_Scene_ProgressView_Create_After isEqualToString:event.eventType]){
        NBProgressViewEvent *progressEvent = (NBProgressViewEvent *)event;
        id<NBProgressViewProtocol> progressView = progressEvent.progressView;
        [progressView setProgressTintColor:[UIColor greenColor]];
    }
```

Customize the navigation bar style of an HTML5 page

You can customize the navigation bar style of an HTML5 page **before the page is loaded** or **after the page is opened**.

- Before the page is loaded: Perform the following steps to customize the navigation bar on the basis of the default navigation bar style.
  i. When the current HTML5 page is loaded, customize startup parameters and specify a customization method.
     - For details about transferring custom startup parameters for switching from one HTML5 page to another HTML5 page, see Open a new page and Start other apps.
     - For details about the method for transferring custom startup parameters for switching from a native page to an HTML5 page, see the following code:

```
#pragma mark: the custom parameter to be passed when you open an HTML5 page. The base class of the HTML5 page will read the parameter to implement your settings of whether to hide the navigation pane.
 - (void)gotoHideNavigator
 {
     // Open an HTML5 page and hide the navigation bar.
     [[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"showTitleBar":@NO,@"transparentTitle":@"auto"}];
 }

 - (void)gotoShowNavigator
 {
     // Open an HTML5 page and show the navigation bar.
     [[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"showTitleBar":@YES}];
 }

 - (void)gotoTransparency
 {
     // Open an HTML5 page and set the navigation bar to transparent.
     [[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"transparentTitle":@"auto"}];
 }

 - (void)gotoUpdateBackgroundColor
 {
     // Modify the background color of the navigation bar.
     [[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"titleBarColor":@"16775138"}];
 }

 - (void)gotoUpdateStatusBarStyle
 {
     // Modify the color of the status bar.
     [[UIApplication sharedApplication] setStatusBarStyle:UIStatusBarStyleLightContent];
 }

 - (void)gotoUpdateBackTitleColor
 {
     // Modify the default text and color of the Back button.
     [[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"backButtonColor":@"ff0000"}];
 }

 - (void)gotoUpdateTitleColor
 {
     // Modify the title color.
     [[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"titleColor":@"ff0000"}];
```

```
}
```

ii. Handle the HTML5 base class. In the `viewWillAppear` method of the base class, call the native API method based on the transferred startup parameters to modify the navigation bar style.

```
- (void)viewWillAppear:(BOOL)animated
{
 [super viewWillAppear:animated];

 // WebView of the current page.
 UIWebView *currentWebView = (UIWebView *)self.psdContentView;
 NSLog(@"[mpaas] webView: %@", currentWebView);

 // Startup parameters of the current page.
 NSDictionary *expandParams = self.psdScene.createParam.expandParams;
 NSLog(@"[mpaas] expandParams: %@", expandParams);

 if ([expandParams count] > 0) {
     [self customNavigationBarWithParams:expandParams];
 }
 WKWebView *webView = (WKWebView *)[self psdContentView];
 if (@available(iOS 11.0, *)) {
   webView.scrollView.contentInsetAdjustmentBehavior =
UIScrollViewContentInsetAdjustmentNever;
 } else {
   self.automaticallyAdjustsScrollViewInsets = NO;
 }
 webView.scrollView.contentInset = UIEdgeInsetsMake(200, 0, 0, 0);
 webView.scrollView.scrollIndicatorInsets = webView.scrollView.contentInset;
}

- (void)customNavigationBarWithParams:(NSDictionary *)expandParams
{
 // Customize the background color of the navigation bar.
 NSString *titleBarColorString = expandParams[@"titleBarColor"];
 if ([titleBarColorString isKindOfClass:[NSString class]] && [titleBarColorString leng
th] > 0) {
     UIColor *titleBarColor = [UIColor colorFromHexString:titleBarColorString];
     [self.navigationController.navigationBar
setNavigationBarStyleWithColor:titleBarColor translucent:NO];
     [self.navigationController.navigationBar
setNavigationBarBottomLineColor:titleBarColor];
 }

 // Set whether to hide the navigation bar, which is not hidden by default. After the
navigation pane is set to be hidden, the WebView needs to be displayed in full-screen
mode.
 NSString *showTitleBar = expandParams[@"showTitleBar"];
 if (showTitleBar && ![showTitleBar boolValue]) {
 self.options.showTitleBar = NO;
     [self.navigationController setNavigationBarHidden:YES];

     // Adjust the position of the WebView.
     UIWebView *webView = (UIWebView *)[self psdContentView];
     CGRect frame = webView.frame;
```

```
     frame.origin.y = [[UIApplication sharedApplication] statusBarFrame].size.height;
     frame.size.height -= [[UIApplication sharedApplication]
statusBarFrame].size.height;
     webView.frame = frame;
     self.automaticallyAdjustsScrollViewInsets = NO;

 }

 // Set whether the navigation bar is transparent. The navigation bar is not transpare
nt by default. After the navigation pane is set to transparent, the WebView needs to b
e displayed in full-screen mode.
 NSString *transparentTitle = expandParams[@"transparentTitle"];
 if ([transparentTitle isEqualToString:@"always"] || [transparentTitle
isEqualToString:@"auto"]) {

     // The navigation bar and bottom line become transparent.
     UIColor *clearColor = [UIColor clearColor] ;
     [self.navigationController.navigationBar setNavigationBarTranslucentStyle];
     [self.navigationController.navigationBar
setNavigationBarStyleWithColor:clearColor translucent:YES];

     // Adjust the position of the WebView.
     self.edgesForExtendedLayout = UIRectEdgeAll;
     if (@available(iOS 11.0, *)) {
         UIWebView *wb = (UIWebView *)[self psdContentView];
         wb.scrollView.contentInsetAdjustmentBehavior =
UIScrollViewContentInsetAdjustmentNever;
     }else{
         self.automaticallyAdjustsScrollViewInsets = NO;
     }
 }

 // Modify the default text and color of the Back button.
 NSString *backButtonColorString = expandParams[@"backButtonColor"];
 if ([backButtonColorString isKindOfClass:[NSString class]] && [backButtonColorString
length] > 0) {
     UIColor *backButtonColor = [UIColor colorFromHexString:backButtonColorString];

     NSArray *leftBarButtonItems = self.navigationItem.leftBarButtonItems;
     if ([leftBarButtonItems count] == 1) {
         if (leftBarButtonItems[0] && [leftBarButtonItems[0] isKindOfClass:
[AUBarButtonItem class]]) {
             AUBarButtonItem *backItem = leftBarButtonItems[0];
             backItem.titleColor = backButtonColor;
             backItem.backButtonColor = backButtonColor;
         }
     }
 }

 // Set the title color.
 NSString *titleColorString = expandParams[@"titleColor"];
 if ([titleColorString isKindOfClass:[NSString class]] && [titleColorString length] >
0) {
     UIColor *titleColor = [UIColor colorFromHexString:titleColorString];
     id<NBNavigationTitleViewProtocol> titleView = self.navigationItem.titleView;
     [[titleView mainTitleLabel] setFont:[UIFont systemFontOfSize:16]];
```

```
      [[titleView mainTitleLabel] setTextColor:titleColor];
  }
}
```

- After the page is opened: Modify the navigation bar style dynamically during user operation. Call the native API method to modify the style through a custom JSAPI.
  - Customize a JSAPI to handle the navigation bar style of the current page.
  - Handle the native navigation bar in the JSAPI based on the API provided in Customize the navigation bar style of an HTML5 page.

```
 - (void)handler:(NSDictionary *)data context:(PSDContext *)context callback:  (PSDJsA
piResponseCallbackBlock)callback
  {
  [super handler:data context:context callback:callback];

  UIViewController *currentVC = context.currentViewController;
  currentVC.navigationItem.titleView = [[AUDoubleTitleView alloc]
initWithTitle:@"Title" detailTitle:@"Subtitle"];
  callback(@{@"success":@YES});
  }
```

1.4.5.2. Automatic HTML5 container tracking

When the Nebula container provided by mPaaS is used to load an HTML5 page, the Nebula container will automatically collect statistics on page loading actions and performance, and capture exceptions to help you track data about HTML5 page loading. This topic describes how to integrate the Nebula container automatic tracking function and view tracking data.

Prerequisite

To integrate the automatic HTML5 container tracking function provided by mPaaS, ensure that:

- You have created an app on the mPaaS console.
- You have completed the steps to add the SDK , and the client project has integrated `NebulaLogging.framework` .

Procedure

Initialize configurations

1. To integrate the automatic tracking function of the Nebula container, enable HTML5 tracking monitoring during container initialization.

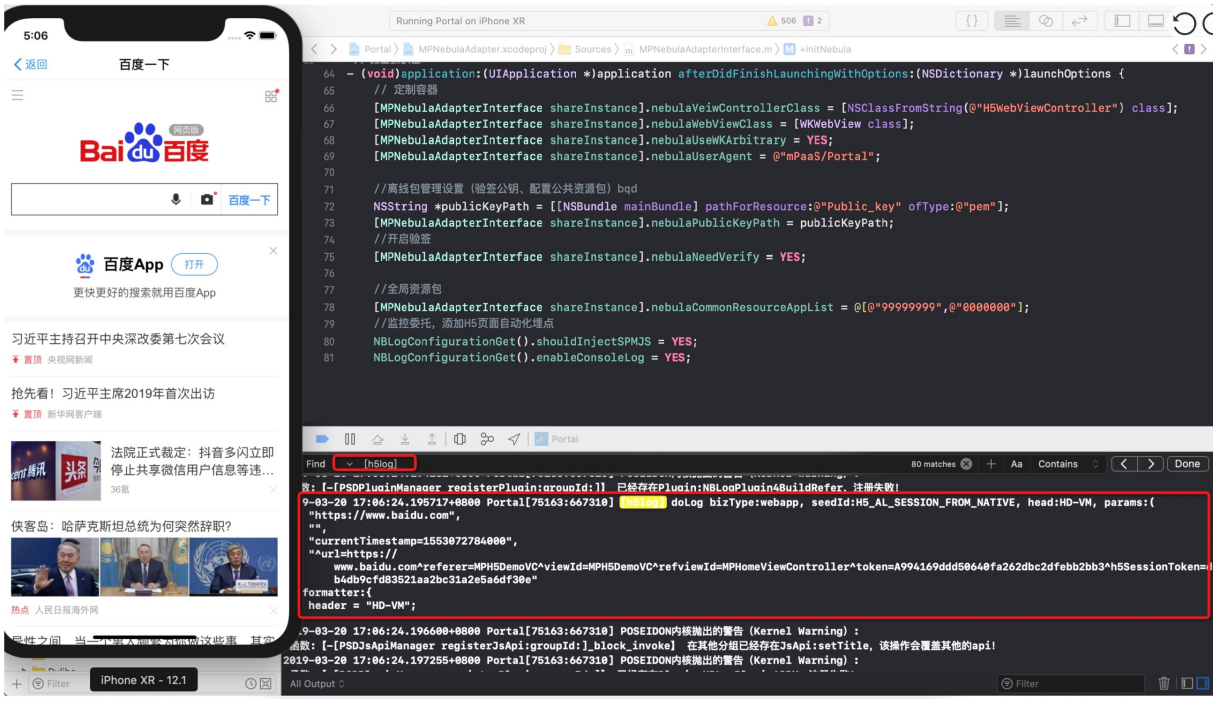
```
// Monitoring delegation. Add automatic HTML5 container tracking.
 NBLogConfigurationGet().shouldInjectSPMJS=YES;
 #ifdef DEBUG
 NBLogConfigurationGet().enableConsoleLog = YES;
 #endif
 [NBLogServiceGet() start];
 [[NBMonitor defaultMonitor] setDelegate:NBLogServiceGet()];
```

2. When the Nebula container is used to load an HTML5 page after HTML5 tracking monitoring is enabled, the container will automatically collect statistics on page loading actions, performance, and exceptions. Based on the scenarios of viewing tracking data, there are two methods:
   - View client logs: View local tracking data on the client. This mode applies to troubleshooting in the app development phase. See View client logs.
   - View server logs: View actual tracking data generated for online users. This mode applies to

online troubleshooting after an app is released online. See View server logs.

View client logs

After an HTML5 page is loaded, search the keyword `[h5log]` on the Xcode console to view key information about tracking data related to page loading, as shown in the following figure.

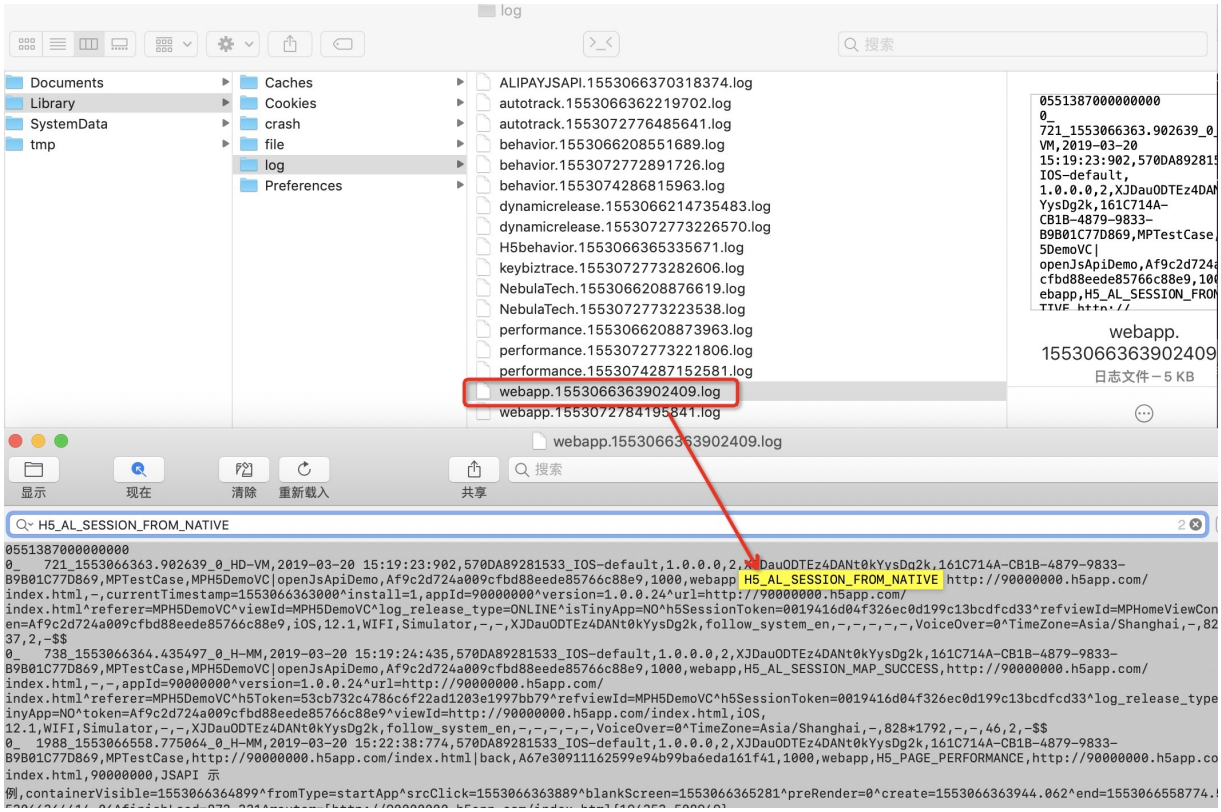


The following table lists the key information included in the HTML5 container tracking data output by the Xcode console.

SeedId	Example	Description
bizType	`webapp`	Log type. Written into the file name of the local log file.
seedId	`H5_AL_SESSION_FROM_NATIVE`	Unique tracking ID. For detailed description, see HTML5 container tracking sets.
head	`HD-VM`	Log model. See Performance tracking.
params	`https://www.baidu.com`	URL of the current page.
	`currentTimestamp=1553072784000`	Custom parameter, such as the timestamp and page loading status.
	-	Custom parameter, such as the page element position.

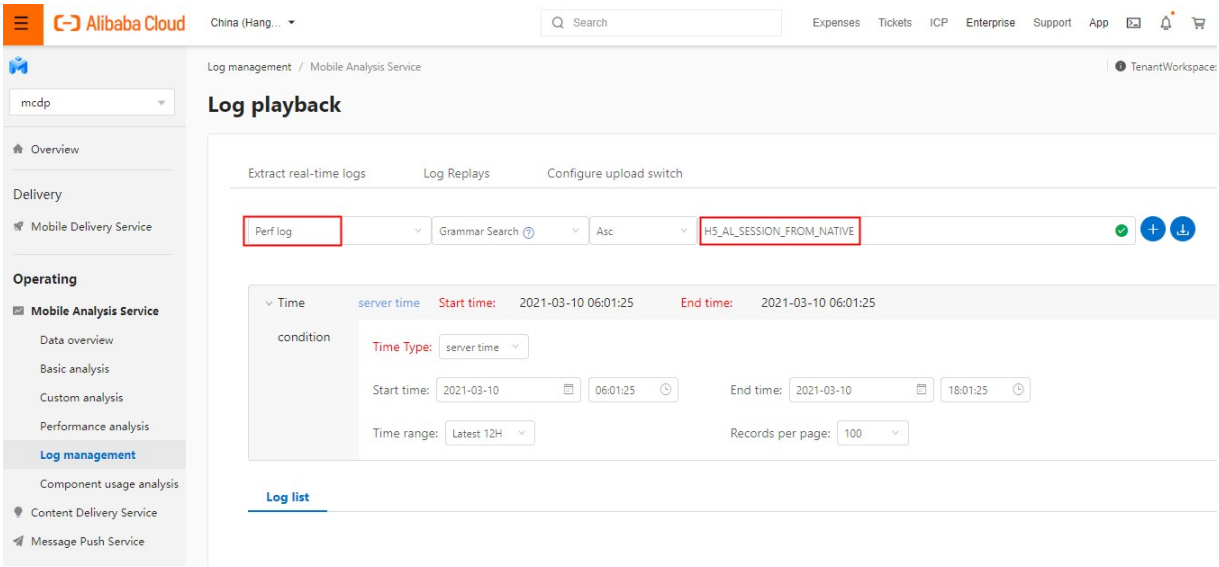
	^url=https://www.baidu.com^referer=MPH5DemoVC^viewId =MPH5DemoVC^refviewId=MPHomeViewController^token= A994169ddd50640fa262dbc2dfebb2bb3^h5SessionToken= db4db9cfd83521aa2bc31a2e5a6df30e/td>	Tracking information: Including offline package app ID, offline package version number, URL loaded by the current HTML5 page, VC class name, and token of the current page.
formatter	header = "HD-VM"	Same as the description of the head field.

To view the complete content of a log record on the Xcode console, find the file whose name begins with `bizType` under the `Library/log` directory in the sandbox, and then search by the keyword seedId.



View server logs

To view HTML5 container tracking data of an online user, search for the data through Log replays on the mPaaS console.



HTML5 container tracking sets

Automatic tracking data measured by the HTML5 container is uniquely identified by seedId. There are three types of tracking sets based on the HTML5 page loading method.

- **Tracking set regarding opening online URLs**

SeedId	Description
H5_AL_SESSION_FROM_NATIVE	The container has been started.
H5_AL_PAGE_START	Page loading begins.
H5_AL_NETWORK_START	The page starts to send network requests.
H5_OPEN_PAGE_FINISH	Page loading is complete.
H5_AL_PAGE_APPEAR	The page appears for the first time.
H5_AL_JSAPI_SENDEVENT	A page calls a JSAPI.
H5_AL_JSAPI_NOTFOUND	JSAPI call on HTML5 page failed.
H5_TITLEBAR_BACK_BT	The Back button on the navigation bar is tapped.
H5_PAGE_PERFORMANCE	The page loading performance is measured.

- **Tracking set regarding opening offline package pages**

SeedId	Description
H5_APP_REQUEST	Request offline package information.
H5_APP_LOAD_DATASOURCE	Load offline package information.
H5_AL_SESSION_FROM_NATIVE	The container has started.
H5_APP_DOWNLOAD	Download an offline package.
H5_APP_UNZIP	Decompress an offline package.

SeedId	Description
H5_APP_POOL	Perform operations on the package management information pool, including adding, deleting, and modifying information in the pool.
H5_APP_VERIFY	Perform signature verification for an offline package.
H5_AL_SESSION_VERIFYTAR_FAIL	Signature verification for an offline package failed.
H5_AL_PAGE_START	Page loading begins.
H5_AL_SESSION_MAP_SUCCESS	A local offline package is loaded successfully.
H5_AL_SESSION_FALLBACK	Loading a local offline package failed. A fallback function is executed to request an online page.
H5_OPEN_PAGE_FINISH	Page loading is complete.
H5_AL_PAGE_APPEAR	The page appears for the first time.

- **Exception tracking set**

seedId	Description
H5_AL_NETWORK_PERFORMANCE_ERROR	Resource request exception.
H5_PAGE_ABNORMAL	Page exception.
H5_AL_PAGE_JSERROR	JS exception.
H5_AL_JSAPI_RESULT_ERROR	JSAPI exception.

1.4.5.3. Custom JSAPI

To call the native function on a page, for example, to display an ActionSheet or contacts dialog box, you need to expand a JavaScript API (JSAPI). With the JSAPI, you can add an entry to call the native function on an HTML5 page. Certain functions are implemented in native mode by implementing the handler method in the custom JSAPI class.

The HTML5 container component has the following capabilities:

- Provides abundant embedded JSAPIs to implement functions such as push, pop, and title setup. For more information, see Built-in JSAPI.
- Allows you to customize JavaScript API (JSAPI) and plug-in to satisfy your business expansion needs.

This topic describes how to use a demo of HTML5 container and offline package to customize a plug-in that modifies the navigation bar when an HTML5 page is loaded.

About this task

You can customize a JSAPI by using two methods:

- **Plist registration**
- **Code registration**

Procedure

Plist registration

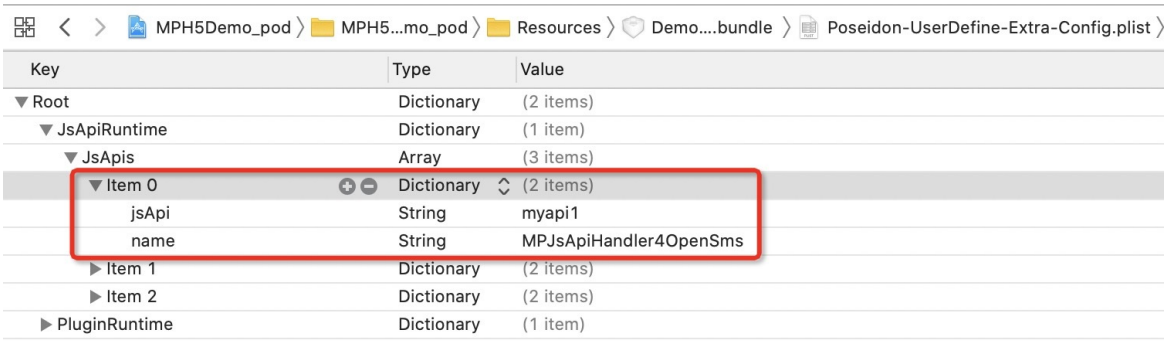
1. Create a JSAPI class:
   - Naming format: For consistency with the names of plug-ins provided by the container by default, the created JSAPI name begins with `XXJsApiHandler4`, where `XX` is a custom prefix.
   - Base class: All JSAPIs are derived from `PSDJsApiHandler`.
   - Basic method: Rewrite method `-(void)handler:context:callback:` in the `.m` file. When this JSAPI is called in the frontend, the call request will be forwarded to this method.
   - The following describes the parameters in this method:

Name	Description
`data`	The parameter passed in when an HTML5 page calls this JSAPI.
`context`	The context of the current HTML5 page. For details, see `PSDContext.h`.
`callback`	The callback method after the JSAPI call is complete. This method returns a dictionary as the calling result to the HTML5 page.

Refer to the following code example:

```
#import <NebulaPoseidon/NebulaPoseidon.h>
@interface MPJsApiHandler4OpenSms : PSDJsApiHandler
@end
@implementation MPJsApiHandler4OpenSms
- (void)handler:(NSDictionary *)data context:(PSDContext *)context callback:(PSDJsA
piResponseCallbackBlock)callback
{
[super handler:data context:context callback:callback];
// Open the SMS.
NSURL *url = [NSURL URLWithString:@"sms://xxx"];
BOOL reasult = [[UIApplication sharedApplication] openURL:url];
callback(@{@"success":@(reasult)});
}
@end
```

2. Register the JSAPI. Register this JSAPI in the custom Plist file.
   - Create a PLIST file for unified management of custom JSAPI and plug-ins. You can download the template file DemoCustomPlugins.bundle.zip and add this file to your project.
   - In the `JsApis` array, register the JSAPI class created in the previous step.



- The registered JSAPI is of the dictionary type and includes two items.

Parameter	Description
jsApi	The name of the JSAPI called on the HTML5 page. **Note** Attach a prefix to the name of a custom JSAPI to distinguish the custom JSAPI from the built-in JSAPI of the container. The purpose is to prevent unavailability due to interaction between the custom JSAPI and the built-in JSAPI.
name	The name of the created JSAPI class.

- Specify the path of the custom PLIST file during container configuration initialization.

  Initialize the HTML5 container. See Quick start of HTML5 container. Refer to the following code example:

```
   - (void)application:(UIApplication *)application
beforeDidFinishLaunchingWithOptions:(NSDictionary *)launchOptions
   {
       // Initialize the container.
       //    [MPNebulaAdapterInterface initNebula];

       // Specify JSAPI path and preset offline package information.
       NSString *presetApplistPath = [[NSBundle mainBundle] pathForResource:[NSString
stringWithFormat:@"DemoCustomPresetApps.bundle/h5_json.json"] ofType:nil];
       NSString *appPackagePath = [[NSBundle mainBundle] pathForResource:[NSString str
ingWithFormat:@"DemoCustomPresetApps.bundle"] ofType:nil];
       NSString *pluginsJsapisPath = [[NSBundle mainBundle] pathForResource:[NSString
stringWithFormat:@"DemoCustomPlugins.bundle/Poseidon-UserDefine-Extra-Config.plist"] o
fType:nil];
       [MPNebulaAdapterInterface
initNebulaWithCustomPresetApplistPath:presetApplistPath
customPresetAppPackagePath:appPackagePath customPluginsJsapisPath:pluginsJsapisPath]
   }
```

Code registration

In addition to customizing a JSAPI using the Plist, you can register a custom JSAPI by calling the interface method provided by the Nebula container.

1. Create a plug-in. See Customize a plug-in.
2. Implement the `addJSApis` method in the plug-in.

```
- (void)addJSApis
{
    [super addJSApis];

    // Register JSAPI by code.
    PSDJsApi *jsApi4DemoTest2 = [PSDJsApi jsApi:@"demoTest2"
                                          handler:^(NSDictionary *data, PSDContext *conte
xt, PSDJsApiResponseCallbackBlock responseCallbackBlock) {
                                              responseCallbackBlock(@{@"result":@"jsapi-d
emoTest2 The result of calling Native"});
                                          }
                                      checkParams:NO
                                        isPrivate:NO
                                            scope:self.scope];
    [self registerJsApi2Target:jsApi4DemoTest2];
}
```

- The following table describes the parameters required for registration.

Parameters	Description
jsApi	The name of the JSAPI called on the HTML5 page.
handler	JSAPI handler function, which is the same as the handler method in the PLIST registration mode.
checkParams	Specifies whether parameters are checked. Set this parameter to NO.
isPrivate	Specifies whether the JSAPI is a private JSAPI. Set this parameter to NO.
scope	Scope of action. Set this parameter to self.scope.

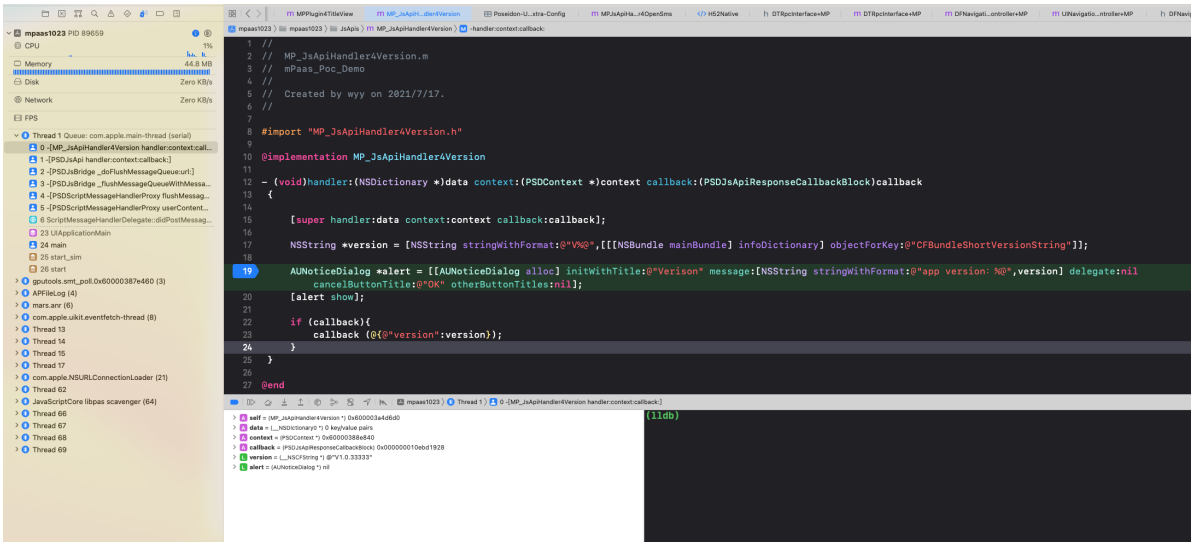
For details, see the implementation of the `MPPlugin4TitleView` class in the code example.

What to do next

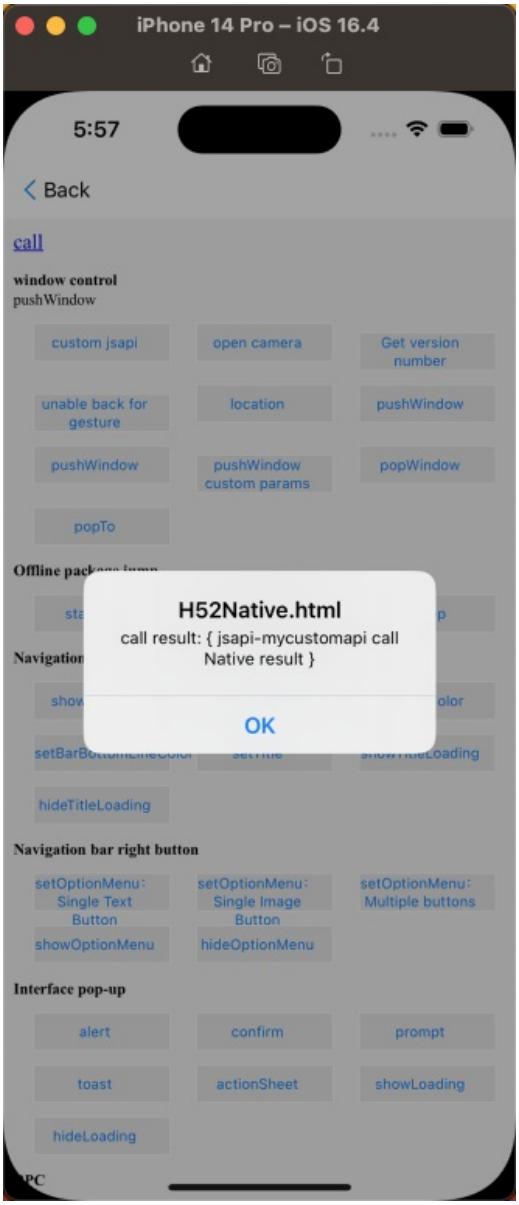
1. Call the custom JSAPI on an HTML5 page.

```
            }
            })">mtBizReport track</button>

        <p class="descript_title">Custom extension JSApi testing</p>
        <button class="jsapiButton" onclick="jsapi_call('myapi1')">openSms</button>
        <button class="jsapiButton" onclick="jsapi_call('myapi2')">setDoubleTitle</button>
        <button class="jsapiButton" onclick="jsapi_call('demoTest2')">demoTest2</button>
        <button class="jsapiButton" onclick="jsapi_call('mycustomapi')">custom jsapi</button>


        <p class="descript_title">Other precautions</p>
        <p class="descript_content">
        Because Android's bridge mechanism cannot ensure early injection, it is recommended to listen to the dom event AlipayJSBridgeReady if you want to call js-api as soon
            as possible, document.addEventListener('AlipayJSBridgeReady', function(e) {xxxx});
        </p>


        <script src="https://a.alipayobjects.com/g/h5-lib/bizlog/1.2.13/bizlog-browser.js"></script>
        <script type="text/javascript">
            function jsapi_call(apiName, params) {
                window.AlipayJSBridge && AlipayJSBridge.call(apiName, params, function(data) {
                    alert('call result'+JSON.stringify(data));
                });
            }

```

2. Add a breakpoint in the handler method and check whether the parameters provided by the HTML5 page meet the expectations.



3. Check whether the results returned by the HTML5 page meet the expectations.



1.4.5.4. Customize plugins

To do something at a certain time, for example, to log events when you enter the page, you need to develop a plugin. When the plugin subscribes to the corresponding events, you can process the data carried with the events in handler.

About this task

The procedure of customizing a plugin falls into 3 steps:

1. Create a plugin
2. Register the plugin
3. Use the plugin

See the code sample to learn how to customize a plugin which modifies the page navigation bar when the system loads H5 page.

Procedure

Create a plugin

Create a plugin class in the format as follows, and the following considerations apply:

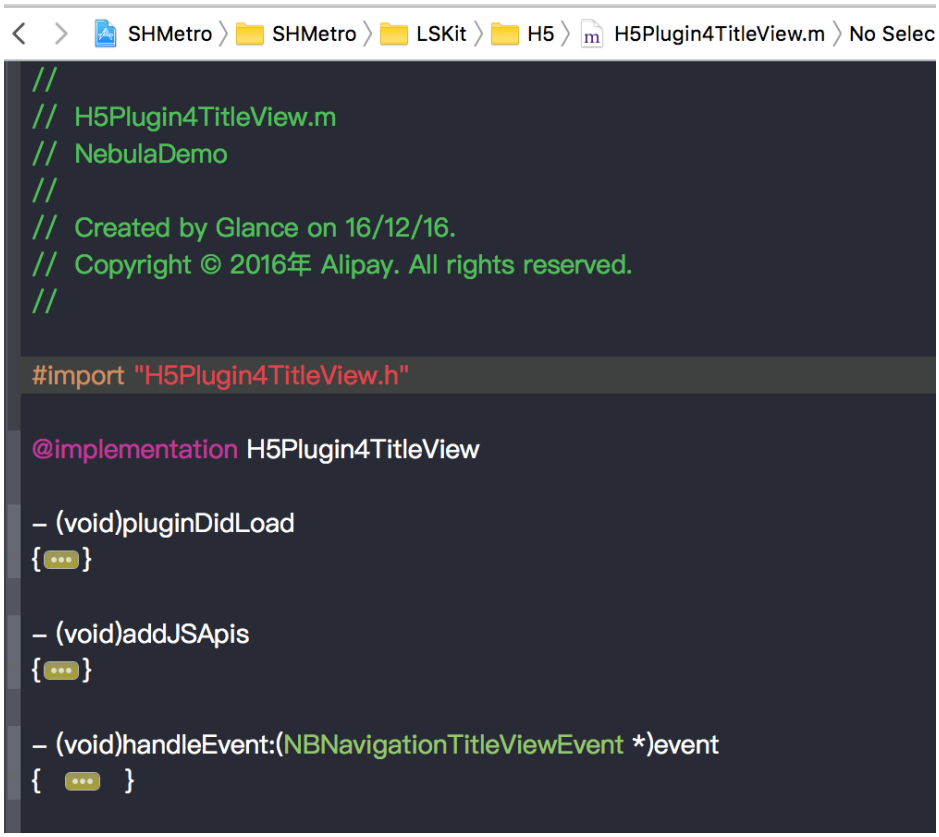
- Name: To keep consistence with the plugin name provided by the container by default, the name starts with `XXPlugin4`, in which the `XX` is your custom prefix.
- Base class: All plugins inherit from `NBPluginBase`.
- Implement basic methods: In the `.m` file, you need to override the following three methods:
  - `- (void)pluginDidLoad`: Required. To monitor events, see header file `NBDefines.h` for eventlist.
  - `- (void)addJSApis`: Optional. Communicate with H5, so it might register JSAPI.
  - `- (void)handleEvent`: Required. Process the logic when the monitored event is triggered.

`.h` file is as follows:

SHMetro 〉 SHMetro 〉 LSKit 〉 H5 〉 H5Plugin4TitleView.h 〉 No Selection

```
//
//  H5Plugin4TitleView.h
//  NebulaDemo
//
//  Created by Glance on 16/12/16.
//  Copyright © 2016年 Alipay. All rights reserved.
//

#import <NebulaSDK/NBPluginBase.h>

@interface H5Plugin4TitleView : NBPluginBase

@end
```

`.m` file is as follows:



Monitor events

Register the events that need to be monitored in `- (void)pluginDidLoad` method.

```
- (void)pluginDidLoad {
    self.scope = kPSDScope_Scene; // 1
    [self.target
    addEventListener:kNBEvent_Scene_TitleView_Title_Click // 2
                     withListener:self // 3
                       useCapture:NO]; // 4
    [super pluginDidLoad];
}
```

`addEventListener` method is used to monitor a certain event, and the description of each step is as follows:

Name	Definition
scope	Set the scope where the event works on. Currently, the supported scope from small to large is Scene, Session, and Service.
event	Set the event name, and the event constant is defined in `NBDefines.h`.

listener	Set the processor of the event, namely the object which provides `- handleEvent:` .
capture	Set whether to broadcast the event by capturing. It is generally `NO` .

Add JSAPI

In the process of registering the plugin, if you need to customize JSAPI to interact with the H5 page, you can register the JSAPI in the `- (void)addJSApis` method in codes, see Customize JSAPI > Register in codes. You can decide whether to implement this method on demand.

```
- (void)addJSApis
{
    [super addJSApis];
    // You can add the custom JSAPI realted to TitleView
}
```

Process monitor

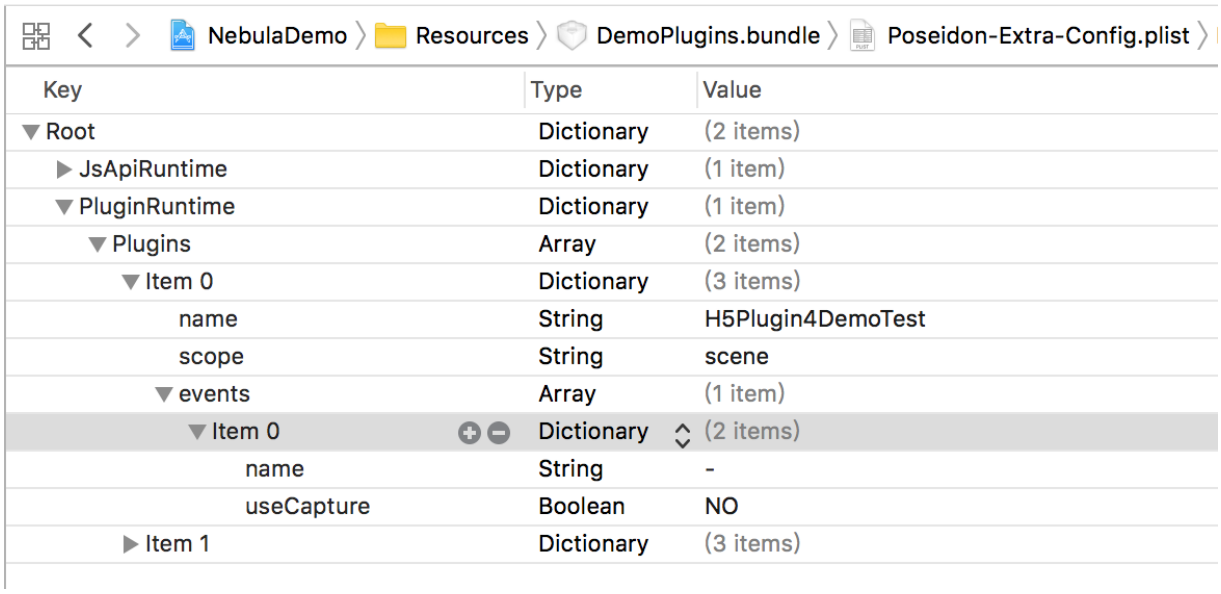
At last, process the logic in `- (void)handleEvent` when the monitored event is triggered.

```
- (void)handleEvent:(NBNavigationTitleViewEvent *)event
{
    [super handleEvent:event];

    if([kNBEvent_Scene_TitleView_Create_Before isEqualToString:event.eventType]) {
        // If you customize the TitleView, the support from the default
kNBEvent_Scene_TitleView_[Title_Set | Title_Click | Subtitle_Click] and other event will b
e lost
        NBNavigationTitleViewEvent *e = (NBNavigationTitleViewEvent *)event;
        H5WebViewController *currentViewController = (H5WebViewController
*)event.context.currentViewController;
        UINavigationController *navi = currentViewController.navigationController ?:
(UINavigationController *)APPDELEGATE.window.rootViewController;
        UINavigationBar *bar = navi.navigationBar;
        H5NavigationTitleView *newTitleView = [self createNavigationTitleView:bar.bounds];
        [newTitleView setMainTitle:@"Recreate" subtitle:nil];
        newTitleView.delegate = [e.titleView delegate];
        currentViewController.navigationItem.titleView = newTitleView;
        e.titleView = newTitleView;
        [e preventDefault];
    }
}
```

Note: Since when the events that have been monitored in the `- (void)pluginDidLoad` method are triggered, the logic upon event triggering is processed in `- handleEvent:` method, so if you have monitored multiple methods, you must process events differently according to the `class` or `eventType` of the `event` .

Register the plugin

After you create the plugin class, you must register this plugin in your custom plist file, see Customize JSAPI > Register JSAPI.

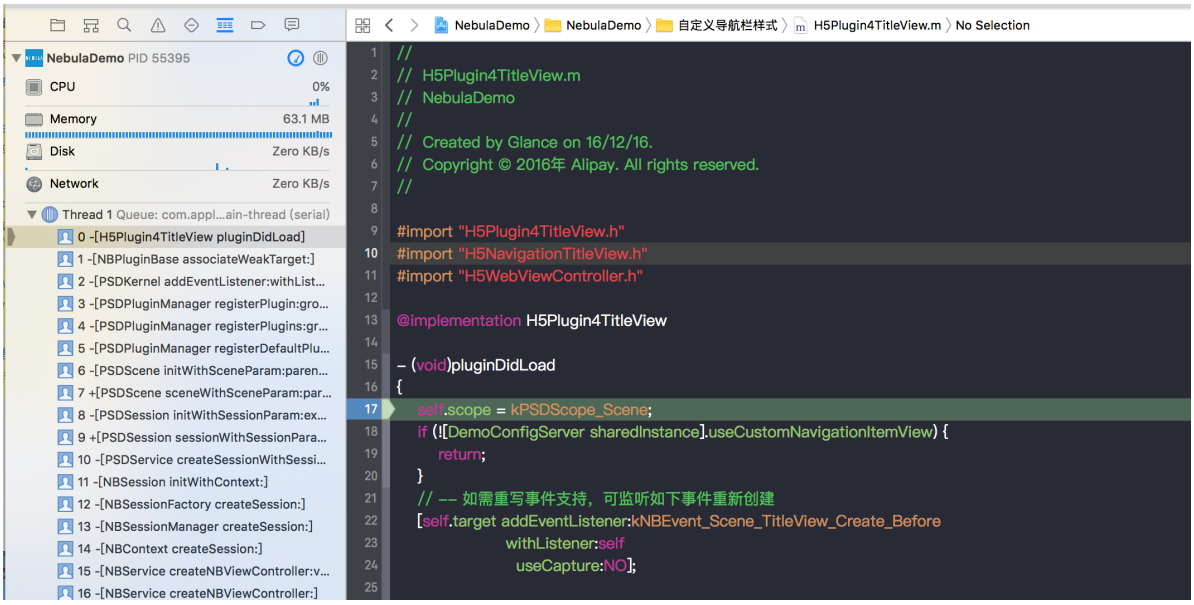


The registered plugin is of dictionary type, including the following 3 items:

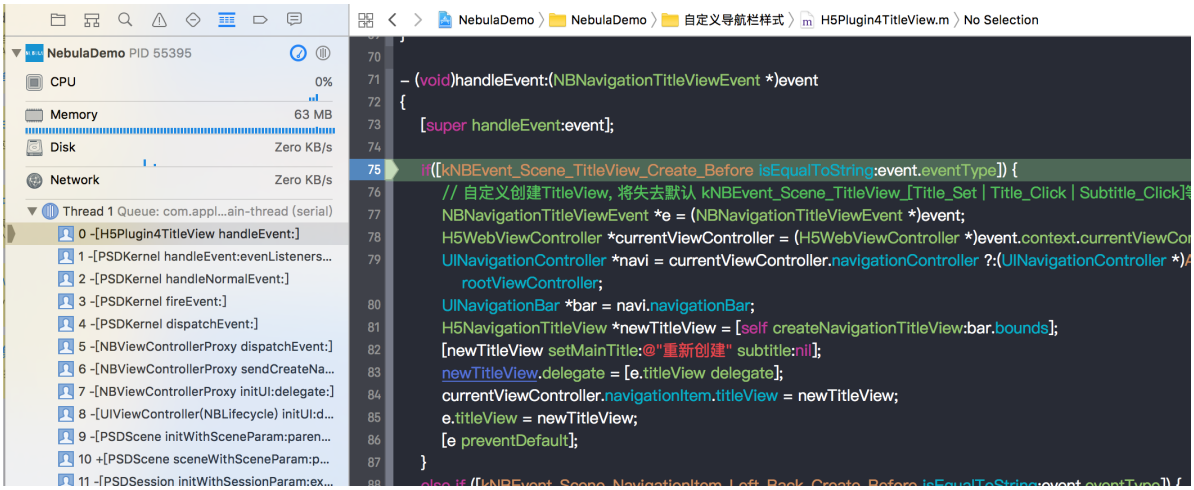
Name	Definition
name	Name of the created plugin class
scope	The scope where the plugin works on
events	Name of the monitored event

Use the plugin

1. Add breakpoints in the `pluginDidLoad` method to observe whether the calling sequence of the triggering time stacks is correct.



2. Add breakpoints in the `handleEvent` method to observe whether the monitored events can be correctly triggered.



3. Observe whether the custom navigation bar style on the H5 page takes effect.

1.4.5.5. Customize error page for H5

When loading the H5 page, if the network fails to load or the website cannot be opened, an error page will appear as follows:

"Network can't connect (-1009)."

The method of customizing the error page likes the above figure on the H5 page is the same as the Mini Program, refer to Customize error page for iOS Mini Program.

1.4.5.6. iOS Mini Program adds extension information

It is necessary to add a Mini Program package when publishing function by using the Mini Program on the console platform. The extension information used to configure the Mini Program must be packaged with launchParams, or it will not be obtained by the iOS terminal.

Obtain extension information

```
NAMApp *app = [NAMServiceGet() findApp:appId version:@"1.0.0.2"];
NSString *extend_info = app.extend_info;
```

1.5. Embedded JSAPI

1.5.1. Startup parameters

The appearance and action of the HTML5 container during running are controlled by a group of parameters. The startup parameters can be specified when you start a new instance or call pushWindow, for example:

```
alipays://platformapi/startapp?
appId=20000067&url=http%3A%2F%2Fm.taobao.com&showOptionMenu=NO&startMultApp=YES
```

In addition, the first URL of a new instance can carry a magic parameter `webview_options`. The content will be fetched by the container and transferred to the container.

```
?webview_options=showOptionMenu%3DNO&startMultApp%3DYES

// urlencode('showOptionMenu=NO&startMultApp=YES') =>
showOptionMenu%3DNO&startMultApp%3DYES
```

The startup parameters passed through by the client can be obtained by the frontend through `AlipayJSBridge.startupParams` or `jsapi:getStartupParams`.

Name	Abbreviation	Type	Description	Default value	pushWindow available	Remarks
url	-	String	Start URL.	“”	Y	
defaultTitle	dt	String	Default title, displayed on the title bar before the page is loaded for the first time.	“”	Y	

Name	Abbreviation	Type	Description	Default value	pushWindow available	Remarks
showLoading	sl	String	Whether to display the global loading animation before the page is loaded. The values include YES and NO.	“NO”	Y	
readTitle	rt	String	Whether to read the page title and display it on the title bar. The values include YES and NO.	“YES”	Y	
bizScenario	bz	String	Business scenario source. The value will be recorded in each event tracking to distinguish different sources.	“”	-	

Name	Abbreviation	Type	Description	Default value	pushWindow available	Remarks
backBehavior	bb	String	Action of the Back button. The values include back, pop, and auto. **back** : Return to the previous page if browser history exists. Otherwise, the current WebView is closed. **pop**: Close the current window. **auto**: Equivalent to pop in iOS. In Android, this action value is equivalent to back when the toolbar is visible and is equivalent to pop when the toolbar is invisible.	The default value is “back” in general browser mode for a non-HTML5 app (app ID: 20000067), and is “pop” for an HTML5 app (started using startApp).	-	
pullRefresh	pr	String	Whether pull-down refresh is supported. Valid values: YES and NO. The value can be set to YES for only local files.	“NO”	Y	

Name	Abbreviation	Type	Description	Default value	pushWindow available	Remarks
showProgress	sp	bool	Whether to display the loading progress bar. The values include YES and NO.	“NO”	-	
canPullDown	pd	String	Whether the screen supports slide-down (to show the black background or domain name). The values include YES and NO, and can be set to NO for only local files.	“YES”	YES	
showDomain	sd	bool	Whether to display the domain name when the screen is slid down. The values include YES and NO. It can be set to NO only for local files, and is forcibly set to NO for offline packages, not allowed to display the domain name.	“YES”	-	

Name	Abbreviation	Type	Description	Default value	pushWindow available	Remarks
backgroundColor	bc	int	Background color (in decimal, for example, bc=16775138).	“”	-	
showOptionMenu	so	bool	Whether to display the ellipsis (...) on the upper right. The values include YES and NO.	The default value is NO for HTML5 apps and YES for non-HTML5 apps.		
showTitleLoading	tl	bool	Whether to display the small loading animation on the left side of the title bar. The values include YES and NO.	NO	Y	
enableScrollBar	es	bool	Whether to use the WebView scrollbars, including the vertical and horizontal ones. This parameter is valid only for Android.	“YES”	-	

1.5.2. Event extension

1.5.2.1. Initialization

After `window.onload`, the container starts initialization to generate the global variable `AlipayJSBridge`, and then triggers the event of JSBridge initialization complete (`AlipayJSBridgeReady`).

> **Important**
> - `AlipayJSBridge` injection is an asynchronous process, and it's necessary to call `AlipayJSBridgeReady` after monitoring the event.
> - Be sure to use the `ready` method to perform initialization, otherwise the HTML5 container may fail to obtain `AlipayJSBridge`.

Use AlipayJSBridgeReady interface

```
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
```

Code sample

The following code sample is the standard writing format for `bridge` entry:

```
<h1>Method to use the Bridge </h1>

<script>
function ready(callback) {
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}

ready(function() {
  alert('bridge ready');
});
</script>
```

1.5.2.2. Click title

Click the navigation bar title (`titleClick`) to trigger callback.

Use titleClick interface

```
document.addEventListener('titleClick', function(e) {
  alert('title clicked')
}, false);
```

Code sample

The following code sample demonstrates the basic functions:

```
<h1>Please click the title to see effect </h1>

<script>
document.addEventListener('titleClick', function(e) {
  alert('title clicked')
}, false);
</script>
```

1.5.2.3. Click subtitle

Click the subtitle of navigation bar (`subtitleClick`) to trigger callback.

Use subtitleClick interface

```
document.addEventListener('subtitleClick', function (e) {
  alert('subtitle clicked')
}, false);
```

Code sample

The following code sample demonstrates the basic functions:

```
<h1>Please click the subtitle to see effect </h1>
<script>
document.addEventListener('subtitleClick', function(e) {
  alert('subtitle clicked')
}, false);

function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  AlipayJSBridge.call('setTitle', {
    title: 'Title',
    subtitle: 'Subtitle'
  });
});
</script>
```

1.5.2.4. Pause

When a webview interface is invisible, such as it is pushed into the background, the screen is locked, or it switches into the next page via pushwindow, the event of page being pushed into the background (`pause`) will be triggered.

> **Note**
>
> Note: For the clients above version 10.0.15, the container newly adds the pagePause and appPause events for the business to distinguish which case has triggered pause. Page being pushed into background (pause) = client invisible due to being pushed into background (appPause) + window invisible due to being siwtched to bottom (pagePause). Since the Android system cannot distinguish whether the native resume and pause event are caused by pushed background or page switch, the pageResume and pagePause event are called back through the JSAPI calling record, and therefore only applicable to the mutual switch among windows in the same session. The method that startApp and other clients directly switch pages is invalid, such as chooseImage.

Use pause interface

```
document.addEventListener('pause', function(e) {
  alert("pause");
}, false);
```

Code sample

The following code sample shows the alert that pops up after the user leave the current page:

```
<h1>Please click the button to open a new window </h1>
<a href="javascript:void(0)" class="btn J_new_window">New window opens the current page</a
>
<script>
function ready(callback) {
  // Call the interface directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.J_new_window').addEventListener('click', function() {
    AlipayJSBridge.call('pushWindow', {
      url: location.pathname,
    });
  });

  document.addEventListener('pause', function(e) {
    alert('paused');
  }, false);
});
</script>
```

1.5.2.5. Resume a page

When a webview interface returns to the top of stack again, such as it is evoked from background, screen is unlocked, or it return from the next page, the event of page resume (`resume`) will be triggered.

If the interface is returned via popWindow or popTo and data parameter is delivered, the page can obtain data.

If the `resume` of App takes place before `resume` of the interface, the event will come with a `resumeParams` containing parameters received during `app resume`.

> **Note**
>
> Note: For the clients above 10.0.15 version, the container newly adds the pageResume and appResume events for the service to distinguish which case has triggered resume. Page resume (resume)= Client being evoked from background, lock screen interface being restored (appResume) + window being restored after returning from the next page (pageResume). Since whether Android native resume and pause event are caused by pushed background or page switch cannot be distinguished, the pageResume and pagePause event are called back through the JSAPI calling record, and therefore only applicable to mutual switch among windows in the same session. The method that startApp and other clients directly switch pages is invalid, such as chooseImage.

Use resume interface

```
document.addEventListener('resume', function(e) {
  console.log("resumed");
}, false);
```

Code sample

The following example is the case with data when returning:

```
<h1>When click "Open new page" and then return, data will be brought to the page</h1>
<a href="#" class="btn J_demo">Open a new page</a>
<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('a').addEventListener('click', function() {
    AlipayJSBridge.call('pushWindow', {
      url: location.pathname
    });
  });

  document.addEventListener('back', function(e) {
    e.preventDefault();

    AlipayJSBridge.call('popWindow', {
      data: {
        from: location.href,
        info: Date.now()
      }
    });
  });

  document.addEventListener('resume', function(event) {
    alert ('contents brought when page rolls back: ' + JSON.stringify(event.data));
  });
});
</script>
```

API

Output parameters

The event parameter of event handling function has the following attributes:

Name	Type	Description
data	object	The contents brought by popTo or popWindow in the current App instance. Data cannot be passed across appId.
resumeParams	object	Contains parameters received during `app resume`.

1.5.2.6. Click upper-right menu buttons

When you call the `setOptionMenu` API to customize menu buttons in the upper-right navigation bar, the `optionMenu` event is triggered when you tap the buttons.

optionMenu API usage instruction

```
document.addEventListener('optionMenu', function (e) {
  alert("option menu clicked");
}, false);
```

Code sample

- The following example shows the basic functions.

```
<h1>Click menu buttons in the upper-right menu bar to view the effect.</h1>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  AlipayJSBridge.call('setOptionMenu', {
    title: 'Button',
    redDot: '5', // -1 indicates not to display, 0 indicates to display a red dot, and a v
alue within the range 1-99 indicates to display the specified number on the red dot.
    color : '#ffff6600', // The ARGB color value must begin with the number (#) sign.
  });
  document.addEventListener('optionMenu', function(e) {
    alert("optionMenu clicked!");
  }, false);
});
</script>
```

- Event with multiple optionMenus:

```
<h1>Click each menu button in the upper-right menu bar to view the effect.</h1>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  AlipayJSBridge.call('setOptionMenu', {
    // Menus are displayed from the last one to the first one.
    menus: [{
      icontype: 'scan',
      redDot: '-1', // -1 indicates not to display, 0 indicates to display a red dot, and
a value within the range 1-99 indicates to display the specified number on the red dot.
    }, {
      icontype: 'user',
      redDot '10', // -1 indicates not to display, 0 indicates to display a red dot, and a
value within the range 1-99 indicates to display the specified number on the red dot.
    }],
    override: true // Indicates whether to retain the default optionMenu when multiple opt
ions need to be set.
  });
  // Call forcibly to refresh the screen.
  AlipayJSBridge.call('showOptionMenu');

  document.addEventListener('optionMenu', function(e) {
    alert(JSON.stringify(e.data));
  }, false);
});
</script>
```

Note: If no `optionMenu` has been set, this event cannot be triggered. That is, tapping the default **...** cannot trigger the `optionMenu` event.

1.5.2.7. Back

When you click the return button on the upper left corner of navigation bar or physical return button on Android devices, the page will receive the (`back`) event.

If `event.preventDefault()` is called in the event handling function, the container will ignore `backBehaviour` and JS will implement rollback or other operations.

> **Note**
>
> Notes: For the sliding return of iOS, since sliding operation can be cancelled halfway, the back event cannot be triggered. The container has built-in protection mechanism. If you repeatedly click the return button but still cannot exit the current page, the page will be forced to exit with preventDefault ignored.

Use back interface

```
// Firstly block the default behavior, and then call the page jump API for manual control
document.addEventListener('back', function(e) {
  e.preventDefault();
  console.log('do something...')
  AlipayJSBridge.call('popWindow');
}, false);
```

Code sample

- Click Back to jump to the specified page:

```
<h1>Please click the return button in the top left corner </h1>
<p>Jump back to the Taobao page after click</p>
<script>
// Note: If you have customized back and used location.href to specify the target addres
s for jump, it's required to package a setTimeout to guarantee that the client thread is
n't blocked.
document.addEventListener('back', function(e) {
e.preventDefault();
setTimeout(function() {
  location.href = "https://m.taobao.com"
}, 10);
}, false);
</script>
```

- Click Back and then the confirmation dialogue box pops up:

```
<h1>Please click the return button in the top left corner </h1>
<script>
document.addEventListener('back', function(e) {
e.preventDefault();

AlipayJSBridge.call('confirm', {
  title: 'Dear',
  message: 'Are you sure to exit',
  okButton: 'Yes',
  cancelButton: 'No'
}, function(e) {
  if (e.ok) {
    AlipayJSBridge.call('popWindow');
  }
});
}, false);
</script>
```

1.5.2.8. Add notification listener

You can call this operation to add native notification listening.

addNotifyListener API usage instruction

```
AlipayJSBridge.call('addNotifyListener', {
  name:'fortest'
}, function (result) {
  console.log(result);
});
```

Code sample

The following example shows the basic functions.

```
<h1>Click the following button to perform a test.</h1>
<p>In this example, multiple operations are called on the same page. This API supports com
munication among different apps.</p>

<a href="#" class="btn start">Start listening.</a>
<a href="#" class="btn stop">Stop listening.</a>
<a href="#" class="btn send">Send a notification.</a>
<script>
function callback(e) {
  alert(JSON.stringify(e));
};

function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.start').addEventListener('click', function() {
    AlipayJSBridge.call('addNotifyListener', {
      name:'NEBULANOTIFY_TEST_EVENT' // Events sent by the HTML5 must be listened to by ad
ding the prefix NEBULANOTIFY_.
    }, callback);
  });

  document.querySelector('.stop').addEventListener('click', function() {
    AlipayJSBridge.call('removeNotifyListener', {
      name:'NEBULANOTIFY_TEST_EVENT' // Events sent by the HTML5 must be listened to by ad
ding the prefix NEBULANOTIFY_.
    }, function(e) {
      alert(JSON.stringify(e));
    });
  });

  document.querySelector('.send').addEventListener('click', function() {
    AlipayJSBridge.call('postNotification', {
      name:'TEST_EVENT' // Events sent by the HTML5 must be listened to by adding the pref
ix NEBULANOTIFY_.
      data: {
        hello: 'world'
      }
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('addNotifyListener', {
  name, keep
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value
name	String	Notification name	Y	“”
keep	String	Indicates whether to keep listening to this notification within the lifecycle of ViewController	N	“true”
fn	Function	Callback function	N	

Output parameters

`result: {}` : The parameter carried by the callback function, corresponding to an event message.

Error codes

Error code	Description
4	Unauthorized call
12	Listen to the same notification multiple times on the same page

Precautions

- If an event being listened to is sent by the HTML5, add the `NEBULANOTIFY_` prefix.
- Before configuring `addNotifyListener` , perform a remove operation to avoid errors caused by multiple add operations.
- If keep is set to false, listening is automatically unregistered after the event is triggered once.
- The data returned by Android contains only one layer of key/value pairs. If the value is Object during `postNotification` , Android performs a `JSON.stringify` operation on the value. We recommend that you perform `try{ JSON.parse }` during usage.

1.5.2.9. Remove notification listener

You can call this interface to remove native notification listening.

removeNotifyListener API usage instruction

```
AlipayJSBridge.call('removeNotifyListener', {
  name: 'fortest'
}, function (result) {
  console.log(result);
});
```

Code sample

The following example shows the basic functions.

```
<h1>Click the following button to perform a test.</h1>
<p>In this example, multiple operations are called on the same page. This API supports com
munication among different apps.</p>

<a href="#" class="btn start">Start listening.</a>
<a href="#" class="btn stop">Stop listening.</a>
<a href="#" class="btn send">Send a notification.</a>
<script>
function callback(e){
  alert(JSON.stringify(e));
};

function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.start').addEventListener('click', function() {
    AlipayJSBridge.call('addNotifyListener', {
      name:'NEBULANOTIFY_TEST_EVENT' // Events sent by the HTML5 must be listened to by ad
ding the prefix NEBULANOTIFY_.
    }, callback);
  });

  document.querySelector('.stop').addEventListener('click', function() {
    AlipayJSBridge.call('removeNotifyListener', {
      name:'NEBULANOTIFY_TEST_EVENT' // Events sent by the HTML5 must be listened to by ad
ding the prefix NEBULANOTIFY_.
    }, function(e) {
      alert(JSON.stringify(e));
    });
  });

  document.querySelector('.send').addEventListener('click', function() {
    AlipayJSBridge.call('postNotification', {
      name:'TEST_EVENT' // Events sent by the HTML5 must be listened to by adding the pref
ix NEBULANOTIFY_.
      data: {
        hello: 'world'
      }
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('removeNotifyListener', {
  name
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value
name	String	Notification name	Y	“”
fn	function	Callback function	N	-

Output parameters

`result: {success}` : the parameter transferred in the callback function.

Name	Type	Description
success	bool	Whether notification listening is removed successfully

Error codes

Error code	Description
4	Unauthorized call

Precautions

- `success: true` is returned regardless of whether the remove operation is registered.

1.5.2.10. Distribute a notification

The frontend can send notifications to the client through this API. The HTML5 adds a prefix `NEBULANOTIFY_` to the transferred name tag and sends it as a notification name.

In versions earlier than Android 10.1.0, the data field of `postNotification` cannot be left empty so that it can be accepted by `addNotifyListener`. However, this restriction has been removed in 10.1.0. `addNotifyListener` can accept an empty data record.

Android sends broadcasts through `LocalBroadcastManager`. The broadcasts can be listened to in the format of `NEBULANOTIFY_` + `name`.

postNotification API usage instruction

```
AlipayJSBridge.call('postNotification', {
  name:'fortest',
  data:{}
}, function (result) {
  console.log(result);
});
```

Code sample

The following example shows the basic functions.

```
<h1>Click the following button to perform a test.</h1>
<p>In this example, multiple operations are called on the same page. This API supports com
munication among different apps.</p>

<a href="#" class="btn start">Start listening.</a>
<a href="#" class="btn stop">Stop listening.</a>
<a href="#" class="btn send">Send a notification.</a>
<script>
function callback(e){
  alert(JSON.stringify(e));
};

function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.start').addEventListener('click', function(){
    AlipayJSBridge.call('addNotifyListener', {
      name:'NEBULANOTIFY_TEST_EVENT' // Events sent by the HTML5 must be listened to by ad
ding the prefix `NEBULANOTIFY_`.
    }, callback);
  });

  document.querySelector('.stop').addEventListener('click', function(){
    AlipayJSBridge.call('removeNotifyListener', {
      name:'NEBULANOTIFY_TEST_EVENT' // Events sent by the HTML5 must be listened to by ad
ding the prefix `NEBULANOTIFY_`.
    }, function(e){
      alert(JSON.stringify(e));
    });
  });

  document.querySelector('.send').addEventListener('click', function(){
    AlipayJSBridge.call('postNotification', {
      name:'TEST_EVENT' // Events sent by the HTML5 must be listened to by adding the pref
ix `NEBULANOTIFY_`.
      data: {
        hello: 'world'
      }
    }, function(e){
      alert(JSON.stringify(e));
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('postNotification', {
  name, data
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value
name	String	Notification name	Y	“”
data	object	Information notified to the client. Android will traverse all JSON data and converts the value of each item into string type for transmission. Ensure that the information is compatible with the data format.	N	-
fn	function	Callback function	N	-

Output parameters

`result: {success}` : the parameter transferred in the callback function.

Name	Type	Description
success	bool	Whether the message is successfully sent

Error codes

Error code	Description
4	Unauthorized call

1.5.3. Page context

1.5.3.1. Open a new page

The `pushWindow` operation is used to open a new page in an active offline package. A transition animation defined in the system is played when the new page is opened. If you need to open a page of another app in an offline package, use the startApp operation instead.

Unlike `location.href` , the `pushWindow` operation is similar to opening a new tab in a browser on a computer. Each window is a new tab. The previous tab is switched to the background, but maintains in the running state. The JavaScript code of the previous page keeps running.

How it works

```
// Open the homepage of Taobao, automatically read the title, and remove the right-side na
vigation bar.
AlipayJSBridge.call('pushWindow', {
  url: 'https://m.taobao.com/', // The URL of the page that you want to open.
  // Configuration parameters of the opened page
  param: {
    readTitle: true,    // Read the title automatically.
    showOptionMenu: false    // Hide the menus on the right side.
  },
  // Optional, used for transferring parameters to a newly opened page.
  // Use AlipayJSBridge.startupParams to obtain passData on a newly opened page.
  passData: {
    key1: "key1Value",
    key2: "key2Value"
  }
});
```

Note: In Android apps, the parameters must be enclosed in `param: { }`. In iOS apps, the parameters must be enclosed in `passData: { }`.

Sample code

- Open the homepage of Taobao and remove the right-side navigation bar.

```
<h1>Open the homepage of Taobao.</h1>
<a class="btn J_demo">Execute</a>
<script>
function ready(callback) {
    // Call JS Bridge if it has been injected.
    if (window.AlipayJSBridge) {
        callback && callback();
    } else {
        // Listen to the injection event if it has not been injected.
        document.addEventListener('AlipayJSBridgeReady', callback, false);
    }
}
ready(function(){
    document.querySelector('a').addEventListener('click', function() {
        // Open the homepage of Taobao, automatically read the title, and remove the right
-side navigation bar.
        AlipayJSBridge.call('pushWindow', {
            url: 'https://m.taobao.com/',
            param: {
                readTitle: true,
                showOptionMenu: false
            }
        });
    });
});
</script>
```

- Open the homepage of Taobao and set the title bar to transparent.

```
<h1>Open the homepage of Taobao.</h1>
<a class="btn J_demo">Execute</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('a').addEventListener('click', function() {
    AlipayJSBridge.call('pushWindow', {
      url: 'https://m.taobao.com',
      param: {
        transparentTitle: 'auto'
      }
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('pushWindow', {
    url, param, passData
})
```

Input parameters

Name	Type	Description	Required	Default value
url	string	The URL of the page that you want to open.	Y	“”
param	dictionary	The value FORMs in the supported key/value pair.	N	{}
passData	dictionary	The parameters that is passed by the new page. This parameter is applicable only to iOS. You can obtain these parameters by using the `AlipayJSBridge.startupParams` parameter on the new page.	N	{}

param parameter description

Name	Type	Description	Default value
`defaultTitle`	string	The default title of the page. The value is displayed in the title bar before the page is loaded the first time.	“”
`showLoading`	bool	Specifies whether to display the loading spinner before the page is 100% loaded.	false
`readTitle`	bool	Specifies whether to read the title of the page and display the title in the title bar.	true
`pullRefresh`	bool	Specifies whether pull-to-refresh is supported. This parameter can be set to true only on local files.	false
`allowsBounceVertical`	bool	Specifies whether the page can be vertically pulled to a blank area out of the content. In Android, pages can only be pulled down to display the background or the domain name. This parameter can be set to false only on local files in `.alipay.com/.alipay.net/` .	true
`bounceTopColor`	int	The color of the top margin when the page is pulled up to a blank area out of the content. The value is a decimal number. For example, the value of the bc color is 16775138.	-
`showTitleLoading`	bool	Specifies whether to display a loading spinner on the left side of the title bar.	false

Name	Type	Description	Default value
`transparentTitle`	string	Specifies whether the title bar is transparent. This parameter cannot be used with the `titleBarColor` parameter. Valid values: 1. `always`: The title bar is all transparent when the page is scrolled up and down. 2. `auto`: The transparency of the title bar gradually increases to 80 pt, which indicates completely transparent, as the page is scrolled down. The transparency gradually decreases to completely non-transparent as the page is scrolled up. When the page is scrolled up to the top, the title bar is completely non-transparent. 3. `none`: If the current page does not need to be transparent, you must set the `param` parameter in the `pushWindow` operation to set `transparentTitle` to none.	-
`titleBarColor`	int	The background color of the custom title bar. The value is a decimal number. For example, the value of the bc color is 16775138. Note: This parameter cannot be used with the `transparentTitle` parameter.	-
`scrollDistance`	int	The distance that the page is scrolled to set the transparency to 0.96 when the `transparentTitle` parameter is set to auto.	80

Name	Type	Description	Default value
`titleImage`	string	The address of the image used on the title. This parameter is supported only in iOS. A 3X PNG image is recommended. Only the current ViewController is affected. The `pushWindow` operation does not pass this parameter. To improve user experience, you can store the image in the global operation resource package.	“”
`closeCurrentWindow`	bool	Specifies whether to close the current window when a new window is opened.	false
`closeAllWindow`	bool	Specifies whether to close all windows of the current app when a new window is opened.	false
`animationType`	string	The type of the animation. Valid values: none and push. Default value: push. Note: This parameter is not supported in Android because animations are not supported in Android pages.	“”

Default inherited parameters of pushWindow

Name	Inherited	Note
url	Y	-
defaultTitle	Y	-
backBehavior	Y	The value is preferentially user-defined. If no values are specified, this parameter is set to pop.
showLoading	Y	-
readTitle	Y	-
pullRefresh	Y	-

Name	Inherited	Note
toolbarMenu	Y	-
showProgress	Y	-
defaultSubtitle	Y	-
backgroundColor	Y	-
canPullDown	Y	-
showOptionMenu	Y	-
showTitleLoading	Y	-
showDomain	Y	-

Differences between pushWindow and location.href

The following content describes the differences between `pushWindow` and `location.href`:

- Understand the Nebula container as a PC browser.
- A window can be understood as a tab, and `location.href` is redirection between tabs.
- The `pushWindow` operation is to open a new tab to superimpose on the previous tab. The page that is `pushed` is displayed at the top layer. All states of the previous tab located underneath are kept.
- The `location.href` operation is to redirect to the destination page on the current tab.
- When the tab pushed by calling the `pushWindow` operation is closed, in which the case backBehavior parameter is set to pop, if other opened windows exist, the previous window is displayed at the top layer and the `resume` operation is triggered.

Note

- When you call the `pushWindow` to open a page that corresponds to a specified URL, existing pages are not closed. To make sure that the performance does not deteriorate, do not open excessive pages. We recommend that you do not impose more than five layers of pages by calling the `pushWindow` operation on the same app. Otherwise, user experience is degraded and the app may stop responding.

1.5.3.2. Close the current page

You can call this interface to close the current page.

popWindow API usage instruction

```
// Close the current page.
AlipayJSBridge.call('popWindow');
```

Code sample

- Close the current page:

```
<h1>Close the current page.</h1>
<a href="#" class="btn J_demo">Execute</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('a').addEventListener('click', function() {
    AlipayJSBridge.call('popWindow');
  });
});
</script>
```

- Close the current page and transfer data:

```
<h1>Click "New window" and then "Return" to view the effect.</h1>
<a href="#" class="btn pop">Return</a>
<a href="#" class="btn new">New window</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.new').addEventListener('click', function() {
    AlipayJSBridge.call('pushWindow', {
      url: location.pathname
    });
  });

  document.querySelector('.pop').addEventListener('click', function() {
    AlipayJSBridge.call('popWindow', {
      data: {
        from: location.href,
        info: Date.now()
      }
    });
  });

  document.addEventListener('resume', function(event) {
    alert('content carried when the system returns to the previous page:' +
JSON.stringify(event.data));
  });
});
</script>
```

API

```
AlipayJSBridge.call('popWindow', {
  data
})
```

Input parameters

Name	Type	Description	Required	Default value

Name	Type	Description	Required	Default value
data	object	Content transferred to the page that is about to pop up within the current app. Data cannot be transferred across apps of different IDs.	N	-

Precautions

For details about how data carried during `popWindow` is received, see Resume a page (resume event).

1.5.3.3. Close multiple pages

This interface is used to roll back multiple levels of pages at a time.

> **Note**
>
> Note: It is only allowed to popTo the pages within the current App instance, and the cross-appId jump is not allowed.

Use popTo interface

```
// Close the currently opened page
AlipayJSBridge.call('popTo', {
  index: -1
});
```

Code example

- Close the current page and pass data:

```
<h1>Click "Execute" to close the current page and return data</h1>
<a href="#" class="btn J_demo">Execute</a>
<script>
function ready(callback) {
// Call directly if JSBridge has been injected
if (window.AlipayJSBridge) {
  callback && callback();
} else {
  // Monitor the injected events if JSBridge hasn't been injected
  document.addEventListener('AlipayJSBridgeReady', callback, false);
}
}
ready(function(){
document.querySelector('.J_demo').addEventListener('click', function() {
  // The passed data object will be received by the resume event on the upcoming page.
  AlipayJSBridge.call('popTo', {
    index: -1, // Return to the previous page, and if there is no previous page at this
time, an error will be reported.
    data: { // Important: data is a dictionary rather than an array
      from: location.href,
      info: Date.now()
    }
  }, function(e) { // Add a callback, because popTo will not necessarily succeed (an err
or will be reported when the current page is the only one opened)
    alert(JSON.stringify(e));
  });
});
});
</script>
```

- Return to the page that complies with regulation match through `urlPattern`:

```
<h1>Return to the URL that complies with a certain rule</h1>
<h3></h3>
<a href="javascript:void(0)" class="btn J_new_window">Open the current page in a new window</a>
<a href="javascript:void(0)" class="btn J_demo">Return</a>
<script>
var query = getQuery();
var depth = (+query.depth) || 0;
document.querySelector('h3').innerHTML = 'current page depth: ' + depth;
function ready(callback) {
// Call directly if JSBridge has been injected
if (window.AlipayJSBridge) {
  callback && callback();
} else {
  // Monitor the injected events if JSBridge hasn't been injected
  document.addEventListener('AlipayJSBridgeReady', callback, false);
}
}
ready(function(){
document.querySelector('.J_demo').addEventListener('click', function() {
  AlipayJSBridge.call('popTo', {
    urlPattern: 'pop-to-url-pattern.html',
  }, function(e) {
    alert(JSON.stringify(e));
  });
});

document.querySelector('.J_new_window').addEventListener('click', function() {
  AlipayJSBridge.call('pushWindow', {
    url: location.pathname + '?depth=' + (1+depth),
  });
});
});
</script>
```

API

```
AlipayJSBridge.call('popTo',{
  index, urlPattern
}, fn)
```

`index` and `urlPattern` refer to two query modes, which should not be used at the same time.

Input parameters

Name	Type	Description	Required	Default value	Version

Name	Type	Description	Required	Default value	Version
index	int	The index of the target interface in the session interface stack. If it is less than 0, it will be added to the index of the current interface.	Y		
urlPattern	string	The URL matching expression for the target interface (if URL contains `urlPattern`, the matching is successful)	Y		
fn	function	When the operation is successful, the callback may not be called; when the operation fails, the callback function is executed	N		

Out parameter

Name	Type	Description
result	undefined	The callback may not be called when the operation is successful; result should not be used

Error code

Error code	Description
10	No arguments are configured; Invalid index; Failed to match `urlPattern`

Attentions

- In general, `popTo` is used for scenarios that can be completed in multiple steps, such as returning when user information is complete, and returning after the 3-level address is selected.
- When returning via `urlPattern`, `popTo` will return to the page furthest from the current page, which is the bottom of the stack. Meanwhile, it will not check whether the URL of the current page matches.
- Please check the pResume event) for information on how the data carried in `popTo` is received.

1.5.3.4. Start other application

The startApp operation is used to start another HTML5 app or another offline package in the current mPaaS app.

startApp API usage instruction

```
AlipayJSBridge.call('startApp', {
  appId: '90000000',
  param: {
    url: '/index.html'
  }
}, function(result) {
  // noop
});

// To open multiple app instances:
// Include both appClearTop and startMultApp in param.
AlipayJSBridge.call('startApp', {
  appId: '90000000',
  param: {
    url: location.href,
    appClearTop: false,
    startMultApp: 'YES' // Note that the value is YES, not a value of the BOOL type.
  }
}, function(result) {
  // noop
});
```

Sample code

- Open an app with a transparent title bar:

```
<h1>Click the button to view the effect.</h1>
<a href="javascript:void(0)" class="btn dream">Open My savings.</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.dream').addEventListener('click', function() {
    AlipayJSBridge.call('startApp', {
      appId: '20000981',
      param: {
        url: '/www/dream-create.html',
        // Input startup parameters.
        canPullDown: true,
        transparentTitle: 'auto'
      }
    }, function(result) {
      // noop
    });
  });
});
</script>
```

- Open a new app and close the current app:

```
<h1>Click the button to open a new app and the current app is closed.</h1>
<a href="javascript:void(0)" class="btn forest">Open Ant Forest.</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.forest').addEventListener('click', function() {
    AlipayJSBridge.call('startApp', {
        appId: '60000002',
        // If no URL is input, the default URL of the app is used.
        param: {
          transparentTitle: 'auto'
        },
        closeCurrentApp: true
      }, function(result) {
        // noop
      });
  });
});
</script>
```

- By default, only one app instance is opened:

```
<h1>Try to open the current page again. Exit the current app and open it again.</h1>
<a href="javascript:void(0)" class="btn self">Open the current page.</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.self').addEventListener('click', function() {
    AlipayJSBridge.call('startApp', {
      // The current page is opened with the general app ID 20000067.
      //Therefore, other 20000067 pages are closed when startApp is called.
      //Actually only the current page is opened.
      appId: '20000067',
      param: {
        url: location.href,
      }
    }, function(result) {
      // noop
    });
  });
});
</script>
```

- Open multiple pages with the same app ID:

```
<h1>Open multiple pages with the same app ID.</h1>
<a href="javascript:void(0)" class="btn multi">Start another app.</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.multi').addEventListener('click', function() {
    AlipayJSBridge.call('startApp', {
      appId: '90000000',
      param: {
        url: '/index.html',
        appClearTop: false,
        startMultApp: 'YES' // Note that the value is YES, not a value of the BOOL type.
      }
    }, function(result) {
      // noop
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('startApp', {
  appId, param: {}, closeCurrentApp
}, fn)
```

Input parameters

Parameter	Data type	Description	Required	Default value	Baseline
appId	string	The ID of the offline package.	Y	“”	-

param	dictionary	The parameters of the app that you want to start. The value is defined by the specific business application. For example, if you want to open an offline package, you must specify the URL of the offline package in the param parameter. If you want to run multiple instances, see Note in this topic.	N	The data type of the value can be CHAR, BOOL, INT, or DOUBLE.	-
closeCurrentApp	bool	Specifies whether to exit the current app before starting a new app. This parameter is used when a page serves as a transition page.	N	-	>10.1.60
fn	function	The callback function that is executed when the operation failed.	N	-	-

Error code description

Error code	Description
10	The specified app ID is invalid.

11	Starting app failed.

Note

- `startApp` is used to open the App, so its positioning is at the App level, which is different from `pushWindow` in this regard.
- By default, if the current App has already been opened, when the App is opened again using `startApp`, an operation similar to restarting will be performed. That is to say, it is impossible to run two instances with the same appId at the same time.
- If an appId needs to run multiple instances, please add the `appClearTop=false&startMultApp=YES` option in the param parameter.

1.5.3.5. Exit the current app

You can call this operation to exit the current stack-top app.

exitApp API usage instruction

```
AlipayJSBridge.call('exitApp');
```

Code sample

- Exit the current page:

```
<h1>Click to exit the current page.</h1>
<a href="#" class="btn J_demo">Execute</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.J_demo').addEventListener('click', function() {
    AlipayJSBridge.call('exitApp');
  });
});
</script>
```

- Page jumping instance:

```
<h1>Click the following button to jump between pages.</h1>
<h3></h3>
<a href="javascript:void(0)" class="btn new">New current page</a>
<a href="javascript:void(0)" class="btn back">Return by 1 level.</a>
<a href="javascript:void(0)" class="btn popTo">Return by 2 levels using popTo.</a>
<a href="javascript:void(0)" class="btn exit">Close all pages.</a>
<script>
var query = getQuery();
var depth = (+query.depth) || 0;
document.querySelector('h3').innerHTML = 'Current page depth: ' + depth;
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.new').addEventListener('click', function() {
    AlipayJSBridge.call('pushWindow', {
      url: location.pathname + '?depth=' + (1+depth),
    });
  });
  document.querySelector('.back').addEventListener('click', function() {
    AlipayJSBridge.call('popWindow', {
      data: {
        method: 'popWindow',
        depth: depth,
      }
    });
  });
  document.querySelector('.popTo').addEventListener('click', function() {
    AlipayJSBridge.call('popTo', {
      index: -2,
      data: {
        method: 'popTo',
        depth: depth,
      }
    }, function(e) {
      if (e.error) {
        alert('Error: ' + JSON.stringify(e));
      }
    });
  });
  document.querySelector('.exit').addEventListener('click', function() {
    AlipayJSBridge.call('exitApp');
  });
});
document.addEventListener('resume', function(event) {
  alert('content carried when the system returns to the previous page:' +
JSON.stringify(event.data));
});
</script>
```

API

```
AlipayJSBridge.call('exitApp', {
  closeActionType, animated
}, fn);
```

Input parameters

Name	Type	Description	Required	Default value
closeActionType	string	`exitSelf` : Exit the local app. `exitTop` : Exit the stack-top app.	N	-
animated	bool	Indicates whether to enable the animation	N	true

Precautions

Pages without an app ID are run with the app ID 20000067. Therefore, calling `exitApp` on any page will close all pages.

1.5.3.6. Start HTML5 application

You can call this operation to open an HTML5 application. This method has been deprecated. The startApp API is recommended.

startH5App API usage instruction

```
AlipayJSBridge.call('startH5App', {
  appId: '20000067',
  url: '/www/index.html',
});
```

Code sample

```
<h1>Click the button to view the effect.</h1>
<a href="javascript:void(0)" class="btn dream">Open My savings.</a>
<script>
function ready(callback) {
// Call JS Bridge if it has been injected.
if (window.AlipayJSBridge) {
  callback && callback();
} else {
  // Listen to the injection event if it has not been injected.
  document.addEventListener('AlipayJSBridgeReady', callback, false);
}
}
ready(function(){
document.querySelector('.dream').addEventListener('click', function() {
  AlipayJSBridge.call('startH5App', {
    appId: '20000981',
      url: '/www/dream-create.html',
      // Input startup parameters.
  });
});
});
</script>
```

API

```
AlipayJSBridge.call('startH5App', {
  appId, url, ...param
})
```

Input parameters

Name	Type	Description	Required
appId	String	HTML5 application ID.	Y
url	String	Parameter for starting an app. This parameter is defined by the specific service app.	N
param	object	The value is in key-value format and no nesting is supported.	N

1.5.4. Native function

1.5.4.1. Scan code

This operation is used to call the scan component. It applies only to Android. Before using this interface, make sure you have added Scan component to your project. `actionType` in the interface is used to get code value.

scan API usage instruction

```
AlipayJSBridge.call('scan', {
  type: 'bar',
  actionType: 'scan'
}, function(result) {
  alert(JSON.stringify(result));
});
```

Code sample

Obtain QR code information:

```
<h1>Click the information corresponding to the output code after scanning.</h1>
<a href="#" class="btn read">Start scanning.</a>
<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.read').addEventListener('click', function() {
    AlipayJSBridge.call('scan', {
      type: 'qr'
    }, function(result) {
      alert(JSON.stringify(result));
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('scan', {
  type, actionType, qrcode
}, fn);
```

Input parameters

Name	Type	Description	Required	Default value
type	string	Target scan type, QR code or bar code.	Y	“”
actionType	string	Operation type: “scan”: scan only the code value.	N	“scan”
qrcode	string	Indicates the code value used for the “route” operation.	N	“”

Name	Type	Description	Required	Default value
fn	function	Callback function after scanning the QR code to obtain code information.	N	-

Output parameters

Parameter carried in the callback function: result: {error, barCode, qrCode, cardNumber}.

Name	Type	Description
barCode	string	Bar code data obtained through scan.
qrCode	string	QR code data obtained through scan.
error	int	Error code (10: operation canceled, 11: operation failed).

1.5.5. Interface

1.5.5.1. Alert

This interface is used for native implementation of alert box.

Use alert interface

```
AlipayJSBridge.call('alert', {
  title: 'Dear',
  message: 'Hello',
  button: 'Confirm'
}, function(e) {
  alert(JSON.stringify(e));
});
```

Code sample

- `alert` and `confirm`:

```
<h1>Click the buttons below to see the different effects</h1>
<a href="javascript:void(0)" class="btn alert">Click Alert</a>
<a href="javascript:void(0)" class="btn confirm">Click Confirm</a>
<script>
function ready(callback) {
// Call directly if JSBridge has been injected
if (window.AlipayJSBridge) {
  callback && callback();
} else {
  // Monitor the injected events if JSBridge hasn't been injected
  document.addEventListener('AlipayJSBridgeReady', callback, false);
}
}
ready(function() {
document.querySelector('.alert').addEventListener('click', function() {
  AlipayJSBridge.call('alert', {
    title: 'Dear',
    message: 'Hello',
    button: 'Confirm'
  }, function(e) {
    alert(JSON.stringify(e));
  });
});
document.querySelector('.confirm').addEventListener('click', function() {
  AlipayJSBridge.call('confirm', {
    title: 'Dear',
    message: 'Are you sure to exit?',
    okButton: 'Yes',
    cancelButton: 'No'
  }, function(e) {
    alert(JSON.stringify(e));
  });
});
});
</script>
```

API

```
AlipayJSBridge.call('alert',{
  title, message, button
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value
title	String	Title of alert box	N	“”
message	String	Text of alert box	N	

Name	Type	Description	Required	Default value
align	String	Alignment of message with left/center/right enumeration available	N	“center” for iOS, “left” for Android
button	String	Button text	N	“Confirm”
fn	function	Callback function, which will be called after clicking button.	N	

Attentions

- Unlike `window.alert`, `alert` is not a blocking interface. It means that if two alert boxes pop up successively, the one you see finally is the latter one.

1.5.5.2. Confirmation

The native implementation of confirmation box.

Use confirm interface

```
AlipayJSBridge.call('confirm', {
  title: 'Dear',
  message: 'Are you sure to exit?',
  okButton: 'Yes',
  cancelButton: 'No'
}, function(e) {
  alert(JSON.stringify(e));
});
```

Code sample

`alert` and `confirm`:

```
<h1>Click the buttons below to see the different effects </h1>
<a href="javascript:void(0)" class="btn alert">Click Alert</a>
<a href="javascript:void(0)" class="btn confirm">Click Confirm</a>
<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.alert').addEventListener('click', function() {
    AlipayJSBridge.call('alert', {
      title: 'Dear',
      message: 'Hello',
      button: 'Confirm'
    }, function(e) {
      e && alert(JSON.stringify(e))
    });
  });
  document.querySelector('.confirm').addEventListener('click', function(){
    AlipayJSBridge.call('confirm', {
      title: 'Dear',
      message: 'Are you sure to exit?',
      okButton: 'Yes',
      cancelButton: 'No'
    }, function(e) {
      alert(JSON.stringify(e))
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('alert',{
  title, message, okButton, cancelButton
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value
title	String	Title of alert box	N	“”
message	String	Text of alert box	N	

Name	Type	Description	Required	Default value
align	String	Alignment of message with left/center/right enumeration available	N	“center” for iOS, “left” for Android
okButton	String	Text of confirm button	N	“Confirm”
cancelButton	String	Text of cancel button	N	“Cancel”
fn	function	Callback function which will be called after clicking button	N	

Attentions

- Similar to `alert`, `confirm` is also not a blocking interface. It means that if two confirm boxes pop up successively, the one you see finally is the latter one.

1.5.5.3. Toast

The interface is used to display a toast, and the toast duration can be configured.

Use toast interface

```
AlipayJSBridge.call('toast', {
  content: 'Operation succeeds.',
  type: 'success',
  duration: 2000
}, function() {
  alert ("Execute after toast disappears");
});

// You can hide the pop-up toast via the hideToast interface

AlipayJSBridge.call('hideToast', {}, function() {
});
```

Code sample

```
<h1>Click the buttons below to see the different effects</h1>
<a href="javascript:void(0)" class="btn success">Display success information</a>
<a href="javascript:void(0)" class="btn fail">Display failure information</a>
<a href="javascript:void(0)" class="btn exception">Display exception information</a>
<a href="javascript:void(0)" class="btn none">Display information only</a>
<a href="javascript:void(0)" class="btn duration">Display information for 3.5s</a>

<script>
```

```
function toast(config, callback){
  AlipayJSBridge.call('toast',config, callback);
}

function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.success').addEventListener('click', function() {
    toast({
      content: 'Operation succeeds',
      type: 'success'
    });
  });

  document.querySelector('.fail').addEventListener('click', function() {
    toast({
      content: 'Network is busy now, please try again later.',
      type: 'fail'
    });
  });

  document.querySelector('.exception').addEventListener('click', function() {
    toast({
      content: 'Attention, an exception has occurred.',
      type: 'exception'
    });
  });

  document.querySelector('.none').addEventListener('click', function() {
    toast({
      content: 'Welcome',
    });
  });

  document.querySelector('.duration').addEventListener('click', function() {
    toast({
      content: 'Please wait a moment',
      duration: 3500,
    }, function(e){
      alert ('Call back after toast disappears');
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('toast', {
  content, type, duration
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value	Version
content	string	Text content	Y	‘’	8.0
type	string	none / success / fail / exception. Text must be passed in case of toast for exceptions	N	‘none’	8.1
duration	int	Duration of display, in milliseconds	N	2000	8.1
xOffset	float	Left is the positive direction, in pixels	N	0	10.0.15
yOffset	float	Up is the positive direction, in pixels	N	0	10.0.15
fn	function	Callback function, which is called when toast disappears	N		

Attentions

- Toast can automatically close, but it can also be closed via `hideLoading`. Although this method is not common, it is necessary to prevent toast from being closed by `hideLoading`.a
- For Android, if the system notification is turned off, the toast will not appear.
- For the Android versions below 10.1.2, `duration` only supports 2000 and 3500. The duration less than or equal to 2000 is equivalent to 2000, and the one more than 2000 is equivalent to 3500.

1.5.5.4. Bottom sheet

The interface provides a list of options, which docks at the bottom of the screen.

Use actionSheet interface

```
AlipayJSBridge.call('actionSheet', {
  'title': 'Title',
  'btns': ['First button ', 'Second button ', 'Third button'],
  'cancelBtn': 'Cancel',
  'destructiveBtnIndex': 2
}, function(data) {
  switch (data.index) { // index indicates the position of the button clicked by users in
actionSheet, starting from 0
    case 0:
      alert ('First button');
      break;
    case 1:
      alert ('Second button');
      break;
    case 2:
      alert ('Third button');
      break;
    case 3:
      alert ('Cancel button');
      break;
  }
});
```

Code sample

```
<h1>Click the button to call actionSheet</h1>
<a href="javascript:void(0)" class="btn actionSheet">Open actionSheet</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.actionSheet').addEventListener('click', function() {
    AlipayJSBridge.call('actionSheet',{
      'title': 'Title',
      'btns': ['First button ', 'Second button ', 'Third button'],
      'cancelBtn': 'Cancel',
      'destructiveBtnIndex': 2
    }, function(data) {
      switch (data.index) { // index indicates the position of the button clicked by users
in actionSheet, starting from 0
        case 0:
          alert ('First button');
          break;
        case 1:
          alert ('Second button');
          break;
        case 2:
          alert ('Third button');
          break;
        case 3:
          alert ('Cancel button');
          break;
      }
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('actionSheet',{
  title, btns, cancelBtn, destructiveBtnIndex
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value	Version
title	string	Title	N		

Name	Type	Description	Required	Default value	Version
btns	array	A set of buttons; the item type is string	Y		
cancelBtn	string	Configure the cancel button and text	N		
destructiveBtnIndex	int	(Special processing in iOS) Specifies the index number of the button, starting from 0 Scenario: Data needs to be deleted or cleared. It is in red color by default	N		
fn	function	It's the callback function after you select an option rather than being called back after API called	N		

Output parameter

The format is `{data: {index: 0}}`. The `index` refers to the position of button clicked by users in actionSheet, starting from 0.

1.5.5.5. Set title

This interface is used to set the title bar of the page, including main title, subtitle, and menu items of the title.

> **Note**
>
> Note: Due to Apple's ATS restrictions, the image URL must be either an HTTPS link or base64. HTTP links will be ignored.

Use setTitle interface

```
AlipayJSBridge.call('setTitle', {
  title: 'Title',
});
```

Code example

Set up title bars:

```
<h1>Click the buttons below to see the different effects</h1>
<a href="javascript:void(0)" class="btn title">Only set the title</a>
<a href="javascript:void(0)" class="btn subTitle">Title + subtitle</a>
<a href="javascript:void(0)" class="btn clear">Clear the title</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.title').addEventListener('click', function() {
    AlipayJSBridge.call('setTitle', {
      title: 'Title'
    });
  });

  document.querySelector('.subTitle').addEventListener('click', function() {
    AlipayJSBridge.call('setTitle', {
      title: 'Title',
      subtitle: 'Subtitle'
    });
  });

  document.querySelector('.clear').addEventListener('click', function() {
    AlipayJSBridge.call('setTitle', {
      title: ' ',
      subtitle: ' ',
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('setTitle',{
  title, subtitle, image
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value	Version
title	string	Main title text	N		

Name	Type	Description	Required	Default value	Version
subtitle	string	Subtitle text	N		
image	string	Supports URL or base64. Please use a 3X image. If you've set an image, the first two parameters will be invalid, and the title will not be read from the `webview` callback	N		9.9.5

Attentions

Empty title is not supported before Android 10.0.18, but it can be bypassed by setting an invisible string. This restriction has been removed since version 10.0.20.

```
AlipayJSBridge.call('setTitle', {
  title: '\u200b',
});
```

1.5.5.6. Set bottom line color

The interface is used to customize the color of the thin line at the bottom of the navigation bar. You can hide it by setting the line to the same color as the navigation bar.

Usee setBarBottomLineColor interface

```
AlipayJSBridge.call("setBarBottomLineColor", {
  color: 16711688
});
```

Code sample

```
<div style="padding-top:80px;">
  <a href="javascript:void(0)" class="btn title">Set the color of the thin line at the bot
tom of the navigation bar</a><br>
</div>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.title').addEventListener('click', function() {
    AlipayJSBridge.call("setBarBottomLineColor", {
      color: parseInt('ff0000', 16)
    });
  }, false);
},false);
</script>
```

API

```
AlipayJSBridge.call('setBarBottomLineColor',{
  color
}, fn)
```

Input parameter

Name	Type	Description	Required	Default value
color	int	Color value, decimal	Y	

1.5.5.7. Set title bar color

This interface is used to set the color of TitleBar.

Use setTitleColor interface

```
AlipayJSBridge.call("setTitleColor", {
  color: 16775138,
  reset: false // (Optional, false by default) Whether or not to reset the title color to
the default one.
});
```

Code sample

```
<div style="padding-top:80px;">
  <a href="javascript:void(0)" class="btn title">Set the title bar color</a>
  <a href="javascript:void(0)" class="btn reset">Reset the title bar color</a>
  <a href="javascript:void(0)" class="btn pushWindow">Open a new window with the transpare
nt title bar</a>
  <a href="javascript:void(0)" class="btn resetTransparent">Reset to the transparent title
bar</a>
</div>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(){
  document.querySelector('.title').addEventListener('click', function() {
    AlipayJSBridge.call("setTitleColor", {
      color: parseInt('108ee9', 16),
      reset: false // (Optional, false by default) Whether or not to reset the title color
to the default one.
    });
  });

  document.querySelector('.reset').addEventListener('click', function() {
    AlipayJSBridge.call("setTitleColor",{
      color: 16775138,
      reset: true
    });
  });

  document.querySelector('.pushWindow').addEventListener('click', function() {
    AlipayJSBridge.call("pushWindow", {
      url: location.pathname + '?__webview_options__=transparentTitle%3Dalways'
    });
  });

  document.querySelector('.resetTransparent').addEventListener('click', function() {
    AlipayJSBridge.call("setTitleColor", {
      color: 16775138,
      resetTransparent: true
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('setTitleColor', {
  color: 16775138,
  reset: false,
  resetTransparent: false
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value	Version
color	int	Color value, decimal	Y	-	-

1.5.5.8. Set option menu

- This interface is used to set the properties of the button on the right side of the title bar.
- This interface is only for setting. To guarantee the display of the button, you need to additionally call `showOptionMenu`.

> **Note**
>
> Note: Due to Apple's ATS restrictions, the icon URL must be either an HTTPS link or base64. HTTP links will be ignored.

Use setOptionMenu interface

```
AlipayJSBridge.call('setOptionMenu', {
  title: 'button',// select one from the three buttons - title, icon, and icontype
  redDot : '-1',// -1 refers to no display, 0 refers to present a badge notification, and
1-99 refers to the number displayed on the badge notification
  color: '#ff00ff00',// ARGB color value that must start with #
});
```

Code sample

Set various types of right-side buttons:

```
<h1>Click the buttons below to see the different effects</h1>

<a href="javascript:void(0)" class="btn button">Single button</a>
<a href="javascript:void(0)" class="btn icon">Single icon</a>
<a href="javascript:void(0)" class="btn menu">Multiple menus (9.9.3)</a>
<a href="javascript:void(0)" class="btn reset">Reset</a>
<a href="javascript:void(0)" class="btn hide">Hide</a>
<a href="javascript:void(0)" class="btn show">Display</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
```

```
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function(e) {
  document.querySelector('.button').addEventListener('click', function() {
    AlipayJSBridge.call('setOptionMenu', {
      title : 'button',
      redDot : '5',// -1 refers to no display, 0 refers to present a badge notification, a
nd 1-99 refers to the number displayed on the badge notification
      color: '#ff00ff00',// ARGB color value that must start with #
    });
    AlipayJSBridge.call('showOptionMenu');
  });

  document.querySelector('.icon').addEventListener('click', function() {
    AlipayJSBridge.call('setOptionMenu', {
      icon : 'http://pic.alipayobjects.com/e/201212/1ntOVeWwtg.png',
      redDot : '-1',// -1 refers to no display, 0 refers to present a badge notification,
and 1-99 refers to the number displayed on the badge notification
    });
    AlipayJSBridge.call('showOptionMenu');
  });

  document.querySelector('.menu').addEventListener('click', function() {
    AlipayJSBridge.call('setOptionMenu', {
      // The display order is from right to left
      menus: [{
        icontype: 'scan',
        redDot: '-1',// -1 refers to no display, 0 refers to present a badge notification,
and 1-99 refers to the number displayed on the badge notification
      }, {
        icontype: 'user',
        redDot: '-1',// -1 refers to no display, 0 refers to present a badge notification,
and 1-99 refers to the number displayed on the badge notification
      }],
      Override: true // Whether or not to reserve the default optionMenu if multiple optio
ns are required.
    });

    // Force to call it to refresh the interface
    AlipayJSBridge.call('showOptionMenu');
  });

  document.querySelector('.reset').addEventListener('click', function() {
    AlipayJSBridge.call('setOptionMenu', {
      reset: true,
    });
    AlipayJSBridge.call('showOptionMenu');
  });

  document.querySelector('.show').addEventListener('click', function() {
    AlipayJSBridge.call('showOptionMenu');
  });
```

```
  document.querySelector('.hide').addEventListener('click', function() {
    AlipayJSBridge.call('hideOptionMenu');
  });

  document.addEventListener('optionMenu', function(e) {
   alert(JSON.stringify(e.data));
  }, false);
});
</script>
```

API

There are several properties that are prioritized: reset > title > icontype > icon. Only one of these four properties is required.

```
AlipayJSBridge.call('setTitle',{
  title, icon, redDot, reset, color, override, menus, icontype
})
```

Input parameters

Name	Type	Description	Required	Default value	Version
title	string	Right-side button text	Y		
icon	string	URL of right-side button icon, base64 (since version 9.0) For version 8.3 and earlier: iOS: 40x40 (no margin left around); Android: 50x50 (5px transparent margin left on each side) For version 8.4 and later: both iOS and Andriod: 40x40 (no margin left around)	Y		
redDot	string	The number displayed on the badge notification	N		8.6

Name	Type	Description	Required	Default value	Version
reset	bool	Reset to the system default, other parameters will be ignored when reset=true	Y	false	8.6
color	string	Text color value	N	#ffffffff	9.0
override	bool	Whether or not to retain the default optionMenu if multiple options are required	N	false	9.9
menus	array	Set multiple buttons	N	[]	9.9
preventDefault	bool	Whether or not to block the default sharing function (the sharing pop-up box by default) ; when preventDefault=true, the sharing function will be blocked	N	[]	9.9

Name	Type	Description	Required	Default value	Version
icontype	string	Load the container preset picture based on picture types. Only one parameter will be select one from title, icon, and icontype. Notes: Only supports color changing of a single optionMenu. The specific types include: User (account), filter, search, add, settings, scan, info, help, locate, more, and mail (mailbox 10.0.8 and above)	N		9.9.3
contentDesc	string	Set reading content for blind readers	N		10.0.18

Attentions

- If the effect is not correct after calling `setOptionMenu`, please call `showOptionMenu` once.

1.5.5.9. Show option menu

This interface is used to display the properties of the button on the right side of the title bar.

Use showOptionMenu interface

```
AlipayJSBridge.call('showOptionMenu');
```

Code sample

See Set option menu.

1.5.5.10. Hide option menu

The interface is used to hide the properties of button on the right side of title bar.

Use hideOptionMenu interface

```
AlipayJSBridge.call('hideOptionMenu');
```

Code sample

See Set option menu.

1.5.5.11. Show loading

This interface is used to display the global loading box.

Use showLoading interface

```
AlipayJSBridge.call('showLoading', {
  text: 'Loading',
});
```

Code sample

Show/hide global loading box:

```
<h1>Click the buttons below to see the different effects</h1>
<p>Note that the entire page will be covered after loading is displayed in Android system.
So, please use the return key to turn off loading box.</p>
<a href="javascript:void(0)" class="btn show">Display loading</a>
<a href="javascript:void(0)" class="btn delay">Delay the loading display by 2 seconds</a>
<a href="javascript:void(0)" class="btn notext">Loading without text</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.show').addEventListener('click', function() {
    AlipayJSBridge.call('showLoading', {
      text: 'Loading',
    });
    setTimeout(function() {
      AlipayJSBridge.call('hideLoading');
    }, 3000);
  });

  document.querySelector('.delay').addEventListener('click', function() {
    AlipayJSBridge.call('showLoading', {
      text: 'Loading',
      delay: 2000,
    });
    setTimeout(function() {
      AlipayJSBridge.call('hideLoading');
    }, 5000);
  });

  document.querySelector('.notext').addEventListener('click', function() {
    AlipayJSBridge.call('showLoading', {
      text: ' ',
    });
    setTimeout(function() {
      AlipayJSBridge.call('hideLoading');
    }, 3000);
  });
});
</script>
```

API

```
AlipayJSBridge.call('showLoading',{
  text, delay
})
```

Input parameters

Name	Type	Description	Required	Default value
text	string	Text contents; To set no text, enter a space key.	N	“Loading”
delay	int	After how many seconds the loading box will be displayed. If `hideLoading` is called before, the loading will not be displayed.	N	0
autoHide	bool	By default, the container will automatically hide the loading box after `pageFinish`, it defaults to True. If it is set to false, auto-hiding will be turned off (for Android only).	N	true
cancelable	bool	Whether the Android return key can close the loading box. The physical return key can close the loading box by default (for Android only).	N	true

Attentions

- After loading box is displayed on Android, the entire page will be covered. Therefore please use the return key to turn off the loading.
- For iOS, when you didn’t set the text value, you can only click the title bar and toolbar. And no contents can be covered when there are texts. In the 9.9.5 version and above, this problem has been fixed.
- `showLoading` is in webview level. So, calling `hideLoading` on webview after `pushwindow` cannot turn off the loading of the previous webview. You must ensure that `showLoading` and `hideLoading` are executed in the same webview workspace.

1.5.5.12. Hide loading

The interface is used to hide the global loading box.

Use hideLoading interface

```
AlipayJSBridge.call('hideLoading');
```

Code sample

Show/hide global loading box:

```
<h1>Click the buttons below to see the different effects</h1>
<p>Note that the entire page will be covered after loading is displayed on Android system.
So, please use the return key to turn off the loading.</p>
<button class="btn show">Display loading</button>
<button class="btn delay">Delay the loading display by 2 seconds</button>
<button class="btn notext">Loading without text</button>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.show').addEventListener('click', function() {
    AlipayJSBridge.call('showLoading', {
      text: 'Loading',
    });
    setTimeout(function() {
      AlipayJSBridge.call('hideLoading');
    }, 3000);
  });

  document.querySelector('.delay').addEventListener('click', function() {
    AlipayJSBridge.call('showLoading', {
      text: 'Loading',
      delay: 2000,
    });
    setTimeout(function() {
      AlipayJSBridge.call('hideLoading');
    }, 5000);
  });

  document.querySelector('.notext').addEventListener('click', function() {
    AlipayJSBridge.call('showLoading', {
      text: ' ',
    });
    setTimeout(function() {
      AlipayJSBridge.call('hideLoading');
    }, 3000);
  });
});
</script>
```

1.5.5.13. Show title loading

This interface is used to display the loading box in the title bar.

Use showTitleLoading interface

```
AlipayJSBridge.call('showTitleLoading');
```

Code sample

Show/hide global loading box:

```
<h1>Click the buttons below to check the effects</h1>
<a href="javascript:void(0)" class="btn show">Display loading</a>
<a href="javascript:void(0)" class="btn hide">Hide loading</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.show').addEventListener('click', function() {
    AlipayJSBridge.call('showTitleLoading');
  });

  document.querySelector('.hide').addEventListener('click', function() {
    AlipayJSBridge.call('hideTitleLoading');
  });
});
</script>
```

1.5.5.14. Hide title loading

The interface is used to hide the loading box of title bar.

Use hideTitleLoading interface

```
AlipayJSBridge.call('hideTitleLoading');
```

Code sample

Show/hide global loading box:

```
<h1>Click the buttons below to check the effect</h1>
<a href="javascript:void(0)" class="btn show">Display loading</a>
<a href="javascript:void(0)" class="btn hide">Hide loading</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.show').addEventListener('click', function() {
    AlipayJSBridge.call('showTitleLoading');
  });

  document.querySelector('.hide').addEventListener('click', function() {
    AlipayJSBridge.call('hideTitleLoading');
  });
});
</script>
```

1.5.6. Tool class

1.5.6.1. Check App

This interface is used to check app availability.

Use checkApp interface

```
AlipayJSBridge.call('checkApp', {
  appId: '20000042',
}, function(result) {
  console.log(result.status);
  if (result && result.status == 'installed') {
    // App is available
  }
});
```

Code sample

Check App availability:

```
<h1>Enter the App ID to check if the App exists</h1>
<p>Note that the performance of `checkApp` API is inconsistent in Android and iOS. Therefo
re, it is not suggested to use it.</p>

<input class="app" type="text" value="20000067">

<a href="javascript:void(0)" class="btn check">Detection</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.check').addEventListener('click', function() {
    var appId = document.querySelector('.app').value;
    AlipayJSBridge.call('checkApp', {
      appId: appId
    }, function(e) {
      alert ('check result of' + appId + ':'  + JSON.stringify(e))
    });
  });;
});
</script>
```

API

```
AlipayJSBridge.call('checkApp',{
  appId, stageCode
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value	Version
appId	string	App ID	Y		

Name	Type	Description	Required	Default value	Version
stageCode	string	Stage code,"parentStageCode": "marketStage", "stageCode": "homeStage", "stageCode": "recommend", "stageCode": "livingConvenience", "stageCode": "capitalTransactions", "stageCode": "shoppingEntertainment", "stageCode": "financialManagement", "stageCode": "educationWelfare", "stageCode": "othersStage"	N		
fn	function	Callback function	N		

Output parameters

Parameter results brought in by callback function: {exist, status, extStatus, version, type}.

Name	Type	Description	Comment
exist	bool	Whether or not the app exists	App existing does not indicate that it is available, as it may be offline
status	string	Value: installed, uninstalled	Installed status indicates the App is available
extStatus	string	Online; uninstall; installing; offline	Supported in the version 8.5 or higher. This field is only available for H5 Apps
version	string	Version of target App	This field is only available for H5 Apps

Name	Type	Description	Comment
type	string	Value is microApp	This return field is only available for native Apps)

Attentions

- The performance of `checkApp` API is inconsistent in Android and iOS. Therefore, it is not suggested to use it.

1.5.6.2. Get startup parameters

This interface is only used to obtain the startup parameters passed when opening the offline package. For parameters passed by `pushWindow`, please use `AlipayJsBridge.startupParams` to obtain.

Use getStartupParams interface

```
AlipayJSBridge.call('getStartupParams', {
  key: ['url','xxx'] // Optional; filter the returned results according to the key value.
If it is left empty, the results will be returned completely
}, function(result) {
  console.log(result);
});
```

Code sample

```
<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}

ready(function() {
  // With key parameter
  AlipayJSBridge.call('getStartupParams', {
    key: ['url']
  }, function(result) {
    alert(JSON.stringify(result));
  });

  // Without key parameter
  AlipayJSBridge.call('getStartupParams', function(result) {
    alert(JSON.stringify(result));
  });
});
</script>
```

API

```
getStartupParams
```

Input parameter

Name	Type	Description	Required	Default value
key	Array	Obtain the value of the corresponding key based on the passed key	N	null

Output parameters

Return the corresponding startup parameters, such as: `{url: 'https://taobao.com', xx: 'Other startup parameter'}` .

- If there are no input parameters, all `startupParams` parameters will be returned.
- If there are input parameters, return the corresponding values based on the input parameters.
- If there is no corresponding key value in the startup parameters, the key is not included in the returned values, and no error is reported.

Error code (Number type)

Error code	Description
2	Parameter exception; key is an empty array or other type.
12	Unknown errors

1.5.6.3. Snapshot

This interface is used for snapshot.

Use snapshot interface

```
AlipayJSBridge.call('snapshot', function(result) {
  console.log(result.success);
});
```

Code sample

```
<h1>Click the buttons below to see different snapshot effects</h1>

<a href="javascript:void(0)" class="btn screen">Capture screen and save to the Gallery</a>
<a href="javascript:void(0)" class="btn viewport">Viewport snapshot returns fileURL</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.screen').addEventListener('click', function() {
    AlipayJSBridge.call('snapshot', function(result) {
      alert(JSON.stringify(result));
    });
  });

  document.querySelector('.viewport').addEventListener('click', function() {
    AlipayJSBridge.call('snapshot', {
      range: 'viewport',
      dataType: 'fileURL',
      saveToGallery: false
    }, function(result) {
      alert(JSON.stringify(result));
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('snapshot', {
  range, saveToGallery, dataType, imageFormat, quality,
  maxWidth, maxHeight
}, fn)
```

Input parameters

Name	Type	Description	Required	Default value

Name	Type	Description	Required	Default value
range	String	Snapshot range: Screen: The whole screen of the current client Viewport: Visible area in the webpage Document: The entire webpage. Note: document captures all webpages. On Android, the browser memory will overflow if the webpage is too large, so please use Screen	N	screen
saveToGallery	bool	Whether or not to save to Gallery	N	true
dataType	String	Result data format: dataURL: Base64-encoded image data fileURL: The URL of image in the file system (the image is stored in the temporary directory and will be cleared when you quit the App) none: No data is returned (for saving to the Gallery)	N	none
imageFormat	String	jpg / png	N	“jpg”
quality	int	The quality of JPG image, with value from 1 to 100	N	75
maxWidth	int	The maximum width of image, which will be reduced by equal ratio if too large	N	Unlimited
maxHeight	int	The maximum height of image, which will be reduced by equal ratio if too large	N	Unlimited

Name	Type	Description	Required	Default value
fn	function	Callback function	N	

Output parameters

Parameter result brought in by the callback function: {success, fileUrl, dataURL}.

Name	Type	Description
success	bool	Whether the processing is successful
fileUrl	String	The URL of image in the file system
fileUrl	String	Base64-encoded image data

Error code

Error code	Description
10	Failed to save Gallery
11	Failed to save image file

1.5.6.4. Call the RPC API

Note: Since the JSON data transferred by JS cannot contain the data type, errors may occur due to the data type when the transferred data is converted into a dictionary at the Native layer. We recommend that precise numeric values be transferred as strings. For example, `{"value":9.45}` will be converted into `{"value":9.449999999999999}` at the Native layer and then uploaded to the server. Instead, using `{"value":"9.45"}` can avoid such error.

RPC API usage instruction

```
AlipayJSBridge.call('rpc', {
  operationType: 'alipay.client.xxxx',
  requestData: [],
  headers: {}
}, function(result) {
  console.log(result);
});
```

Code sample

```
<h1>Click the button to initiate an RPC request.</h1>

<a href="javascript:void(0)" class="btn rpc">Initiate a request.</a><br/>
<a href="javascript:void(0)" class="btn rpcHeader">Initiate a request with a response head
er returned.</a>

<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.rpc').addEventListener('click', function() {
    AlipayJSBridge.call('rpc', {
      operationType: 'alipay.client.xxxx',
      requestData: [],
      headers: {}
    }, function(result) {
      alert(JSON.stringify(result));
    });
  });

  document.querySelector('.rpcHeader').addEventListener('click', function() {
    AlipayJSBridge.call('rpc', {
      operationType: 'alipay.client.xxxx',
      requestData: [],
      headers: {},
      getResponse: true
    }, function(result) {
      alert(JSON.stringify(result));
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('rpc', {
  operationType:,
  requestData:,
  headers
}, fn);
```

Input parameters

Name	Type	Description	Required	Default value

operationType	string	RPC service name.	Y	-
requestData	array	Parameter of the RPC request. You need to build the parameter based on the specific RPC API.	N	-
headers	object	Headers set for the RPC request. headers	N	{}
gateway	string	Gateway address.	N	Alipay gateway
compress	boolean	Whether request gzip compression is supported.	N	true
disableLimitView	boolean	Whether to prevent the unified traffic-limiting window from popping up when the RPC gateway is subject to throttling.	N	false
timeout	int	RPC timeout duration, in seconds.	N	Timeout durations are set by the framework in a unified manner and the policy is complex. Specifically, the timeout duration is set to 20s in a Wi-Fi environment for iOS and 30s in other environments. For Android, the timeout duration is set to a value between 12s and 42s in a Wi-Fi or 4G environment, and to a value between 32s and 60s in other environments.

getResponse	boolean	Whether to obtain the RPC response header (note: if it is set to true, the response data is nested by one more layer and can be used to obtain the trace ID/entity ID during data backflow reporting).	N	false
fn	function	Callback function.	N	-

Output parameters

Parameter carried in the callback function: result: {error }

Name	Type	Description
error	string	Error code

Error code description

Error code	Description
10	Network error
11	Request timeout
Others	Defined by the Mobile Gateway

Native RPC error codes

Error code	Description
1000	Success
0	Unknown error
1	The client cannot find the communication object.
2	Network access is unavailable on the client (the error code is converted to 10 in JSAPI).

3	The client certificate is incorrect.
4	The network connection on the client times out.
5	The speed of the network connection on the client is too low.
6	The server does not give a response upon a request from the client.
7	A network IO error occurs on the client.
8	A network request scheduling error occurs on the client.
9	A processing error occurs on the client.
10	A data deserialization error occurs on the client, or the data format on the server is incorrect.
11	Logging in to the client failed.
12	The login account on the client is changed.
13	The request is interrupted. For example, the network request will be interrupted when the thread is interrupted.
14	A network cache error occurs on the client.
15	A network authorization error occurs on the client.
16	A DNS resolution error occurs.
17	operationType is not on the whitelist.
1001	The access is denied.
1002	The call times exceed the limit: The system is busy. Please try again later.
2000	Login has timed out. Please log in again.

3000	No operation type is present, or the operation type is not supported.
3001	The requested data is empty: The system is busy. Please try again later.
3002	The data format is incorrect.
4001	The service request has timed out. Please try again later.
4002	An exception occurs in remotely calling the service system: The network is busy. Please try again later.
4003	Creating a remote call agent failed: The network is busy. Please try again later.
5000	Unknown error: Sorry. Operations are not allowed for now. Please try again later.
6000	The RPC service is not found.
6001	the RPC target method is not found.
6002	The RPC parameter quantity is incorrect.
6003	The RPC target method is not accessible.
6004	An RPC JSON parsing exception occurs.
6005	RPC parameters are invalid when the target method is called.
6666	An exception occurs on the RPC service.
7000	No public key is set.
7001	The parameters for signature verification are insufficient.
7002	Signature verification failed.

7003	Signature verification timestamp check failed.
7004	The operationType parameter in the signature verification RPC API is empty.
7005	The productId parameter is empty.
7006	Signature verification API: The did parameter is empty.
7007	Signature verification API: The request sending time parameter t is empty.
7008	Signature verification API: The IMEI (client device identifier) parameter is empty.
7009	Signature verification API: The IMSI (client user identifier) parameter is empty.
7010	Signature verification API: The API version number is empty.
7011	Signature verification API: The user is not authorized.
7012	Signature verification API: The RPC API is not opened up.
7013	Signature verification API: The product ID is not registered, or no key is obtained.
7014	Signature verification API: The signature data is empty.
7015	Signature verification API: The subscription is invalid.
7016	Signature verification API: The transferred sid in the login request RPC API is empty.
7017	Signature verification API: The transferred sid in the login request RPC API is invalid.
7018	Signature verification API: The transferred token in the login request RPC API is invalid.

7019	Signature verification API: The alipayuserid obtained by the login request RPC API is empty.
8001	etag: The response data has no changes.

RPC custom gateway

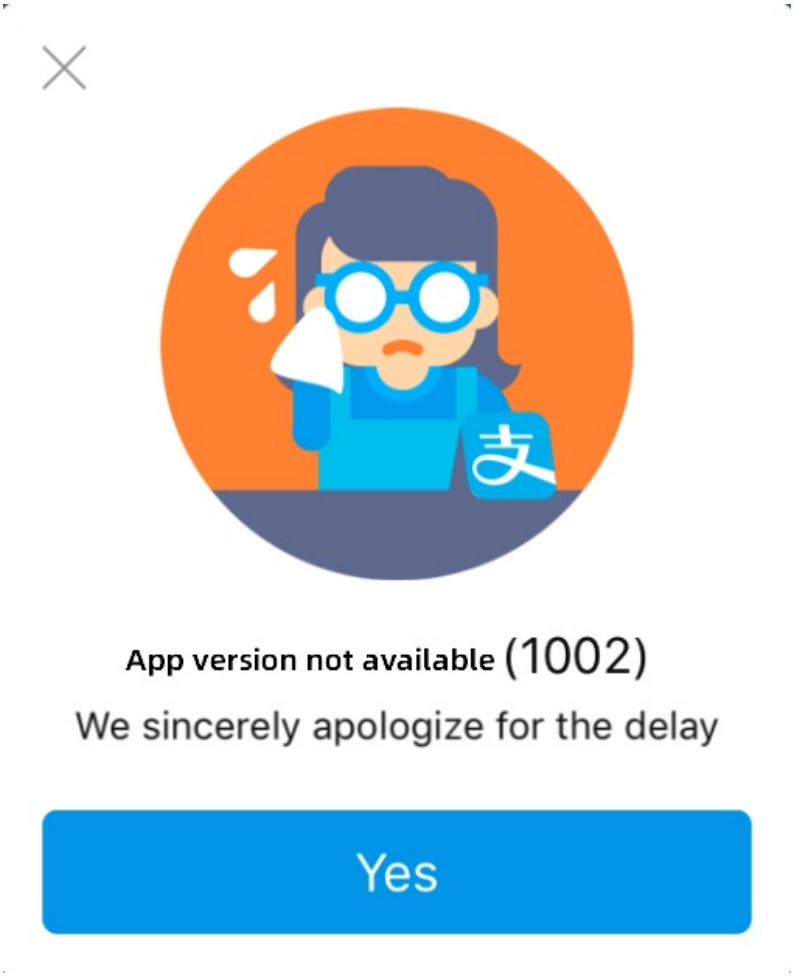
You can specify the gateway address to send request to in the RPC.

RPC throttling logic

Container version	disableLimitView	Action	Callback parameter
<=9.9.5	true	Silent	1002
<=9.9.5	false	Alert	1002
>=9.9.6	true	Silent	1002
>=9.9.6	false	Gateway handling	100201

Action Type	Description
Silent	None
Alert	A unified throttling box is displayed, as shown in the following figure.
Toast	A system toast is displayed. If the user exits the system, no toast is displayed.
Gateway handling	The action is silent, alert, or toast, depending on the RPC configuration for the gateway.

RPC throttling pop-up box



FAQ

Q: How to handle `ESLint: 'AlipayJSBridge' is not defined` ?

A: For questions not defined by AlipayJSBridge, try the following two solutions :

- Solution 1:

  ```
  window.AlipayJSBridge.call('rpc');
  ```

- Solution 2:

  ```
  const { AlipayJSBridge } = window;
  AlipayJSBridge.call('rpc');
  ```

1.5.6.5. Event tracking

The interface is the most primitive event tracking interface that can be used by the front end.

Use remoteLog interface

```
AlipayJSBridge.call('remoteLog', {
  bizType: "Nebula",    // Business type
  logLevel: 1,          // 1 - high, 2 - medium, 3 - low
  actionId: "event",    // Event tracking type, fixed to "event"
  seedId: "Login",      // Unique identifier of event tracking
  param1: "",
  param2: "",
  param3: "",
  param4:  {key1:"value1",key2:"value2"}, // Custom parameter
});
```

Note

- If you need to add a custom buried point parameter, you can add it to the param4 of the above code in the format of `key:"value"`, such as: `key1:"value1"`.
- When adding multiple custom buried point parameters, the format of the content added in param4 is as follows: `param4: "key1:"value1",key2:"value2",key3:"value3"`.

Code example

```
<h1>Clicking on the button records the relevant information</h1>

<a href="javascript:void(0)" class="btn read">Click</a>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('.read').addEventListener('click', function() {
    AlipayJSBridge.call('remoteLog', {
      type: "behavior",
      bizType: "Nebula",
      spmId: "a1.b2",
      logLevel: 1, // 1 - high, 2 - medium, 3 - low
      actionId: "event"
      seedId: "xxx",
      param1: "xxx",
      param2: "xxx",
      param3: "xxx",
      param4: "xxx",
    });
  });
});
</script>
```

API

```
AlipayJSBridge.call('remoteLog', {
  type, seedId, ucId, bizType, logLevel,
  actionId, spmId, param1, param2, param3, param4
});
```

Input parameters

Name	Type	Description	Required	Default value
type	String	Type of event tracking: monitor (monitoring type), monitorWithLocation (monitoring type, automatically attached with latitude and longitude in param4), behavior (behavior type), behaviorAuto (automatic behavior type), performance (performance type), error (exception type, supported since version 9.6.8), 135 (135 related service, supported since version 9.9).	N	“monitor”
seedId	String	Event tracking ID	Y	“”
ucId	String	Use case ID	N	“”
bizType	float	Business type identifier; when the value is passed by the parameter, a separate log file is generated.	N	“”
logLevel	int	1-high, 2-medium, 3-low; low level logs may be subject to traffic limitation.	N	“”

Name	Type	Description	Required	Default value
actionId	String	Event tracking type, fixed to “event”.	Y	“”
spmId	String	SPM encoding. Ignore seedId when spmId is programmed.	Y	“”
param1	String	Event tracking parameter 1	N	“”
param2	String	Event tracking parameter 2	N	“”
param3	String	Event tracking parameter 3	N	“”
param4	String	Event tracking parameter 4	N	“”

1.5.6.6. Set AP data

`setAPDataStorage` interface is used to save a String to the client unified storage; the length of the string is cannot exceed 200×1024.

> **Note**
> - The implementation of underlying storage service components is inconsistent in iOS and Android. The Android unified storage component does not support `type=user` attribute. In order to be consistent with the front end interface, when `type=user` is set, `key=key + "_" +MD5 (userId + userId + userId)` is set and stored in the underlying of Android. Key must also be processed accordingly when the service is obtained via client.
> - In the 10.1.60 and lower baselines, the client needs to do adaptation work to enable the interface to obtain the userId, otherwise the storage interface will not be able to distinguish storage by userId, see below Implement H5LoginProvider interface.
> - In the 10.1.68 and above baselines, userId uses the value in `MPLogger.setUserId` by default. If H5LoginProvider is implemented, H5LoginProvider is used.

Implement H5LoginProvider interface

Android

Implement the `H5LoginProvider` interface and set the instance class to `H5ProviderManager`.

Code example

```
package com.mpaas.nebula.provider;

import android.os.Bundle;

import com.alipay.mobile.common.logging.api.LoggerFactory;
import com.alipay.mobile.nebula.provider.H5LoginProvider;

public class H5LoginProviderImpl implements H5LoginProvider {
    // Other codes omitted

    @Override
    public String getUserId() {
        // This method returns userId
        return LoggerFactory.getLogContext().getUserId();
    }

    // Other codes omitted
}
```

Set H5LoginProvider

```
H5Utils.setProvider(H5LoginProvider.class.getName(), new H5LoginProviderImpl());
```

Use setAPDataStorage interface

```
AlipayJSBridge.call('setAPDataStorage', {
  type: "common",
  business: "customBusinessKey",
  key: "customKey",
  value: "customValue"
}, function(result) {
  alert(JSON.stringify(result));
});
```

Code example

```
<button id="J_saveDataBtn" class="btn">Save data</button>
<button id="J_getDataBtn" class="btn">View data</button>
<button id="J_removeDataBtn" class="btn">Delete data</button>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('#J_saveDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('setAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey",
      value: "customValue"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);

  document.querySelector('#J_getDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('getAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);

  document.querySelector('#J_removeDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('removeAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);
}, false);
</script>
```

API

```
AlipayJSBridge.call('setAPDataStorage', {
  type, business, key, value
});
```

Input parameters

Name	Type	Description	Required	Default value	Version
type	string	(user/common) User dimension storage or common storage; it defaults to common	N	'common'	
business	string	The customized service identifier, which can be agreed with the corresponding access code of the client. It defaults to NebulaBiz	N	''	
key	string	Key of custom data	Y	''	
value	string	Values to be stored; only the string type is supported. JSON data must be stringified first	Y	''	

Output parameter

Parameter result brought in by callback function: {success}.

Name	Type	Description
success	bool	Whether the saving is successful

Error code

Error code	Description
11	String length exceeds limit

1.5.6.7. Obtain AP data

You can call this operation to obtain data from Data Center and supports only the string type.

> **Note** If the data obtained on the iOS App end is stored directly on the native App end , make sure to use setString to store the data, otherwise the Get API data interface will fail.

getAPDataStorage API usage instruction

```
AlipayJSBridge.call('getAPDataStorage', {
  type: "common",
  business: "customBusinessKey",
  key: "customKey",
}, function(result) {
  alert(JSON.stringify(result));
});
```

Code sample

```
<button id="J_saveDataBtn" class="btn">Save data</button>
<button id="J_getDataBtn" class="btn">View data</button>
<button id="J_removeDataBtn" class="btn">Delete data</button>

<script>
function ready(callback) {
  // Call JS Bridge if it has been injected.
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Listen to the injection event if it has not been injected.
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('#J_saveDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('setAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey",
      value: "customValue"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);

  document.querySelector('#J_getDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('getAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);

  document.querySelector('#J_removeDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('removeAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);
}, false);
</script>
```

API

```
AlipayJSBridge.call('getAPDataStorage', {
  type, business, key
});
```

Input parameters

Name	Type	Description	Required	Default value
type	string	Storage type. Values include user and common, and the default value is common.	N	“common”
business	string	Custom service identifier, which can be written into the corresponding client access code. The default value is NebulaBiz. (In Android, this service identifier corresponds to `GROUP_ID` input when creating APSharedPreferences.)	N	“”
key	string	Key of custom data.	Y	“”

Output parameters

Parameter carried by the callback function: `result: {data}`

Name	Type	Description
Data	string	Data.
errorMessage	string	Data not found.

Error code description

Error code	Description
11	Data not found.

1.5.6.8. Remove AP data

This interface is used to remove data from unified storage.

Use removeAPDataStorage interface

```
AlipayJSBridge.call('removeAPDataStorage', {
  type: "common",
  business: "customBusinessKey",
  key: "customKey",
}, function(result) {
  alert(JSON.stringify(result));
});
```

Code example

```
<button id="J_saveDataBtn" class="btn">Save data</button>
<button id="J_getDataBtn" class="btn">View data</button>
<button id="J_removeDataBtn" class="btn">Delete data</button>

<script>
function ready(callback) {
  // Call directly if JSBridge has been injected
  if (window.AlipayJSBridge) {
    callback && callback();
  } else {
    // Monitor the injected events if JSBridge hasn't been injected
    document.addEventListener('AlipayJSBridgeReady', callback, false);
  }
}
ready(function() {
  document.querySelector('#J_saveDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('setAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey",
      value: "customValue"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);

  document.querySelector('#J_getDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('getAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);

  document.querySelector('#J_removeDataBtn').addEventListener('click', function(e) {
    AlipayJSBridge.call('removeAPDataStorage', {
      type: "common",
      business: "customBusinessKey",
      key: "customKey"
    }, function(result) {
      alert(JSON.stringify(result));
    });
  }, false);
}, false);
</script>
```

API

```
AlipayJSBridge.call('removeAPDataStorage', {
  type, business, key
});
```

Input parameters

Name	Type	Description	Required	Default value
type	String	(user/common) User dimension storage or common storage; it defaults to common	N	“common”
business	String	The customized service identifier, which can be agreed with the corresponding access code of client. It defaults to NebulaBiz	N	“”
key	String	Key of customized data	Y	“”

Output parameter

Parameter result brought in by callback function: {success}.

Name	Type	Description
success	bool	Whether it is successfully deleted

1.6. API Description

1.6.1. Android API reference

1.6.1.1. 10.1.68

This topic describes the API operations for the Android HTML5 container and offline package SDKs in mPaaS 10.1.68.

Common functions

H5TitleView

getTitle

- Declaration

  String getTitle();

- Description

  Queries the text of the main title.

- Parameters

  None.
- Returned value

  The value is the main title of the STRING type.

setTitle

- Declaration

  void setTitle(String title);
- Description

  Sets the text of the main title.
- Parameters

Parameters	Data type	Description
title	String	Title text

- Returned value

  None.

setSubTitle

- Declaration

  void setSubTitle(String subTitle);
- Description

  Sets the subtitle.
- Parameters

Parameters	Data type	Description
subTitle	String	The text of the subtitle.

- Returned value

  None.

setImgTitle

- Declaration

  void setImgTitle(Bitmap imgTitle);
- Description

  Sets the image icon of the title.
- Parameters

Parameters	Data type	Description

Parameters	Data type	Description
imgTitle	Bitmap	The information about the image icon.

- Returned value

  None.

setImgTitle

- Declaration

  void setImgTitle(Bitmap imgTitle,String contentDescription);
- Description

  Sets the image icon and content description of the title.
- Parameters

Parameters	Data type	Description
imgTitle	Bitmap	The information about the image icon.
contentDescription	String	The content description of the title.

- Returned value

  None.

showCloseButton

- Declaration

  void showCloseButton(boolean visible);
- Description

  Specifies whether to show the Close button.
- Parameters

Parameters	Data type	Description
visible	boolean	Specifies whether to show the Close button. Valid values: true: Shows the Close button. false: Hides the Close button.

- Returned value

  None.

getContentView

- Declaration

View getContentView();

- Description

  Queries the view of the title bar.

- Parameters

  None.

- Returned value

  View: the view of the title bar.

getContentBgView

- Declaration

  ColorDrawable getContentBgView();

- Description

  Queries the background of the title bar.

- Parameters

  None.

- Returned value

  ColorDrawable: the background of the title bar.

getMainTitleView

- Declaration

  TextView getMainTitleView();

- Description

  Queries the view of the main title.

- Parameters

  None.

- Returned value

  TextView: the view of the main title.

getSubTitleView

- Declaration

  TextView getSubTitleView();

- Description

  Queries the view of the subtitle.

- Parameters

  None.

- Returned value

  TextView: the view of the subtitle.

showBackButton

- Declaration

  void showBackButton(boolean visible);

- Description

  Specifies whether to show the Back button.

- Parameters

Parameters	Data type	Description
visible	boolean	Valid values: true: Shows the Back button. false: Hides the Back button.

- Returned value

  None.

showBackHome

- Declaration

  void showBackHome(boolean visible);

- Description

  Specifies whether to show the Home button.

- Parameters

Parameters	Data type	Description
visible	boolean	Valid values: true: Shows the Home button. false: Hides the Home button.

- Returned value

  None.

showOptionMenu

- Declaration

  void showOptionMenu(boolean visible);

- Description

  Specifies whether to show the upper-right menu.

- Parameters

Parameters	Data type	Description
visible	boolean	Valid values: true: Shows the upper-right menu. false: Hides the upper-right menu.

- Returned value

  None.

setOptionType

- Declaration

  void setOptionType(H5Param.OptionType type);

- Description

  Sets the display type of upper-right menu items.
- Parameters

Parameters	Data type	Description
type	H5Param.OptionType	The display type of the menu items.

- Returned value

  None.

setOptionType

- Declaration

  void setOptionType(H5Param.OptionType type, int num, boolean byIndex);
- Description

  Sets the display type of upper-right menu items.
- Parameters

Parameters	Data type	Description
type	H5Param.OptionType	The display type of the menu items.
num	int	Specifies the order of the specified icon from right to left. The value starts from 0.
byIndex	boolean	Specifies whether to set the display type of a specified menu item.

- Returned value

  None.

showTitleLoading

- Declaration

  void showTitleLoading(boolean visible);
- Description

  Specifies whether to show the loading status in the title bar. You can select an implementation method as needed.
- Parameters

Parameters	Data type	Description
visible	boolean	Specifies whether to show the loading status in the title bar.

- Returned value

  None.

setBtIcon

- Declaration

  void setBtIcon(Bitmap btIcon, int index);
- Description

  Sets the icon of a specified upper-right button.
- Parameters

Parameters	Data type	Description
btIcon	Bitmap	The icon of the button.
index	int	Specifies the order of the specified icon from right to left. The value starts from 0.

- Returned value

  None.

setH5Page

- Declaration

  void setH5Page(H5Page h5Page);
- Description

  Sets the HTML5 page of a container.
- Parameters

Parameters	Data type	Description
h5Page	H5Page	The HTML5 page.

- Returned value

  None.

setOptionMenu

- Declaration

  void setOptionMenu(JSONObject params);

- Description

  Sets the upper-right menu based on the parameters passed by JavaScript objects.
- Parameters

Parameters	Data type	Description
params	JSONObject	The parameters passed by JavaScript.

- Returned value

  None.

getDivider

- Declaration

  View getDivider();
- Description

  Queries the separator between the Back button and the title content. An empty value can be returned.
- Parameters

  None.
- Returned value

  View: the view of the separator.

getHdividerInTitle

- Declaration

  View getHdividerInTitle();
- Description

  Queries the separator between the title bar and the web page. The returned value cannot be empty.
- Parameters

  None.
- Returned value

  View: the view of the separator.

getPopAnchor

- Declaration

  View getPopAnchor();
- Description

  Queries the view of the anchor in the place whether the drop-down list appears.
- Parameters

  None.
- Returned value

  View: the view of the anchor in the place whether the drop-down list appears.

resetTitleColor

- Declaration

  void resetTitleColor(int color);
- Description

  Resets the background color of the title bar.
- Parameters

Parameters	Data type	Description
color	int	The value of the color.

- Returned value

  None.

releaseViewList

- Declaration

  void releaseViewList();
- Description

  Releases a referenced view. This operation is triggered when a container page is destroyed.
- Parameters

  None.
- Returned value

  None.

openTranslucentStatusBarSupport

- Declaration

  void openTranslucentStatusBarSupport(int color);
- Description

  Sets the color of a translucent title bar.
- Parameters

Parameters	Data type	Description
color	int	The value of the color.

- Returned value

  None.

setTitleTxtColor

- Declaration

  void setTitleTxtColor(int color);
- Description

  Sets the font color in the title bar.
- Parameters

Parameters	Data type	Description
color	int	The value of the color.

- Returned value

  None.

getOptionMenuContainer

- Declaration

  View getOptionMenuContainer();

- Description

  Queries the views of the upper-right menu. The ViewGroup parameter and its subclass must be returned.

- Parameters

  None.

- Returned value

  View: the views of the upper-right menu.

getOptionMenuContainer

- Declaration

  View getOptionMenuContainer(int index);

- Description

  Queries the views of the upper-right menu based on a specified position. The ViewGroup parameter and its subclass must be returned.

- Parameters

Parameters	Data type	Description
index	int	The position of the icon from right to left. The value starts from 0.

- Returned value

  View: the views of the upper-right menu.

setBackgroundAlphaValue

- Declaration

  void setBackgroundAlphaValue(int alpha);

- Description

  Sets the transparency of the background.

- Parameters

Parameters	Data type	Description
alpha	int	The value of the transparency.

- Returned value

  None.

setBackgroundColor

- Declaration

  void setBackgroundColor(int color);
- Description

  Sets the color of the background.
- Parameters

Parameters	Data type	Description
color	int	The value of the background color.

- Returned value

  None.

H5AppCenterPresetProvider

getCommonResourceAppList

- Declaration

  `Set<String> getCommonResourceAppList();`
- Declaration

  Queries the global resource packages.
- Parameters

  None.
- Returned value

  `Set<String>` : the set of global resource packages.

getH5PresetPkg

- Declaration

  H5PresetPkg getH5PresetPkg();
- Description

  Queries preset resource packages.
- Parameters

  None.
- Returned value

  H5PresetPkg: the preset resource packages for HTML5 pages.

getTinyCommonApp

- Declaration

  String getTinyCommonApp();

- Description

  Queries the appId of the common resource package of Mini Programs.

- Parameters

  None.

- Returned value

  The value is of the STRING type. It indicates the appId of the common resource package for Mini Programs.

H5Plugin

onPrepare

- Declaration

  void onPrepare(H5EventFilter filter);

- Description

  Register an HTML5 event filter in the preparation phase.

- Parameters

Parameters	Data type	Description
filter	H5EventFilter	The HTML5 plug-in for filtering events.

- Returned value

  None.

interceptEvent

- Declaration

  boolean interceptEvent(final H5Event event, final H5BridgeContext context);

- Description

  Intercepts an event.

- Parameters

Parameters	Data type	Description
event	H5Event	The HTML5 event.
context	H5BridgeContext	The bridge context of the HTML5 event.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The operation is successful. false: The operation failed.

handleEvent

- Declaration

  boolean handleEvent(final H5Event event, final H5BridgeContext context);

- Description

  Specifies whether to process an event.

- Parameters

Parameters	Data type	Description
event	H5Event	The read-only event that can be processed.
context	H5BridgeContext	The bridge context that is used to process some JSAPI-related events.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The plug-in has processed the event. false: The plug-in has not processed the event.

MPNebula

downloadApp

- Declaration

  public static void downloadApp(final String appId, final MpaasNebulaDownloadCallback mpaasNebulaDownloadCallback)

- Description

  Downloads an offline package.

- Parameters

Parameters	Data type	Description
appId	String	The ID of the offline package.
mpaasNebulaDownloadCallback	MpaasNebulaDownloadCallback	The callback function that triggers download.

- Returned value

  None.

installApp

- Declaration

  public static void installApp(final String appId, final MpaasNebulaInstallCallback mpaasNebulaInstallCallback)

- Description

  Installs an offline package.

- Parameters

Parameters	Data type	Description
appId	String	The ID of the offline package.
mpaasNebulaInstallCallback	MpaasNebulaInstallCallback	The callback function that triggers installation.

- Returned value

  None.

loadOfflineNebula

- Declaration

  public static void loadOfflineNebula(String jsonFileName, MPNebulaOfflineInfo... mpNebulaOfflineInfos)

- Description

  Loads a preset offline package.

- Parameters

Parameters	Data type	Description
jsonFileName	String	The preset offline package in JSON. This package can be downloaded in the console.
mpNebulaOfflineInfos	MPNebulaOfflineInfo	Information about the preset offline package.

- Returned value

  None.

registerH5Plugin

- Declaration

  public static void registerH5Plugin(String className, String bundleName, String scope, String[] events)

- Description

  Registers a custom HTML5 plug-in that is implemented based on the JSAPI.

- Parameters

Parameters	Data type	Description

Parameters	Data type	Description
className	String	The name of the class to which the plug-in belongs. The name must consist of the full path, including the package name and the class name.
bundleName	String	The name of the bundle where the plug-in is located. To view the bundle name, go to Main module/build/intermediates/bundle/META-INF/BUNDLE.MF.
scope	String	The scope of the plug-in. In general, the value is page.
events	String[]	The event of the registration operation.

- Returned value

  None.

enableAppVerification

- Declaration

  public static void enableAppVerification(final String publicKey)

- Description

  Enables key verification of offline packages. To make sure that the public key can be set, you must enable key verification of offline packages before you open the first offline package.

- Parameters

Parameters	Data type	Description
publicKey	String	The public key used for verification.

- Returned value

  None.

setCustomViewProvider

- Declaration

  public static void setCustomViewProvider(H5ViewProvider viewProvider)

- Description

  Sets custom views of a container, such as the title bar, menu bar, web layout, and pull-to-refresh views.

- Parameters

Parameters	Data type	Description
viewProvider	H5ViewProvider	The provider of custom views.

- Returned value

  None.

getH5View

- Declaration

  public static View getH5View(Activity activity, Bundle param)

- Description

  Queries the view of the HTML5 container.

- Parameters

Parameters	Data type	Description
activity	Activity	The page context.
param	Bundle	The startup parameters, which may include the app ID or URL.

- Returned value

  View: the view of the HTML5 container.

getH5ViewAsync

- Declaration

  public static void getH5ViewAsync(Activity activity, Bundle param, H5PageReadyListener h5PageReadyListener)

- Description

  Asynchronously queries the view of the HTML5 container.

- Parameters

Parameters	Data type	Description
activity	Activity	The page context.
param	Bundle	The startup parameters, which may include the app ID or URL.
h5PageReadyListener	H5PageReadyListener	The callback function that triggers asynchronous query.

- Returned value

None.

AbsTitleView

resetTitle

- Declaration

  public abstract void resetTitle();
- Description

  Resets the navigation bar.
- Parameters

  None.
- Returned value

  None.

H5ViewProvider

API operations related to custom views of full-screen HTML5 pages

createTitleView

- Declaration

  H5TitleView createTitleView(Context context);
- Description

  Creates the title of a custom title bar.
- Parameters

Parameters	Data type	Description
context	Context	The bridge context of the HTML5 event.

- Returned value

  H5TitleView: the title of the custom title bar.

createNavMenu

- Declaration

  public H5NavMenuView createNavMenu()
- Description

  Creates a custom navigation pane.
- Parameters

  None.
- Returned value

  H5NavMenuView: the custom navigation pane.

createPullHeaderView

- Declaration

  H5PullHeaderView createPullHeaderView(Context context, ViewGroup viewGroup);
- Description

Creates the header of the custom pull-to-refresh view.

- Parameters

Parameters	Data type	Description
context	Context	The bridge context of the HTML5 event.
viewGroup	ViewGroup	The ViewGroup control.

- Returned value

  H5PullHeaderView: the header of the custom pull-to-refresh view.

createWebContentView

- Declaration

  H5WebContentView createWebContentView(Context context);

- Description

  Creates the layout of WebView.

- Parameters

Parameters	Data type	Description
context	Context	The bridge context of the HTML5 event.

- Returned value

  H5WebContentView: the layout of the custom WebView.

ITinyOptionMenuView

getView

- Declaration

View getView();

- Description

  Queries the view of the current UI.

- Parameters

None.

- Returned value

  View: the view of the current UI.

setH5Page

- Declaration

void setH5Page(H5Page h5Page);

- Description

  Sets the HTML5 page of a container.

- Parameters

Parameters	Data type	Description
h5Page	H5Page	The HTML5 page.

- Returned value

  None.

setOptionMenuOnClickListener

- Declaration

void setOptionMenuOnClickListener(View.OnClickListener listener);

- Description

Sets a listener on events of tapping the option menu.

- Parameters

Parameters	Data type	Description
listener	View.OnClickListener	The event of the view that you want to listen to.

- Returned value

None.

setCloseButtonOnClickListener

- Declaration

void setCloseButtonOnClickListener(View.OnClickListener listener);

- Description

Sets a listener on events of tapping the Close button.

- Parameters

Parameters	Data type	Description
listener	View.OnClickListener	The event of the view that you want to listen to.

- Returned value

None.

setCloseButtonOnLongClickListener

- Declaration

void setCloseButtonOnLongClickListener(View.OnLongClickListener listener);

- Description

  Sets a listener on events of tapping and holding the Close button.

- Parameters

Parameters	Data type	Description
listener	View.OnLongClickListener	The event of the view that you want to listen to.

- Returned value

None.

H5Utils

setProvider

- Declaration

void setProvider(String name, Object provider);

- Description

  Sets the provider.

- Parameters

Parameters	Data type	Description
name	String	The name of the provider.
provider	Object	The object of the provider.

- Returned value

None.

AbsTinyOptionMenuView

onTitleChange

- Declaration

void onTitleChange(H5TitleView title);

- Description

  Responds to changes in the title bar.

- Parameters

Parameters	Data type	Description
title	H5TitleView	The title bar of HTML5 pages.

- Returned value

  None.

H5ReplaceResourceProvider

API operations related to dynamic loading of custom resources

getReplaceResourcesBundleName

- Declaration

  String getReplaceResourcesBundleName();

- Description

  Queries the name of the bundle where the title bar resources are located.

- Parameters

  None.

- Returned value

  The value is of the STRING type. The returned value indicates the name of the bundle where the title bar resources are located.

H5ErrorPageView

API operations related to custom network error pages

enableShowErrorPage

- Declaration

  boolean enableShowErrorPage(H5Page h5Page, APWebView view, String errorUrl, int statusCode, String errorMsg, String subErrorMsg, Bundle extInfo, Object extObj);

- Description

  Specifies whether to show a custom error page.

- Parameters

Parameters	Data type	Description
h5Page	H5Page	The page object.
view	APWebView	The WebView object.
errorUrl	String	The URL of the error.
statusCode	int	The error code.
errorMsg	String	The description of the error.
subErrorMsg	String	The subclass of the error description.
extInfo	Bundle	Extension information.
extObj	Object	The type of the extension.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: Shows the custom page. In this case, the errorPageCallback method is executed. false: Hides the custom page.

H5BridgeContext

API operations used by the HTML5 plug-in to return results to JSAPI requests

sendBridgeResult

- Declaration

  boolean sendBridgeResult(JSONObject data);

- Description

  Returns results to the JavaScript layer. The results are consumed each time an event is returned. To use the same BridgeContext to return an event multiple times, you must call the sendBridgeResultWithCallbackKept operation.

- Parameters

Parameters	Data type	Description
data	JSONObject	The data that you want to return to the JavaScript layer.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The operation is successful. false: The operation failed.

sendToWeb

- Declaration

  void sendToWeb(String action, JSONObject param, H5CallBack callback);

- Description

  Returns results to the JavaScript layer. You can add the callback parameter to the request.

- Parameters

Parameters	Data type	Description
action	String	The action that is performed to transfer data.
param	JSONObject	The data that you want to transfer to the JavaScript layer.
callback	H5CallBack	-

- Returned value

  None.

sendError

- Declaration

  boolean sendError(H5Event event, Error code);

- Description

  Sends error information.

- Parameters

Parameters	Data type	Description
event	H5Event	The HTML5 event.
code	Error	The information about the error.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The data is transferred. false: The data fails to be transferred.

sendSuccess

- Declaration

  void sendSuccess();
- Description

  Calls back an HTML5 page without returning values.
- Parameters

  None.
- Returned value

  None.

sendError

- Declaration

  void sendError(int error, String errorMessage);
- Description

  Customizes an error code and an error message to return.
- Parameters

Parameters	Data type	Description
error	int	The error code you want to return.
errorMessage	String	The error message you want to return.

- Returned value

  None.

Callback function

errorPageCallback

- Declaration

  void errorPageCallback(H5Page h5Page, APWebView view, String errorUrl, int statusCode, String errorMsg, String subErrorMsg, Bundle extInfo, Object extObj)
- Description

Calls back an error page.

- Parameters

Parameters	Data type	Description
h5Page	H5Page	The page object.
view	APWebView	The WebView object.
errorUrl	String	The URL of the error.
statusCode	int	The error code.
errorMsg	String	The description of the error.
subErrorMsg	String	The subclass of the error description.
extInfo	Bundle	Extension information.
extObj	Object	The type of the extension.

- Returned value

  None.

1.6.1.2. 10.1.60

This topic describes the API operations for the Android HTML5 container and offline package SDKs in mPaaS 10.1.60.

Common functions

H5TitleView

getTitle

- Declaration

  String getTitle();

- Description

  Queries the text of the main title.

- Parameters

  None.

- Returned value

  The value is the main title of the STRING type.

setTitle

- Declaration

void setTitle(String title);

- Description

  Sets the text of the main title.

- Parameters

Parameters	Data type	Description
title	String	Title text

- Returned value

  None.

setSubTitle

- Declaration

  void setSubTitle(String subTitle);

- Description

  Sets the subtitle.

- Parameters

Parameters	Data type	Description
subTitle	String	The text of the subtitle.

- Returned value

  None.

setImgTitle

- Declaration

  void setImgTitle(Bitmap imgTitle);

- Description

  Sets the image icon of the title.

- Parameters

Parameters	Data type	Description
imgTitle	Bitmap	The information about the image icon.

- Returned value

  None.

setImgTitle

- Declaration

  void setImgTitle(Bitmap imgTitle,String contentDescription);

- Description

  Sets the image icon and content description of the title.
- Parameters

Parameters	Data type	Description
imgTitle	Bitmap	The information about the image icon.
contentDescription	String	The content description of the title.

- Returned value

  None.

showCloseButton

- Declaration

  void showCloseButton(boolean visible);
- Description

  Specifies whether to show the Close button.
- Parameters

Parameters	Data type	Description
visible	boolean	Specifies whether to show the Close button. Valid values: true: Shows the Close button. false: Hides the Close button.

- Returned value

  None.

getContentView

- Declaration

  View getContentView();
- Description

  Queries the view of the title bar.
- Parameters

  None.
- Returned value

  View: the view of the title bar.

getContentBgView

- Declaration

  ColorDrawable getContentBgView();

- Description

  Queries the background of the title bar.
- Parameters

  None.
- Returned value

  ColorDrawable: the background of the title bar.

getMainTitleView

- Declaration

  TextView getMainTitleView();
- Description

  Queries the view of the main title.
- Parameters

  None.
- Returned value

  TextView: the view of the main title.

getSubTitleView

- Declaration

  TextView getSubTitleView();
- Description

  Queries the view of the subtitle.
- Parameters

  None.
- Returned value

  TextView: the view of the subtitle.

showBackButton

- Declaration

  void showBackButton(boolean visible);
- Description

  Specifies whether to show the Back button.
- Parameters

Parameters	Data type	Description
visible	boolean	Valid values: true: Shows the Back button. false: Hides the Back button.

- Returned value

  None.

showBackHome

- Declaration

  void showBackHome(boolean visible);
- Description

  Specifies whether to show the Home button.
- Parameters

Parameters	Data type	Description
visible	boolean	Valid values: true: Shows the Back button. false: Hides the Back button.

- Returned value

  None.

showOptionMenu

- Declaration

  void showOptionMenu(boolean visible);
- Description

  Specifies whether to show the upper-right menu.
- Parameters

Parameters	Data type	Description
visible	boolean	Valid values: true: Shows the upper-right menu. false: Hides the upper-right menu.

- Returned value

  None.

setOptionType

- Declaration

  void setOptionType(H5Param.OptionType type);
- Description

  Sets the display type of upper-right menu items.
- Parameters

Parameters	Data type	Description
type	H5Param.OptionType	The display type of the menu items.

- Returned value

None.

setOptionType

- Declaration

  void setOptionType(H5Param.OptionType type, int num, boolean byIndex);

- Description

  Sets the display type of upper-right menu items.

- Parameters

Parameters	Data type	Description
type	H5Param.OptionType	The display type of the menu items.
num	int	Specifies the order of the specified icon from right to left. The value starts from 0.
byIndex	boolean	Specifies whether to set the display type of a specified menu item.

- Returned value

  None.

showTitleLoading

- Declaration

  void showTitleLoading(boolean visible);

- Description

  Specifies whether to show the loading status in the title bar. You can select an implementation method as needed.

- Parameters

Parameters	Data type	Description
visible	boolean	Specifies whether to show the loading status in the title bar.

- Returned value

  None.

setBtIcon

- Declaration

  void setBtIcon(Bitmap btIcon, int index);

- Description

  Sets the icon of a specified upper-right button.

- Parameters

Parameters	Data type	Description
btIcon	Bitmap	The icon of the button.
index	int	Specifies the order of the specified icon from right to left. The value starts from 0.

- Returned value

  None.

setH5Page

- Declaration

  void setH5Page(H5Page h5Page);

- Description

  Sets the HTML5 page of a container.

- Parameters

Parameters	Data type	Description
h5Page	H5Page	The HTML5 page.

- Returned value

  None.

setOptionMenu

- Declaration

  void setOptionMenu(JSONObject params);

- Description

  Sets the upper-right menu based on the parameters passed by JavaScript objects.

- Parameters

Parameters	Data type	Description
params	JSONObject	The parameters passed by JavaScript.

- Returned value

  None.

getDivider

- Declaration

  View getDivider();

- Description

  Queries the separator between the Back button and the title content. An empty value can be returned.
- Parameters

  None.
- Returned value

  View: the view of the separator.

getHdividerInTitle

- Declaration

  View getHdividerInTitle();
- Description

  Queries the separator between the title bar and the web page. The returned value cannot be empty.
- Parameters

  None.
- Returned value

  View: the view of the separator.

getPopAnchor

- Declaration

  View getPopAnchor();
- Description

  Queries the view of the anchor in the place whether the drop-down list appears.
- Parameters

  None.
- Returned value

  View: the view of the anchor in the place whether the drop-down list appears.

resetTitleColor

- Declaration

  void resetTitleColor(int color);
- Description

  Resets the background color of the title bar.
- Parameters

Parameters	Data type	Description
color	int	The value of the color.

- Returned value

  None.

releaseViewList

- Declaration

void releaseViewList();

- Description

  Releases a referenced view. This operation is triggered when a container page is destroyed.

- Parameters

  None.

- Returned value

  None.

openTranslucentStatusBarSupport

- Declaration

  void openTranslucentStatusBarSupport(int color);

- Description

  Sets the color of a translucent title bar.

- Parameters

Parameters	Data type	Description
color	int	The value of the color.

- Returned value

  None.

setTitleTxtColor

- Declaration

  void setTitleTxtColor(int color);

- Description

  Sets the font color in the title bar.

- Parameters

Parameters	Data type	Description
color	int	The value of the color.

- Returned value

  None.

getOptionMenuContainer

- Declaration

  View getOptionMenuContainer();

- Description

  Queries the views of the upper-right menu. The ViewGroup parameter and its subclass must be returned.

- Parameters

  None.

- Returned value

  View: the views of the upper-right menu.

getOptionMenuContainer

- Declaration

  View getOptionMenuContainer(int index);

- Description

  Queries the views of the upper-right menu based on a specified position. The ViewGroup parameter and its subclass must be returned.

- Parameters

Parameters	Data type	Description
index	int	The position of the icon from right to left. The value starts from 0.

- Returned value

  View: the views of the upper-right menu.

setBackgroundAlphaValue

- Declaration

  void setBackgroundAlphaValue(int alpha);

- Description

  Sets the transparency of the background.

- Parameters

Parameters	Data type	Description
alpha	int	The value of the transparency.

- Returned value

  None.

setBackgroundColor

- Declaration

  void setBackgroundColor(int color);

- Description

  Sets the color of the background.

- Parameters

Parameters	Data type	Description
color	int	The value of the background color.

- Returned value

  None.

H5AppCenterPresetProvider

getCommonResourceAppList

- Declaration

```
Set<String> getCommonResourceAppList();
```

- Declaration

  Queries the global resource packages.

- Parameters

  None.

- Returned value

  `Set<String>`: the set of global resource packages.

getH5PresetPkg

- Declaration

  H5PresetPkg getH5PresetPkg();

- Description

  Queries preset resource packages.

- Parameters

  None.

- Returned value

  H5PresetPkg: the preset resource packages for HTML5 pages.

getTinyCommonApp

- Declaration

  String getTinyCommonApp();

- Description

  Queries the appId of the common resource package of Mini Programs.

- Parameters

  None.

- Returned value

  The value is of the STRING type. It indicates the appId of the common resource package for Mini Programs.

H5Plugin

onPrepare

- Declaration

  void onPrepare(H5EventFilter filter);

- Description

  Register an HTML5 event filter in the preparation phase.

- Parameters

Parameters	Data type	Description
filter	H5EventFilter	The HTML5 plug-in for filtering events.

- Returned value

  None.

interceptEvent

- Declaration

  boolean interceptEvent(final H5Event event, final H5BridgeContext context);

- Description

  Intercepts an event.

- Parameters

Parameters	Data type	Description
event	H5Event	The HTML5 event.
context	H5BridgeContext	The bridge context of the HTML5 event.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The operation is successful. false: The operation failed.

handleEvent

- Declaration

  boolean handleEvent(final H5Event event, final H5BridgeContext context);

- Description

  Specifies whether to process an event.

- Parameters

Parameters	Data type	Description
event	H5Event	The read-only event that can be processed.
context	H5BridgeContext	The bridge context that is used to process some JSAPI-related events.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The plug-in has processed the event. false: The plug-in has not processed the event.

H5ReplaceResourceProvider

API operations related to dynamic loading of custom resources

getReplaceResourcesBundleName

- Declaration

  String getReplaceResourcesBundleName();

- Description

  Queries the name of the bundle where the title bar resources are located.

- Parameters

  None.

- Returned value

  The value is of the STRING type. The returned value indicates the name of the bundle where the title bar resources are located.

H5ErrorPageView

API operations related to custom network error pages

enableShowErrorPage

- Declaration

  boolean enableShowErrorPage(H5Page h5Page, APWebView view, String errorUrl, int statusCode, String errorMsg, String subErrorMsg, Bundle extInfo, Object extObj);

- Description

  Specifies whether to show a custom error page.

- Parameters

Parameters	Data type	Description
h5Page	H5Page	The page object.
view	APWebView	The WebView object.
errorUrl	String	The URL of the error.
statusCode	int	The error code.
errorMsg	String	The description of the error.
subErrorMsg	String	The subclass of the error description.
extInfo	Bundle	Extension information.
extObj	Object	The type of the extension.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: Shows the custom page. In this case, the errorPageCallback method is executed. false: Hides the custom page.

H5BridgeContext

API operations used by the HTML5 plug-in to return results to JSAPI requests

sendBridgeResult

- Declaration

  boolean sendBridgeResult(JSONObject data);

- Description

  Returns results to the JavaScript layer. The results are consumed each time an event is returned. To use the same BridgeContext to return an event multiple times, you must call the sendBridgeResultWithCallbackKept operation.

- Parameters

Parameters	Data type	Description
data	JSONObject	The data that you want to return to the JavaScript layer.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The operation is successful. false: The operation failed.

sendToWeb

- Declaration

  void sendToWeb(String action, JSONObject param, H5CallBack callback);

- Description

  Returns results to the JavaScript layer. You can add the callback parameter to the request.

- Parameters

Parameters	Data type	Description
action	String	The action that is performed to transfer data.
param	JSONObject	The data that you want to transfer to the JavaScript layer.
callback	H5CallBack	-

- Returned value

  None.

sendError

- Declaration

boolean sendError(H5Event event, Error code);

- Description

  Sends error information.

- Parameters

Parameters	Data type	Description
event	H5Event	The HTML5 event.
code	Error	The information about the error.

- Returned value

  The value is of the BOOLEAN type. Valid values: true: The data is transferred. false: The data fails to be transferred.

sendSuccess

- Declaration

  void sendSuccess();

- Description

  Calls back an HTML5 page without returning values.

- Parameters

  None.

- Returned value

  None.

sendError

- Declaration

  void sendError(int error, String errorMessage);

- Description

  Customizes an error code and an error message to return.

- Parameters

Parameters	Data type	Description
error	int	The error code you want to return.
errorMessage	String	The error message you want to return.

- Returned value

  None.

Callback function

errorPageCallback

- Declaration

  void errorPageCallback(H5Page h5Page, APWebView view, String errorUrl, int statusCode, String errorMsg, String subErrorMsg, Bundle extInfo, Object extObj)

- Description

  Calls back an error page.

- Parameters

Parameters	Data type	Description
h5Page	H5Page	The page object.
view	APWebView	The WebView object.
errorUrl	String	The URL of the error.
statusCode	int	The error code.
errorMsg	String	The description of the error.
subErrorMsg	String	The subclass of the error description.
extInfo	Bundle	Extension information.
extObj	Object	The type of the extension.

- Returned value

  None.

1.7. Tutorial

1.7.1. Android tutorial - Native AAR

1.7.1.1. Overview

The HTML5 container is supported in the native AAR mode, the mPaaS Inside mode, and the component mode. If you want to connect to and use mPaaS as other SDKs, we recommend that you use the native AAR mode. The native AAR mode uses the native Android AAR-based packaging solution, and therefore is closer to the technology stack of Android developers. This access method frees you from the need to understand mPaaS-related packaging knowledge. You can integrate mPaaS into your projects by using the mPaaS Android Studio plug-in. This method reduces your access costs and makes it easier for you to get started with mPaaS.

To help you get familiar with and master the native AAR mode, this tutorial uses the native AAR mode as an example to provide guidance on how to connect to the HTML5 container component and use HTML5 offline packages.

This tutorial consists of the following parts:

1. Create an app in Android Studio.
2. Create an app in the mPaaS console.
3. Connect mPaaS to your projects in the native AAR mode.
4. Use the HTML5 container.
5. Use HTML5 offline packages.

What you will learn

- How to create an Android app that displays Toast when a button is tapped
- How to connect to mPaaS based on native AAR
- How to use the HTML5 container service of mPaaS
- How to use HTML5 offline package service of mPaaS

Prerequisites

1. The development environment is configured. In this tutorial, the development environment in Windows is taken as an example.
2. A web browser is installed. The Chrome browser is recommended.
3. An Android phone and the supporting data cable are prepared. The operating system version is Android 4.3 or later. You can also use a simulator for debugging. This tutorial uses a simulator as an example.

1.7.1.2. Create App in Android Studio

In this section, you will create an app that causes a Toast to appear when a user clicks a button, and obtain the installation package in APK format.

This process mainly includes the following four steps:

1. Create a project
2. Write the code
3. Create a signature file and add the signature to the project
4. Install the app on a cell phone

If you already have a native Android development project and it is already signed, you may skip this tutorial and directly Create an app on the mPaaS console.

Create a project

1. Start Android Studio and choose **File** > **New** > **New Project**.
2. In the **Create New Project** dialog box that appears, select **Empty Activity** and click **Next**.
3. Enter the **Name**, **Package name (the default value may be used)** , and **Save location**. Enter **H5 Application** as the **Name** here. Select **API 18: Android 4.3 (Jelly Bean)** as the **Minimum SDK**.

   > **Note**
   >
   > “API 18: Android 4.3 (Jelly Bean)” is the earliest version supported by mPaaS. You can select a version based on your needs in the actual production environment.

4. Click **Finish** to complete **creation of the project**.

Write the code

1. Open the file `activity_main.xml` and add a button by referring to the following code.

```
<Button
    android:id="@+id/button"
    android:layout_width="101dp"
    android:layout_height="50dp"
    android:layout_marginStart="142dp"
    android:layout_marginTop="153dp"
    android:layout_marginBottom="151dp"
    android:text="Button"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

2. Open the class `MainActivity` and add the click event for the button.

```
findViewById(R.id.button).setOnClickListener(new View.OnClickListener(){
        @Override
        public void onClick(View v) {
            Toast.makeText(MainActivity.this, "Hello mPaaS!",
Toast.LENGTH_SHORT).show();
        }
    });
```

3. After the compilation succeeds, you will have completed **writing the code**.

Create a signature file and add the signature to the project

1. In Android Studio, choose **Build** > **Generate Signed Bundle / APK**.
2. In the dialog box that appears, select **APK** and click **Next**.
3. Click **Create new**.
4. After entering all the required information, click **OK** and you will complete creating the signature. You can obtain the generated signature file in the specified **Key store path**.
5. After the contents are automatically entered, click **Next** to start adding the signature to the project.
6. Based on your needs, select **Build Variants** and then **V1 (Jar Signature)** as the signature version. Note that V1 (Jar Signature) is required while V2 (Full APK Signature) is optional.
7. Click **Finish**. After the packaging is complete, you will find the signed APK installation package for this app in the `debug` folder of the project directory (`~\MyHApplication\app\debug`). In this tutorial, the name of the installation package is `app-debug.apk` .

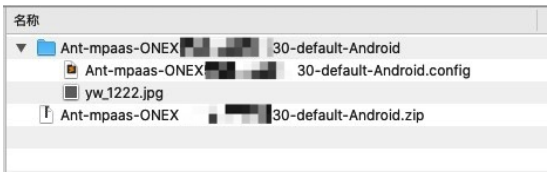
Install the app on a cell phone

1. Connect a cell phone to the computer and enable the USB Debugging mode of the cell phone.
2. Run the project.
3. Click **BUTTON**, pop up Toast, then it means the app was successfully installed and the expected function was implemented. Now, you have completed **installing the app on a cell phone**.

1.7.1.3. Create an application in the console

1. Start a web browser and log on to the mPaaS console.
2. Create an mPaaS app.
3. Enter a project name and click **OK**.

4. On the **App List** page, click **HTML5 Application** (**App name**).
5. On the **App Details** page, click **Configure now** to open the **Configure App** page.
6. On the **Configure App** page, click **Download now** to open the **Code Configuration** page.On the **Code Configuration** page, enter a **Package Name** (**package name of the app**) (for example, **com.example.h5application**), and upload the compiled and signed APK package. To learn how to quickly generate a signed APK, see Generate a signed APK for the console.
7. On the **Code Configuration** page, after you have entered all the information, click **Download Configuration** and you will obtain the configuration file of mPaaS.
8. The configuration file is a compressed file. This compressed file contains a `.config` file and `yw_1222.jpg`, which is an encryption image.



1.7.1.4. Integrate HTML5 Container into project

This topic describes how to connect mPaaS to a project in the native AAR mode.

1. In Android Studio, choose **mPaaS** > **Native AAR mode**.
2. In the dialog box that appears on the right side of the UI, click **Start Importing** under **Import App Configurations**.
3. In the **Import the configuration file of mPaaS** dialog box, select **I have downloaded the configuration file in the console and imported the file into the project.** Click **Next**.
4. Select the Configuration file that you download after the mPaaS app is created in the console. Click **Finish**.
5. The system sends a message that indicates that the configuration file is imported.
6. Click **Start Configuration** under **Access/upgrade the baseline** on the right side of the window.
7. In the **Select an mPaaS Baseline Version** dialog box that appears, select the baseline 10.1.68, click **OK**, and you will be able to access the mPaaS SDK.

> **Note**
>
> You can click **Start Configuration** again to upgrade the baseline.

8. Click **Start Configuration** under **Configure/update components** on the right side of the window.
9. From the list of components that appears, select **H5 Container**, click **OK**, and then you will be able to add the component H5 Container to the project.

Now, you have completed connecting the project to mPaaS by using the native AAR method.

1.7.1.5. Use HTML5 container

You can use the HTML5 container to perform the following operations:

- Open an online web page within an app
- Call the Native APIs on the front end
- Call a custom JSAPI on the front end
- Customize the title bar for an HTML5 page

Open an online web page within an app

1. Add a custom class `MyApplication`, which inherits from `Application`, to the project.
2. In the custom class `MyApplication`, perform initialization. The initialization method is shown in the following code:

```
@Override
public void onCreate() {
    super.onCreate();

    MP.init(this);
}
```

3. In the file `app/src/main/AndroidManifest.xml`, add `android:name=".MyApplication"`.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.h5application">

    <application
        android:name=".MyApplication"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

4. In the file `activity_main.xml`, set the style of the button again and change the id of the button to `start_url_btn`.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/start_url_btn"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dp"
        android:background="#108EE9"
        android:gravity="center"
        android:text="Start an Online Web Page"
        android:textColor="#ffffff"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

5. In the class `MainActivity`, rewrite the code for the clicking button event to implement the function to open the official website of Ant Group Financial Technology. The following code shows how this can be implemented:

```
findViewById(R.id.start_url_btn).setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View v) {
                MPNebula.startUrl("https://tech.antfin.com/");
            }
        });
```

6. Add the following configurations to `build.gradle(:app)` in the main module of the project.

```
apply plugin: 'com.android.application'
apply plugin: 'com.alipay.apollo.baseline.config'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.3"

    defaultConfig {
        applicationId "com.example.h5application"
        minSdkVersion 18
        targetSdkVersion 26
        ndk{
            abiFilters 'armeabi'
        }
        multiDexEnabled true
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation platform("com.mpaas.android:$mpaas_artifact:$mpaas_baseline")
    implementation fileTree(dir: "libs", include: ["*.jar"])
    implementation 'androidx.appcompat:appcompat:1.1.0'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    implementation 'com.mpaas.android:nebula'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.1'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'

}
```

7. After compiling the project, install the app on a cell phone.
8. You will open the home page of the official website of Ant Group Financial Technology within the app when you click the button. This means the API operation was successful. Now, you have completed **opening an online web page within an app** .

Call the Native APIs on the front end

1. When implementing a front-end page, through the bridge provided by the Nebula container, you can communicate with Native by using the JSAPI method to obtain the related information or data processed by Native. The Nebula container provides some basic preset JSAPIs. For more details, see Link. In a `.js` file of an HTML5 page, you can call these JSAPIs by using `AlipayJSBridge.call` . See the following sample code:

```
AlipayJSBridge.call('alert', {
  title: 'Native Alert Dialog',
  message: 'This is an Alert Dialog from Native.',
  Button: 'OK'
}, function (e) {
  alert("The button was tapped.");
});
```

> **Note**
>
> "https://mcube-prod.oss-cn-hangzhou.aliyuncs.com/570DA89281533-default/20200121/0.0.0.8_all/nebula/fallback/mPaaSComponentTestWebview.html" is a fully functional front-end page. You can use this page to try out calling the Native APIs on the front end.

2. In the file `activity_main.xml`, add a button and set the button id to `h5_to_native_btn`.

```
<Button
     android:id="@+id/h5_to_native_btn"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_marginTop="20dp"
     android:background="#108EE9"
     android:gravity="center"
     android:text="Call Native on the Front End"
     android:textColor="#ffffff"
     app:layout_constraintEnd_toEndOf="parent"
     app:layout_constraintHorizontal_bias="0.0"
     app:layout_constraintStart_toStartOf="parent"
     app:layout_constraintTop_toBottomOf="@+id/start_url_btn" />
```

3. In the class `MainActivity`, define the behavior after the button `h5_to_native_btn` is tapped, to implement the function of opening a front-end page. The following code shows how this can be implemented:

```
 findViewById(R.id.h5_to_native_btn).setOnClickListener(new View.OnClickListener() {
     @Override
     public void onClick(View v) {
         MPNebula.startUrl("https://mcube-prod.oss-cn-
hangzhou.aliyuncs.com/570DA89281533-
default/80000000/1.0.XX.XX_all/nebula/fallback/h5_to_native.html");
     }
 });
```

4. After compiling the project, install the app on a cell phone.
5. Tap the button to open the front-end page. Then tap the button **"Show Native Alert Dialog"** and a Native alert dialog box will appear. The title of this alert dialog box is **"Native Alert Dialog"** and the content of it is **"This is an Alert Dialog from Native."**. If you tap the **OK** button in this alert dialog box, another alert dialog box with no title will appear and its content is **"The button was tapped."**. This means the API operation was successful. Now, you have completed **calling the Native APIs on the front end** .

Call a custom JSAPI on the front end

1. Create a custom class `MyJSApiPlugin` to define a custom JSAPI.

```
package com.example.h5application;

import com.alibaba.fastjson.JSONObject;
import com.alipay.mobile.h5container.api.H5BridgeContext;
import com.alipay.mobile.h5container.api.H5Event;
import com.alipay.mobile.h5container.api.H5EventFilter;
import com.alipay.mobile.h5container.api.H5SimplePlugin;

public class MyJSApiPlugin extends H5SimplePlugin {
    private static final String API = "myapi";
    @Override
    public void onPrepare(H5EventFilter filter) {
        super.onPrepare(filter);
        filter.addAction(API);
    }
    @Override
    public boolean handleEvent(H5Event event, H5BridgeContext context) {
        String action = event.getAction();
        if (API.equalsIgnoreCase(action)) {
            JSONObject params = event.getParam();
            String param1 = params.getString("param1");
            String param2 = params.getString("param2");
            JSONObject result = new JSONObject();
            result.put("success", true);
            result.put("message", API + " with " + param1 + "," + param2 + " was handle
d by native.");
            context.sendBridgeResult(result);
            return true;
        }
        return false;
    }
}
```

2. Register the custom JSAPI named `MyJSApiPlugin` in the project. We recommend that you register it when the app starts. In this example, we register it in `MyApplication`.

```
public class MyApplication extends Application {
        @Override
        protected void attachBaseContext(Context base) {
            super.attachBaseContext(base);
            // Suggestion: Check if this is the main process. Initialize only in the
main process.
            QuinoxlessFramework.setup(this, new IInitCallback() {
                @Override
                public void onPostInit() {
                    // Start to use the mPaaS functions here.
        // Call registerCustomJsapi() to complete registering the custom JSAPI.
                    registerCustomJsapi();
                }
            });
        }

    @Override
    public void onCreate() {
        super.onCreate();
        QuinoxlessFramework.init();
    }

    private void registerCustomJsapi(){
        MPNebula.registerH5Plugin(
                // Class name of the plug-in.
                MyJSApiPlugin.class.getName(),
                // Just use an empty string.
                "",
                // Applicable context. Just use "page".
                "page",
                // Name of the registered JSAPI.
                new String[]{"myapi"});
    }
}
```

3. In a front-end page, call this custom JSAPI. See the following sample code:

```
AlipayJSBridge.call('myapi', {
  param1: 'JsParam1',
  param2: 'JsParam2'
}, function (result) {
  alert(JSON.stringify(result));
});
```

> **Note**
>
> “https://mcube-prod.oss-cn-hangzhou.aliyuncs.com/570DA89281533-default/80000001/1.0.XX.XX_all/nebula/fallback/custom_jsapi.html“ is a fully functional front-end page. You can use this page to try out calling a **custom JSAPI** on the front end.

4. In the file `activity_main.xml`, add a button and set the `button id` to `custom_jsapi_btn`.

```
<Button
android:id="@+id/custom_jsapi_btn"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="20dp"
android:background="#108EE9"
android:gravity="center"
android:text="Custom JSAPI"
android:textColor="#ffffff"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="1.0"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/h5_to_native_btn" />
```

5. In the class `MainActivity`, define the behavior after the button `custom_jsapi_btn` is tapped, to implement the function of calling a custom JSAPI on the front end. The following code shows how this can be implemented:

```
findViewById(R.id.custom_jsapi_btn).setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        MPNebula.startUrl("https://mcube-prod.oss-cn-
hangzhou.aliyuncs.com/570DA89281533-
default/80000001/1.0.XX.XX_all/nebula/fallback/custom_jsapi.html");
    }
});
```

6. After compiling the project, install the app on a cell phone.
7. Tap the **Custom JSAPI** button and a front-end page containing a button named **Custom JSAPI** will appear. If you tap the **Custom JSAPI** button on this page, an alert dialog box with no title will appear. This dialog box shows the handled parameters that were passed in while the API was called on the front end, based on the function defined by the custom API. Now, you have **called the custom JSAPI from the front end** .

Customize the title bar for an HTML5 page

The HTML5 container provides you with methods to configure a custom title bar. You can use the default title bar `MpaasDefaultH5TitleView` that is provided by mPaaS. You can rewrite some of these methods based on your needs. You can also implement `H5TitleView`. This section describes how to use `MpaasDefaultH5TitleView` to configure a title bar.

1. Construct an `H5ViewProvider` implementation class. Set your custom `H5TitleView` as the return result of the `createTitleView` method.

```
package com.example.h5application;

import android.content.Context;
import android.view.ViewGroup;

import com.alipay.mobile.nebula.provider.H5ViewProvider;
import com.alipay.mobile.nebula.view.H5NavMenuView;
import com.alipay.mobile.nebula.view.H5PullHeaderView;
import com.alipay.mobile.nebula.view.H5TitleView;
import com.alipay.mobile.nebula.view.H5WebContentView;
import com.mpaas.nebula.adapter.view.MpaasDefaultH5TitleView;

public class H5ViewProviderImpl implements H5ViewProvider {
    @Override
    public H5TitleView createTitleView(Context context) {
        return new MpaasDefaultH5TitleView(context);
    }
    @Override
    public H5NavMenuView createNavMenu() {
        return null;
    }
    @Override
    public H5PullHeaderView createPullHeaderView(Context context, ViewGroup viewGroup)
{
        return null;
    }
    @Override
    public H5WebContentView createWebContentView(Context context) {
        return null;
    }
}
```

2. In the `activity_main.xml` file, create a button and set the `button id` to `custom_title_btn`.

```
<Button
<Button
android:id="@+id/custom_title_btn_before"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="20dp"
android:background="#108EE9"
android:gravity="center"
android:text="Before Customizing a Title"
android:textColor="#ffffff"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/custom_jsapi_btn" />

<Button
android:id="@+id/custom_title_btn_after"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="20dp"
android:background="#108EE9"
android:gravity="center"
android:text="After Customizing a Title"
android:textColor="#ffffff"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.0"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/custom_title_btn_before" />
```

3. In the `MainActivity` class, specify the system behavior upon a tap on the `custom_title_btn` button: specify the custom View Provider for the container and open an online page. The following code shows how this can be implemented:

```
 findViewById(R.id.custom_title_btn_before).setOnClickListener(new
View.OnClickListener() {
     @Override
     public void onClick(View v) {
         MPNebula.startUrl("https://www.cloud.alipay.com/docs/2/49549");
     }
 });
 findViewById(R.id.custom_title_btn_after).setOnClickListener(new View.OnClickListener()
{
     @Override
     public void onClick(View v) {
         // Set a custom title. You only need to set the title for once.
         MPNebula.setCustomViewProvider(new H5ViewProviderImpl());
         // Start a URL. The title changes after the URL is started.
         MPNebula.startUrl("https://www.cloud.alipay.com/docs/2/49549");
     }
 });
```

4. After compiling the project, install the app on a cell phone.
5. Tap the **Before Customizing a Title** and **After Customizing a Title** buttons in order. The same online page appears after you tap either of the two buttons. You can find that both the color of the title bar and font color are changed. Now, you have **customized the title bar for an HTML5 page**.

Sample code

Click here to download the sample code in this topic.

1.7.1.6. Use HTML5 offline package

The usage of an HTML5 offline package involves four steps:

- Release an offline package
- Preset an offline package
- Start an offline package
- Update an offline package

To demonstrate the features of HTML5 offline packages, this topic describes all of the four steps. However, not every step is necessary for using an HTML5 offline package. In actual production, you can perform specific steps based on your needs.

Release an offline package

This section describes how to release an offline package.

Prerequisites

You need to prepare a ZIP package of your front-end app. Otherwise, you can download the sample offline package.

Procedure

1. Log on to the mPaaS console and configure the information about the offline package of your app. For more information, see Configure an offline package.
2. Generate the offline package of your front-end app. You can also use the sample offline package. For more information, see Generate an offline package.
3. In the mPaaS console, create an HTML5 Application and upload your offline package. For more information, see Create an offline package.
4. Release the configured offline package to your app on the client. For more information, see Release an offline package.

Preset an offline package

This section describes how to preset an offline package.

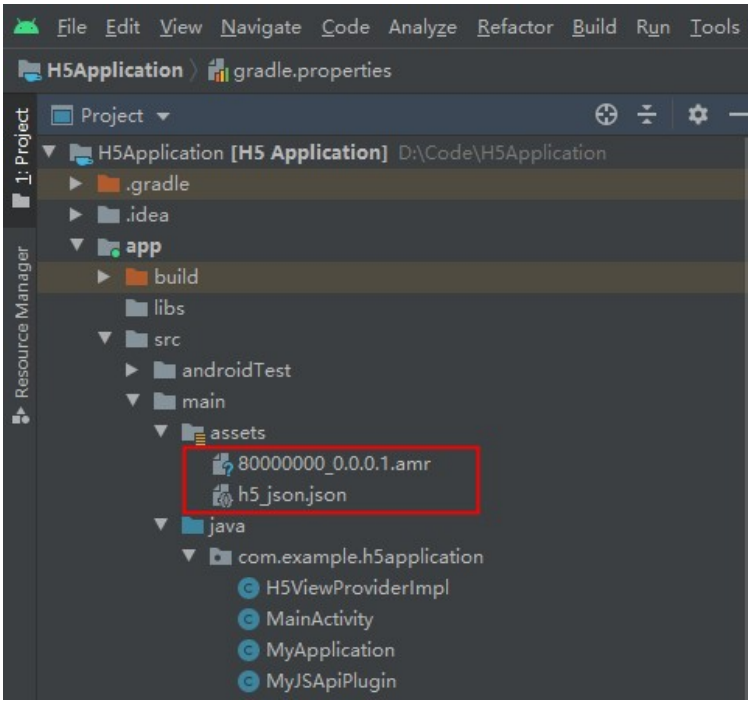
Prerequisites

Your offline package is released in the mPaaS console.

Procedure

1. In the mPaaS console, download the AMR file and the configuration file of the offline package to your local system.

2. Put the downloaded AMR file and the configuration file in the assets directory of your project.



3. Preset the offline package into your app. We recommend that you register the offline package during the startup of the app. In this tutorial, the offline package is registered in the `MyApplication` class. Now, you have preset the offline package.

```
public class MyApplication extends Application {

    @Override
    public void onCreate() {
        super.onCreate();

        MP.init(this,
            MPInitParam.obtain().setCallback(new MPInitParam.MPCallback() {
                @Override
                public void onInit() {
                    registerCustomJsapi();
                    loadOfflineNebula();
                }
            })
        );
    }

    private void loadOfflineNebula() {
        new Thread(new Runnable() {
            @Override
            public void run() {
            // This method is a blocking call. Please do not call the built-in offline p
ackage method on the main thread. If multiple amr packages are built in, make sure the f
ile exists. If it does not exist, other built-in offline packages will fail.
            // This method can only be called once. Only the first call is valid if
multiple calls are made.
            MPNebula.loadOfflineNebula("h5_json.json", new
MPNebulaOfflineInfo("80000000_1.0.0.0.amr", "80000000", "1.0.0.0"));
            }
        }).start();
    }
}
```

Start an offline package

This section describes how to start an offline package.

Prerequisite

Your offline package is preset on the client.

Procedure

1. In the activity_main.xml file, create a button and set the button id to `start_app_btn`.

```
<Button
    android:id="@+id/start_app_btn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:background="#108EE9"
    android:gravity="center"
    android:text="Start Offline Package"
    android:textColor="#ffffff"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/custom_title_btn_after" />
```

2. In the `MainActivity` class, specify the system behavior upon a tap on the `start_app_btn` button: start the offline package. In the following sample code, the parameter “80000000” is the app ID of the offline package.

```
findViewById(R.id.start_app_btn).setOnClickListener(new View.OnClickListener() {
 @Override
 public void onClick(View v) {
     MPNebula.startApp("80000000");
 }
});
```

3. After compiling the project, install and open the application on the phone.
4. Tap the **Start Offline Package** button. The web page that is preset in the offline package appears. Now, you have **started the offline package**.

Update an offline package

This section describes how to start an offline package.

Prerequisites

Your offline package is preset in the app on your client. A new HTML5 Application is created in the mPaaS console and a new offline package is uploaded.

Procedure

1. In the activity_main.xml file, create a button and set the button id to `update_app_btn`.

```
<Button
        android:id="@+id/update_app_btn"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:background="#108EE9"
        android:gravity="center"
        android:text="Update Offline Package"
        android:textColor="#ffffff"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/start_app_btn" />
```

2. In the `MainActivity` class, specify the system behavior upon a tap on the `update_app_btn` button: update the offline package. In the following sample code, the parameter “80000000” is the app ID of the offline package.

```
findViewById(R.id.update_app_btn).setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        MPNebula.updateAllApp(new MpaasNebulaUpdateCallback() {
            @Override
            public void onResult(final boolean success, final boolean isLimit) {
                // The "success?" in the following code indicates whether the update is
successful.
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(MainActivity.this, success ? "Offline package upda
te succeeded": "Offline package update failed", Toast.LENGTH_SHORT).show();
                    }
                });
            }
        });
    }
});
```

3. After compiling the project, install and open the application on the phone.
4. Tap the **Update Offline Package** button to open and update the offline package. After a prompt appears and indicates that the update is successful, tap the **Start Offline Package** button. The web page that is preset in the updated offline package appears. Now, you have **updated the offline package**.

Sample code

Click here to download the sample code in this topic.

1.7.2. Android tutorial-mPaaS Inside

1.7.2.1. Overview

mPaaS Inside is a new access method of mPaaS. This new access method is almost the same as the native AAR method. The difference is that the new method allows you to use the capabilities of mPaaS. mPaaS Inside is applicable to scenarios in which you do not need the component-based Bundle solution but urgently need mPaaS capabilities. We provide this tutorial to help you become familiar with this new access method and get started quickly.

This tutorial contains the following six parts:

1. Create an app in Android Studio.
2. Create an app in the mPaaS console.
3. Connect a native project to mPaaS Inside.
4. Add the HTML5 container component to the project.
5. Use the HTML5 container
6. Use HTML5 offline packages

What you will learn

- How to create an Android app in which a toast appears upon a tap on a button.
- How to connect to mPaaS Inside.
- How to use the HTML5 container service of mPaaS
- How to use HTML5 offline package service of mPaaS

Prerequisites

1. Configure a development environment. This tutorial describes how to configure a development environment in macOS as an example. For more information, see Configure the environment for mPaaS Inside connection and Portal & Bundle connection.
2. Network access.
3. A web browser is installed. The Chrome browser is recommended.
4. An Android phone and an accessory data bus. The operating system must be Android 4.3 or later. You can also use an emulator for commissioning. This tutorial uses a mobile phone as an example.

1.7.2.2. Create an app in Android Studio

In this section, you will create an app that causes a Toast to appear when a user taps a button, and obtain the installation package in APK format.

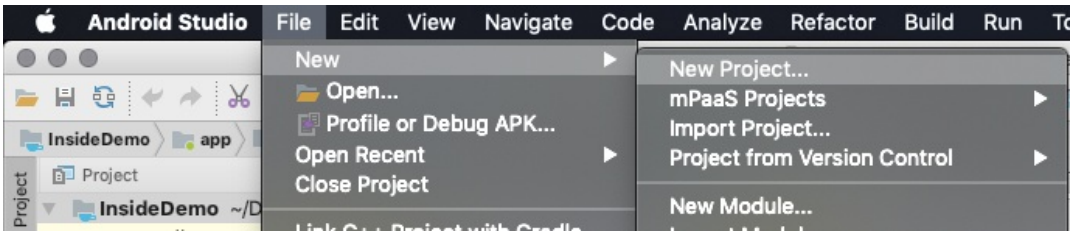
This process mainly includes the following four steps:

1. Create a project
2. Write the code
3. Create a signed file.
4. Install the app on a cell phone

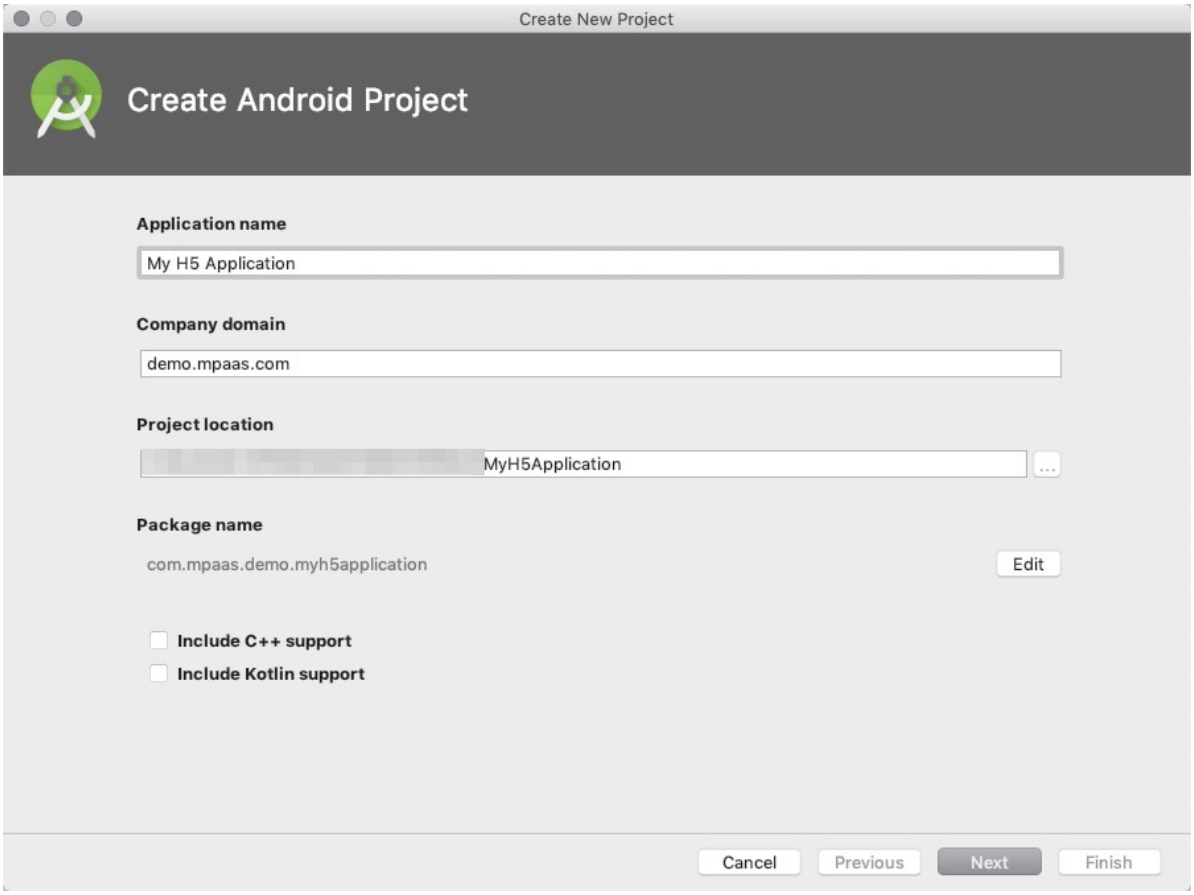
If you already have a signed native Android development project, you can skip this tutorial and directly create an app in the mPaaS console.

Create a project

1. Start Android Studio.

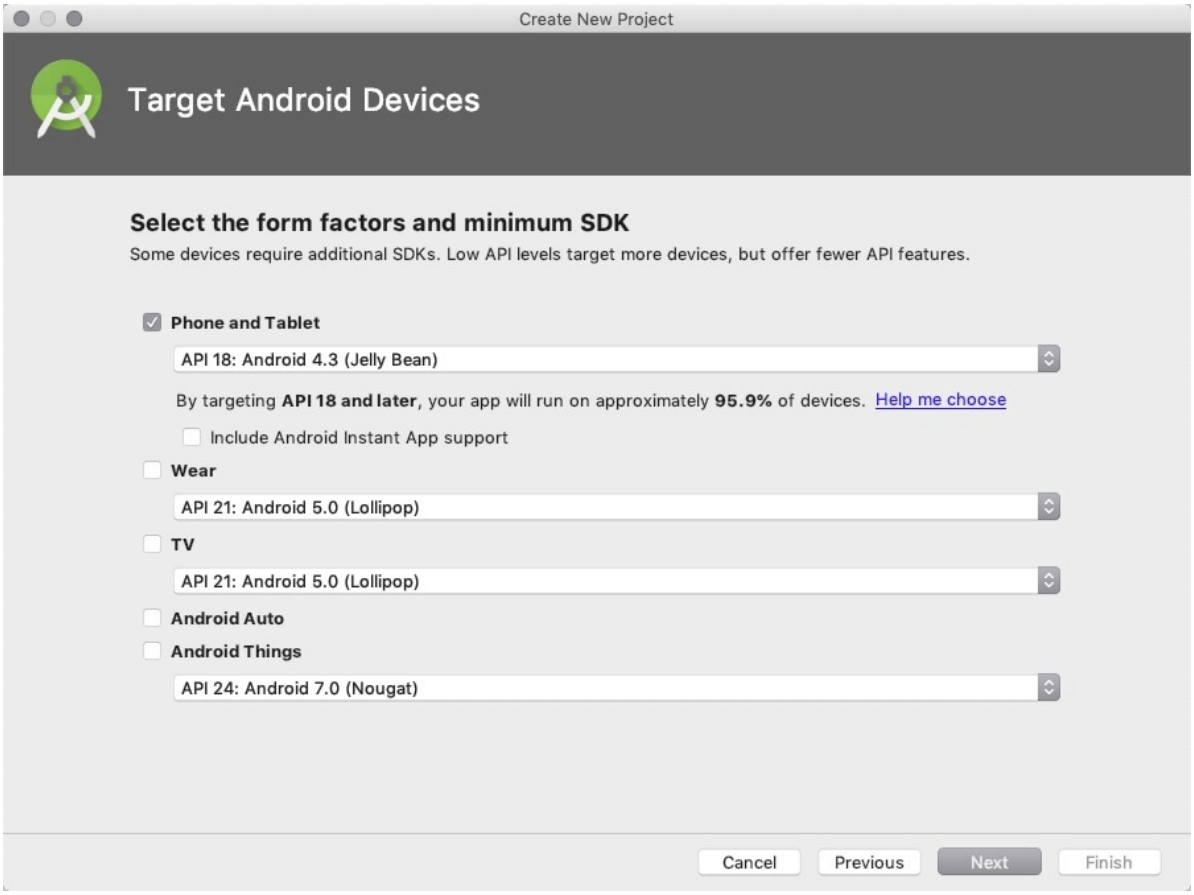


2. Choose **File** > **New** > **New Project**. Set the **Application Name**, **Company domain**, and **Project Location** parameters. If you do not have a company domain, you can use the default value. In this example, set **Application Name** to **My HTML5 Application**.

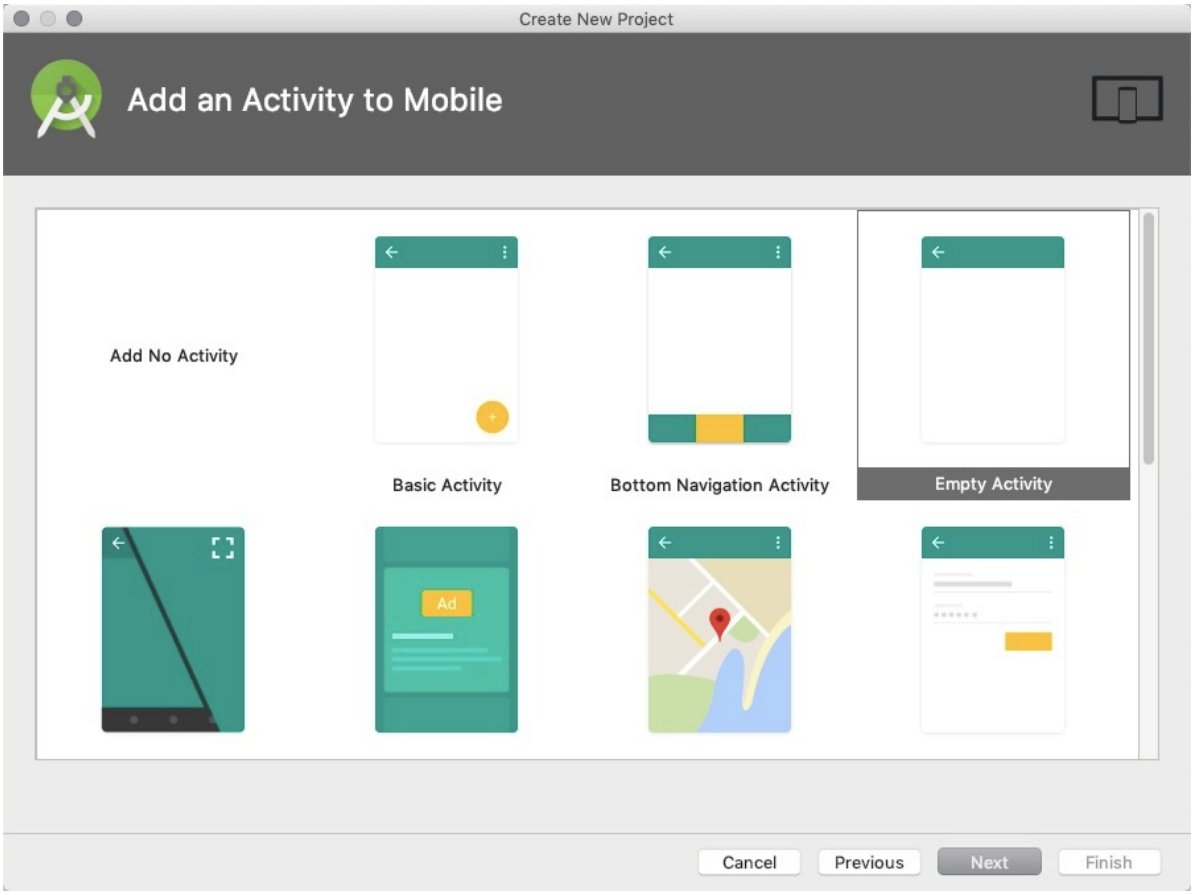


3. Set **Target Android Devices** to **Phone and Tablet** and **API Target level** to **API 18: Android 4.3 (Jelly Bean)**.

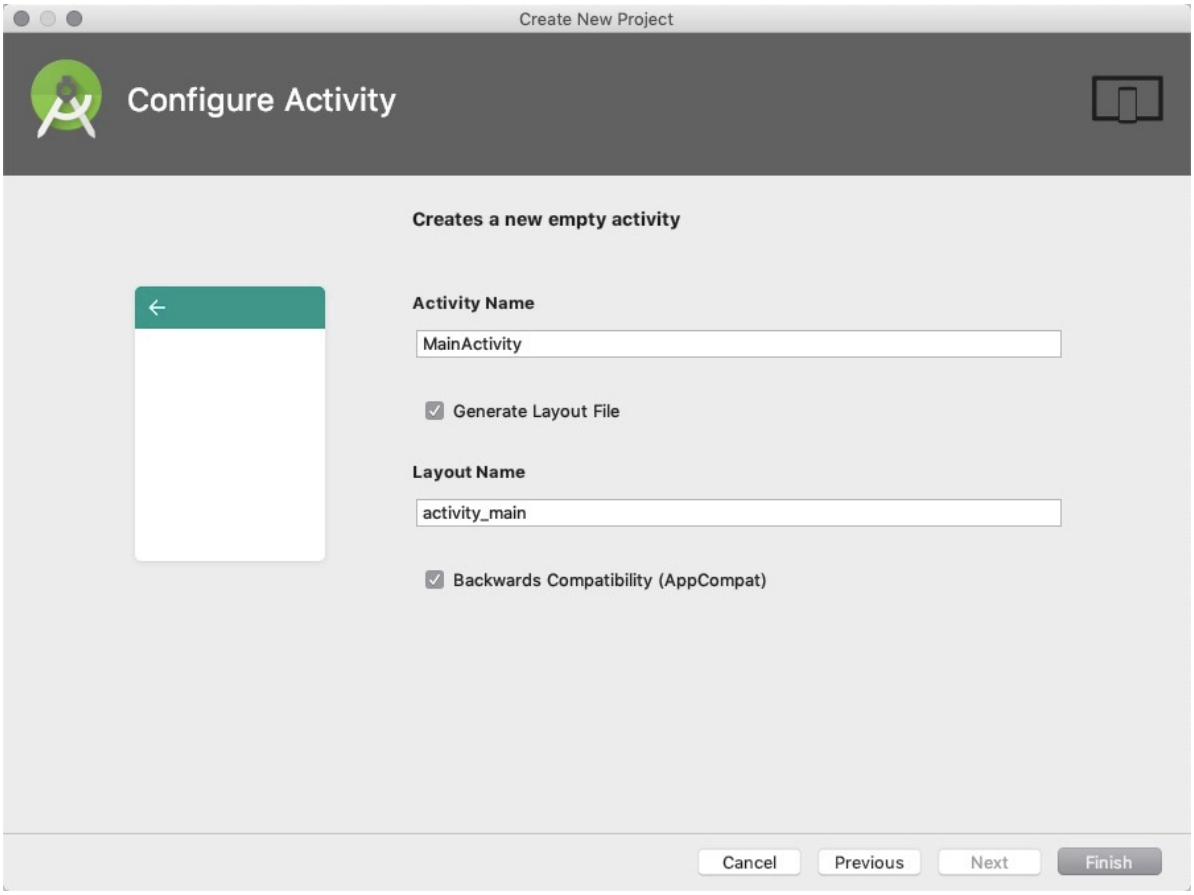
   Note: API 18: Android 4.3 (Jelly Bean) is the earliest version that mPaaS and mPaaS Inside can support. In actual production, you can select a version based on your needs.



4. Select **Empty Activity**.



5. Select **Generate Layout File** and **Backwards Compatibility (AppCompat)**. For Activity Name and Layout Name, you can use the default values.

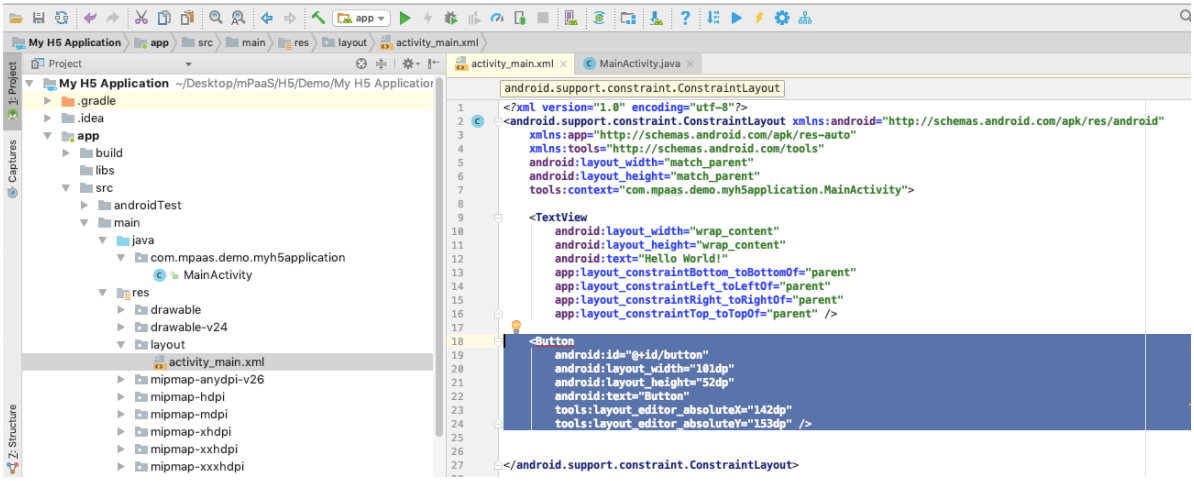


6. Click **Finish**. Now, you have **created a project**.

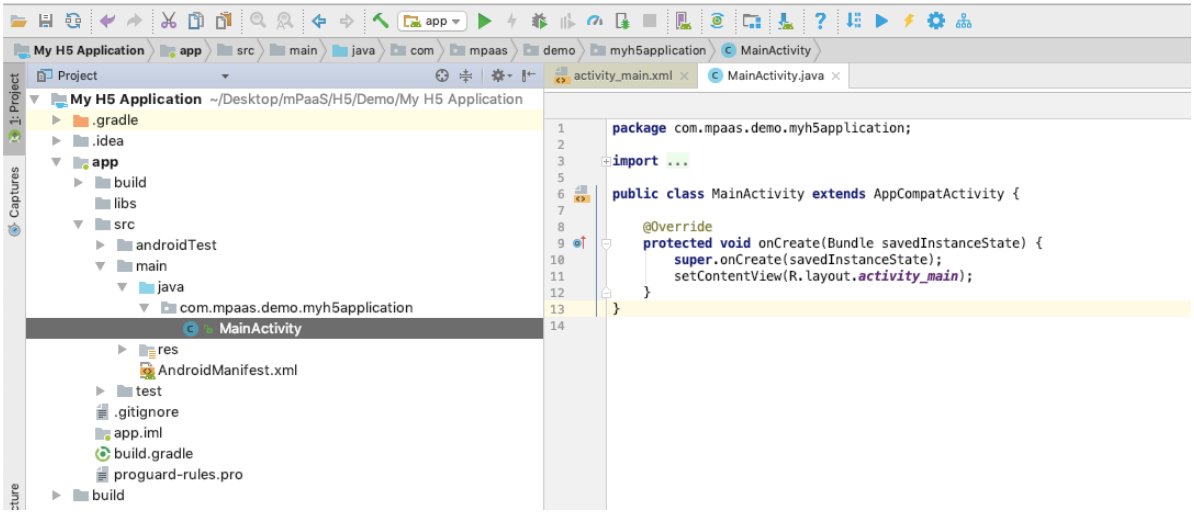
Write the code

1. Open the `activity_main.xml` file and write the following code in the file.

```
<Button
 android:id="@+id/button"
 android:layout_width="101dp"
 android:layout_height="52dp"
 android:text="Button"
 tools:layout_editor_absoluteX="142dp"
 tools:layout_editor_absoluteY="153dp" />
```

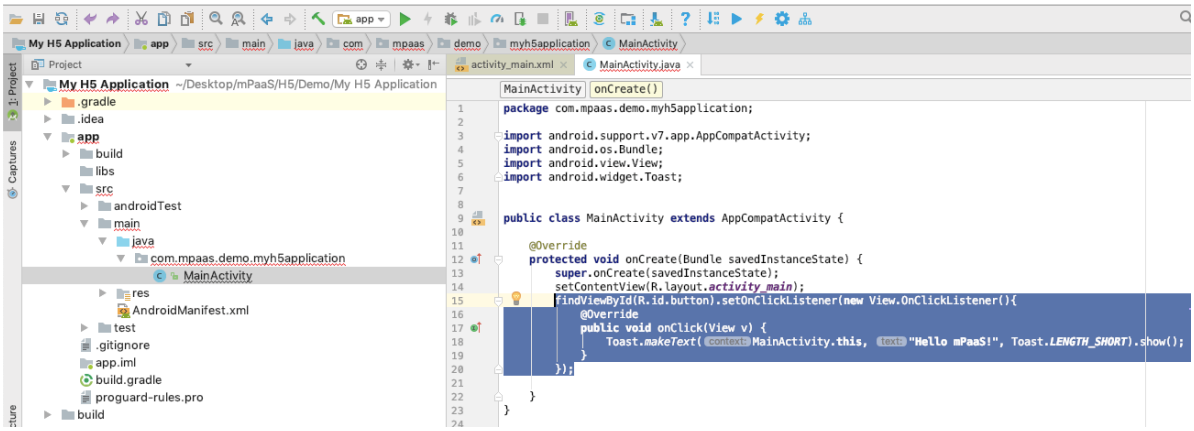


2. Open the `MainActivity` class.

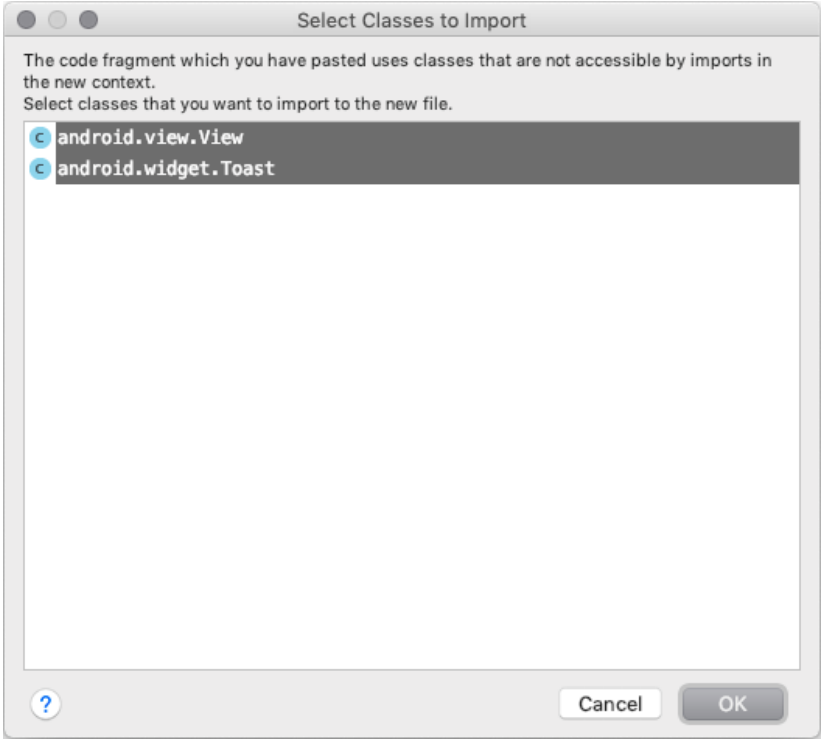


3. Write the following code in the MainActivity class.

```
findViewById(R.id.button).setOnClickListener(new View.OnClickListener(){
 @Override
 public void onClick(View v) {
     Toast.makeText(MainActivity.this, "Hello mPaaS!", Toast.LENGTH_SHORT).show();
 }
});
```



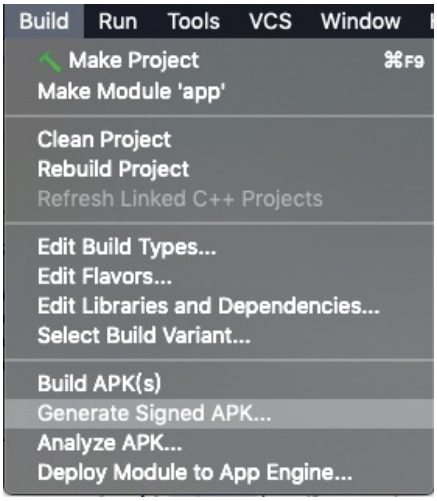
Note: If the following dialog box appears after you paste the preceding code in Android Studio, click **OK**.



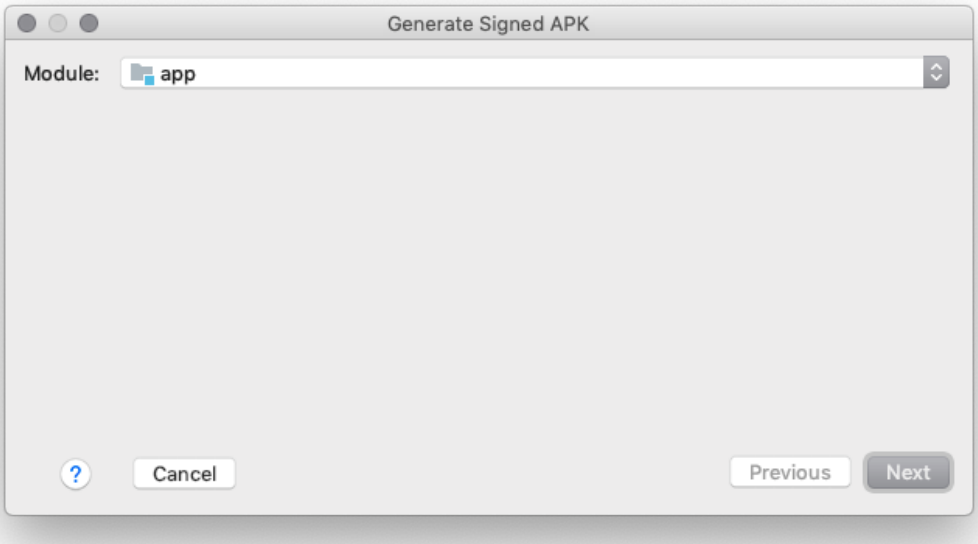
4. Compile and run the project. Make sure that the code runs normally. Now, you have **written the code**.

Create a signed file

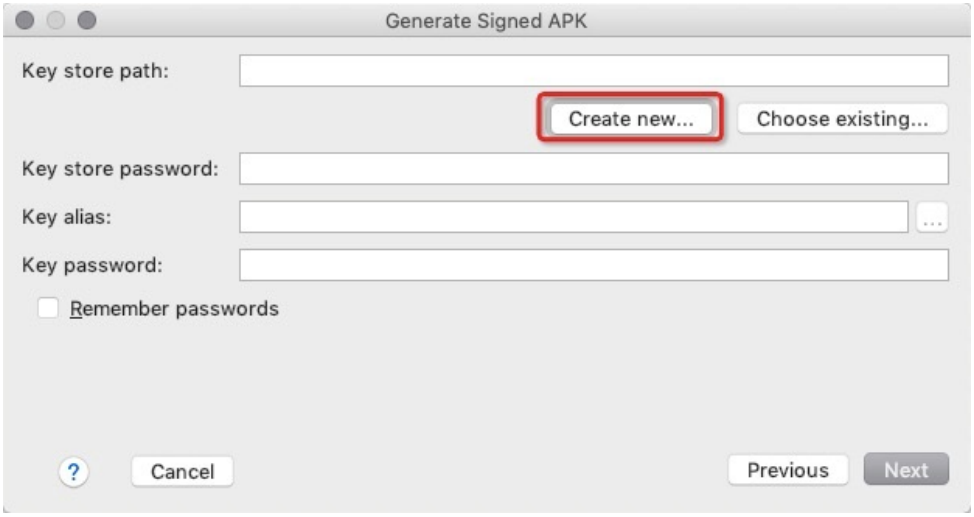
1. Choose **Build** > **Generate Signed APK**.



2. Click **Next**.



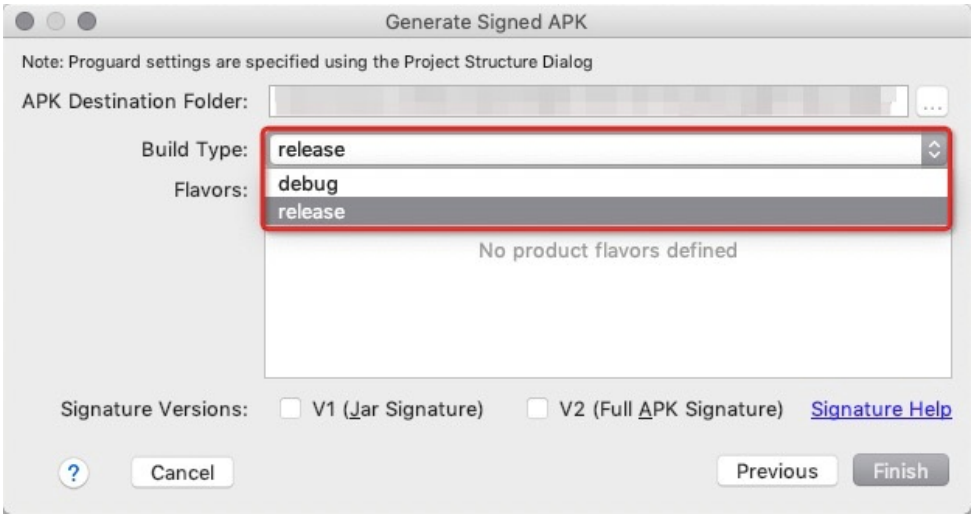
3. Click **Create new...**. If you already have a signed file, click **Choosing existing...**, select the signed file, and then skip to Step 5.



4. Set the parameters and click **OK**.

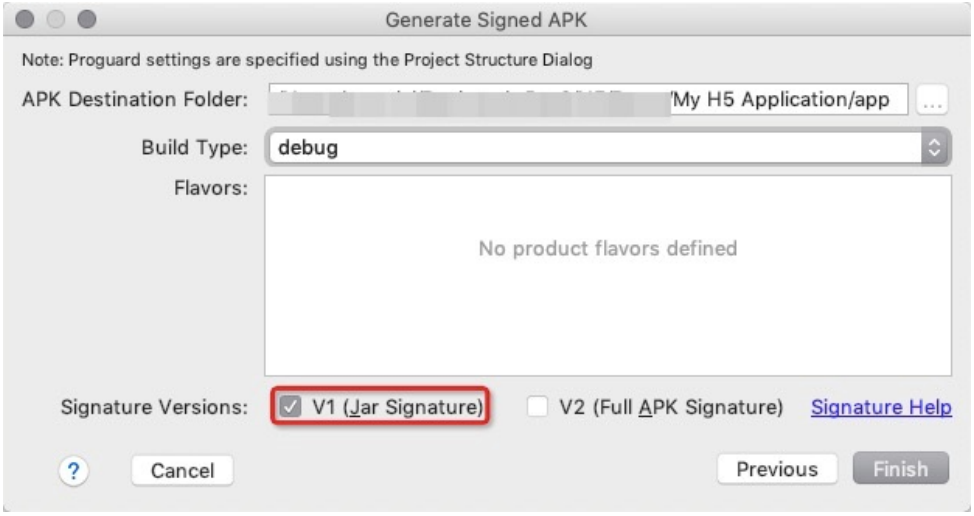
5. Click **Next**.

6. Set the **Build Type** parameter based on your needs.



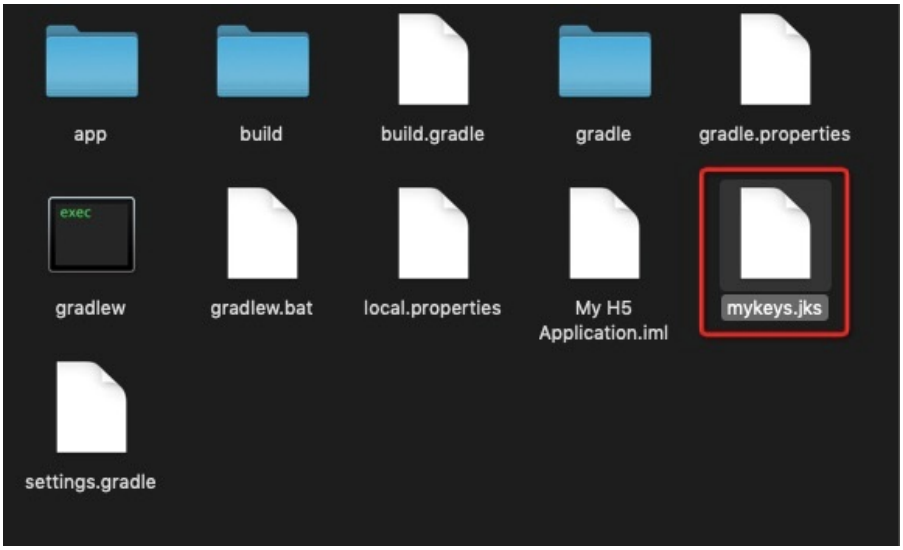
7. Select **V1 (Jar Signature)** for the Signature Versions parameter.

   V1 (Jar Signature) is required and **V2 (Full APK Signature)** is optional.



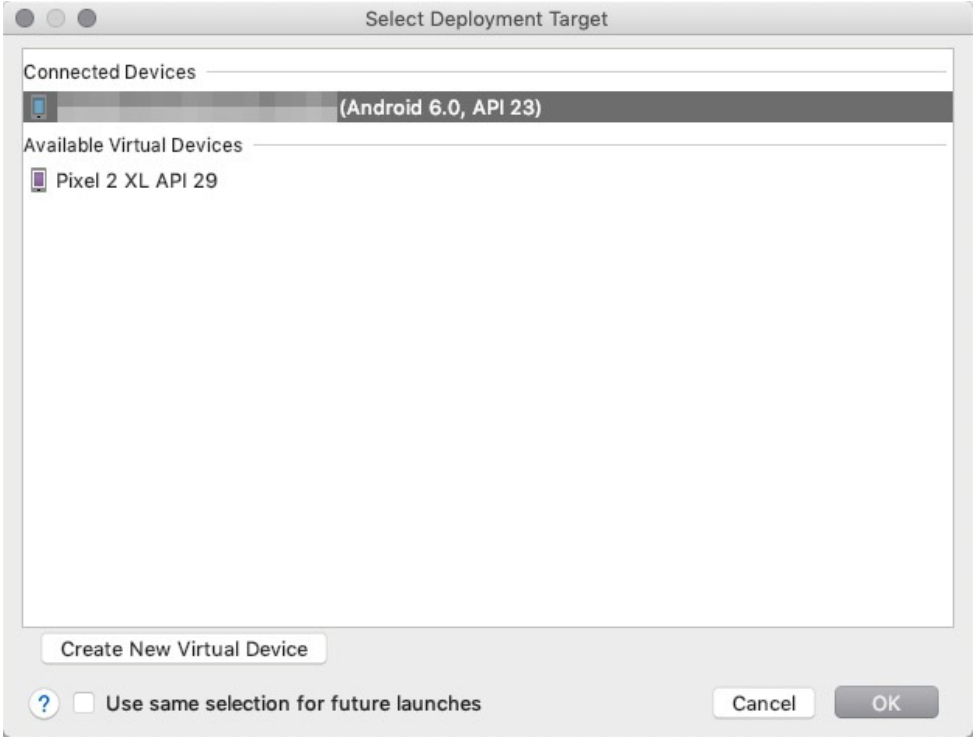
8. Click **Finish**. Wait until the signed file appears in the specified directory. Now, you have **created a signed file**. You can find the signed APK package of your app in the `debug` folder of your project. The directory is `/My HTML5 Application/app/build/outputs/apk/debug`.

   Note: After the new signed file is generated, Android Studio automatically compiles and runs the project.

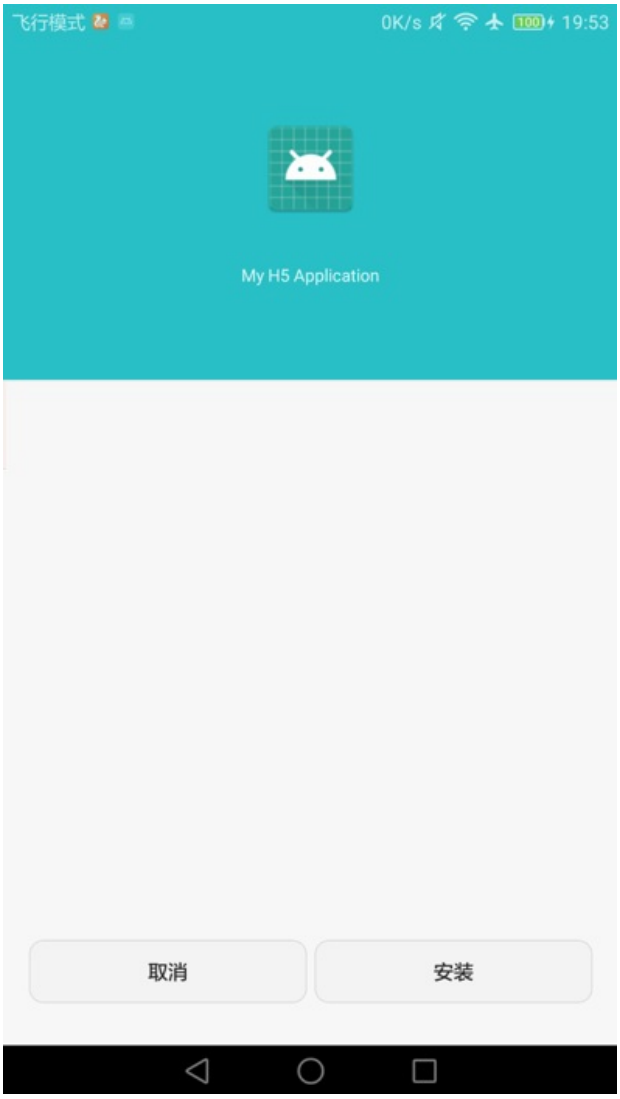


Install the app on a cell phone

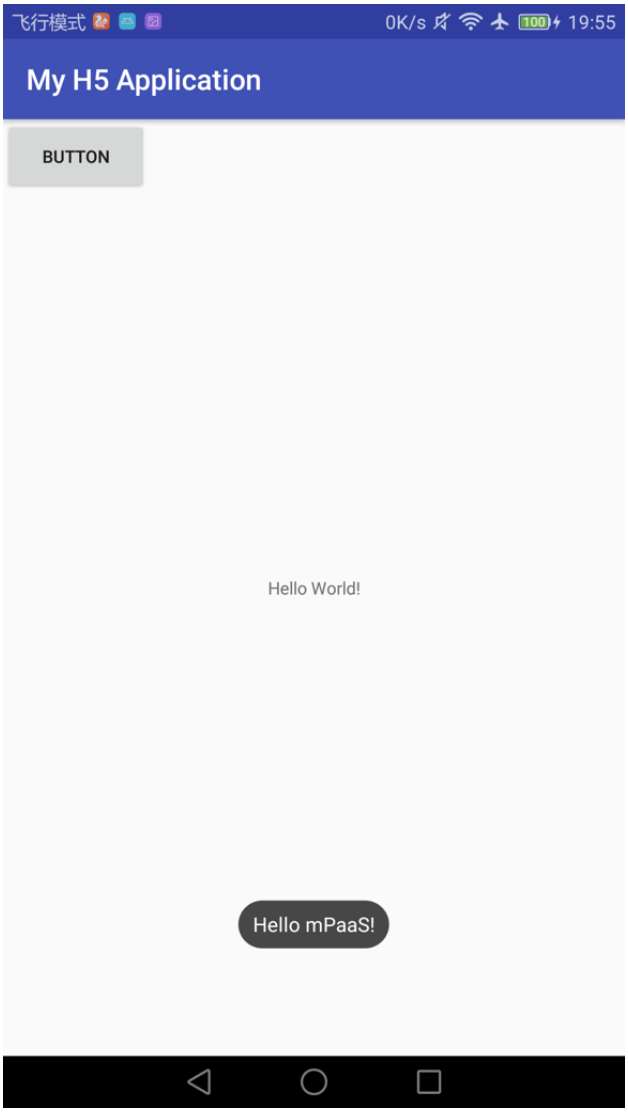
1. Connect a cell phone to the computer and enable the USB Debugging mode of the cell phone.
2. Run the project. The **Select Deployment Target** dialog box appears. Select your mobile phone and click **OK**.



3. On your mobile phone, tap **Install** to install the app.

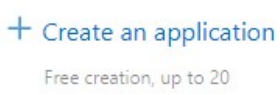


4. Open the app and tap **BUTTON**. The toast **“Hello mPaaS!”** appears, as shown in the following figure. The display of this toast indicates that the app is installed and can function as expected. Now, you have **installed the app on your mobile phone**.

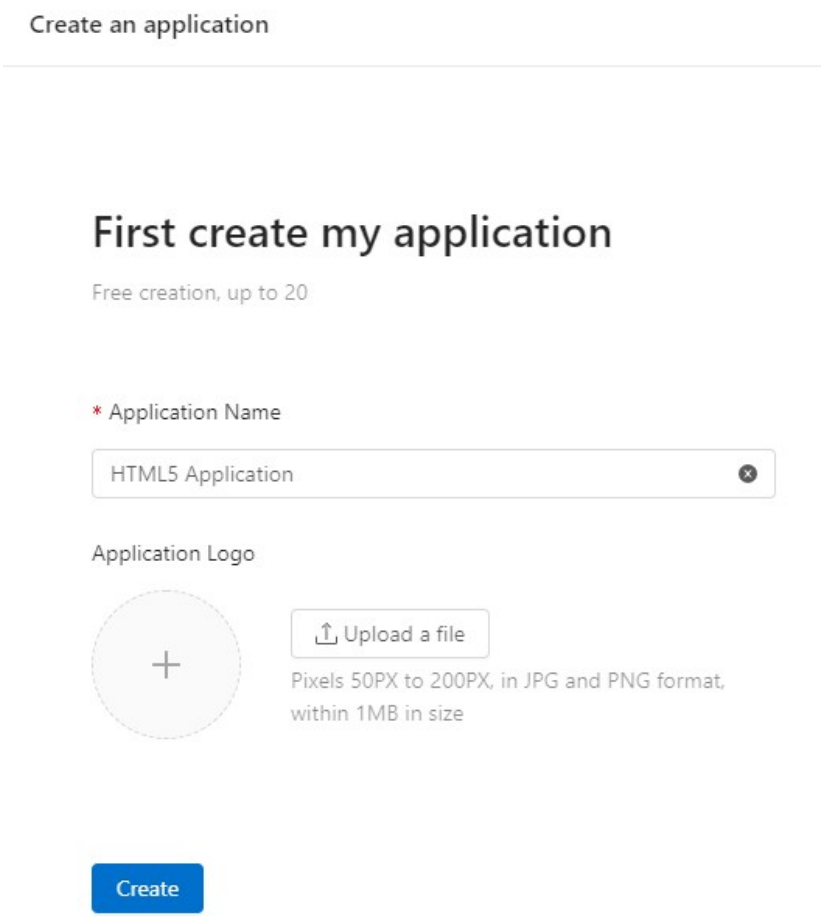


1.7.2.3. Create an application in the mPaaS console

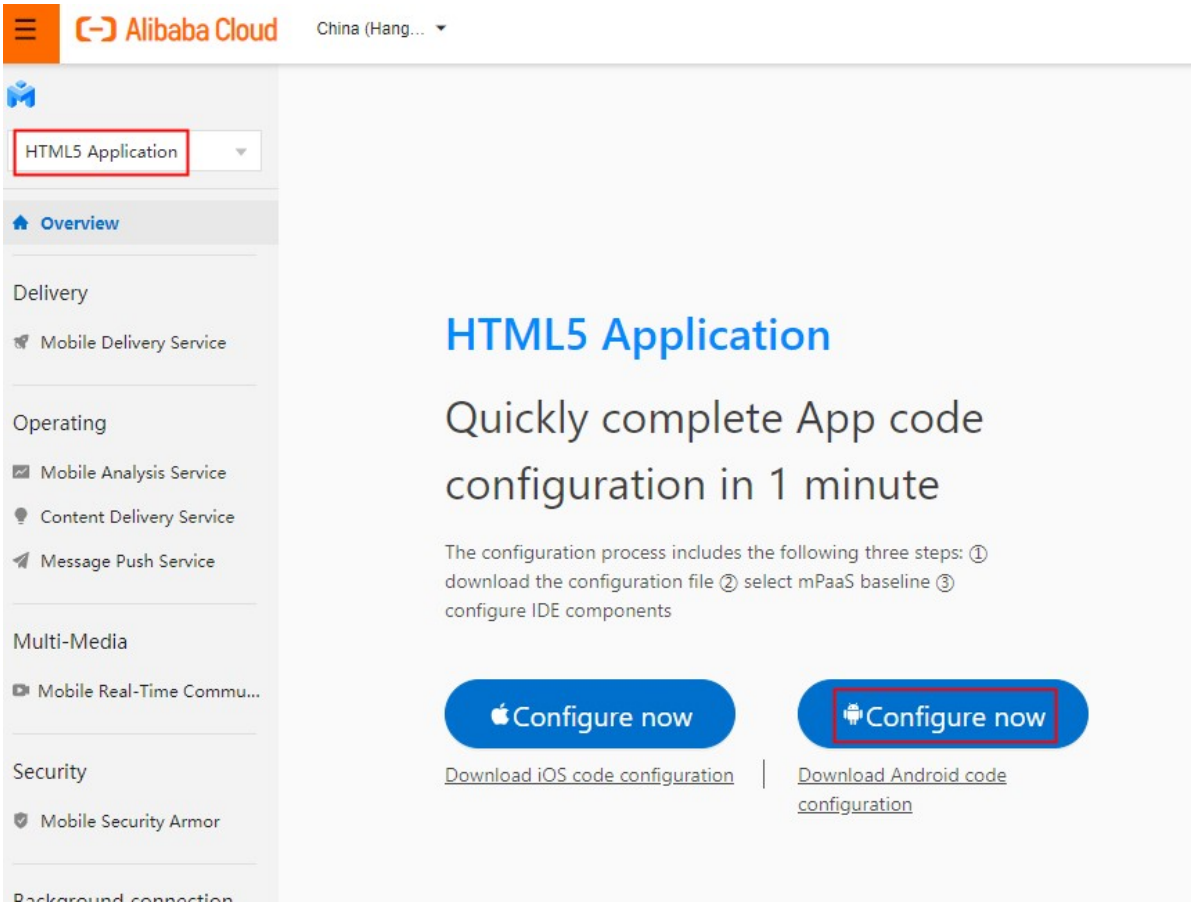
1. Start a web browser and log on to the mPaaS console.
2. Create an mPaaS app.



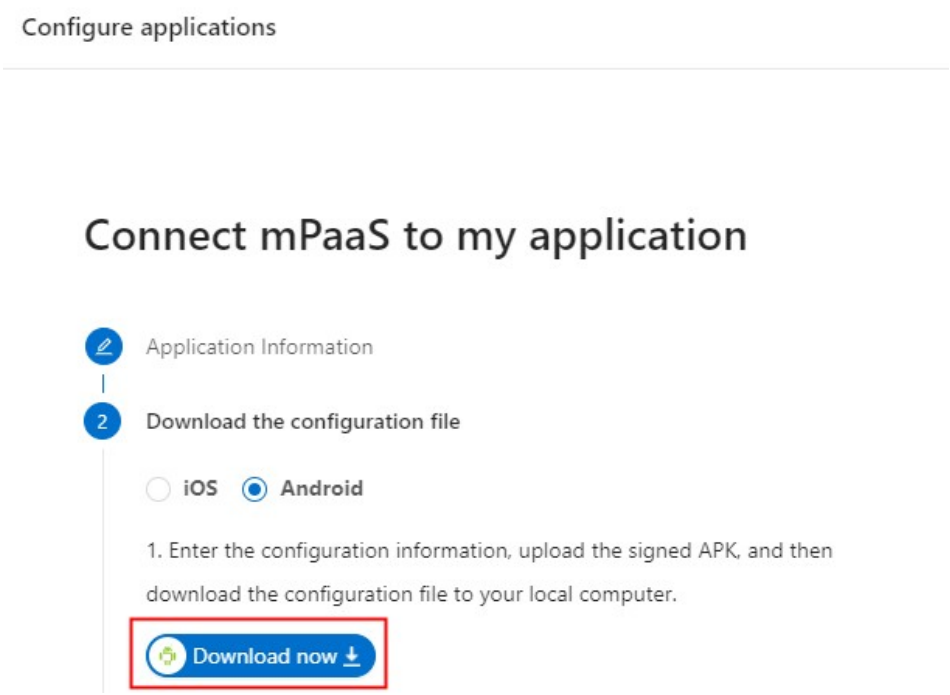
3. Enter an app name and click **Create**.



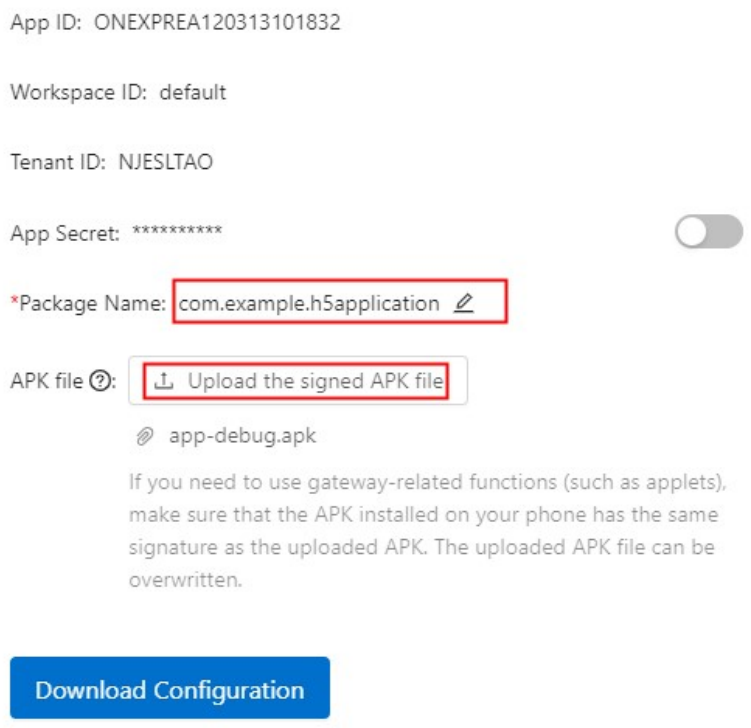
4. On the **App List** page, click **HTML5 Application**.



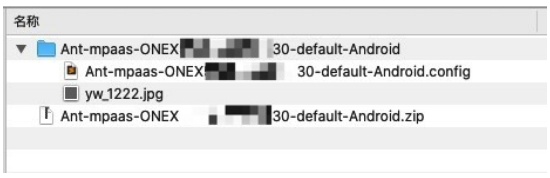
5. On the **App Details** page, click **Configure now** to open the **Configure App** page.



6. On the **Configure App** page, click **Download now** to open the **Code Configuration** page.In the **Code Configuration** pane, set the **Package Name** parameter. In this example, set the parameter to **com.mpaas.demo**. Then, upload a compiled and signed APK package. For information about how to generate a signed APK, see Generate a signed APK for the console.



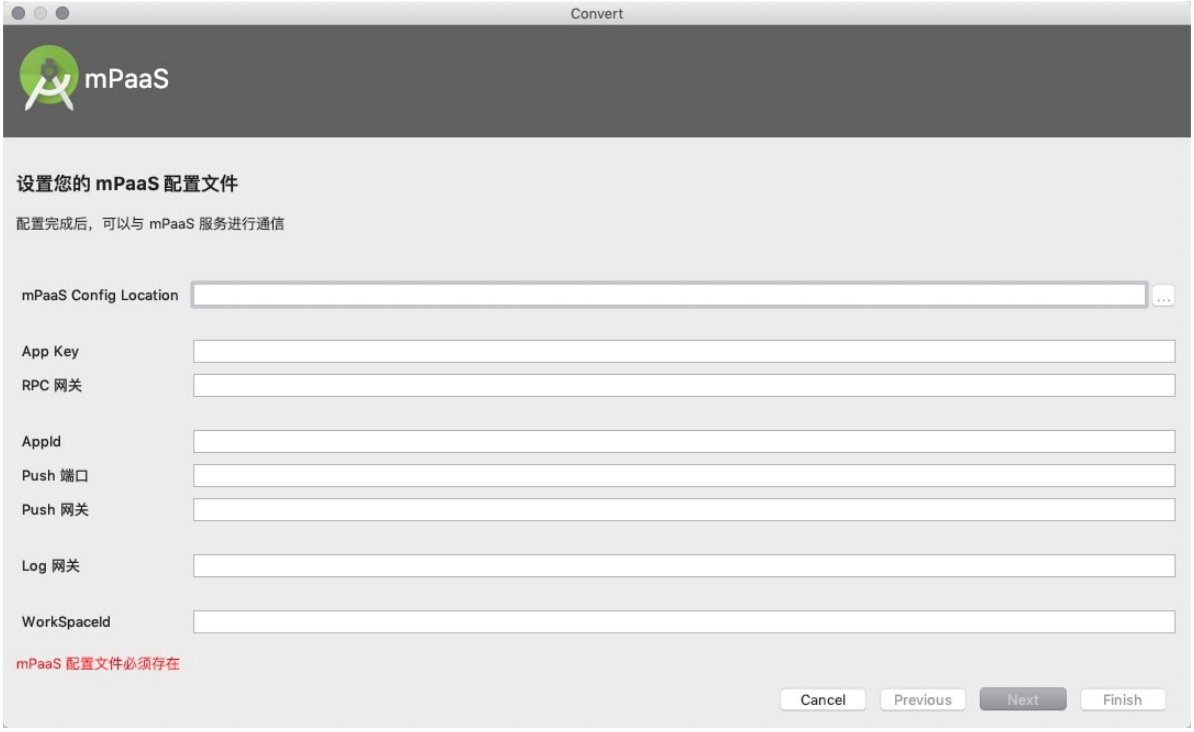
7. On the **Code Configuration** page, after you have entered all the information, click **Download Configuration** and you will obtain the configuration file of mPaaS.
8. The configuration file is a compressed file. The compressed package contains a `.config` file and an encrypted image named `yw_1222.jpg`.



1.7.2.4. Integrate mPaaS Inside into project

This topic describes how to integrate mPaaS Inside into your My HTML5 Application project.

1. In Android Studio, choose **mPaaS** > **mPaaS Inside Access** to open the mPaaS integration page. Select the configuration file that you download from the mPaaS console. Then, Android Studio automatically loads the information from the configuration file and modifies the `build.gradle` file based on the obtained information. The purpose is to introduce a Maven repository into the project and modify the version of Gradle.



Take note of the following items:

- If your project is connected in Portal&Bundle mode, you also need to modify the `build.gradle` file to introduce the following extra plug-ins. The plug-ins help you make better use of mPaaS capabilities.

```
apply plugin: 'com.alipay.library'
apply plugin: 'com.alipay.apollo.baseline.update'
```

- If you do not need specific components of mPaaS, write the components into strings in the `group:artifact` format and add the strings to the `excludeDependencies` array.

```
mpaascomponents{
// If you do not need specific components of mPaaS, write the components into strings
in the "group:artifact" format and add the strings to the "excludeDependencies"
array.
//Example   excludeDependencies=[
//Example    "com.alipay.android.phone.thirdparty:androidsupport-build",
//Example    "com.alipay.android.phone.thirdparty:androidsupportrecyclerview-build"
//Example    ]
     excludeDependencies=[

     ]
}
```

2. Compile and run your project. If the compilation succeeds and the project can run normally, the project is connected to mPaaS Inside. Now, you have **integrated mPaaS Inside into your project**.

1.7.2.5. Add H5 components to the project

1. Choose **mPaaS** > **Component Management**.

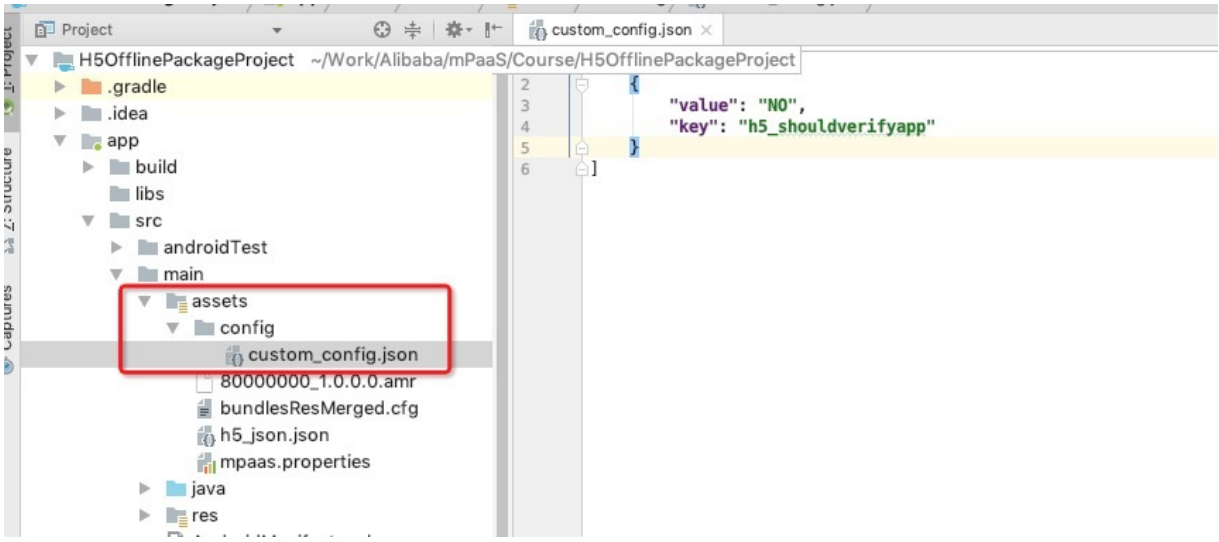
2. Find the **NEBULA - HTML5 Container** component and click **Install**. After the installation succeeds, the text on the button changes to **Uninstall**.

ANTUI	AntUI	AntUI	未安装	安装
FRAMEWORK	框架	Framework	已安装	卸载
HOTFIX	热修复	Hotfix	未安装	安装
LBS	定位	Location Service	未安装	安装
LOGGING	日志	Mobile Analytics Service	未安装	安装
MAP	地图	Map service	未安装	安装
MEDIA	多媒体	Media	未安装	安装
NEBULA	H5 容器	H5 Container	未安装	安装
PUSH	推送	Mobile Push Service	未安装	安装
RPC	移动网关	Mobile Gateway Service	未安装	安装
SCAN	扫码	Code Scanner	未安装	安装
STORAGE	存储	Storage	未安装	安装
SYNC	同步服务	Mobile Sync Service	未安装	安装
TINYPROGRAM	小程序	Tiny Program	未安装	安装
UPGRADE	升级	Upgrade	未安装	安装
UTDID	设备标识符	Device ID	未安装	安装
CONFIGSERVICE	开关	Config Service	未安装	安装
ESSENTIAL	必备组件	Essential	已安装	卸载
MINIPROGRAM-BLUETOOTH	小程序-蓝牙	Mini Program - Bluetooth	未安装	安装
MINIPROGRAM-MEDIA	小程序-媒体	Mini Program - Media	未安装	安装
CDP	智能投放	Content Delivery Platform	未安装	安装

3. After the installation, compile the project. Now, you have added the HTML5 container component to your project.
4. (Optional) Disable signature verification for the HTML5 container.

   By default, signature verification is enabled for the HTML5 container. If you need to disable this feature, add a configuration file named `custom_config.json` in the `/app/src/main/assets/config` directory. Write the following code in the file to disable signature verification.

```
[
{
 "value": "NO",
 "key": "h5_shouldverifyapp"
}
]
```



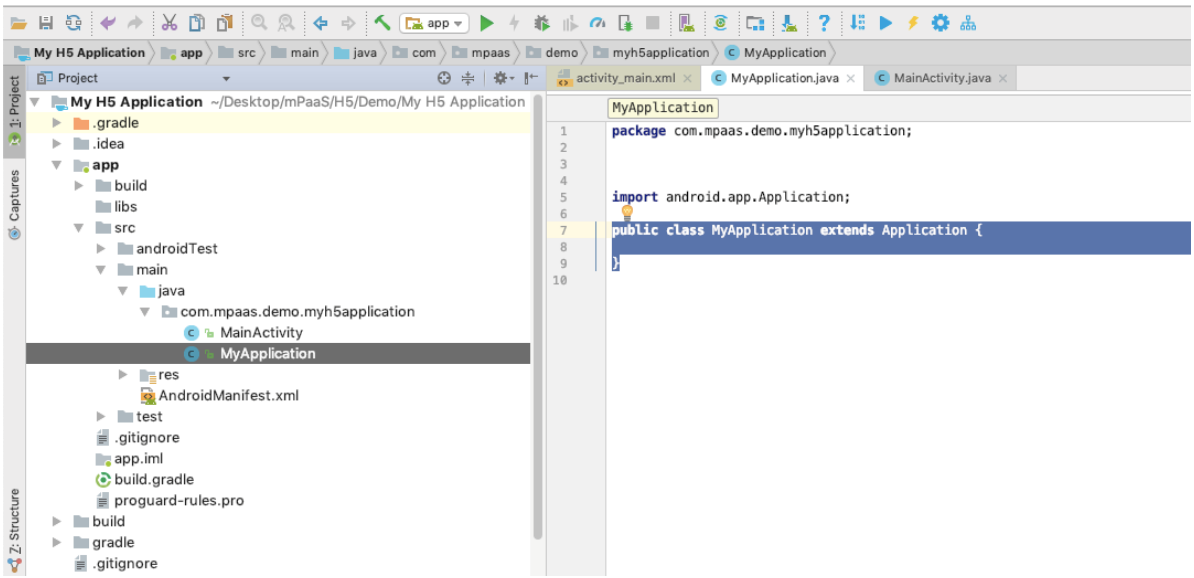
1.7.2.6. Use HTML5 container

You can use the HTML5 container to perform the following operations:

- Open an online web page within an app
- Call the Native APIs on the front end
- Call a custom JSAPI on the front end
- Customize the title bar for an HTML5 page

Open an online web page within an app

1. Add a custom class `MyApplication`, which inherits from `Application`, to the project.



2. Initialize mPaaS Inside. In the custom class `MyApplication`, perform initialization. The initialization method is shown in the following code:

```
package com.mpaas.demo.myh5application;

 import android.app.Application;
 import android.content.Context;

 import com.alipay.mobile.framework.quinoxless.IInitCallback;
 import com.alipay.mobile.framework.quinoxless.QuinoxlessFramework;

 public class MyApplication extends Application {
 @Override
 protected void attachBaseContext(Context base) {
 super.attachBaseContext(base);
 QuinoxlessFramework.setup(this, new IInitCallback() {
 @Override
 public void onPostInit() {
 // Start to use the mPaaS functions here.
 }
 });
 }

 @Override
 public void onCreate() {
 super.onCreate();
 QuinoxlessFramework.init();
 }
 }
```

3. In the file `app/src/main/AndroidManifest.xml`, add `android:name=".MyApplication"`.

```
<?xml version="1.0" encoding="utf-8"?>
 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
     package="com.mpaas.demo.myh5application">

     <application
         android:name=".MyApplication"
         android:allowBackup="true"
         android:icon="@mipmap/ic_launcher"
         android:label="@string/app_name"
         android:roundIcon="@mipmap/ic_launcher_round"
         android:supportsRtl="true"
         android:theme="@style/AppTheme">
         <activity android:name=".MainActivity">
             <intent-filter>
                 <action android:name="android.intent.action.MAIN" />

                 <category android:name="android.intent.category.LAUNCHER" />
             </intent-filter>
         </activity>
     </application>

 </manifest>
```

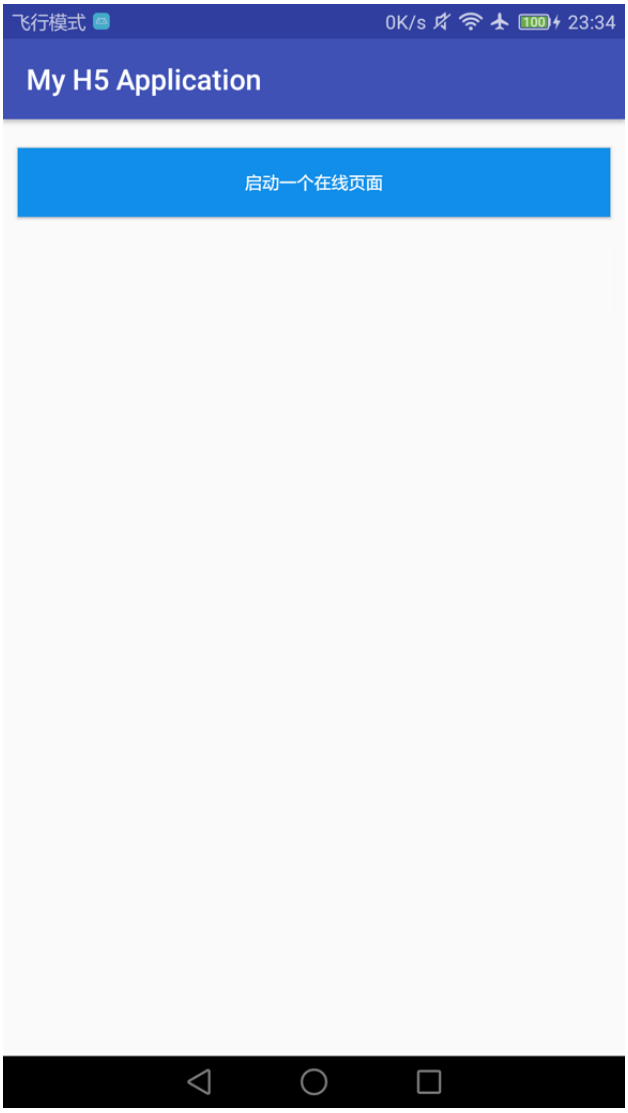
4. In the `activity_main.xml` file, rewrite the code of the button. Change the `button id` to `start_url_btn` and make some simple adjustments to the style of the button.

```
<?xml version="1.0" encoding="utf-8"?>
 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
     android:layout_width="match_parent"
     android:layout_height="match_parent"
     android:orientation="vertical">

     <Button
         android:id="@+id/start_url_btn"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:layout_marginBottom="10dp"
         android:layout_marginLeft="10dp"
         android:layout_marginRight="10dp"
         android:layout_marginTop="20dp"
         android:background="#108EE9"
         android:gravity="center"
         android:text="Start an Online Web Page"
         android:textColor="#ffffff" />

 </LinearLayout>
```

5. In the `MainActivity` class, rewrite the code of the system behavior upon a tap on the button. Make sure that a tap on the button opens the official website of Ant Group Financial Technology. The following code shows how this can be implemented:

```
findViewById(R.id.start_url_btn).setOnClickListener(new View.OnClickListener(){
             @Override
             public void onClick(View v) {
                 MPNebula.startUrl("https://tech.antfin.com/");
             }
         });
```

6. After compiling the project, install the app on a cell phone. You will see the following user interface after starting the app:



7. You will open the home page of the official website of Ant Group Financial Technology within the app when you click the button. This means the API operation was successful. Now, you have completed **opening an online web page within an app** .

Call the Native APIs on the front end

1. When implementing a front-end page, through the bridge provided by the Nebula container, you can communicate with Native by using the JSAPI method to obtain the related information or data processed by Native. The Nebula container provides some basic JSAPIs. You can use `AlipayJSBridge.call` to call the JSAPIs in the `js` file of an HTML5 page. For more information, see API description. See the following sample code:

```
AlipayJSBridge.call('alert', {
   title: 'Native Alert Dialog',
   message: 'This is an Alert Dialog from Native.',
   Button: 'OK'
 }, function (e) {
   alert("The button was tapped.");
 });
```

Note: `https://mcube-prod.oss-cn-hangzhou.aliyuncs.com/570DA89281533-default/80000000/1.0.XX.XX_all/nebula/fallback/h5_to_native.html` is a ready-to-use front-end page. You can use this page to test calling native APIs from a front-end page.

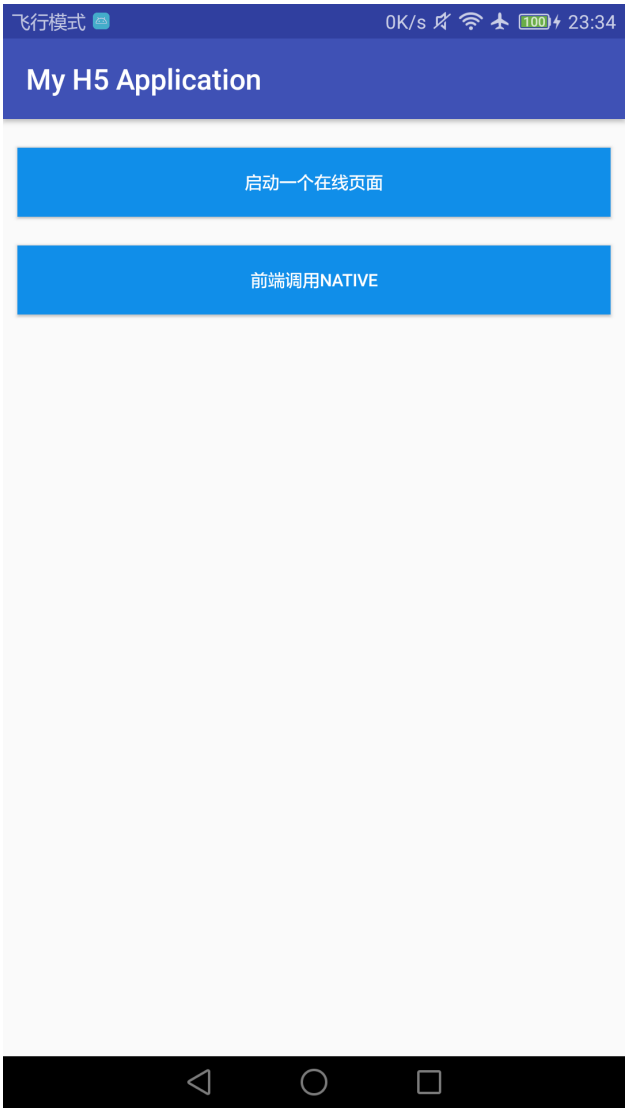
2. In the `activity_main.xml` file, create a button and set the `button id` to `h5_to_native_btn`.

```
<Button
     android:id="@+id/h5_to_native_btn"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_margin="10dp"
     android:background="#108EE9"
     android:gravity="center"
     android:text="Call Native on the Front End"
     android:textColor="#ffffff" />
```

3. In the class `MainActivity`, define the behavior after the button `h5_to_native_btn` is tapped, to implement the function of opening a front-end page. The following code shows how this can be implemented:

```
findViewById(R.id.h5_to_native_btn).setOnClickListener(new View.OnClickListener() {
     @Override
     public void onClick(View v) {
         MPNebula.startUrl("https://mcube-prod.oss-cn-
hangzhou.aliyuncs.com/570DA89281533-
default/80000000/1.0.XX.XX_all/nebula/fallback/h5_to_native.html");
     }
 });
```

4. After compiling the project, install the app on a cell phone. You will see the following user interface after starting the app:



5. Tap the button for calling native APIs from the front end. The front-end page appears. Tap the **Display Native Alert Dialog**. A message titled **Native Alert Dialog** appears, reading **This is a native Alert Dialog**. Click **OK** in the message. The message disappears and another untitled message appears, reading **You tapped the button**. This means the API operation was successful. Now, you have completed **calling the Native APIs on the front end** .


飞行模式
13B/s
23:32
前端页面调用原生
显示原生 Alert Dialog
原生 Alert Dialog
这是一个来自原生的 Alert Dialog
确定

Call a custom JSAPI on the front end

1. Create a custom class `MyJSApiPlugin` to define a custom JSAPI.

```
package com.mpaas.demo;

 import com.alibaba.fastjson.JSONObject;
 import com.alipay.mobile.h5container.api.H5BridgeContext;
 import com.alipay.mobile.h5container.api.H5Event;
 import com.alipay.mobile.h5container.api.H5EventFilter;
 import com.alipay.mobile.h5container.api.H5SimplePlugin;

 public class MyJSApiPlugin extends H5SimplePlugin {

     private static final String API = "myapi";

     @Override
     public void onPrepare(H5EventFilter filter) {
         super.onPrepare(filter);
         filter.addAction(API);
     }

     @Override
     public boolean handleEvent(H5Event event, H5BridgeContext context) {
         String action = event.getAction();
         if (API.equalsIgnoreCase(action)) {
             JSONObject params = event.getParam();
             String param1 = params.getString("param1");
             String param2 = params.getString("param2");
             JSONObject result = new JSONObject();
             result.put("success", true);
             result.put("message", API + " with " + param1 + "," + param2 + " was handle
d by native.");
             context.sendBridgeResult(result);
             return true;
         }
         return false;
     }

 }
```

2. Register the custom JSAPI named MyJSApiPlugin in the project. We recommend that you register it when the app starts. In this example, register the JSAPI in the `MyApplication` class.

```
public class MyApplication extends Application {
         @Override
         protected void attachBaseContext(Context base) {
             super.attachBaseContext(base);
             // Suggestion: Check if this is the main process. Initialize only in the
main process.
             QuinoxlessFramework.setup(this, new IInitCallback() {
                 @Override
                 public void onPostInit() {
                     // Start to use the mPaaS functions here.
         // Call registerCustomJsapi() to complete registering the custom JSAPI.
                     registerCustomJsapi();
                 }
             });
         }

     @Override
     public void onCreate() {
         super.onCreate();
         QuinoxlessFramework.init();
     }

     private void registerCustomJsapi(){
         MPNebula.registerH5Plugin(
                 // Class name of the plug-in.
                 MyJSApiPlugin.class.getName(),
                 // Just use an empty string.
                 "",
                 // Applicable context. Just use "page".
                 "page",
                 // Name of the registered JSAPI.
                 new String[]{"myapi"});
     }
 }
```

3. In a front-end page, call this custom JSAPI. See the following sample code:

```
AlipayJSBridge.call('myapi', {
   param1: 'JsParam1',
   param2: 'JsParam2'
 }, function (result) {
   alert(JSON.stringify(result));
 });
```

Note: “https://mcube-prod.oss-cn-hangzhou.aliyuncs.com/570DA89281533-default/80000001/1.0.XX.XX_all/nebula/fallback/custom_jsapi.html“ is a fully functional front-end page. You can use this page to try out calling a **custom JSAPI** on the front end.

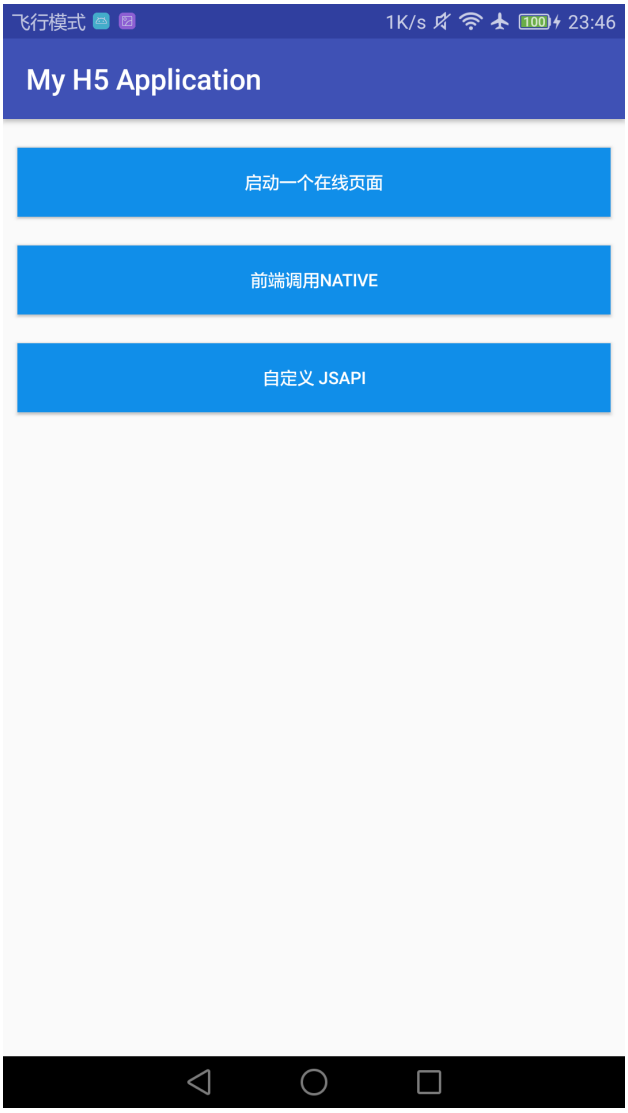
4. In the `activity_main.xml` file, create a button and set the `button id` to `custom_jsapi_btn`.

```
<Button
     android:id="@+id/custom_jsapi_btn"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_margin="10dp"
     android:background="#108EE9"
     android:gravity="center"
     android:text="Custom JSAPI"
     android:textColor="#ffffff" />
```

5. In the class `MainActivity`, define the behavior after the button `custom_jsapi_btn` is tapped, to implement the function of calling a custom JSAPI on the front end. The following code shows how this can be implemented:

```
findViewById(R.id.custom_jsapi_btn).setOnClickListener(new View.OnClickListener() {
     @Override
     public void onClick(View v) {
         MPNebula.startUrl("https://mcube-prod.oss-cn-
hangzhou.aliyuncs.com/570DA89281533-
default/80000001/1.0.XX.XX_all/nebula/fallback/custom_jsapi.html");
     }
 });
```

6. After compiling the project, install the app on a cell phone. You will see the following user interface after starting the app:



7. Tap the **Custom JSAPI** button and a front-end page containing a button named **Custom JSAPI** will appear. If you tap the **Custom JSAPI** button on this page, an alert dialog box with no title will appear. This dialog box shows the handled parameters that were passed in while the API was called on the front end, based on the function defined by the custom API. Now, you have **called the custom JSAPI from the front end** .

Customize the title bar for an HTML5 page

The HTML5 container provides you with methods to configure a custom title bar. You can use the default title bar `MpaasDefaultH5TitleView` that is provided by mPaaS. You can rewrite some of these methods based on your needs. You can also implement `H5TitleView`. This section describes how to use `MpaasDefaultH5TitleView` to configure a title bar.

1. Construct an `H5ViewProvider` implementation class. Set your custom `H5TitleView` as the return result of the `createTitleView` method.

```
package com.mpaas.demo.myh5application;

 import android.content.Context;
 import android.view.ViewGroup;

 import com.alipay.mobile.nebula.provider.H5ViewProvider;
 import com.alipay.mobile.nebula.view.H5NavMenuView;
 import com.alipay.mobile.nebula.view.H5PullHeaderView;
 import com.alipay.mobile.nebula.view.H5TitleView;
 import com.alipay.mobile.nebula.view.H5WebContentView;
 import com.mpaas.nebula.adapter.view.MpaasDefaultH5TitleView;

 public class H5ViewProviderImpl implements H5ViewProvider {
     @Override
     public H5TitleView createTitleView(Context context) {
         return new MpaasDefaultH5TitleView(context);
     }

     @Override
     public H5NavMenuView createNavMenu() {
         return null;
     }

     @Override
     public H5PullHeaderView createPullHeaderView(Context context, ViewGroup viewGroup)
{
         return null;
     }

     @Override
     public H5WebContentView createWebContentView(Context context) {
         return null;
     }
 }
```

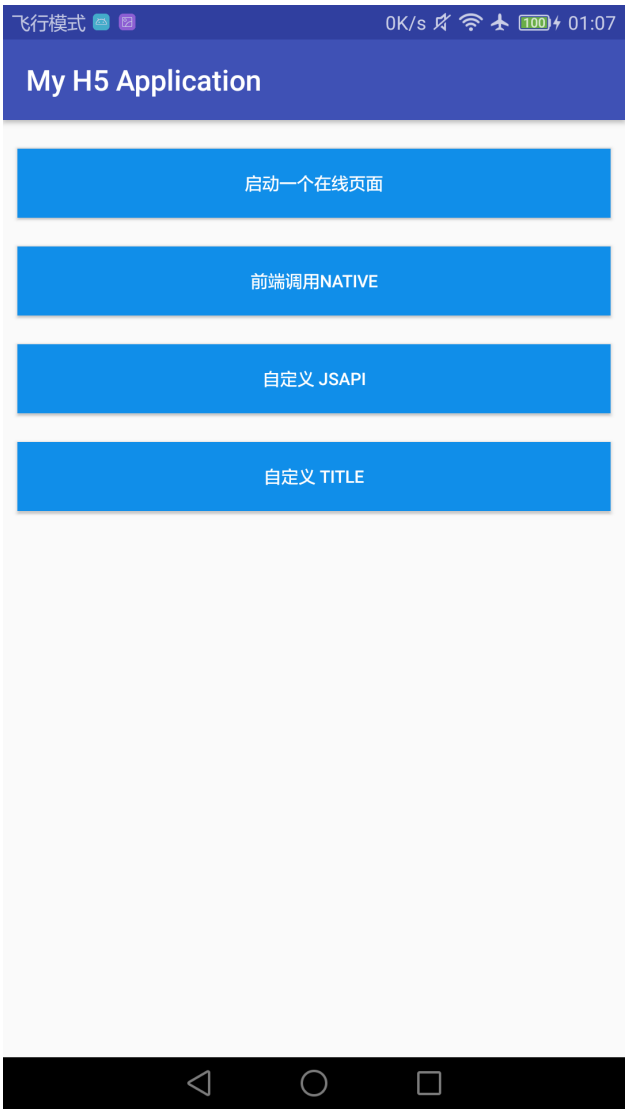
2. In the `activity_main.xml` file, create a button and set the `button id` to `custom_title_btn`.

```
<Button
     android:id="@+id/custom_title_btn"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_margin="10dp"
     android:background="#108EE9"
     android:gravity="center"
     android:text="Custom TITLE"
     android:textColor="#ffffff" />
```

3. In the `MainActivity` class, specify the system behavior upon a tap on the `custom_title_btn` button: specify the custom View Provider for the container and open an online page. The following code shows how this can be implemented:

```
findViewById(R.id.custom_title_btn).setOnClickListener(new View.OnClickListener() {
     @Override
     public void onClick(View v) {
         // Set a custom title. You only need to set the title for once.
         MPNebula.setCustomViewProvider(new H5ViewProviderImpl());
         // Start a URL. The title changes after the URL is started.
         MPNebula.startUrl("https://www.cloud.alipay.com/docs/2/49549");
     }
 });
```

4. After compiling the project, install the app on a cell phone. You will see the following user interface after starting the app:



5. Tap the **Custom TITLE** and **Custom JSAPI** buttons in order. The same online page appears after you tap either of the two buttons. You can find that both the color of the title bar and font color are changed. Now, you have **customized the title bar for an HTML5 page** .
   - Before you customize the title bar

- After you customize the title bar

1.7.2.7. Use HTML5 offline package

The usage of an HTML5 offline package involves four steps: release, preset, start, and then update the offline package. To demonstrate the features of HTML5 offline packages, this topic describes all of the four steps. However, not every step is necessary for using an HTML5 offline package. In actual production, you can perform specific steps based on your needs.

Release an offline package

This section describes how to release an offline package.

Prerequisites

You need to prepare a ZIP package of your front-end Application. Otherwise, you can download the sample offline package.

Procedure

1. Log on to the mPaaS console and configure the information about the offline package of your Application. For more information, see Configure an offline package.
2. Generate the offline package of your front-end Application. You can also use the sample offline package. For more information, see Generate an offline package.

3. In the mPaaS console, create an HTML5 Application and upload your offline package. For more information, see Create an offline package.
4. Release the configured offline package to your Application on the client. For more information, see Release an offline package.

Preset an offline package

This section describes how to preset an offline package.

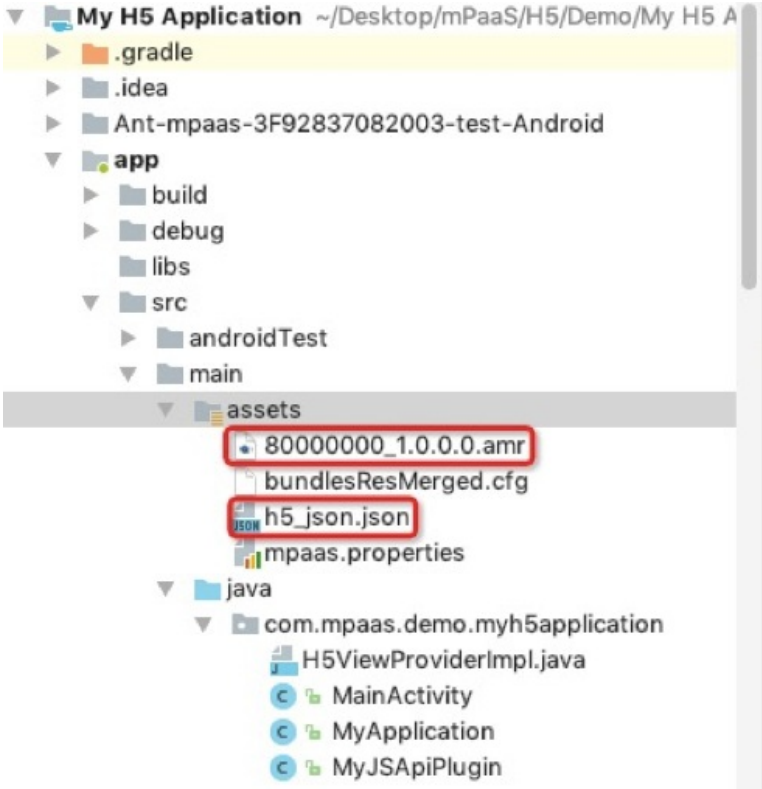
Prerequisites

Your offline package is released in the mPaaS console.

Procedure

1. In the mPaaS console, download the AMR file and the configuration file of the offline package to your local system.

2. Put the downloaded AMR file and the configuration file in the assets directory of your project.



3. Preset the offline package into your Application. We recommend that you register the offline package during the startup of the Application. In this tutorial, the offline package is registered in the `MyApplication` class. Now, you have preset the offline package.

```
 public class MyApplication extends Application {
         @Override
         protected void attachBaseContext(Context base) {
             super.attachBaseContext(base);
             // Suggestion: Check if this is the main process. Initialize only in the
main process.
             QuinoxlessFramework.setup(this, new IInitCallback() {
                 @Override
                 public void onPostInit() {
                     // Start to use the mPaaS functions here.
                     registerCustomJsapi();
         // Call loadOfflineNebula() to load the offline package.
                     loadOfflineNebula();
                 }
             });
         }

     @Override
     public void onCreate() {
     super.onCreate();
     QuinoxlessFramework.init();
     }
     private void loadOfflineNebula() {
     new Thread(new Runnable() {
     @Override
     public void run() {
     // This method blocks calls of methods. Do not call the methods that are embedded i
n the offline package from the main thread. If multiple AMR packages are embedded, make
sure that all the files exist. If one file does not exist, all the other embedded
offline packages fail.
     // This method can be called only once. If it is called multiple times, only the fi
rst call is valid.
     MPNebula.loadOfflineNebula("h5_json.json", new
MPNebulaOfflineInfo("80000000_1.0.0.0.amr", "80000000", "1.0.0.0"));
     }
     }).start();
     }

 }
```

Start an offline package

This section describes how to start an offline package.

Prerequisite

Your offline package is preset on the client.

Procedure

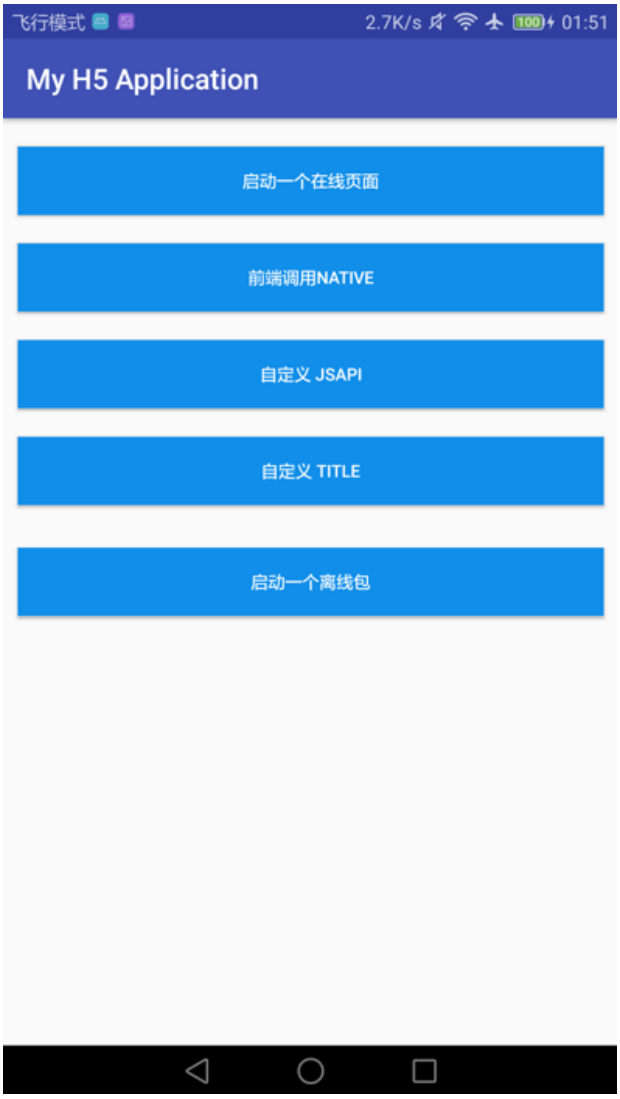
1. In the activity_main.xml file, create a button and set the button id to `start_app_btn`.

```
<Button
 android:id="@+id/start_app_btn"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:layout_marginBottom="10dp"
 android:layout_marginLeft="10dp"
 android:layout_marginRight="10dp"
 android:layout_marginTop="20dp"
 android:background="#108EE9"
 android:gravity="center"
 android:text="Start Offline Package"
 android:textColor="#ffffff" />
```

2. In the `MainActivity` class, specify the system behavior upon a tap on the `start_app_btn` button: start the offline package. In the following sample code, the parameter “80000000” is the Application ID of the offline package.

```
findViewById(R.id.start_app_btn).setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        MPNebula.startApp("80000000");
    }
});
```

3. After compiling the project, install the Application on a cell phone. After you start the Application, the following user interface appears.



4. Tap the **Start Offline Package** button. The web page that is preset in the offline package appears. Now, you have **started the offline package**.

Update an offline package

This section describes how to update an offline package.

Prerequisites

Your offline package is preset in the Application on your client. A new HTML5 Application is created in the mPaaS console and a new offline package is uploaded.

Procedure

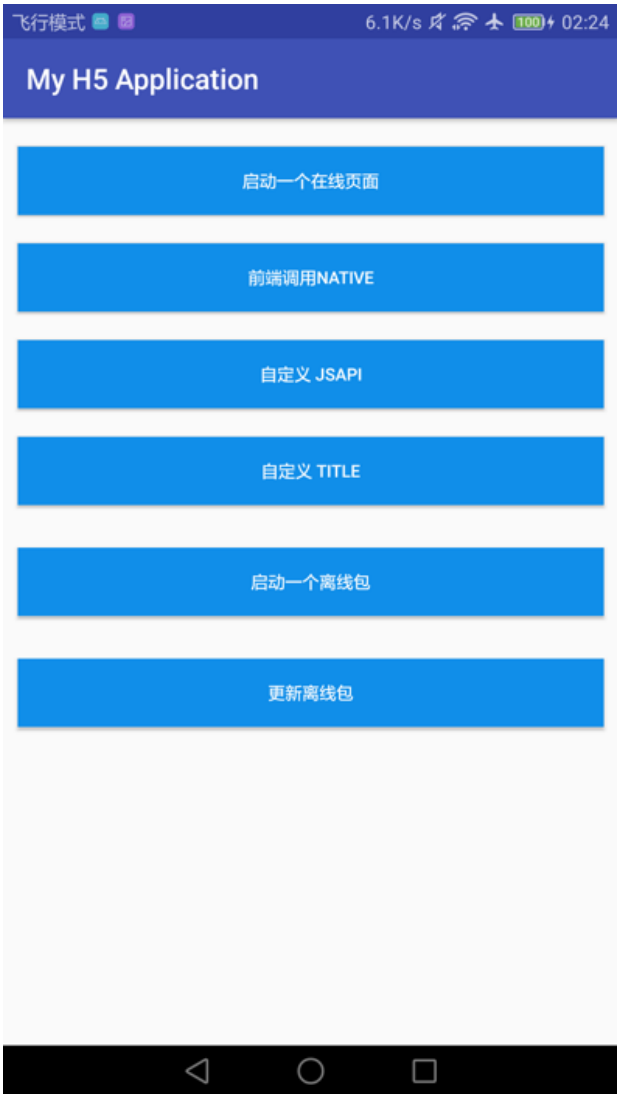
1. In the activity_main.xml file, create a button and set the button id to `update_app_btn`.

```
<Button
 android:id="@+id/update_app_btn"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:layout_marginBottom="10dp"
 android:layout_marginLeft="10dp"
 android:layout_marginRight="10dp"
 android:layout_marginTop="20dp"
 android:background="#108EE9"
 android:gravity="center"
 android:text="Update Offline Package"
 android:textColor="#ffffff" />
```

2. In the `MainActivity` class, specify the system behavior upon a tap on the `update_app_btn` button: start the offline package. In the following sample code, the parameter “80000000” is the Application ID of the offline package.

```
findViewById(R.id.update_app_btn).setOnClickListener(new View.OnClickListener() {
 @Override
 public void onClick(View v) {
     MPNebula.updateAllApp(new MpaasNebulaUpdateCallback() {
         @Override
         public void onResult(final boolean success, final boolean isLimit) {
             // The "success?" in the following code indicates whether the update is
successful.
             runOnUiThread(new Runnable() {
                 @Override
                 public void run() {
                     Toast.makeText(MainActivity.this, success ? "Offline package update
succeeded": "Offline package update failed", Toast.LENGTH_SHORT).show();
                 }
             });
         }
     });
 }
});
```

3. After compiling the project, install the Application on a cell phone. After you start the Application, the following user interface appears.



4. Tap the **Update Offline Package** button to open and update the offline package. After a prompt appears and indicates that the update is successful, tap the **Start Offline Package** button. The web page that is preset in the updated offline package appears. Now, you have **updated the offline package**.

1.8. FAQ

1.8.1. Android FAQ

What is the difference between interceptEvent and handleEvent in the process of customizing JSAPI?

A: If you want to monitor other events of the container, write them in `interceptEvent`; if you want to handle the events, write them in `handleEvent`, and that is OK if the interface returns true.

If the interface returns true, the system stops passing the events; if false, the events are continuously passed to other plugins.

The events have been added in config.setEvents("event"); in the process of customizing JSAPI, why it is still required to add the events once again in onPrepare of the plugin ?

A: Because the container plugin is lazily loaded, namely it is loaded upon the page creation. HTML5 Container injects the events to be monitored through external `config.setEvents`, so only when JS call occurs, the corresponding plugin object can be instantiated. For the actual instantiated plugin, the events distributed are those in `onPrepare` in the plugin. So, you must ensure that the event of `config.setEvents("event")` must be consistent with the event in internal `onPrepare`.

What is the difference among page, session and service in the process of customizing JSAPI plugin?

A: A page corresponds to a webview; a session corresponds to an App object in mPaaS; a service is a global singleton.

- If the plugin is registered as page level, a plugin instance is created every time you create a WebView, and the `onRelease` of the corresponding plugin is called back when the WebView is disposed.
- If the plugin is registered as session level, a plugin is created every time you create an App object.
- If the plugin is registered as service level, only one plugin is created globally, and the plugin is created when your open the container for the first time.

1.8.2. iOS FAQ

This article introduces common problems and corresponding solutions during the process of integrating to iOS.

Note:

- To facilitate unified processing, we recommend that you specify the base class of all HTML5 pages and WebViews during initialization of the HTML5 container.

```
H5Demo 〉 H5Demo 〉 Sources 〉 NebulaDemo 〉 SDK接入 〉 H5Container.m 〉 +configNebulaSDK

71
72  + (void)configNebulaSDK
73  {
74      // 控件创建
75      [NBServiceConfigurationGet() setViewControllerClass:[H5WebViewController class]];
76      [NBServiceConfigurationGet() setContentViewClass:[MPWebView class]];
77
78      // 自定义的插件配置，所有自定义的jsapi和plugin都保存在DemoPresetApps.bundle/Poseidon-Extra-Config.plist中
79      NSString *filePath = [[NSBundle mainBundle].bundlePath stringByAppendingFormat:@"/%@/%@",
            @"DemoPlugins.bundle", @"Poseidon-Extra-Config.plist"];
80      [NBServiceConfigurationGet() setExtraPluginsFilePath:filePath];
81
82      // 预置包配置，所有预置的离线包在DemoPresetApps.bundle中，预置的包信息在 DemoPresetApps.bundle/
            NAMApplist.plist 中
83      [NBServiceConfigurationGet().appConfig setPresetApplistPath:[[NSBundle mainBundle] pathForResource:
            [NSString stringWithFormat:@"DemoPresetApps.bundle/NAMApplist.plist"] ofType:nil]];
84      [NBServiceConfigurationGet().appConfig setPresetAppPackagePath:[[NSBundle mainBundle]
            pathForResource:@"DemoPresetApps.bundle" ofType:nil]];
85
86      // 包管理服务器配置
87      NSString *version = [[[NSBundle mainBundle] infoDictionary] objectForKey:@"Product Version"];
88      [NBServiceConfigurationGet() setClientVersion:version];
89      NSString *bundleId = [[[NSBundle mainBundle] infoDictionary] objectForKey:@"CFBundleIdentifier"];
90      [NBServiceConfigurationGet() setClientBundleId:bundleId];
91      [NBServiceConfigurationGet() setUserAgent:[NSString stringWithFormat:@"H5Demo %@", version]];//指定
            userAgent
92
93      // 离线包管理设置（验签公钥、公共资源包、离线包下载器）
94      NSString *pubpem = [[NSBundle mainBundle] pathForResource:@"public_key" ofType:@"pem"];
95      [NBServiceConfigurationGet().appConfig setSignPublicKey: [NSString stringWithContentsOfFile:pubpem
            encoding:NSUTF8StringEncoding error:nil]];// 设置离线包验签公钥
```

- When the frontend is developing an offline package, make sure that Chinese characters are not contained in the paths and file names. Otherwise, the client fails to decompress the offline package.
- The absolute length of each resource path in offline packages must not exceed 100 characters. Otherwise, the client fails to decompress .tar packages and the pages are blank.

Why does the offline package file named H5_json.json not take effect?

Answer: Baseline 10.1.32 supports only the PList format. Baseline 10.1.60 supports both the PList and JSON formats.

How do I obtain the app information of an installed offline package?

Answer: Execute the code defined based on the following sample code: `NSDictionary *installedApps = [NAMServiceGet() installApps:nil];` .

How do I force update the information about all offline packages?

Answer: Use the packaged `requestAllNebulaApps` method to perform a full update.

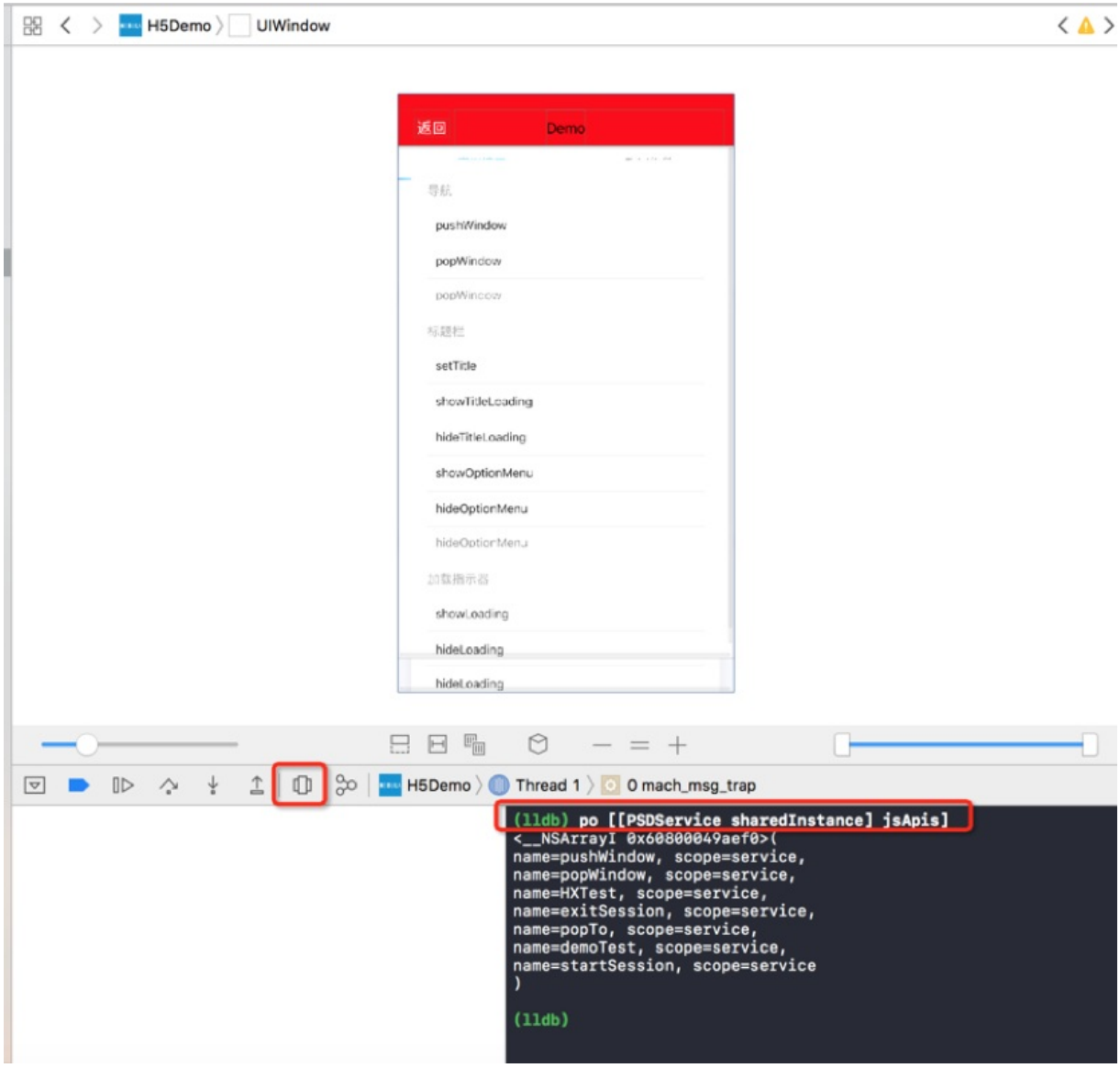
```
[[MPNebulaAdapterInterface shareInstance]requestAllNebulaApps:^(NSDictionary *data, NSEror *error) {

 }];
```

In iOS 13, why is the tvux-ui pulldown component freezes in the HTML5 container during swiping?

Answer: This problem is resulted from the bug of `UIWebview` in iOS. You can use WKWebview instead of the component or change the frontend component. For more information about how to use WKWebview instead of the component, see Adapt mPaaS 10.1.60 to WKWebView.

How do I check all JSAPI operations and plug-ins that are registered in the current app?

Answer: Open an HTML5 page. Go to a view-layer page in Xcode. In the lldb console, run `po [[PSDService sharedInstance] jsApis]` to view all registered JSAPI operations. Similarly, run `po [[PSDService sharedInstance] plugins]` to view all registered plug-ins.



How do I obtain the UIViewController and WebView objects of the current HTML5 page in the JSAPI or plug-in?

Answer: During execution, the plug-in can directly obtain the `event.context` parameter and the JSAPI operation can obtain the `context` parameter. In the `PSDContext` object, you can obtain all the information or references that you want. Such references include the references to the current `event.currentViewController` controller and the referenced `context.currentViewController.psdContentView` of the current WebView.

H5Demo 〉 H5Demo 〉 Sources 〉 NebulaDemo 〉 自定义扩展插件 〉 H5JsApi4DemoTest.m 〉 -handler:context:callback:

```
2  // H5JsApi4DemoTest.m
3  // NebulaDemo
4  //
5  // Created by Glance on 16/12/20.
6  // Copyright © 2016年 Alipay. All rights reserved.
7  //
8
9  #import "H5JsApi4DemoTest.h"
10 // 该JSApi已在Poseidon-Extra-Config.plist中注册
11
12 @implementation H5JsApi4DemoTest
13
14 - (void)handler:(NSDictionary *)data
15         context:(PSDContext *)context
16        callback:(PSDJsApiResponseCallbackBlock)callback
17 {
18     [super handler:data context:context callback:callback];
19     UIViewController *vc = context.currentViewController;
20     UIWebView *webView = (UIWebView *)context.currentViewController.psdContentView;
21     NSLog(@"[NEBULADEMO]:JSAPI调用，当前viewController %@, webView:%@, jsApi实例为 %@",vc, webView, self);
22
23     if (!data) {
24         ErrorCallback(callback, e_inavlid_params);
25         return;
26     }
27     else {
28         callback(@{@"result":@"jsapi-demoTest来自Native的处理结果"});
29     }
30
31 }
```

```
H5Demo 〉 H5Demo 〉 Sources 〉 NebulaDemo 〉 自定义扩展插件 〉 H5Plugin4DemoTest.m 〉 -handleEvent:
                withListener:self
                 useCapture:NO];
    [super pluginDidLoad];
}

- (void)addJSApis
{
    [super addJSApis];
    // 这种jsapi的注册是在plist中配置之外的另一种方式
    PSDJsApi *jsApi4DemoTest2 = [PSDJsApi jsApi:@"demoTest2"
                                 handler:^(NSDictionary *data, PSDContext *context, PSDJsApiResponseCallbackBlock responseCall
                                    {
                                    responseCallbackBlock(@{@"result":@"jsapi-demoTest2调用Native的处理结果"});
                                 }
                              checkParams:NO
                               isPrivate:NO
                                  scope:self.scope];
    [self registerJsApi2Target:jsApi4DemoTest2];
}

- (void)handleEvent:(PSDEvent *)event
{
    [super handleEvent:event];
    UIViewController *vc = event.context.currentViewController;
    UIWebView *webView = (UIWebView *)event.context.currentViewController.psdContentView;
    NSString *eventType = event.eventType;
    NSLog(@"[NEBULADEMO]:有事件抛出，当前viewController %@, webView:%@, 事件名为 %@",vc, webView, eventType);

}
```

How do I obtain the WebView of the current HTML5 page?

Answer: The WebView of the current HTML5 page is an attribute of the current ViewController. You can obtain the WebView by using `vc.psdContentView`. In the JSAPI or plug-in, the ViewController of the current page can be obtained by using the preceding method.

> **Note**
>
> When you want to obtain the WebView of the current page in the base class of HTML5 pages, use the viewWillAppear method. The WebView parameter is not created in the viewDidLoad method. If you use this method, the returned value is nil.

```
//
// H5WebViewController.m
// NebulaDemo
//
// Created by Glance on 16/12/14.
// Copyright © 2016年 Alipay. All rights reserved.
//

#import "H5WebViewController.h"

@interface H5WebViewController ()<PSDPluginProtocol>

@end

@implementation H5WebViewController

#pragma mark – UIViewController LifeCycle

– (void)viewDidLoad {
    [super viewDidLoad];
    NSLog(@"[NebulaDemo]: 容器中的一个Scene被打开");
}

– (void)viewWillAppear:(BOOL)animated
{
    [super viewWillAppear:animated];

    UIWebView *webView = (UIWebView *)self.psdContentView;
}
```

Unused variable 'webView'

How do I obtain the startup parameters passed in by the frontend when the current page is loaded?

Answer: Obtain the `psdScene.createParam.expandParams` attribute of the current ViewController.

```
H5Demo 〉 H5...mo 〉 Sources 〉 Ne...mo 〉 SD...入 〉 H5WebViewController.m 〉 H5WebViewController
6  //
7  //
8
9  #import "H5WebViewController.h"
10
11 @interface H5WebViewController ()<PSDPluginProtc
12
13 @end
14
15 @implementation H5WebViewController
16
17 #pragma mark - UIViewController LifeCycle
18
19 - (void)viewDidLoad {
20     [super viewDidLoad];
21     NSLog(@"[NebulaDemo]: 容器中的一个Scene被打开");
22 }
23
24 - (void)viewWillAppear:(BOOL)animated
25 {
26     [super viewWillAppear:animated];
27
28
29     UIWebView *webView = (UIWebView * self.psdContentView;        ⚠ Unused variable 'webView'
30
31
32     NSDictionary *expandParams = self.psdScene.createParam.expandParams;  ⚠ Unused variable 'expandPar...
33 }
34
```

How do I limit JSAPI to taking effect only for a specified offline package?

Answer: In the execution method of JSAPI, obtain the appId of the offline package to which the current page belongs and determine whether to execute the logic.

```
H5Demo 〉 H5...mo 〉 So...es 〉 Ne...mo 〉 自...件 〉 H5JsApi4DemoTest.m 〉 -handler:context:callback:
3  // NebulaDemo
4  //
5  // Created by Glance on 16/12/20.
6  // Copyright © 2016年 Alipay. All rights reserved.
7  //
8
9  #import "H5JsApi4DemoTest.h"
10 // 该JSApi已在Poseidon-Extra-Config.plist中注册
11
12 @implementation H5JsApi4DemoTest
13
14 - (void)handler:(NSDictionary *)data
15         context:(PSDContext *)context
16        callback:(PSDJsApiResponseCallbackBlock)callback
17 {
18     [super handler:data context:context callback:callback];
19     UIViewController *vc = context.currentViewController;
20     UIWebView *webView = (UIWebView *)context.currentViewController.psdContentView;
21     NSLog(@"[NEBULADEMO]:JSAPI调用，当前viewController %@, webView:%@, jsApi实例为 %@",vc, webView,
           self);
22
23     NSString *appId = context.currentSession.createParam.expandParams[@"appId"];   ⚠ Unused variable 'appId'
24
25     if (!data) {
26         ErrorCallback(callback, e_inavlid_params);
27         return;
28     }
29     else {
30         callback(@{@"result":@"jsapi-demoTest来自Native的处理结果"});
31     }
32
33 }
```

How do I intercept a page URL?

Answer: You can customize the plug-in to listen to events.

- Listen to specified events: `[PSDProxy addEventListener:kEvent_Proxy_Request_Start_Handler withListener:self useCapture:YES];` .
- Intercept and process the events.

```
else if ([kEvent_Proxy_Request_Start_Handler isEqualToString:event.eventType]
        && [event isKindOfClass:[PSDProxyEvent class]] ){

        NSLog(@"-----kNBEvent_Scene_NavigationItem_Right_Setting_Click----");
        PSDProxyEvent *proxyEvent = (PSDProxyEvent*) event;
        NSMutableURLRequest *redirectReq = proxyEvent.request.mutableCopy;
        NSString *appId = event.context.currentSession.createParam.expandParams[@"appId"];

        NAMApp *app = [NAMServiceGet() findApp:appId version:nil];
        NSString *fallBackUrl = app.fallback_host;
        NSString *vhost = app.vhost;;
        NSString *url = redirectReq.URL.absoluteString;


       NSLog(@"url__%@_______fallBackUrl:%@",url,fallBackUrl);
       if ([url containsString:@"www.baidu.com"]) {
           [proxyEvent preventDefault];

 }}
```

How do I intercept the URL of the current page before the current HTML5 page is loaded?

Answer: In the base class of HTML5 pages, in the proxy method that implements the lifecycle of UIWebView, listen to the `kEvent_Navigation_Start` event, intercept it before the page is loaded, and obtain the WebView and URL of the current page to perform processing.

```
#pragma mark - 对应 UIWebViewDelegate 的委托实现

-(void)handleEvent:(PSDEEvent *)event
{
  if(![[event.context currentViewController] isEqual:self]){
    return;
  }
  if([kEvent_Navigation_Start isEqualToString:event.eventType]){
    BOOL shouldStart = [self handleContentViewShouldStartLoad:(id)event];

    if(!shouldStart){
      [event preventDefault];
    }
  }
  else if([kEvent_Page_Load_Start isEqualToString:event.eventType]){
    [self handleContentViewDidStartLoad:(id)event];
  }
  else if([kEvent_Page_Load_Complete isEqualToString:event.eventType]){
    [self handleContentViewDidFinishLoad:(id)event];
  }
  else if([kEvent_Navigation_Error isEqualToString:event.eventType]){
    [self handleContentViewDidFailLoad:(id)event];
  }
  else if([kEvent_Sence_NavigationItem_Left_Back_Click isEqualToString:event.eventType]){

  }

  -(BOOL)handleContentViewShouldStartLoad:(PSDNavigationEvent *)event{
    return YES;
  }
}
```

How do I manually call a JSAPI operation in Native?

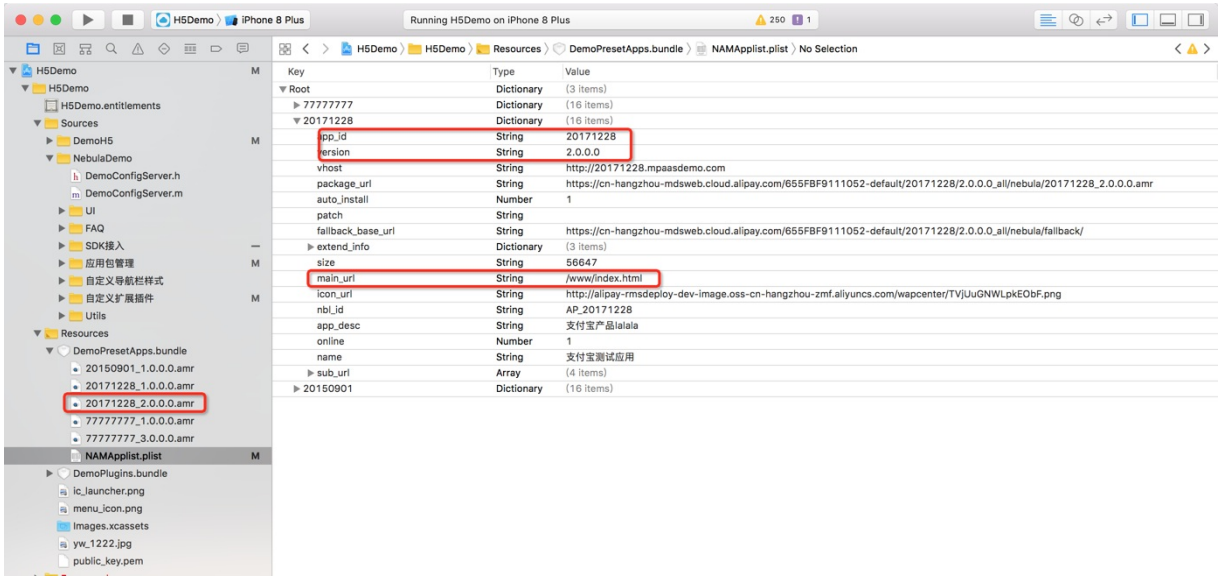
Answer: Sometimes, on the Native side, you may need to manually call a specified JSAPI operation on the current page. This can be achieved by calling the operation of the current ViewController shown in the following figure.

```
  if(self.navigationController){
    NSMutableArray *aar = [[self.navigationController viewControllers] mutableCopy];
    [arr removelastObject];

    [self.navigationController setViewControllers:aee animated:YES];
  }
  else{
    [self dismissViewControllerAnimated:YES completion:NULL];
  }
}

-(IBAction)btnRightItemClicked:(id)sender
{
  NSLog(@"[NebuiaDemo]:导航栏右侧按钮被点击":);

  //native 手动触发 JSAPI
  [self callHandler:@"HXTest" data:@{@"key":@"value"} responseCallback:^(id responseData){

  }];
}
```

Why does a local preset offline package fail to be loaded?

Answer: The failure in loading a preset resource package is generally caused by the mismatching between the preset package version and the package information. Before you test the local preset offline package, close the network connection to prevent the offline package from being updated. This ensures that version that is locally preset is loaded.

Check whether the information about the locally preset offline package is consistent with the package information configured in the PList file. Such information includes the appID, version, and main_url.

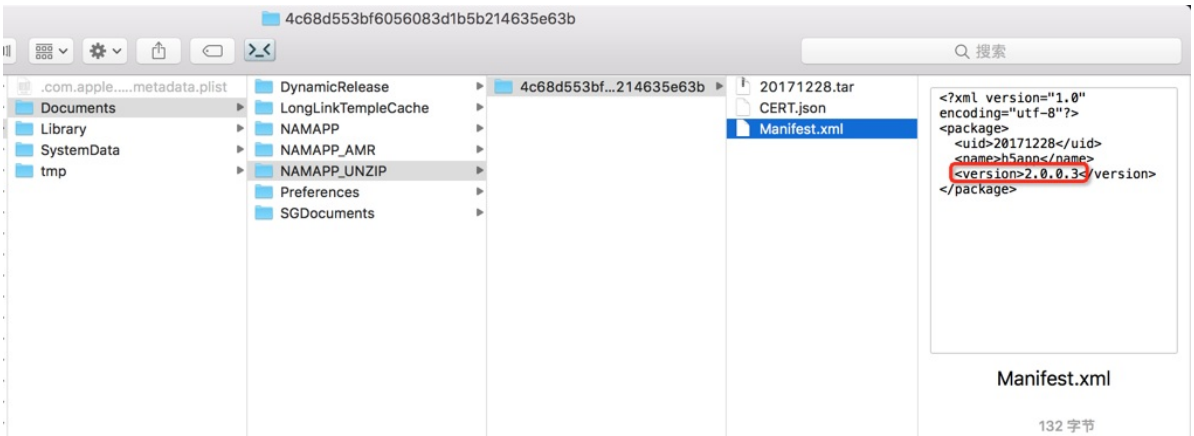


On the client, why do I fail to load a package with a new version that is released in the console?

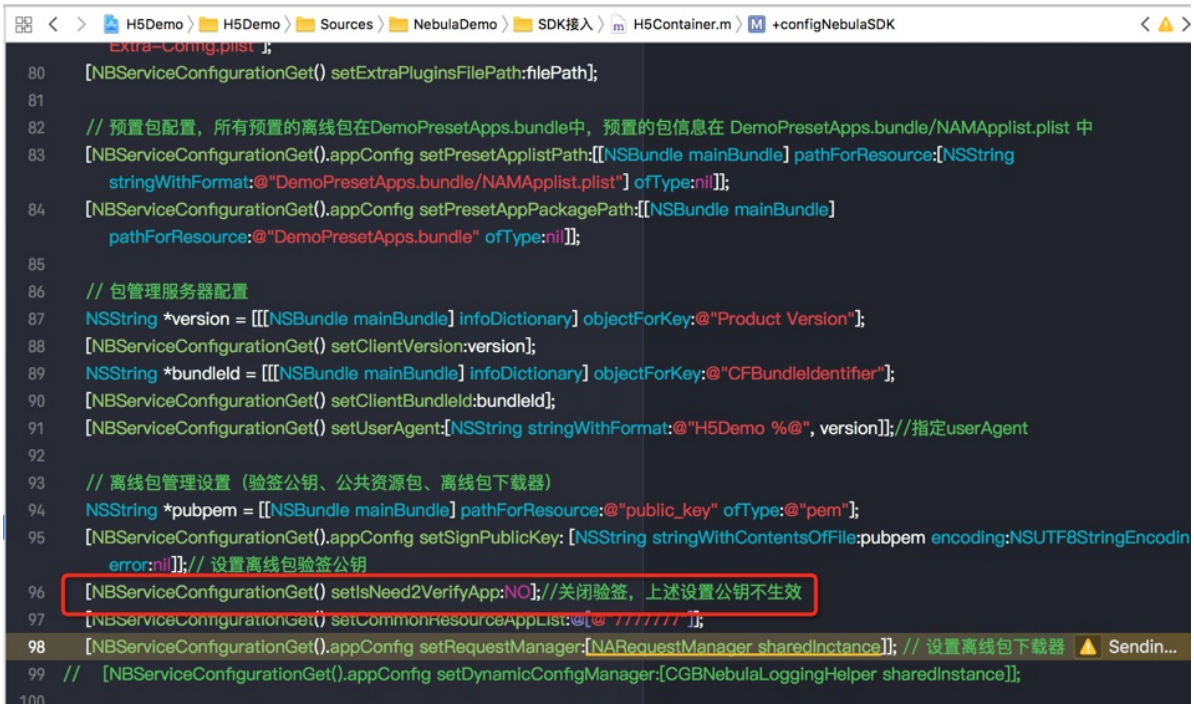
Answer: Before you view the solution to this problem, ensure that you have understood How to update an offline package. If the client cannot load a new package, errors may occur in all rendering phases of the offline package. The following code shows how to locate the error.

1. Check the response of the RPC to fully update the offline package. In the console, search for “bizType: 4” to view the details of the returned offline package. Make sure that the latest package information released in the console is pulled.

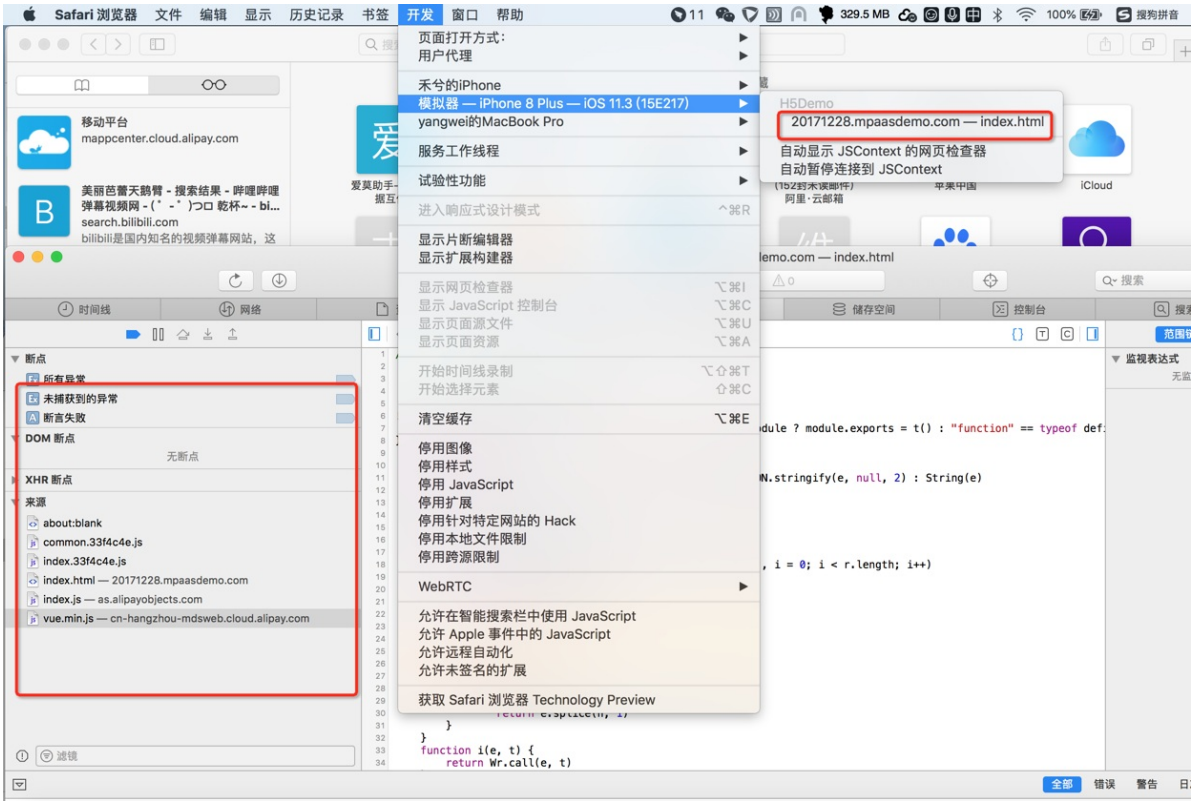
2. View the offline package information in the finish callback method for loading the offline package. Make sure that the offline package is the latest package released by the console, and the error parameter is set to nil. Make sure that the appID, version, and main_url of the offline package are correct.
3. Check whether the offline package is decompressed in the sandbox directory. If the current offline package references the content of the global resource package, the global resource package must also exist in the directory.



4. If the offline package does not exist in the sandbox directory in the previous step, you can temporarily disable signature verification for the offline package, delete the app, and run the client again. If loading is normal after signature verification is disabled, the private key of the offline package that is used for signature is inconsistent with the public key of the client that is used for verification. Update the public key information of the client.



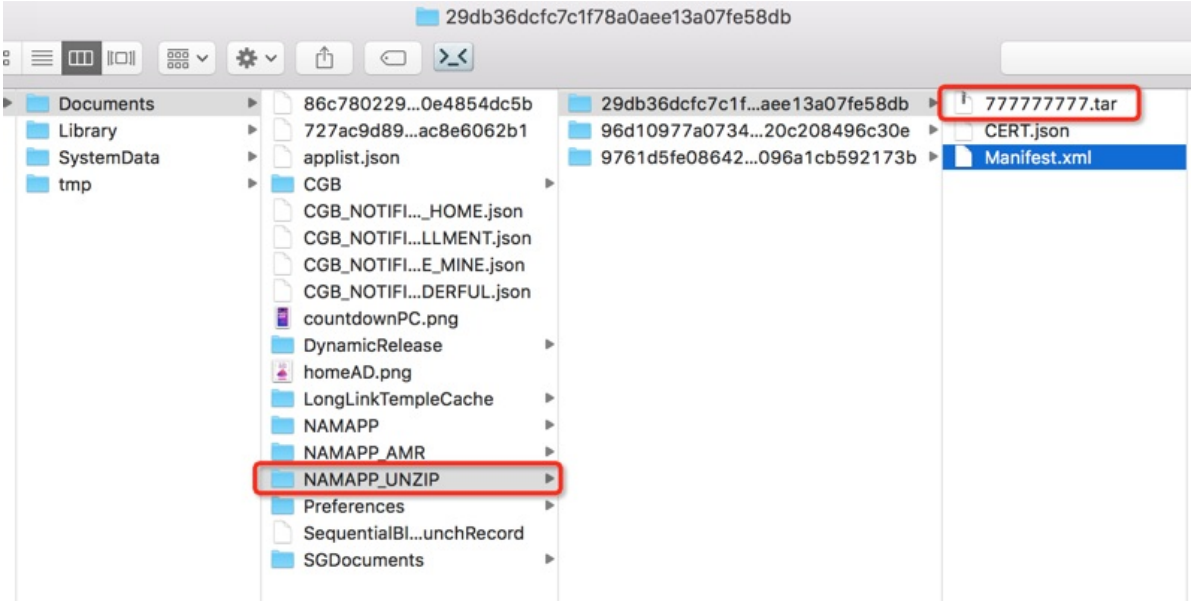
5. After all the preceding steps, if the offline package still fails to be updated, you can use Safari to debug the HTML5 page to locate the cause for the error. For example, check whether the path to the global resource package is correct and whether the URL for loading the offline package exists.



Why does a global resource package fail to be loaded?

Answer: If the debugging results on Safari show that the global resource package fails to be loaded, you can perform the following steps to troubleshoot:

1. Verify that the global resource package is registered.
2. Check whether the offline package is decompressed in the sandbox directory.



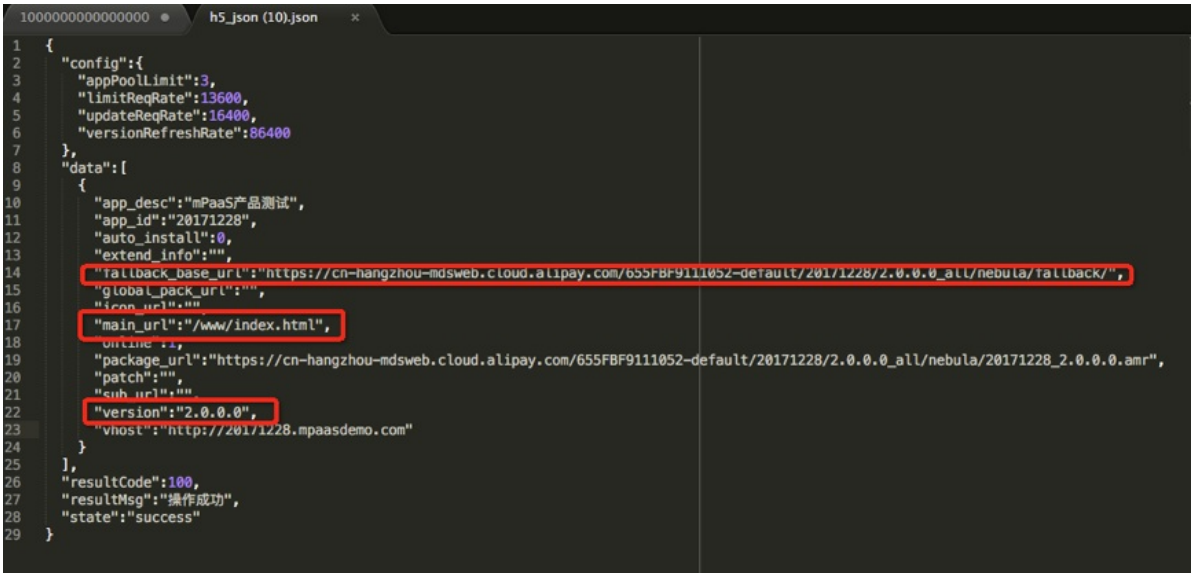
3. Check whether the path to the referenced global resource package is valid. Ensure that no Chinese characters exist in the reference path.

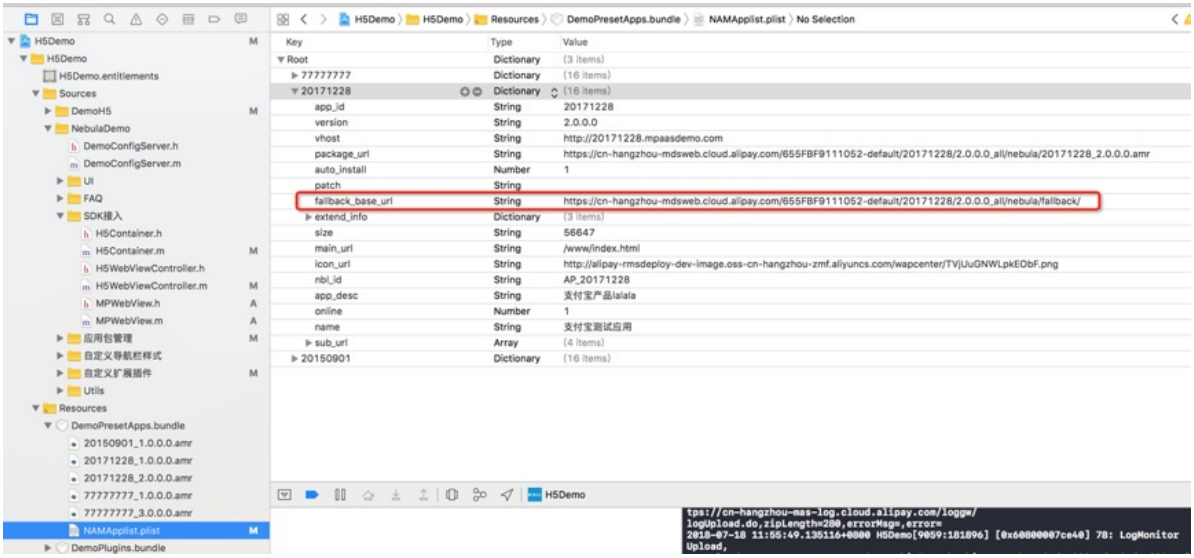
Why does a white screen or a 400 error occurs after the HTML5 page of an offline package is opened?

Answer: A white screen or a 400 error is generally caused by the failure in loading a local offline package. When the failure occurs, the online fallback address is used, while the fallback address of the corresponding page does not exist.As a result, the HTML5 page fails to be loaded. Perform the following steps to troubleshoot:

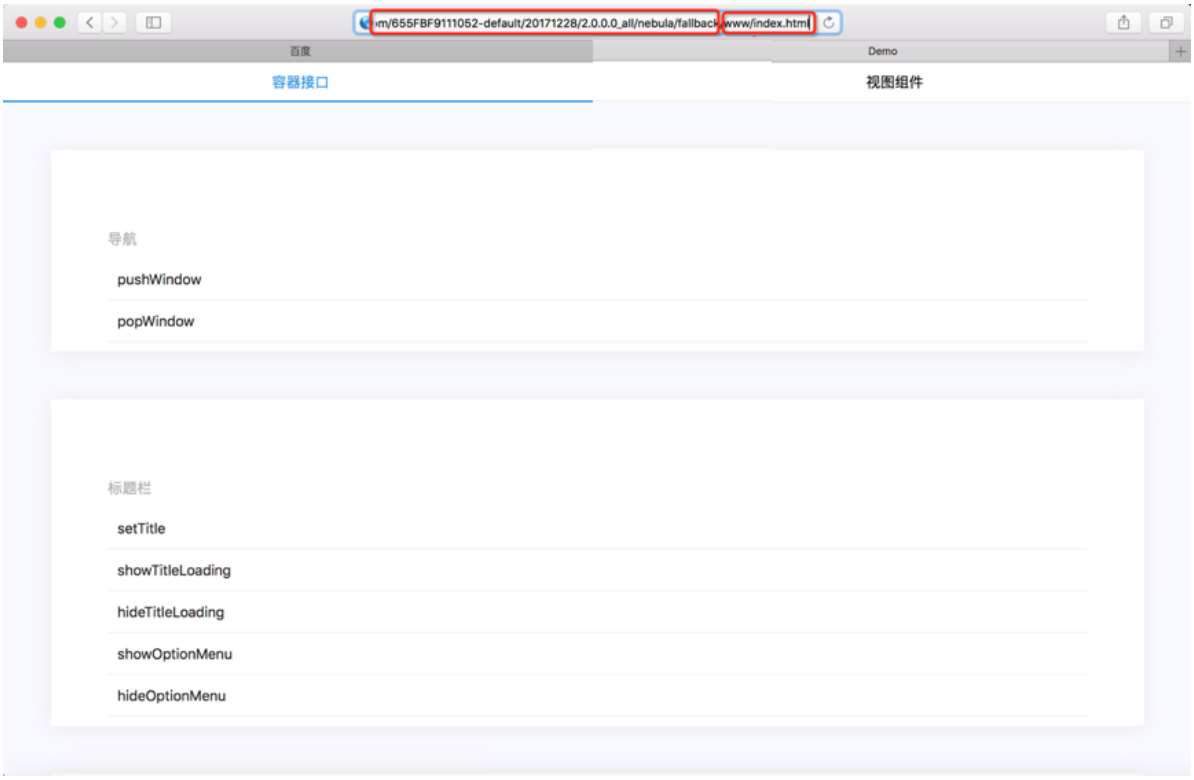
1. Locate the cause for the loading failure of the offline package on the client based on the preceding troubleshooting steps for offline packages.
2. If the online fallback address fails to be loaded, verify that the offline package with the specified version is generated and both the common offline package and the global resource package are uploaded in the console. For more information, see Generating an offline package.
3. If the online fallback address fails to be loaded for the preset offline package, make sure that the preset offline package is also uploaded in the console.

4. Make sure that the `fallback_base_url` parameter in the information about the local preset package is consistent with the `fallback_base_url` parameter in the configuration file h5_json.json that is downloaded in the console.





5. In addition, verify that the spliced address specified by `fallback_base_url + main_url` can be loaded in browsers.



How do I disable the Swipe Back gesture on HTML5 pages?

Answer: You can disable the gesture on a frontend HTML5 page and the base class of the native HTML5 container.

1. Disable the gesture on a frontend HTML5 page: Call the setGestureBack JSAPI. This method is applicable to the scenario where you want to disable the Swipe Back gesture on a specified page. `AlipayJSBridge.call('setGestureBack',{val:false});` .
2. Disable the gesture on the base class of the native HTML5 container: In the viewDidAppear method of the base class, call the system API operation for disabling the Swipe Back gesture. In addition, set the guestBack parameter. This method is applicable to scenarios where you want to disable the Swipe Back gesture on multiple or all HTML5 pages.

```
- (void)viewDidAppear:(BOOL)animated {
     [super viewDidAppear:animated];

     self.options.gestureBack = NO;
     if ([self.navigationController
respondsToSelector:@selector(interactivePopGestureRecognizer)]) {
         self.navigationController.interactivePopGestureRecognizer.enabled = NO;
     }
 }
```

How do I check whether the current page is a Mini Program page?

Answer: Obtain the session where the current page is located and then call the `isTinyAppWithSession` operation to check whether the current page is a Mini Program page. The following sample code is for your reference.

```
PSDSession *session = self.psdSession;
BOOL isTinyApp = [NBUtils isTinyAppWithSession:session];
```

How do I pass custom parameters on an HTML5 page?

Answer: The methods for passing custom parameters vary depending on scenarios.

- Native - HTML5 page: Call the `startH5ViewControllerWithParams` method to pass the following parameters: `[[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithParams:@{@"url": @"https://tech.antfin.com", @"key1":@"value1"}];` .
- Native - offline package: Call the `startH5ViewControllerWithNebulaApp` method to pass the following parameters: `[[MPNebulaAdapterInterface shareInstance] startH5ViewControllerWithNebulaApp:@{@"appId":@"70000000",@"param":@{@"key2":@"value2"}}];` .
- HTML5 page - HTML5 page: Call the `pushWindow` operation with the custom parameter specified in `passData` .

```
AlipayJSBridge.call('pushWindow', {
// URL of the page to be opened.
url: 'https://m.taobao.com/',
// Configuration parameters of the opened page
param: {
  readTitle: true, // Automatically reads the title.
  showOptionMenu: false, // Hides the right-side menu.
transparentTitle:'always',
},
// Optional, used for transferring parameters to a newly opened page.
// Use AlipayJSBridge.startupParams to obtain passData on a newly opened page.
passData: {
  key1: "key1Value",
  key2: "key2Value"
}
});
```

- HTML5 page - offline package: Call the JSAPI operation startApp with the custom parameter specified in the param parameter.

```
AlipayJSBridge.call('startApp', {
appId: '70000000',
param: {
key1:'value1'
}
}, function(result) {
// noop
});
```

How do I obtain a passed parameter from an HTML5 page?

Answer: The methods vary depending on scenarios.

- Obtain the passed parameter from the frontend: Call the startupParams method.

```
// The startup parameters of the current page.
AlipayJSBridge.startupParams
```

- Obtain the passed parameter from the native component: Obtain the passed parameter from the ViewController object where the current page is located.

```
// The startup parameters of the current page.
NSDictionary *expandParams = self.psdScene.createParam.expandParams;
NSLog(@"[mpaas] expandParams: %@", expandParams);
```

How do I intercept a JSAPI call?

Answer: You can customize the plug-in to listen to events.

- Listen to the event specified by using `kEvent_Invocation_Event_Start`.
- Intercept and process the event. Obtain the name of the JSAPI operation and the passed parameters.

```
else if([kEvent_Invocation_Event_Start isEqualToString:event.eventType]) {
      PSDInvocationEvent * invocationEvent = (PSDInvocationEvent *)event;
      NSString * apiName = invocationEvent.invocationName;
      if([apiName isEqualToString:@"setOptionMenu"] || [apiName
isEqualToString:@"showOptionMenu"] ) {
          NSLog(@"wwww");
      }
  }
```