Common error codes

The following table lists all common error codes for Mini Program OpenAPIs. If you do not find an error code in the following table, it means that the error code is not common, but dedicated to a specific OpenAPI.

| | | | | --- | --- | --- | --- | | **Error code** | **Result status** | **Error message** | **Further action** | | PROCESS_FAIL | F | A general business failure occurred. Do not retry. | Contact technical support to troubleshoot the issue. | | PARAM_ILLEGAL | F | Illegal parameters. For example, non-numeric input, invalid date. | Check and verify whether the request fields (including the header fields and body fields) of the current API are correct and valid. For example, /v1/authorizations/applyToken. | | INVALID_API | F | The called API is invalid or not active. | Check whether the current API name is used by mistakes when the API is called. | | ACCESS_DENIED | F | Access is denied | Need to check the *resultMessage* of the current API specification for details. For example, /v1/authorizations/applyToken. | | REQUEST_TRAFFIC_EXCEED_LIMIT | F | The request traffic exceeds the limit. | - The party that calls APIs needs to reduce the API calling frequency, or
- The API service provider needs to increase the traffic limit or threshold. | | UNKNOWN_EXCEPTION | U | An API calling is failed, which is caused by unknown reasons. | Trying to recall the API might help to resolve the issue. |

API-specific error codes

For error codes that are dedicated to a specific API, see the **Error codes** section in each API specification, for example, /v1/authorizations/applyToken.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/error_codesv2

Event Introduction {#event-introduction}

Last updated: 2021-05-10

Path: miniprogram_gcash

Event Introduction

2021-05-10 03:43

What is event?

- Event is the way of communication from the view layer to logic layer.
- The event feeds user behaviors back to logic layer for handling.

- Event can be bound to component. When the trigger condition happens, the corresponding event handler is executed in the logic layer.
- Event object can carry additional information, such as id, dataset and touches.

Usage

If you want to bind an event handler in a component, for example `onTap`, you need to define the `onTap` function in the `Page` object in the relative `.js` file.

copy

```
<view id="tapTest" data-hi="Mini Program" onTap="tapName">
  <view id="tapTestInner" data-hi="Mini Program Inner">
    Click me!
  </view>
</view>
```

In the relative `Page`, `tapName` should be defined to handle the event, and the parameter of the function is `event`.

copy

```
Page({
  tapName(event) {
    console.log(event);
  },
});
```

The console output

copy

```
{
  "type": "tap",
  "timeStamp": 1550561469952,
  "target": {
    "id": "tapTestInner",
    "dataset": {
      "hi": "Mini Program"
    },
    "targetDataset": {
      "hi": "Mini Program Inner"
    }
  },
  "currentTarget": {
    "id": "tapTest",
    "dataset": {
      "hi": "Mini Program"
    }
  }
}
```

In the use of components (basic component, extended component and custom component), the events available in the component depends on the support of the component itself. The supported events are specified clearly in the document of the specific component.

Event Type

The events fall into bubbling events and non-bubbling events:

1. **Bubbling events**: With `on` as the prefix, when the event of a component is triggered, the event is transferred to parent node.

2. **Non-bubbling events**: With `catch` as the prefix, when the event of a component is triggered, the event is not transferred to parent node.

The event binding is written is the same as component attribute in form of `key, value`.

- The key starts with `on` or `catch`, followed by event type, such as onTap and catchTap.

- The value is a string corresponding to the name of function defined in the Page. Error is reported when no trigger event exists.

copy

```
<view id="outter" onTap="handleTap1">
 view1
 <view id="middle" catchTap="handleTap2">
   view2
   <view id="inner" onTap="handleTap3">
     view3
   </view>
 </view>
</view>
```

In the above codes, clicking view3 triggers handleTap3 and handleTap2 (because tap event bubbles to view2m while view2 prevents tap event bubbling and does not transfer to parent node). Clicking view2 triggers handleTap2; clicking view1 triggers handleTap1.

All bubbling events:

| | | --- | --- | | **Type** | **Trigger condition** | | touchStart | Start of touch action. | | touchMove | Move after touch. | | touchEnd | End of touch action. | | touchCancel | Touch action interrupted, such as incoming call and pop-up. | | tap | Touch and leave immediately. | | longTap | Touch and leave after 500ms. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_event-system_event-introduction

---

Event Introduction {#event-introduction}

Last updated: 2022-07-03

Path: miniprogram_gcash

Event Introduction

2022-07-03 18:44

What is event?

- Event is the way of communication from the view layer to logic layer.
- The event feeds user behaviors back to logic layer for handling.
- Event can be bound to component. When the trigger condition happens, the corresponding event handler is executed in the logic layer.
- Event object can carry additional information, such as id, dataset and touches.

Usage

If you want to bind an event handler in a component, for example `onTap`, you need to define the `onTap` function in the `Page` object in the relative `.js` file.

copy

```
<view id="tapTest" data-hi="Mini Program" onTap="tapName">
  <view id="tapTestInner" data-hi="Mini Program Inner">
    Click me!
  </view>
</view>
```

In the relative `Page`, `tapName` should be defined to handle the event, and the parameter of the function is `event`.

copy

```
Page({
  tapName(event) {
    console.log(event);
  },
});
```

The console output

copy

```
{
  "type": "tap",
  "timeStamp": 1550561469952,
  "target": {
    "id": "tapTestInner",
    "dataset": {
      "hi": "Mini Program"
    },
    "targetDataset": {
      "hi": "Mini Program Inner"
    }
  },
  "currentTarget": {
    "id": "tapTest",
    "dataset": {
      "hi": "Mini Program"
    }
  }
}
```

In the use of components (basic component, extended component and custom component), the events available in the component depends on the support of the component itself. The supported events are specified clearly in the document of the specific component.

Event Type

The events fall into bubbling events and non-bubbling events:

1. **Bubbling events**: With `on` as the prefix, when the event of a component is triggered, the event is transferred to parent node.

2. **Non-bubbling events**: With `catch` as the prefix, when the event of a component is triggered, the event is not transferred to parent node.

The event binding is written is the same as component attribute in form of `key, value`.

- The key starts with `on` or `catch`, followed by event type, such as onTap and catchTap.

- The value is a string corresponding to the name of function defined in the Page. Error is reported when no trigger event exists.

copy

```
<view id="outter" onTap="handleTap1">
 view1
 <view id="middle" catchTap="handleTap2">
   view2
   <view id="inner" onTap="handleTap3">
     view3
   </view>
```

```
 </view>
</view>
```

In the above codes, clicking view3 triggers handleTap3 and handleTap2 (because tap event bubbles to view2m while view2 prevents tap event bubbling and does not transfer to parent node). Clicking view2 triggers handleTap2; clicking view1 triggers handleTap1.

All bubbling events:

| | | --- | --- | | **Type** | **Trigger condition** | | touchStart | Start of touch action. | | touchMove | Move after touch. | | touchEnd | End of touch action. | | touchCancel | Touch action interrupted, such as incoming call and pop-up. | | tap | Touch and leave immediately. | | longTap | Touch and leave after 500ms. |

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_event-system_event-introduction

---

Event Object {#event-object}

Last updated: 2021-05-09

Path: miniprogram_gcash

Event Object

2021-05-09 18:43

When the component triggers the event, the event handler bounded with the logic layer receives an event object.

BaseEvent

BaseEvent basic event object attribute list:

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | type | String | Event type. | | timeStamp | Integer | Event generated timestamp. | | target | Object | Attribute value set of the component triggering the event. |

Type

Type: Event type

Timestamp

timeStamp: Event generated timestamp

Target

`dataset` define data in component, and the data is transferred via event to the logic layer. Start with `data-` and use hyphen `-` to connect multiple words which must be in lower case (upper case automatically converted into lower case). For example, the `data-element-type` will eventually convert the hyphen into hump elementType in the `event.target.dataset`.

Sample codes:

copy

```
<view data-alpha-beta="1" data-alphaBeta="2" onTap="bindViewTap">
DataSet Test </view>
```

copy

```
Page({
 bindViewTap:function(event){
   event.target.dataset.alphaBeta === 1 // - Will convert into hump
writing
   event.target.dataset.alphabeta === 2 // Upper case converted into
lower case
 }
})
```

Target: source component object that triggers the event, attribute list:

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | id | String | Event source component id. | | tagName | String | Current component type. | | dataset | Object | Set of custom attributes starting with **data-** on component bound with the event. | | targetDataset | Object | Set of custom attributes starting with **data-** on component actually triggering the event. |

CustomEvent

CustomEvent custom event object attribute list (inherited from BaseEvent)

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | detail | Object | Additional information. |

Detail

Data carried in custom event The form component event carries user entry information. For example, the switch component, when onChange trigger, gets user selected status value via event.detail.value. The media error event carries error information. **For details, see the component document event description**.

TouchEvent

TouchEvent touch event object attribute list (inherited from BaseEvent)

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | touches | Array | Array of touch point information staying current on the screen. | | changedTouches | Array | Array of touch point information changing currently. |

The touches is an array. Each of its elements is a Touch object (the touches carried in the **canvas** touch event is the CanvasTouch array), indicating the touch point staying on the screen.

changedTouches data format is the same as touches. Indicates changing touch point, such as from none to start (touchstart), location change (touchmove), from touch to end (touchend, touchcancel).

Touch Object

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | identifier | Number | Touch point identifier. | | pageX, pageY | Number | Distance to the document upper-left corner, the upper-left corner as origin, horizontal direction as x axis and vertical direction as y axis. | | clientX, clientY | Number | Distance to the displayable region of page (screen except for navigation bar), the upper-left corner as origin, horizontal direction as x axis and vertical direction as y axis. |

CanvasTouch Object

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | identifier | Number | Touch point identifier. | | x, y | Number | Distance to the Canvas upper-left corner, the Canvas upper-left corner as origin, horizontal direction as x axis and vertical direction as y axis. |

Sample

Take `touchMove` event as an example, when user touch the following component.

copy

```
<view class="move-view" onTouchMove="touchMoveHandle">
</view>
```

The `touchMoveHandle` will be invoked in the page, the TouchEvent will act as the parameter.

copy

```
Page({
  touchMoveHandle(e){
    console.log(e)
  }
});
```

The console output

copy

```
{
  "type": "touchMove",
  "timeStamp": 1562241425847,
  "target": {
    "targetDataset": {},
    "tagName": "view",
    "dataset": {},
    "offsetLeft": 0,
    "offsetTop": 0
  },
  "currentTarget": {
    "tagName": "view",
    "dataset": {},
    "offsetLeft": 0,
    "offsetTop": 0
  },
  "touches": [\
    {\
      "clientX": 49.69140625,\
      "clientY": 54.1640625,\
      "identifier": 0,\
      "pageX": 49.69140625,\
      "pageY": 54.1640625\
    }\
  ],
  "changedTouches": [\
    {\
      "clientX": 49.69140625,\
      "clientY": 54.1640625,\
      "identifier": 0,\
      "pageX": 49.69140625,\
      "pageY": 54.1640625\
    }\
  ]
}
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_event-system_event-object

---

Event Object {#event-object}

Last updated: 2022-07-03

Path: miniprogram_gcash

Event Object

2022-07-03 18:44

When the component triggers the event, the event handler bounded with the logic layer receives an event object.

BaseEvent

BaseEvent basic event object attribute list:

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | type | String | Event type. | | timeStamp | Integer | Event generated timestamp. | | target | Object | Attribute value set of the component triggering the event. |

Type

Type: Event type

Timestamp

timeStamp: Event generated timestamp

Target

`dataset` define data in component, and the data is transferred via event to the logic layer. Start with `data-` and use hyphen `-` to connect multiple words which must be in lower case (upper case automatically converted into lower case). For example, the `data-element-type` will eventually convert the hyphen into hump elementType in the `event.target.dataset`.

Sample codes:

copy

```
<view data-alpha-beta="1" data-alphaBeta="2" onTap="bindViewTap">
DataSet Test </view>
```

copy

```
Page({
 bindViewTap:function(event){
   event.target.dataset.alphaBeta === 1 // - Will convert into hump
writing
   event.target.dataset.alphabeta === 2 // Upper case converted into
lower case
 }
})
```

Target: source component object that triggers the event, attribute list:

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | id | String | Event source component id. | | tagName | String | Current component type. | | dataset | Object | Set of custom attributes starting with **data-** on component bound with the event. | | targetDataset | Object | Set of custom attributes starting with **data-** on component actually triggering the event. |

CustomEvent

CustomEvent custom event object attribute list (inherited from BaseEvent)

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | detail | Object | Additional information. |

Detail

Data carried in custom event The form component event carries user entry information. For example, the switch component, when onChange trigger, gets user selected status value via event.detail.value. The media error event carries error information. **For details, see the component document event description**.

TouchEvent

TouchEvent touch event object attribute list (inherited from BaseEvent)

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | touches | Array | Array of touch point information staying current on the screen. | | changedTouches | Array | Array of touch point information changing currently. |

The touches is an array. Each of its elements is a Touch object (the touches carried in the **canvas** touch event is the CanvasTouch array), indicating the touch point staying on the screen.

changedTouches data format is the same as touches. Indicates changing touch point, such as from none to start (touchstart), location change (touchmove), from touch to end (touchend, touchcancel).

Touch Object

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | identifier | Number | Touch point identifier. | | pageX, pageY | Number | Distance to the document upper-left corner, the upper-left corner as origin, horizontal direction as x axis and vertical direction as y axis. | | clientX, clientY | Number | Distance to the displayable region of page (screen except for navigation bar), the upper-left corner as origin, horizontal direction as x axis and vertical direction as y axis. |

CanvasTouch Object

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | identifier | Number | Touch point identifier. | | x, y | Number | Distance to the Canvas upper-left corner, the Canvas upper-left corner as origin, horizontal direction as x axis and vertical direction as y axis. |

Sample

Take `touchMove` event as an example, when user touch the following component.

copy

```
<view class="move-view" onTouchMove="touchMoveHandle">
</view>
```

The `touchMoveHandle` will be invoked in the page, the TouchEvent will act as the parameter.

copy

```
Page({
  touchMoveHandle(e){
    console.log(e)
  }
});
```

The console output

copy

```
{
  "type": "touchMove",
  "timeStamp": 1562241425847,
  "target": {
    "targetDataset": {},
    "tagName": "view",
    "dataset": {},
    "offsetLeft": 0,
    "offsetTop": 0
  },
  "currentTarget": {
    "tagName": "view",
    "dataset": {},
    "offsetLeft": 0,
    "offsetTop": 0
  },
  "touches": [\
    {\
      "clientX": 49.69140625,\
      "clientY": 54.1640625,\
      "identifier": 0,\
      "pageX": 49.69140625,\
```

```
      "pageY": 54.1640625\
    }\
  ],
  "changedTouches": [\
    {\
      "clientX": 49.69140625,\
      "clientY": 54.1640625,\
      "identifier": 0,\
      "pageX": 49.69140625,\
      "pageY": 54.1640625\
    }\
  ]
}
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_event-system_event-object

---

Event management and analysis {#event-management-and-analysis}

Last updated: 2022-07-07

Path: miniprogram_gcash

Event management and analysis

2022-07-07 17:08

This topic introduces the event management and analysis functions that are provided by the Mini Program platform. To learn more about how to manage and analyze events, see How to customize your analysis.

Event management

What are events?

Events are triggered when users interact with your mini programs. When an event is triggered, data are collected and you can analyze user behavior. Based on the data analysis, you can visualize the performance of your mini programs and get continuous feedback on how to improve your product strategy.

For example, by adding events for a shopping mini program, you can identify the most popular products among your users. This information can then inform your strategy on developing similar products that will convert users.

What are event fields?

Data of an event consist of default fields and customized fields. The data of default fields are collected from systems on user devices, such as location and device type. The data of customized fields are collected from user actions, such as the act of clicking a product details page.

Under **My Analysis** > **Manage Event** > **Field List**, you can view the list of default fields and customized fields.

Currently, the field type supports integer and string.

Features

You can manage events in the following ways:

- View the event and field list

On the **Manage Event** page, you can view all events and fields. If no event is available, you can click the **+ New Event** button to create an event.

- Define an event

You can define an event by specifying the event name and data reporting method of the event. Currently, Data Reporting by Self-Defined Actions is supported.

- Publish an event

After confirming all required information of an event, you can publish the event. Once an event is live, it will start collecting data immediately.

- Modify an event

You can modify the event configuration regardless of whether the event is published or not. If a published event is modified, you must publish the event again to update the changes.

- Delete an event

You can delete the event. This action cannot be reversed, and a deleted event cannot be republished. However, you can create a new event with the same event name and configurations. In this way, you can see the data collected previously under **My Analysis** and continue to use this event to collect data. If either the event name or the configuration is inconsistent with the original event, all the data collected previously won't be displayed under **My Analysis**.

Event analysis

What is event analysis?

Event analysis is the customized analysis based on events. By selecting the required analysis conditions, you can see the data that reflects user's behavior, such as count of placing orders.

What are analysis conditions?

During the event analysis, the following analysis conditions might be involved:

- **Event**: The event to be analyzed.
- **Metric**: The dimension that you want to analyze for the event. For all events, you can select the following two default metrics:
- Page view: A default metric that indicates the number of times the page is viewed.
- Unique visitor: A default metric that indicates the number of users who view the page. One user with multiple views is still counted as one unique visitor.

In addition to these two default metrics, you can also select customized metrics, which are the fields defined in the event, for analysis.

- **Filter**: Filter conditions of the customized metrics that are selected. For example, the metric *productPrice* is selected and you can set the filter condition as *productPrice* equal to 10.
- **Date Range**: The period when events are triggered.
- **Time Granularity**: Currently the time granularity supports daily and hourly.

Features

The event analysis function enables you to do the following:

- Set conditions for the event analysis

You can select an event for data analysis and set the following conditions:

- Metric
- Filter conditions of the metric
- Date range
- Time granularity
- View analysis result

You can view the analysis result in the form of table or graph.

More information

Analytics

Data Reporting by Self-Defined Actions

How to customize your analysis

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/manage-event

---

Extended Component Reference {#extended-component-reference}

Last updated: 2022-07-07

Path: miniprogram_gcash

Extended Component Reference

2022-07-07 17:08

Layout Navigation

| | | | --- | --- | | **Component Name** | **Function Description** | | Container | Make the style of all elements in the container more consistent, such as margins between elements. | | Title | Display the title of each page. | | List | List | | List-item | Customize items in a list. | | List-secondary | Display the additional information on the right side of the list item. | | Tabs | Tabs allow the user to switch between different views in landscape. | | VTabs | Vtabs allow the user to switch between different views in vertical. | | Card | Card. | | Coupon | Display the coupon, red packet, and ticket that can be redeemed by users. | | Grid | Grid. | | Steps | Show the progress bar as per the steps. | | Footer | Show the page footer. | | Terms | Users must agree with terms before using or activating the service. | | Flex | Flex layout. | | Pagination | Pagination. | | Collapse | Collapse panel. |

Float

| | | | --- | --- | | **Component Name** | **Function Description** | | Popover | Popover. | | Filter | Use as tab filter. | | Modal | Dialog box. | | Popup | Popup menu. |

Result

| | | | --- | --- | | **Component Name** | **Function Description** | | PageResult | Page result. | | Message | Message. |

Guide

| | | | --- | --- | | **Component Name** | **Function Description** | | Tips | Tips. | | Notice | Notice. | | Badge | Red dot, number or text used to tell the user the number of things or updates to be handled. | | Tag | Highlight the information, such as the warning. | | Mask | Display the pop-up element with a mask. | | Guide | Teach users to use features on a mask. | | Avatar | Display avatars. |

Form

| | | | --- | --- | | **Component Name** | **Function Description** | | InputItem | Text input. | | Verify Code | Display the input box of the verification code. | | PickerItem | Pick to input. | | Long password | Display the input box for the password. | | Multi Liner | Allow users to enter multiple lines of content in an input box. | | AmountInput | Input money number. | | Button | Allow users to perform actions or make choices. | | Am-switch | Change the state on or off. | | SearchBar | Search box. | | AMRadio | Allow users to select radio buttons. | | AMCheckBox | Multiple check box. |

Gesture

| | | | --- | --- | | **Component Name** | **Function Description** | | SwipeAction | Slide cell. |

Other

| | | | --- | --- | | **Component Name** | **Function Description** | | Calendar | Calendar. | | Stepper | Stepper. | | AMIcon | Icons. | | Alphabet | Arrange content in the alphabetical order. | | Loading | Display the loading animation. |

Experience Mini Program

Developers can use the Android Demo App to scan the QR code shown in the extended component documents.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_extended-component-reference

---

FAQs {#faqs}

Last updated: 2021-05-10

Path: miniprogram_gcash

FAQs

2021-05-10 13:12

Get answers to common questions about mini programs.

Mini program platform

Why does the request parameter error occur when I modify the mini program slogan and name?

The local CDN is not stable when the mini program image is uploaded. You can upload a new image and update the mini program information again.

Why do wallet admins not receive the notification of the approval request when mini program developers release mini programs in the sandbox environment?

Approval is required only when the mini program is released in the production environment. In other environments the mini program release is approved automatically.

Component

Why does the *No Privilege* error occur when using the WebView component?

You can try to solve the problem by the following checklist:

- Add required H5 domains to the whitelist under **Mini Program** > **Configuration** > **H5 Domain Whitelist**.
- Release the mini program in your environment, which can be sandbox environment, test environment, or development environment. Otherwise, the updated configuration does not take effect.

If you can't solve the problem with the provided solution, contact the technical support team of your site.

API

Why does the *JSAPI call denied* error occur when calling the my.request JSAPI?

You can try to solve the problem by the following checklist:

- Add the required server domains to the whitelist under **Mini Program** > **Configuration** > **Server D omain Whitelist**.
- Release the mini program in the sandbox environment, test environment, or development environment. Otherwise, the updated configuration does not take effect.
- Check whether the JSAPI is private. If the called JSAPI is private, get the permission to call the JSAPI from the wallet first.

Why is the error code 4006 returned when I call the getAuthCode JSAPI?

This error occurs because of the incorrect configurations at the merchant server side. You need to contact the server end developer for help.

Why does the *No privilege* error occur when I call some JSAPIs?

Check whether the JSAPI is private. If the called JSAPI is private, get the permission to call the JSAPI from the wallet first.

IDE

What's the maximum size of the mini program when uploading mini programs from IDE to the mini program platform?

The maximum size limit at the server side is 8MB. However, it's difficult to define the size limit of the source code because many factors impact the size of the packaged source code, such as image compressing. Currently the package for debugging on real devices has the maximum size. The file is stored in the directory as highlighted in the following figure. Except for this file, generally all files that are no more than 8MB can be uploaded.

Basic information

Does the mini program support SVG and gradient?

Yes. Progressive web apps (also referred as H5) and the image component in native mini programs support SVG. Progressive web apps support gradient. SVG is used in the user authentication, such as sliding the verification picture.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/about/faq

---

Features {#features}

Last updated: 2022-07-07

Path: miniprogram_gcash

Features

2022-07-07 17:08

You can define specific features according to your business requirements, in addition to the payment and user information capabilities that are available by default. For more information, see Capabilities.

Features

You can perform the following activities:

- Create new features with the following details:
- `Feature Name` (Mandatory)
- `Activation Access` (Mandatory)
- `Direct Access`: You can have access to this feature immediately, without further requirements.
- `Activation Needed`: You need to fulfill the activation requirements before you can have access to this feature.
- Select APIs to construct the content of the feature

For more information on how APIs can combine to form a capability, see Capabilities.

- View features in a list
- Change the following details of a feature
- `API Name`

Choose the API name, you can see the details of the API

You can have a overview of the usage of this API, such as the API description, scenario, wallet name, security level, and so on.

Under the `Info` tab, you get basic information of the APIs, such as the input parameters, output parameters, sample codes, and error codes.

Under the `Version History` tab, , you can see the version and status of the JSAPI.

- `Latest Version`
- `Created by`
- `Edited Date`
- Track the version history of a feature

More Information

Overview

Manage Mini Programs

Capabilities

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/feature

---

File reference {#file-reference}

Last updated: 2022-07-04

Path: miniprogram_gcash

File reference

2022-07-04 03:44

The axml provides two file reference methods: `import` and `include`

import

The import can load a defined `template`.

For example, a `template` named `item` was defined in the item.axml.

copy

```
<!-- item.axml -->
<template name="item">
  <text>{{text}}</text>
</template>
```

When item.axml is referenced in index.axml, the `item` template can be used.

copy

```
<import src="./item.axml"/>
<template is="item" data="{{text: 'forbar'}}"/>
```

The import has the concept of action scope. Only the template defined in the target file is imported, but the one imported into the target file is not imported.

For example, C import B, B import A, in C it is possible to use the template defined in B, in B it is possible to use the template defined in A, but in C it is impossible to use the template defined in A.

copy

```
<!-- a.axml -->
<template name="A">
  <text> A template </text>
</template>
```

copy

```
<!-- b.axml -->
<import src="./a.axml"/>
<template name="B">
  <text> B template </text>
</template>
```

copy

```
<!-- c.axml -->
<import src="./b.axml"/>
<template is="A"/>  <!-- Note: cannot use import A -->
<template is="B"/>
```

Note that the sub-node of template can be one only. For example:

Allowed example:

copy

```
<template name="x">
  <view />
</template>
```

Disallowed example:

copy

```
<template name="x">
  <view />
  <view />
</template>
```

include

The include may introduce the whole codes except for `<template/>`, which is equivalent to copy to the include position.

Code sample:

copy

```
<!-- index.axml -->
<include src="./header.axml"/>
<view> body </view>
<include src="./footer.axml"/>
```

copy

```
<!-- header.axml -->
<view> header </view>
```

copy

```
<!-- footer.axml -->
<view> footer </view>
```

Introduction Path

The template introduction path supports relative path, absolute path and third-party module loaded from node_modules directory.

copy

```
<import src="./a.axml"/> <!-- relative path -->
<import src="/a.axml"/> <!-- absolute path of the project -->
<import src="third-party/x.axml"/> <!-- path of the third-party npm
package -->
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_axml-reference_file-reference

---

File reference {#file-reference}

Last updated: 2021-05-09

Path: miniprogram_gcash

File reference

2021-05-09 18:43

The axml provides two file reference methods: `import` and `include`

import

The import can load a defined `template`.

For example, a `template` named `item` was defined in the item.axml.

copy

```
<!-- item.axml -->
<template name="item">
  <text>{{text}}</text>
</template>
```

When item.axml is referenced in index.axml, the `item` template can be used.

copy

```
<import src="./item.axml"/>
<template is="item" data="{{text: 'forbar'}}"/>
```

The import has the concept of action scope. Only the template defined in the target file is imported, but the one imported into the target file is not imported.

For example, C import B, B import A, in C it is possible to use the template defined in B, in B it is possible to use the template defined in A, but in C it is impossible to use the template defined in A.

copy

```
<!-- a.axml -->
<template name="A">
  <text> A template </text>
</template>
```

copy

```
<!-- b.axml -->
<import src="./a.axml"/>
<template name="B">
  <text> B template </text>
</template>
```

copy

```
<!-- c.axml -->
<import src="./b.axml"/>
<template is="A"/>  <!-- Note: cannot use import A -->
<template is="B"/>
```

Note that the sub-node of template can be one only. For example:

Allowed example:

copy

```
<template name="x">
  <view />
</template>
```

Disallowed example:

copy

```
<template name="x">
  <view />
  <view />
</template>
```

include

The include may introduce the whole codes except for `<template/>`, which is equivalent to copy to the include position.

Code sample:

copy

```
<!-- index.axml -->
<include src="./header.axml"/>
<view> body </view>
<include src="./footer.axml"/>
```

copy

```
<!-- header.axml -->
<view> header </view>
```

copy

```
<!-- footer.axml -->
<view> footer </view>
```

Introduction Path

The template introduction path supports relative path, absolute path and third-party module loaded from node_modules directory.

copy

```
<import src="./a.axml"/> <!-- relative path -->
<import src="/a.axml"/> <!-- absolute path of the project -->
<import src="third-party/x.axml"/> <!-- path of the third-party npm
package -->
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_axml-reference_file-reference

---

Filter {#filter}

Last updated: 2022-07-03

Path: miniprogram_gcash

Filter

2022-07-03 18:44

Use as tab filter.

Filter

| | | | | | | --- | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | **Required** | | show | Show or not, optional, `show` `hide`. | String | hide | No | | max | Maximum choices, 1 for single choice. | Number | 10000 | No | | onChange | Submit selection callback upon multiple choice. | (e: Object) => void | | No |

Filter-item

| | | | | | | --- | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | **Required** | | className | Custom style. | String | | No | | value | Value. | String | | Yes | | id | Custom identifier. | String | | No | | selected | Selected by default. | Boolean | false | No | | onChange | Submit selection callback upon single choice. | (e: Object) => void | | No |

Example

copy

```
{
  "defaultTitle": "AntUI Component Library",
  "usingComponents": {
    "filter": "mini-antui/es/filter/index",
    "filter-item": "mini-antui/es/filter/filter-item/index"
  }
}
```

copy

```
<filter show="{{show}}" max="{{5}}" onChange="handleCallBack">
  <block a:for="{{items}}">
    <filter-item value="{{item.value}}" id="{{item.id}}" selected="
{{item.selected}}"/>
  </block>
</filter>
```

copy

```
Page({
  data: {
    show: true,
    items: [\
      { id: 1, value: 'Clothes', selected: true },\
      { id: 1, value: 'Cupboard' },\
      { id: 1, value: 'Hanger' },\
      { id: 3, value: 'Digital' },\
```

```
      { id: 4, value: 'Door' },\
      { id: 5, value: 'Chair' },\
      { id: 7, value: 'Monitor' },\
      { id: 6, value: 'Game' },\
      { id: 8, value: 'Food' },\
    ]
  },
  handleCallBack(data) {
    my.alert({
      content: data
    });
  },
  toggleFilter() {
    this.setData({
      show: !this.data.show,
    });
  }
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_floating-layer_filter

---

Flex {#flex}

Last updated: 2022-07-03

Path: miniprogram_gcash

Flex

2022-07-03 18:44

CSS flex layout encapsulation

Sample Code

copy

```
  {
    "defaultTitle": "Mini Program AntUI component library",
    "usingComponents": {
      "flex": "mini-antui/es/flex/index",
      "flex-item": "mini-antui/es/flex/flex-item/index"
    }
  }
```

copy

```
<view class="flex-container">
  <view class="sub-title">Basic</view>
  <flex>
    <flex-item><view class="placeholder">Block</view></flex-item>
    <flex-item><view class="placeholder">Block</view></flex-item>
  </flex>
  <view style="height: 20px;" />
  <flex>
    <flex-item><view class="placeholder">Block</view></flex-item>
    <flex-item><view class="placeholder">Block</view></flex-item>
    <flex-item><view class="placeholder">Block</view></flex-item>
  </flex>
  <view style="height: 20px;" />
  <flex>
    <flex-item><view class="placeholder">Block</view></flex-item>
    <flex-item><view class="placeholder">Block</view></flex-item>
    <flex-item><view class="placeholder">Block</view></flex-item>
    <flex-item><view class="placeholder">Block</view></flex-item>
  </flex>
  <view className="sub-title">Wrap</view>
  <flex wrap="wrap">
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
  </flex>
  <view className="sub-title">Align</view>
  <flex justify="center">
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
  </flex>
  <flex justify="end">
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
  </flex>
  <flex justify="between">
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline">Block</view>
  </flex>
  <flex align="start">
    <view class="placeholder inline">Block</view>
    <view class="placeholder inline small">Block</view>
    <view class="placeholder inline">Block</view>
  </flex>
  <flex align="end">
```

```
      <view class="placeholder inline">Block</view>
      <view class="placeholder inline small">Block</view>
      <view class="placeholder inline">Block</view>
    </flex>
    <flex align="baseline">
      <view class="placeholder inline">Block</view>
      <view class="placeholder inline small">Block</view>
      <view class="placeholder inline">Block</view>
    </flex>
  </view>
```

copy

```
.flex-container {
  padding: 10px;
}

.sub-title {
  color: #888;
  font-size: 14px;
  padding: 30px 0 18px 0;
}

.placeholder {
  background-color: #ebebef;
  color: #bbb;
  text-align: center;
  height: 30px;
  line-height: 30px;
  width: 100%;
}

.placeholder.inline {
  width: 80px;
  margin: 9px 9px 9px 0;
}

.placeholder.small {
  height: 20px;
  line-height: 20px
}
```

copy

```
Page({});
```

Attributes

| | | | | | --- | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | **Required** | | direction | Direction of item, including row, row-reverse, column and olumn-reverse. | String | row | No | | wrap | Sub-elementwrap mode, including nowrap, wrap and rap-

reverse. | String | nowrap | No | | justify | Sub-elementjustify on main axis, including start, end, center, between and around. | String | start | No | | align | Sub-element justify on cross axis, including start, center, end, baseline and stretch. | String | center | No | | alignContent | Sub-element justify on multiple axises, including start, end, center, between, around and stretch. | String | stretch | No |

Flex-item

The flex-item component has a default style flex:1, which ensures all items have equal width. The children of the flex container may not be a flex-item.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_layout-navigation_flex

---

Footer {#footer}

Last updated: 2022-07-03

Path: miniprogram_gcash

Footer

2022-07-03 18:44

Show the page footer.

| | | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | copyright | Copyright information. | String | No | | links | Footer link. | `Array<text, url>` | No |

Example

copy

```
{
  "defaultTitle": "AntUI Component Library",
  "usingComponents":{
    "footer": "mini-antui/es/footer/index"
  }
}
```

copy

```
<view>
  <footer
    copyright="{{copyright}}"
```

```
    links="{{links}}" />
</view>
```

copy

```
Page({
 data: {
    copyright: '© 2004-2017 *.com. All rights reserved.',
    links: [\
      { text: 'Bottom link', url: '../../list/demo/index' },\
      { text: 'Bottom link', url: '../../card/demo/index' },\
    ],
 },
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_layout-navigation_footer

---

Funnel management and analysis {#funnel-management-and-analysis}

Last updated: 2022-07-07

Path: miniprogram_gcash

Funnel management and analysis

2022-07-07 17:08

This topic introduces the funnel management and analysis functions that are provided by the Mini Program platform. To learn more about how to manage and analyze funnels, see How to customize your analysis.

Funnel management

What are funnels?

A funnel consists of multiple events. Based on the funnel analysis, you can visualize the steps your users take to go through a task and quickly see the conversion rate between two steps. For example, how do general visitors become buyers? How do one-time buyers become repeat buyers? With this information, you can improve user journeys specifically.

Features

You can manage funnels in the following ways:

- View the funnel list

You can modify individual funnels under the funnel list page. All your funnels are displayed on the Funnel List page, and you can modify individual funnels by clicking **Edit** or **Delete**.

- Create a funnel

You can create a new funnel. A funnel must consist of at least two steps. Each step is defined by an event.

- Modify a funnel

You can modify a created funnel.

- Delete a funnel

You can delete a funnel.

Funnel analysis

Features

The funnel analysis function enables you to do the following:

- Set conditions for the funnel analysis

To set conditions for a funnel analysis, click on the **Manage Funnel** tab at the **< page name >** page.

- View the analysis result

You can view the analysis result in the form of table or graph.

More information

Analytics

Event management and analysis

How to customize your analysis

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/manage-funnel

---

GCash Mini Program Development Platform {#gcash-mini-program-development-platform}

Welcome to

Mini Program Platform

Email

PasswordForgot Password?

Log InDon’t have an account?Sign up >

Source: https://miniprogram.gcash.com/login?goto=https%3A%2F%2Fminiprogram.gcash.com%2Fdocs%2Fminiprogram_gcash%2Fplatform_guide%2Fmanage-templates%3FssrData%3Dtrue

---

GCash Mini Program Development Platform {#gcash-mini-program-development-platform}

Welcome to

Mini Program Platform

Email

PasswordForgot Password?

Log InDon’t have an account?Sign up >

Source: https://miniprogram.gcash.com/login?goto=https%3A%2F%2Fminiprogram.gcash.com%2Fdocs%2Fminiprogram_gcash%2Fplatform_guide%2Fnavigate-to-operation-platform%3FssrData%3Dtrue

---

Generate QR codes {#generate-qr-codes}

Last updated: 2022-11-13

Path: miniprogram_gcash

Generate QR codes

2022-11-13 15:01

This topic provides steps to generate different promotion QR codes. A default QR code is available when a mini program is released. Depending on business requirements, you can also general multiple QR codes based on this default QR code so that multiple users can quickly run your mini program by scanning these QR codes.

Procedures

1. Go to the **Manage Mini Programs** page and choose the mini program that you want to generate promotion QR codes. Then Click the **QR Code** tab and then **+Generate QR Code**.
2. Fill in the page URL, query parameter, and description of the mini program and click **Create**.

Now you have finished generating a promotion QR code.

Next steps

Remove mini programs

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/generate-qrcode

---

Getting Started Guide {#getting-started-guide}

Path: miniprogram_gcash

Getting Started Guide

A quick-start guide for digital wallets and their merchants to learn the basic steps to get onboarded to the Mini Program Platform.

Get to know Mini Program

Learn the basics about the latest technology and services of Mini Programs.

Watch this video to quickly learn how to manage Mini Programs on the platform.

Onboarding to the Platform

Apply an account and KYB for merchants or acquirers.

Quickstart creating a mini program after onboarding on the Mini Program Platform for the first time.

Build your first Mini Program

Download Mini Program Studio and install the IDE.

Create a project and debug your demo Mini Program in the IDE.

Understand the file structure, global variables and page directory.

Preview the Mini Program on a device and upload the ready-to-go Mini Program to the Platform.

Release Mini Programs

Customize your Mini Programs according to cross-border or local businesses.

Start compatibility testing and go through an approval process to release the Mini Program.

Dive Deeper

Get the development resources and references such as components, framework, JSAPIs and Open APIs.

Learn product features to use the Mini Program Console.

Dip into the rich UI guidelines to design a Mini Program.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/getting-started/getting-started-guide

---

Getting Started with Mini Program Platform {#getting-started-with-mini-program-platform}

Last updated: 2022-07-07

Path: miniprogram_gcash

Getting Started with Mini Program Platform

2022-07-07 17:08

The Mini Program Platform is a comprehensive solution to empower digital wallets to quickly start Mini Program businesses. It not only helps you to save the hardware deployment costs but also provides you with operation and maintenance support. You can also enjoy the regular service updates and new product features.

Procedures

Before you start, follow the integration steps below:

1. Contact us to create a Mini Program workspace for your app. You're required to provide the following:
2. Your valid business license and point of contact
3. Your APK signature, bundle ID, and package name
4. Once your Mini Program workspace creation is approved, you will receive a dedicated URL to access your Mini Program Developer Portal. You then make the following configuration settings:
5. Replace the default URL to your business domain by configuring a reverse proxy server
6. Customize the UI by replacing the default logo/favicon with your preferred logo
7. Customize the email service
8. Integrate the SDK
9. You will need to integrate the SDK (Android and iOS) in order to run the Mini Programs in your application.
10. Download the configuration from your Developer Portal and use it to initialize the SDK.
11. Implement the standard JSAPIs. While the SDK provides the core Mini Program functionality, there are some features that are best presented with a customized implementation. For instance, when a Mini Program invokes the payment function, it is better to launch the existing payment flow instead of a completely new one.
12. Implement the standard backend APIs

You need to implement some standard backend APIs in order to support the Mini Program OAuth and Payment flow.

Now you can start using the Mini Program Platform.

Contact Us

If you're interested in using Mini Programs, please send us an email.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/getting-started/getting-started-with-platform

---

Git Management {#git-management}

Last updated: 2021-05-09

Path: miniprogram_gcash

Git Management

2021-05-09 18:43

Mini Program IDE supports use of visualized Git management tool to manage the project codes.

1. Initialize Git

When the project directory has no Git warehouse, you can directly initialize Git (`git init`).

2. Stage Change

You can choose to stage all changes by using one-key (`git add .`), or hover over individual file and choose to stage.

3. Submit Change

Submit the staged change locally (`git commit –m 'xxx'`)

Confirm the code diff and then do the submission operation.

4. Push to Remote Warehouse

Add to the warehouse (`git remote add origin xxxxxx`)

Confirm and push (`git push origin`)

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/mini-program-studio_function-panel_git-management

---

Git Management {#git-management}

Last updated: 2022-07-03

Path: miniprogram_gcash

Git Management

2022-07-03 18:44

Mini Program IDE supports use of visualized Git management tool to manage the project codes.

1. Initialize Git

When the project directory has no Git warehouse, you can directly initialize Git (`git init`).

2. Stage Change

You can choose to stage all changes by using one-key (`git add .`), or hover over individual file and choose to stage.

3. Submit Change

Submit the staged change locally (`git commit -m 'xxx'`)

Confirm the code diff and then do the submission operation.

4. Push to Remote Warehouse

Add to the warehouse (`git remote add origin xxxxxx`)

Confirm and push (`git push origin`)

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/mini-program-studio_function-panel_git-management

---

Global Configuration {#global-configuration}

Last updated: 2021-05-09

Path: miniprogram_gcash

Global Configuration

2021-05-09 18:43

`app.json` is used to implement global configuration and set up the path of page files, window display, multiple tabs and so on.

Here is a basic configuration example:

copy

```
{
  "pages": [\
    "pages/index/index",\
    "pages/logs/index"\
  ],
  "window": {
    "defaultTitle": "Demo"
  }
}
```

Complete configuration item:

| | | | | | --- | --- | --- | --- | | **Attribute** | **Type** | **Required** | **Description** | | pages | Array | Yes | Set page path. | | window | Object | No | Set default window display for all page. | | tabBar | Object | No | Set the display of bottom tabbar. |

pages

In the app.json, the `pages` have the array attribute, with string for each member in the array, used to specify the `pages` of the Mini Program. In the Mini Program, adding or deleting pages modifies the `pages` array.

Each member of the pages array represents the path of the related page. The first member indicates the home of the Mini Program.

The page path does not need any suffix. The framework automatically loads the `.json`, `.js`, `.axml` and `.acss` files of the same name. For example, if the development directory is:

copy

```
├── pages
│   ├──index
│   │    ├── index.json
│   │    ├── index.js
│   │    ├── index.axml
│   │    └── index.acss
│   ├──logs
│   │    ├── logs.json
│   │    ├── logs.js
```

```
|   |    └── logs.axml
├── app.json
├── app.js
└── app.acss
```

The following configuration shall be done in the `app.json`:

copy

```
{
  "pages":[\
    "pages/index/index",\
    "pages/logs/logs"\
  ]
}
```

window

The window is used to set up the status bar, navigation bar, title, window background color, etc. for the Mini Program.

Sample codes:

copy

```
{
  "window":{
    "defaultTitle": "Default Title"
  }
}
```

| | | | | | | --- | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | **Minimum version** | | defaultTitle | String | NO | Default title of page. | - | | pullRefresh | String | NO | Allow pull-to-refresh or not, `NO` by default. Remarks: The precondition for pull-to-refresh to take effect is allowsBounceVertical `YES`. | - | | allowsBounceVertical | String | NO | Allow bounce vertical or not `YES` by default, supporting `YES / NO`. | - | | transparentTitle | String | NO | Navigation bar transparency setting `none` by default, supporting `always` (always transparent)/ `auto` (sliding adaptation)/ `none` (not transparent). | - | | titlePenetrate | String | NO | Allow navigation bar click-through or not `No` by default, supporting `YES / NO`. | - | | showTitleLoading | String | NO | Show navigation bar loading or not upon entrance `No` by default, supporting `YES / NO`. | - | | titleImage | String | No | Navigation bar picture address. | - | | titleBarColor | HexColor | NO | Background color of navigation bar, decimal color value (0-255). | - | | backgroundColor | HexColor | NO | Background color of page, decimal color value (0-255). | - | | backgroundImageColor | HexColor | NO | Background bottom color to display during pull-down, decimal color value (0-255). | - | | backgroundImageUrl | String | NO | URL of background to display during pull-down. | - | | gestureBack | String | NO | For iOS only, supporting gesture return or not `No` by default, supporting `YES / NO`. | - | | enableScrollBar | Boolean | NO | For Android only, showing WebView scroll bar or not `YES` by default, supporting `YES / NO`. | - | | onReachBottomDistance | Number | NO | Distance to page bottom for triggering bottom-out during pull-up, in px. Related documents Page event handler. | Currently, the `iOS` does not support settings in the `page.json`, and only global settings are supported. |

tabBar

If the Mini Program is a multi-tab application (pages switchable in the window bottom bar of the client), the `tabBar` configuration item can be used to specify the tab presentation pattern and the corresponding page in case of tab switch.

Note:

• On the destination page via page jump (my.navigateTo) or page redirection (my.redirectTo), the bottom tab bar is not displayed even when the page is defined in the tabbar configuration.

The tabBar has the following configuration items:

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | textColor | HexColor | NO | Text color. | | selectedColor | HexColor | NO | Highlighted text color. | | backgroundColor | HexColor | NO | Background color. | | items | Array | Yes | Each tab configuration. |

Each item configuration:

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | pagePath | String | Yes | Set page path. | | name | String | Yes | Name. | | icon | String | NO | Common icon path. | | activeIcon | String | NO | Highlighted icon path. |

The recommended size of icon is 60×60 px. The pictures not in the recommended size will be stretched or scaled in unequal proportion.

Sample codes:

copy

```
{
  "tabBar": {
    "textColor": "#dddddd",
    "selectedColor": "#49a9ee",
    "backgroundColor": "#ffffff",
    "items": [\
      {\
        "pagePath": "pages/index/index",\
        "name": "Home\
      },\
      {\
        "pagePath": "pages/logs/logs",\
        "name": "Log\
      }\
    ]
  }
}
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_app_global-configuration

---

Global Configuration {#global-configuration}

Last updated: 2022-07-03

Path: miniprogram_gcash

Global Configuration

2022-07-03 18:44

`app.json` is used to implement global configuration and set up the path of page files, window display, multiple tabs and so on.

Here is a basic configuration example:

copy

```
{
  "pages": [\
    "pages/index/index",\
    "pages/logs/index"\
  ],
  "window": {
    "defaultTitle": "Demo"
  }
}
```

Complete configuration item:

| | | | | --- | --- | --- | --- | | **Attribute** | **Type** | **Required** | **Description** | | pages | Array | Yes | Set page path. | | window | Object | No | Set default window display for all page. | | tabBar | Object | No | Set the display of bottom tabbar. |

pages

In the app.json, the `pages` have the array attribute, with string for each member in the array, used to specify the `pages` of the Mini Program. In the Mini Program, adding or deleting pages modifies the `pages` array.

Each member of the pages array represents the path of the related page. The first member indicates the home of the Mini Program.

The page path does not need any suffix. The framework automatically loads the `.json`, `.js`, `.axml` and `.acss` files of the same name. For example, if the development directory is:

copy

```
├── pages
│   ├──index
```

```
│   │   ├── index.json
│   │   ├── index.js
│   │   ├── index.axml
│   │   └── index.acss
│   ├──logs
│   │   ├── logs.json
│   │   ├── logs.js
│   │   └── logs.axml
├── app.json
├── app.js
└── app.acss
```

The following configuration shall be done in the `app.json`:

copy

```
{
  "pages":[\
    "pages/index/index",\
    "pages/logs/logs"\
  ]
}
```

window

The window is used to set up the status bar, navigation bar, title, window background color, etc. for the Mini Program.

Sample codes:

copy

```
{
  "window":{
    "defaultTitle": "Default Title"
  }
}
```

| | | | | | | --- | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | **Minimum version** | | defaultTitle | String | NO | Default title of page. | - | | pullRefresh | String | NO | Allow pull-to-refresh or not, `NO` by default. Remarks: The precondition for pull-to-refresh to take effect is allowsBounceVertical `YES`. | - | | allowsBounceVertical | String | NO | Allow bounce vertical or not `YES` by default, supporting `YES / NO`. | - | | transparentTitle | String | NO | Navigation bar transparency setting `none` by default, supporting `always` (always transparent)/ `auto` (sliding adaptation)/ `none` (not transparent). | - | | titlePenetrate | String | NO | Allow navigation bar click-through or not `No` by default, supporting `YES / NO`. | - | | showTitleLoading | String | NO | Show navigation bar loading or not upon entrance `No` by default, supporting `YES / NO`. | - | | titleImage | String | No | Navigation bar picture address. | - | | titleBarColor | HexColor | NO | Background color of navigation bar, decimal color value (0-255). | - | | backgroundColor | HexColor | NO | Background color of page, decimal color value (0-255). | - | | backgroundImageColor | HexColor | NO | Background bottom color to display during pull-down, decimal color value (0-255). | - | |

backgroundImageUrl | String | NO | URL of background to display during pull-down. | - | | gestureBack | String | NO | For iOS only, supporting gesture return or not `No` by default, supporting `YES / NO`. | - | | enableScrollBar | Boolean | NO | For Android only, showing WebView scroll bar or not `YES` by default, supporting `YES / NO`. | - | | onReachBottomDistance | Number | NO | Distance to page bottom for triggering bottom-out during pull-up, in px. Related documents Page event handler. | Currently, the `iOS` does not support settings in the `page.json`, and only global settings are supported. |

tabBar

If the Mini Program is a multi-tab application (pages switchable in the window bottom bar of the client), the `tabBar` configuration item can be used to specify the tab presentation pattern and the corresponding page in case of tab switch.

Note:

• On the destination page via page jump (my.navigateTo) or page redirection (my.redirectTo), the bottom tab bar is not displayed even when the page is defined in the tabbar configuration.

The tabBar has the following configuration items:

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | textColor | HexColor | NO | Text color. | | selectedColor | HexColor | NO | Highlighted text color. | | backgroundColor | HexColor | NO | Background color. | | items | Array | Yes | Each tab configuration. |

Each item configuration:

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | pagePath | String | Yes | Set page path. | | name | String | Yes | Name. | | icon | String | NO | Common icon path. | | activeIcon | String | NO | Highlighted icon path. |

The recommended size of icon is 60×60 px. The pictures not in the recommended size will be stretched or scaled in unequal proportion.

Sample codes:

copy

```
{
  "tabBar": {
    "textColor": "#dddddd",
    "selectedColor": "#49a9ee",
    "backgroundColor": "#ffffff",
    "items": [\
      {\
        "pagePath": "pages/index/index",\
        "name": "Home\
      },\
      {\
        "pagePath": "pages/logs/logs",\
        "name": "Log\
```

```
      }\
    ]
  }
}
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_app_global-configuration

---

Global Style {#global-style}

Last updated: 2022-07-03

Path: miniprogram_gcash

Global Style

2022-07-03 18:44

`app.acss` is for global style, and is effective for all the pages in the Mini Program.

Please refer to ACSS Reference for further `acss` details.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_app_global-style

---

Global Style {#global-style}

Last updated: 2021-05-09

Path: miniprogram_gcash

Global Style

2021-05-09 18:43

`app.acss` is for global style, and is effective for all the pages in the Mini Program.

Please refer to ACSS Reference for further `acss` details.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_app_global-style

---

Glossary {#glossary}

Last updated: 2021-05-10

Path: miniprogram_gcash

Glossary

2021-05-10 04:12

A

ACL

An ACL (Access Control List) is a list of instructions that are applied to the router interface. These instruction lists are used to tell the router which packets can be received and which packets need to be rejected. Whether a packet is received or rejected can be determined by specific indications such as the source address, destination address, port number, and so on.

ACSS

ACSS is a style language that describes the style of AXML components and determines how AXML components should be displayed.

Android

Android is a free and open source operating system based on the Linux kernel which does not contain GNU components. The system is mainly used in mobile devices, such as smart phones and tablets. It is It is led and developed by Google Inc. and the Open Handset Alliance.

API

API (Application Programming Interface) is a set of rules and specifications that software programmes can follow to communicate with each other. An API serves as an interface between different software programs and facilitates their interaction.

App

An app, also known as mobile app is a software application that is designed to run on smartphones and other mobile devices.

App Container

App Container is a productive and secure runtime system to execute Mini Programs on mobile platforms of Android and iOS in any apps that are integrated with Mini Programs.

AXML

AXML is a set of tag languages designed by the Mini Program framework to describe the structure of Mini Program pages. The AXML syntax can be broken down into five parts: data binding, conditional rendering, list rendering, templates, and references.

app. json

App. json is a global configuration file of Mini Program, which is used to configure the general navigation bar title, window background color and other configurations of a Mini Program.

B

back-end running

The status when a Mini Program is not destroyed but still runs in the back end after a user closes the Mini Program or press the Home button to leave mobile App.

C

cold startup

A cold startup is the startup when a user opens a Mini Program that has not been started or has been destroyed.

cookies

Cookies are designed to be a reliable mechanism for websites to remember stateful information (such as items added in the shopping cart in an online store) or to record the user's browsing activity (including clicking particular buttons, logging in, or recording which pages were visited in the past).

CSS

CSS (Cascading Style Sheets) is a style sheet language to describe the presentation of a document written in a markup language such as HTML and XML. It can not only modify the web page statically, but also cooperate with a variety of scripting languages to dynamically format the elements of a web page.

CDN

CDN (Content Delivery Network or Content Distribution Network) is a geographically distributed network of proxy servers and their data centers, in order to provide high availability and performance by distributing the service spatially relative to end users.

D

DSL

DSL (Domain Specific Language) is a computer language specialized to a particular application domain. This is in contrast to a general-purpose language (GPL), which is broadly applicable across domains.

DOM

DOM (The Document Object Model) is a cross-platform and language-independent interface that treats an XML or HTML document as a tree structure wherein each node is an object representing a part of the document. The DOM represents a document with a logical tree.

E

ES

ES (ECMAScript) is a general-purpose programming language, standardized by Ecma International according to the document ECMA-262. It is a JavaScript standard meant to ensure the interoperability of Web pages across different Web browsers.

F

front-end running

The status when a Mini Program runs in the front end after a user opens the Mini Program for the first time.

H

hot startup

A hot startup is the startup when a user opens a Mini Program that has been opened but is running in back end.

HTML

An application of the Standard Generalized Markup Language that uses tags to mark elements, such as text and graphics, in a document to indicate how Web browsers should display these elements to the user and should respond to user actions.

I

IDE

IDE (Integrated Development Environment) is a software application that provides comprehensive facilities to computer programmers for software development.
An IDE normally consists of at least a source code editor, build automation tools and a debugger. The IDE for Mini Program development is called Mini Program Studio.

iOS

iOS (formerly iPhone OS) is a mobile operating system that is created and developed by Apple Inc. exclusively for its hardware.

ISV

ISV (Independent Software Vendor) is an individual or organization that develops, markets, and sells software solutions that run on one or more computer hardware providers like Macintosh, operating systems like iOS, or cloud platforms like Amazon Web Services. In a Mini Program, ISV usually act as the third-party service developer to develop and operate the Mini Program on behalf of merchants.

icon

A pictogram used on the user interface to assist with navigation.

J

JS

JS (Java Scripting) is a scripting language that is used by Web page authors in designing interactive sites. It can interact with HTML source code, enabling Web developers to enhance their websites with dynamic content.

JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate.

JSAPI

JSAPI is the client-side (front-end) API of Mini Programs. It can be divided into 14 categories according to the functions that can be realized, including interface, multimedia, cache, file, location, network, equipment, data security, sharing, collection, custom general menu, the type of small program currently running version, custom analysis, and update management.

L

lifecycle

Lifecycle is the life and death of an object. It usually refers to the process by which a program is created, started, paused, evoked, stopped, and uninstalled. The lifecycle of a Mini Program can be divided into application lifecycle and page lifecycle.

M

mini program

Mini Programs are sub-applications that run inside an mobile app.

mini program component

Mini program components are mini program view controls which allow developers to combine components for development.

mini program framework

The Mini Program is divided into two layers: `app` and `page`. The app describes the whole program; the page describes the individual pages.

Mini program Developer Portal

Mini program developer portal is a website where the external developers can get tools, resources, and documentation for mini program development. To get started, the 3rd-party developer needs to register a developer account and submit necessary KYC document for verification purpose.

Mini Program Studio

Mini Program Studio is the Integrated Development Environment (IDE) for developers to write code for Mini Programs. It has a built-in Simulator for Preview, and a debug console to monitor the logs, as well as a full-fledged code editor that provides syntax lighting and auto-completion.

Mini Program Platform

Mini Program Platform is a platform that supports full mini program development lifecycle and runtime operations.

N

NPM

NPM is the default package manager for the JavaScript runtime environment Node. js. It consists of a command line client (also called NPM) and an online database of public and paid-for private packages (called the NPM registry).

O

Open API

Open API is the server-side (back-end) API of Mini Programs. With open APIs, Mini Programs can realize a number of functions such as user authorization, acquisition of member's basic information, and acquisition of user's phone number.

P

page lifecycle

Page lifecycle refers to the process of a page from entry to departure or from one page to another in a mini program.

PV

A PV (page view) is a Web analytics term that refers to each time when a Web page is successfully loaded onto a user's Web browser.

Q

QR code

Quick Response code that is used to provide access to information or mini programs on a smartphone.

R

RPX

Responsive Pixel that can be adapted with screen widths.

S

silence mode

Silence mode is used to get sensitive user information, which requires the user's permission on a wallet app.

SJS

SJS (Safe/Subset Javascript) is a set of custom scripting language of Mini Program that is used to build page structure in AXML.

SMTP

SMTP (Simple Mail Transfer Protocol) is a communication protocol for electronic mail transmission.

SSL

SSL (Secure Sockets Layer) is a standard security technology to establish an encrypted link between a server and a client.

U

user consent mode

User consent mode is the mode to get public user information without further permission from wallets.

UI

UI (User Interface) refers to the overall design of man-machine interaction, manipulationlogic and Interface of software.

URI

URI (Uniform Resource Identifiers)is a unique identifier used by web technologies. URIs may be used to identify anything, including real-world objects, such as people and places, concepts, or information resources such as web pages and books.

UTC

UTC (Coordinated Universal Time) is the primary time standard by which the world regulates clocks and time. It is within about 1 second of mean solar time at 0° longitude, and is not adjusted for daylight saving time.

URL

A URL (Uniform Resource Locator), colloquially termed a web address, is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it. A URL is a specific type of Uniform Resource Identifier (URI), although many people use the two terms interchangeably.

UV

UV (Unique Visitor) is a person who visits a site at least once in a reporting period as determined by their IP address.

W

webview

A view that displays web content inside the application. The mini program provides a web-view component to implement this function.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/about/glossary

---

Grid {#grid}

Last updated: 2022-07-03

Path: miniprogram_gcash

Grid

2022-07-03 18:44

Grid.

| | | | | | --- | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | **Required** | | list | Grid data. | `Array<icon, text>` | [] | Yes | | onGridItemClick | Callback when the grid is clicked. | (index: Number) => void | | No | | columnNum | Number of columns displayed per row | `2`, `3`, `4`, `5` | `3` | No | | circular | Circular or not, take effect when the columnName value is 4. | Boolean | `false` | No | | hasLine | Have borderline or not. | Boolean | `true` | No |

Example

copy

```
{
  "defaultTitle": "AntUI Component Library",
  "usingComponents": {
    "grid": "mini-antui/es/grid/index"
  }
}
```

copy

```
<grid onGridItemClick="onItemClick" columnNum="{{3}}" list="{{list3}}"
/>
```

copy

```
Page({
  data: {
    list3: [\
      {\
        icon: 'https://img.example.com/example1.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example2.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example3.png',\
        text: 'Title',\
```

```
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example4.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example5.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example6.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example7.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example8.png',\
        text: 'Title',\
        desc: 'text',\
      },\
      {\
        icon: 'https://img.example.com/example9.png',\
        text: 'Title',\
        desc: 'text',\
      },\
    ],
  },
  onItemClick(ev) {
    my.alert({
      content: ev.detail.index,
    });
  },
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_layout-navigation_grid

---

Guide {#guide}

Last updated: 2022-07-03

Path: miniprogram_gcash

Guide

2022-07-03 18:44

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_guide_guide

---

How to add members to workspaces {#how-to-add-members-to-workspaces}

Last updated: 2022-07-08

Path: miniprogram_gcash

How to add members to workspaces

2022-07-08 02:08

Overview

This topic provides steps for the wallet and merchants to add members to their own workspaces. After onboarding on Mini Program Platform, the wallet and merchants can add members to collaborate to manage their workspaces.

For members who have permission to add members to workspaces, see the following figure:

- Wallet: Only workspace admins have permission to add members to their workspaces.
- Merchant: Only developer admins have permission to add members to their workspaces.

Procedures

To add members to a workspace, you can follow the steps below:

Step 1: Navigate to Members tab

Log in to the platform and click **Members** on the navigation panel to the left. Then, click **+ New Members** to initiate the process.

Step 2: Invite members

Fill in the member's name and email and then set the role for the member. Roles on the platform are different for the wallet and merchants, see Workspace member roles for more information.

For the wallet

- You can set the member as a workspace admin, workspace reviewer, workspace developer, or workspace operator. If set the member as a workspace developer, you need to assign mini programs for the role.

For merchants

You can set the member as a developer admin, developer, or operator. If set the member as a developer or operator, you need to assign mini programs for the role.

After setting the role for the member, click **Invite** to send an invitation email to the member.

Step 3: Manage invitation

A member invitation is valid for3 days since you send it to the member. During this period, you can withdraw the invitation before the member accepts it. Once withdraw, you need to initiate the process again to add the member to the platform. If the invitation is expired, you can resend the invitation to the member.

Step 4: Manage members

After the member joins the workspace, you can block, delete, and set roles for the member. In addition, you can also check the statuses of members.

Block members

To block a member, click **Block** to continue the process. The blocked member cannot perform operations based on the assigned role but the member still displays on the member list. Once blocked, the member status will change from **Active** to **Inactive**.

Delete members

To delete a member, click **Delete** to continue the process. Once deleted, the number will be removed from the workspace.You can invite the member again.

Set roles for members

To set roles for a member, click **Set Role** and select the role that you want to set for the member. Then confirm the action to change member's role to the new one.

Check member status

Members who are invited to join a workspace move through different statuses from the time they are invited to when they are blocked.

| | | | --- | --- | --- | | **Status** | **Description** | **Possible Actions** | | Invitation Sent | Indicates that a workspace admin or developer admin has sent the invitation to a member and waits for the member to join the workspace. | Withdraw the invitation. | | Invitation Expired | Indicates that a workspace admin or developer admin has sent the invitation to a member but the member didn't accept it within the valid period. | Resend or withdraw the invitation. | | Active | Indicates that a member accepted the invitation to join the workspace. The member can perform operations normally based on the assigned role. | Block, delete, or set roles for the member. | | Inaction | Indicates that a member accepted the invitation to join a workspace but was blocked later. The member cannot perform operations normally until he or she is unblocked. | Unblock or set roles for the member. |

More information

Workspace member roles

Member role authorization in tenant workspace

Member role authorization in developer workspace

Mini program member roles

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/add-member-to-workspace

---

How to apply for authorization {#how-to-apply-for-authorization}

Last updated: 2022-07-07

Path: miniprogram_gcash

How to apply for authorization

2022-07-07 17:08

This procedure is available for ISV. When the ISV registers as a Third-Party Developer, the ISV can apply for the authorization.

Procedure

To apply for the authorization, complete the following steps:

1. Apply for an account.
2. Log into your account and go to the **Authorization** page.
3. Provide the following information to apply for the authorization:

- You company name
- Merchant name
- Merchant email
- Client ID
- Reason

Your approval request is now submitted. You will be automatcially authorized after your request is approved by workspace admins.

More information

Overview

Authorization

Member Role

Workflow Procedures

Manage Mini Program

Manage Workspace

Settings

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/apply-for-authorization

---

How to customize your analysis {#how-to-customize-your-analysis}

Last updated: 2022-07-07

Path: miniprogram_gcash

How to customize your analysis

2022-07-07 17:08

This document is intended for mini program developers to customize their analysis and track user behavior in a mini program.

What is customized analysis?

With the customized analysis function, you can perform multi-dimensional and near real-time analysis of user behavior in a mini program. In addition to the standard analysis, such as page views (PV) and unique visitors (UV), you can define different conditions and metrics to measure the mini program's performance according to user behavior.

For example, for an e-commerce mini program, you can configure the settings and obtain detailed analysis on the following scenarios:

1. What is the conversion rate for users who have taken a certain path, from viewing the product page, product details, user reviews, to placing an order, and finally paying to complete the purchase?
2. What is the participation rate of online activities across different hourly time periods?

Prerequisites

To get the most of your customized analysis, you need to have basic statistics to measure the analysis against. For example, you can analyze an e-commerce mini program from the following aspects:

- Order quantity and total payment.
- The conversion rate and churn rate at each step in the purchase process.

Procedures

To customize your analysis to meet your business requirements, complete the following steps:

1. Define an event

Define an event andconfigure the data reporting method based on your requirements. An event is an action made by the user in the mini program.

2. Publish an event

Collect data after publishing an event.

3. Analyze events and funnels

Analyze data collected according to events.

Related information

You can read more details about the analysis features in the following topics:

- Event management and analysis
- Funnel management and analysis

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/customize-analysis

---

How to release mini programs {#how-to-release-mini-programs}

Last updated: 2022-07-07

Path: miniprogram_gcash

How to release mini programs

2022-07-07 17:08

This topic introduces how to release mini programs in a workspace and what are preparations you need to make before starting the release process.

Before you begin

Before you release mini programs, you need to get ready with some prerequisites and check how members join the creation and release process:

- Apply for an account
- Complete the on-boarding process:
- For wallets: create the tenant workspace
- For merchants: onboard to the tenant workspace
- Download Mini Program Studio (IDE)
- Check how mini program member roles collaborate to construct the mini program life cycle.

Procedures

For a guided tour on how to release mini programs, see the following steps:

1. Create mini programs
2. Add mini program members
3. Upload versions
4. Configure mini programs
5. Release versions
6. Generate QR codes
7. Delete mini programs

More information

Manage mini programs

Members

Video Tutorials for Mini Program Platform

Develop mini programs

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/how-to-manage-miniprogram

---

How to send your feedback {#how-to-send-your-feedback}

Last updated: 2022-07-07

Path: miniprogram_gcash

How to send your feedback

2022-07-07 17:08

If you have any questions or suggestions on the mini program platform, you can send us your feedback in the header toolbar.

Procedures

1. Open your workspace
2. Select `Feedback` in the header toolbar and you can see `Feedback to the Console`.
3. Enter the following details:

- `Feedback Title`
- `Feedback Type`
- `Description`
- `Attachment`

4. Choose `Submit`, and your feedback is now sent to us.

More Information

Overview

Manage Mini Program

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/feedback

---

How to transform an HTML 5 mobile app into an HTML 5 mini program {#how-to-transform-an-html-5-mobile-app-into-an-html-5-mini-program}

Last updated: 2022-07-07

Path: miniprogram_gcash

How to transform an HTML 5 mobile app into an HTML 5 mini program

2022-07-07 17:08

This topic describes the process to transform an HTML5 mobile app (native app) into an HTML5 mini program. With the web development approach, Progressive Web Application (PWA), this transformation makes it possible to have a scalable solution while keeping the digital experience consistent.

The following two types of mini programs are supported on Mini Program Platform:

- **Mini Program based on DSL (Domain Specific Language)**

We recommend that you develop the default DSL-based mini programs. For more information, see About Mini Program.

- **Mini Program based on HTML5**

However, to help you integrate your existing HTML5 mobile apps with the wallet app, you can choose the HTML5 mini program type.

What are HTML5 mini programs?

HTML5 mini programs are the mini programs that are transformed from HTML5 mobile apps. HTML5 mini programs run in the WebView environment, so you can develop HTML5 mini programs as you do for web pages. Compared with HTML5 mobile apps, HTML5 mini programs are powered with more capabilities by calling mini program JSAPIs, thus improving user experience.

Benefits

There is no need to start from scratch if you already have an existing HTML5 mobile app. Different roles can enjoy the following benefits by transforming your HTML5 app into a mini program:

- **Convenient**

As compared with the time-consuming learning of the Mini Program framework, developers can reuse their familiar web development knowledge to get started with Mini Program.

- **Cost-efficient**

Merchants, ISVs, and wallet developers just need to add a few lines of JavaScript code to the existing HTML5 app, and the HTML5 mini programs become ready.

- **Consistent and user-friendly**

End users can also benefit from a consistent user experience of mini programs and native apps.

Before you start

Before you start, make sure you have applied for an account and get on-boarded.

Also, you can see the following process flow to get an overview of the detailed steps:

Note:

As compared with the steps of DSL mini programs, the following steps that are shown in red in the above process flow are required:

- **HTML5 mini program type**
- **Entrance URL**

Entrance of the HTML5 mini program. You can modify the entrance URL after the HTML5 mini program is created.

The following steps that are shown in gray in the above process flow are optional:

- **IDE**

You can develop the HTML5 mini programs just like you develop HTML5 web pages.

- **Build package process**
- **Quality review**

See procedures for a step-by-step guide on how to transform an HTML5 mobile app (native app) into an HTML5 mini program.

Procedures

To transform HTML5 apps to mini programs, complete the following steps:

Step 1: Create the HTML 5 mini program

When you create a new mini program, choose **Mini Program Based on HTML5** and enter the URL of your HTML5 mobile app into the **Entrance URL** field.

For more information on creating mini programs, see Create Mini Programs.

Step 2: Use wallet capabilities

To use the mini program JSAPIs, integrate the SDK according to your business requirements.

a. Mini Program JSAPIs in HTML5

Follow the steps below to use mini program JSAPIs in HTML5:

1. Import the JSBridge SDK by importing the following JS file to the HTML file:

copy

```
<script src="https://cdn.marmot-cloud.com/npm/hylid-
bridge/1.0.5/index.js"></script>
```

See the following HTML sample codes for reference:

copy

```
<!DOCTYPE html>
<html>
<head>
  <title>Miniprogram JSAPI Demo</title>
  <script src="https://cdn.marmot-cloud.com/npm/hylid-
bridge/1.0.5/index.js"></script>
</head>
<body>
  <div>
    <button id="alert">my.alert</button>
  </div>
  <script>
    var alertButton = document.getElementById('alert');
    alertButton.addEventListener('click', function () {
      my.alert({
        title: 'Test Alert!!',
        content: window.navigator.userAgent,
        buttonText: 'Alert Button',
```

```
        success: function (res) {
          my.alert({
            content: 'success!' + JSON.stringify(res),
          });
        },
        fail: function () {
          my.alert({
            content: 'fail!',
          });
        },
        complete: function () {
          my.alert({
            content: 'complete!',
          });
        },
      });
    });
  </script>
</body>
</html>
```

2. Call an API from the following available API list. The parameter in the API is an object. Each API has the following properties:

- **success**: callback function that indicates the successful API calling.
- **fail**: callback function that indicates the failed API calling.
- **complete**: callback function that indicates the completed API calling.

Take the **my.alert** JSAPI as an example. In this API, **success**, **fail**, and **complete** are the common properties. Specific properties are **title**, **content**, and **buttonText**.

For example, see the following sample codes to know how the **IAlertOptions** interface can be defined:

copy

```
interface IAlertOptions {
  title?: string;
  content?: string;
  buttonText?: string;
  success?: () => void;
  fail?: () => void;
  complete?: () => void;
}
```

Available API list

The following table lists all available JSAPIs. You can apply for your own JSAPIs if required.

| | | --- | --- | | **JSAPI Name** | **Description** | | my.alert | Displays alert box. | | my.confirm | Displays confirmation box. | | my.prompt | Prompts a user input | | my.showToast | Displays a toast, which disappears within certain seconds. | | my.hideToast | Hides a toast. | | my.showLoading | Displays the loading message. | | my.hideLoading | Hides the loading message. | | my.setNavigationBar | Sets the navigation bar text and style. | | my.showNavigationBarLoading | Shows navigation bar loading. | | my.hideNavigationBarLoading | Hides navigation bar loading. | | my.request | Network request | | my.getSystemInfo | Gets system information. | | my.getNetworkType | Gets current network status. | | my.showAuthGuide | Guides users to grant the authorization when the permission is needed. | | my.getAuthCode | Gets user's authorization code. | | my.saveImage | Saves online images to cellphone album. | | my.chooseImage | Takes a photo or chooses a photo from the album on the phone. | | my.previewImage | Previews images. | | my.getImageInfo | Gets image information. | | my.hideKeyboard | Hides keyboard. | | my.setStorage | Stores the data in the specified key in the local cache, which overlaps with the original data corresponding to the key. | | my.getStorage | Gets cached data. | | my.removeStorage | Removes cached data. | | my.clearStorage | Clears local data cache. | | my.tradePay | Initiates a payment. | | my.navigateToMiniProgram | Jumps to other mini programs. | | my.navigateBackMiniProgram | Returns to the previous Mini Program, only when another Mini Program jumps to the current Mini Program. | | my.getClipboard | Gets the clipboard data. | | my.setClipboard | Sets the clipboard data. | | my.vibrate | Invokes the vibrate ability of the device. | | my.makePhoneCall | Makes a phone call. | | my.multiLevelSelect | Selects in multiple levels. | | my.choosePhoneContact | Chooses the phone number of a contact person | | my.getLocation | Gets the current geographical location of the user. |

b. Open APIs

Depending on the wallet capabilities, you can use different open APIs.

Step 3: Release the mini program

You can request to release the mini program after the development. Before you submit the release request for approval, you need to check if all the required information is filled in. Enter the required information under **Mini Programs > Info** if you find some required information is missing. For more information, see Release Mini Programs and Approvals.

Now you have completed the process of transforming the HTML app into a mini program. You can start to manage the mini program on the platform. For more information, see Manage Mini Program.

Precautions

Browser compatibility

When the page is running in a browser, mini program JSAPIs cannot be called. You need to identify whether the page is running in a mini program or a browser by UserAgent, and then take actions accordingly.

See the following sample codes for reference:

copy

```
if (/MiniProgram/.test(navigator.userAgent)) {
    my.tradePay(params)
} else {
    // process payment in other browsers.
}
```

UserAgent

UserAgent which is retrieved via navigator.userAgent has some specific fields like **MiniProgram**, which can be used to identify whether the web page runs as an HTML mini program.

For a whole UserAgent, see the following sample codes for reference:

copy

```
Mozilla/5.0 (iPhone; CPU iPhone OS 14_4 like Mac OS X)
AppleWebKit/605.1.15 (KHTML, like Gecko) Mobile/18D46 ChannelId(35)
Ariver/1.0.11 Griver/2.23.0 AppContainer/1.9.1 MiniProgram
```

Restrictions

Reserved objects

my.JSBridge is reserved for global objectsdefined in the PWA runtime, and you cannot modify them.

console.log

The PWA runtime strongly depends on the default console.log object in the web context, which cannot be overridden or modified.

More information

Manage Mini Programs

Mini Program types

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/transform-html5

How-to videos {#how-to-videos}

Last updated: 2022-07-07

Path: miniprogram_gcash

How-to videos

2022-07-07 17:08

The Mini Program Platform provides comprehensive functionalities for you to develop and manage mini programs. To have a quick start for the mini program platform, you can watch the video tutorials.

Manage mini programs

For a quick overview of how to manage mini programs, you can watch a video here:

For more information about the introduction to the Mini Program Manage functionality, see Manage Mini programs.

Customize your analysis

For a quick overview of how to customize your analysis on mini programs, you can watch a video here:

For more information about the Customized Analysis functionality, see Event management and analysis and How to customize your analysis.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/how-to-videos

---

IDE Debugging {#ide-debugging}

Last updated: 2022-07-03

Path: miniprogram_gcash

IDE Debugging

2022-07-03 18:44

Overview

We provide two debugging methods:

- Simulator + Debugging tool
- Remote real machine debugging

The simulator can simulate most of the real machine APIs and has an integrated debugging tool. We suggest completing basic function and style debugging in the simulator and then performing the verification and debugging on real machine. However, the final running result shall be based on the real machine.

Simulator

The simulator provides the following functions:

- Device simulation (dimension, precision, etc.)
- Compiling log, compiling error prompt, refresh
- JSAPI simulation, custom configuration of simulation interface including position, Bluetooth, startup parameter, etc.

Debugging Tool

Together with the simulator, we provide customized Chrome devtool, and provide extension, such as axml, on basis of that. Default extensions include:

- AXML: Mini Program element based dom, css debugging
- Console: operation log, error viewing
- Storage: buffer data viewing and editing
- Sources: source code viewing and breakpoint debugging
- Network: network resource and requests viewing

Remote Debugging

In the remote debugging mode, a connection is established between IDE and cellphone. At the IDE end, you may perform breakpoint check, runtime information check, Network/Storage information viewing, remote log viewing, and so on. See more details in **Remote Debugging** document.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/mini-program-studio_debugging_ide-debugging

---

IDE Debugging {#ide-debugging}

Last updated: 2021-05-09

Path: miniprogram_gcash

IDE Debugging

2021-05-09 18:43

Overview

We provide two debugging methods:

- Simulator + Debugging tool
- Remote real machine debugging

The simulator can simulate most of the real machine APIs and has an integrated debugging tool. We suggest completing basic function and style debugging in the simulator and then performing the verification and debugging on real machine. However, the final running result shall be based on the real machine.

Simulator

The simulator provides the following functions:

- Device simulation (dimension, precision, etc.)
- Compiling log, compiling error prompt, refresh
- JSAPI simulation, custom configuration of simulation interface including position, Bluetooth, startup parameter, etc.

Debugging Tool

Together with the simulator, we provide customized Chrome devtool, and provide extension, such as axml, on basis of that. Default extensions include:

- AXML: Mini Program element based dom, css debugging
- Console: operation log, error viewing
- Storage: buffer data viewing and editing
- Sources: source code viewing and breakpoint debugging

- Network: network resource and requests viewing

Remote Debugging

In the remote debugging mode, a connection is established between IDE and cellphone. At the IDE end, you may perform breakpoint check, runtime information check, Network/Storage information viewing, remote log viewing, and so on. See more details in **Remote Debugging** document.

九色鹿

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/mini-program-studio_debugging_ide-debugging

---

Idempotency {#idempotency}

Last updated: 2021-05-09

Path: miniprogram_gcash

Idempotency

2021-05-09 18:43

If a request timeout error occurs when you call an API, you might attempt to resend the request. In this case, you can configure the specified idempotency fields in the request to help avoid unwanted duplication in case of failures and retries.

An API call is idempotent if it has the same result no matter how many times the API call is applied. For example, idempotency can guarantee that the payment is charged only once if the same API payment call is retried multiple times in the case of a timeout error. You can retry the request via using the same idempotency field to guarantee that no more than one charge is created.

Idempotency fields

The following table lists the idempotency fields of specific OpenAPI.

| | | | --- | --- | --- | | **Interface name** | **Idempotency field** | **Rule** | | - /v1/payments/pay - /v2/payments/pay | paymentRequestId | This field is used for the idempotence control. For the payment requests which are initiated with the same `paymentRequestId` and reach a final status (S or F), the must return the unique result. | | - /v1/payments/refund - /v2/payments/refund | refundRequestId | This field is used for the idempotence control. For the refund requests which are initiated with the same `refundRequestId` and reach a final status (S or F), themust return the unique result. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/api_idempotency

---

Idempotency {#idempotency}

Last updated: 2022-07-07

Path: miniprogram_gcash

Idempotency

2022-07-07 17:08

If a request timeout error occurs when you call an API, you might attempt to resend the request. In this case, you can configure the specified idempotency fields in the request to help avoid unwanted duplication in case of failures and retries.

An API call is idempotent if it has the same result no matter how many times the API call is applied. For example, idempotency can guarantee that the payment is charged only once if the same API payment call is retried multiple times in the case of a timeout error. You can retry the request via using the same idempotency field to guarantee that no more than one charge is created.

Idempotency fields

The following table lists the idempotency fields of specific OpenAPI.

| | | | --- | --- | --- | | **API name** | **Idempotency field** | **Rule** | | - /v1/payments/pay | paymentRequestId | This field is used for the idempotence control. For the payment requests which are initiated with the same `paymentRequestId` and reach a final status (S or F), the native app must return the unique result. | | - /v1/payments/refund | refundRequestId | This field is used for the idempotence control. For the refund requests which are initiated with the same `refundRequestId` and reach a final status (S or F), the native must return the unique result. |

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_idempotency

---

InputItem {#inputitem}

Last updated: 2022-07-03

Path: miniprogram_gcash

InputItem

2022-07-03 18:44

Text input.

Sample Code

copy

```
// API-DEMO page/component/input-item/input-item.json
{
  "defaultTitle": "Mini Program AntUI component library",
  "usingComponents": {
    "list": "mini-antui/es/list/index",
    "list-item": "mini-antui/es/list/list-item/index",
    "input-item": "mini-antui/es/input-item/index",
    "picker-item": "mini-antui/es/picker-item/index"
  }
}
```

copy

```
<!-- API-DEMO page/component/input-item/input-item.axml -->
<view>
  <view style="margin-top: 10px;" />
  <list>
    <input-item
      data-field="cardNo"
      clear="{{true}}"
      value="{{cardNo}}"
      className="dadada"
      placeholder="Bank card number"
      focus="{{inputFocus}}"
      onInput="onItemInput"
      onFocus="onItemFocus"
      onBlur="onItemBlur"
      onConfirm="onItemConfirm"
      onClear="onClear"
    >
      Card number
      <view slot="extra" class="extra" onTap="onExtraTap"></view>
    </input-item>
    <picker-item
      data-field="bank"
      placeholder="Select issuing bank"
      value="{{bank}}"
      onPickerTap="onPickerTap"
    >
      Issuing bank
```

```
      </picker-item>
      <input-item
        data-field="name"
        placeholder="Name"
        type="text"
        value="{{name}}"
        clear="{{true}}"
        onInput="onItemInput"
        onClear="onClear"
      >
        Name
      </input-item>
      <input-item
        data-field="password"
        placeholder="Password"
        password
      >
        Password
      </input-item>
      <input-item
        data-field="remark"
        placeholder="Remarks"
        last="{{true}}"
      />
    </list>
    <view style="margin: 10px;">
      <button type="primary" onTap="onAutoFocus">Focus</button>
    </view>
  </view>
```

copy

```
// API-DEMO page/component/input-item/input-item.js
const banks = ['Mybank', 'CCB', 'ICBC', 'SPDB'];

Page({
  data: {
    cardNo: '1234****',
    inputFocus: true,
    bank: '',
    name: '',
  },
  onAutoFocus() {
    this.setData({
      inputFocus: true,
    });
  },
  onExtraTap() {
    my.alert({
      content: 'extra tapped',
    });
  },
```

```
  onItemInput(e) {
    this.setData({
      [e.target.dataset.field]: e.detail.value,
    });
  },
  onItemFocus() {
    this.setData({
      inputFocus: false,
    });
  },
  onItemBlur() {},
  onItemConfirm() {},
  onClear(e) {
    this.setData({
      [e.target.dataset.field]: '',
    });
  },
  onPickerTap() {
    my.showActionSheet({
      title: 'Select issuing bank',
      items: banks,
      success: (res) => {
        this.setData({
          bank: banks[res.index],
        });
      },
    });
  },
});
```

copy

```
/* API-DEMO page/component/input-item/input-item.acss */
.extra {
  background-image: url('https://img.example.com/example.svg');
  background-size: contain;
  background-repeat: no-repeat;
  background-position: right center;
  opacity: 0.2;
  height: 20px;
  width: 20px;
  padding-left: 10px;
}
```

Attributes

| | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | | className | Customized class. | String | '' | | labelCls | Customized label class. | String | '' | | inputCls | Customized input class. | String | '' | | last | Is the last row or not. | Boolean | false | | value | Initial contents. | String | '' | | name | Component name, used for getting data via form submission. | String | '' | | type | Type of input, effective values including text, number,

idcard and digit (see table below for details). | String | text | | password | Is password type or not. | Boolean | false | | placeholder | Placeholder. | String | '' | | placeholderStyle | Specify the style of the placeholder. | String | '' | | placeholderClass | Specify the style class of the placeholder. | String | '' | | disabled | Disable or not. | Boolean | false | | maxlength | Maximum length. | Number | 140 | | focus | Get focus. | Boolean | false | | clear | Have clear function or not, taking effect only when disabled is false. | Boolean | false | | onInput | Trigger the input event on keyboard input. | (e: Object) => void | - | | onConfirm | Trigger on clicking keyboard completion. | (e: Object) => void | - | | onFocus | Trigger on getting focus. | (e: Object) => void | - | | onBlur | Trigger on losing focus. | (e: Object) => void | - | | onClear | Trigger on clicking the clear icon. | () => void | - |

Description of type attribute value

• text: Character input box

• number: Pure number input box (number within 0-9)

• idcard: Input box for ID card number (number within 0-9 and character x)

• digit: number input box (number within 0-9 and decimal point . used for number containing a decimal)

Note: The type attribute value affects the keyboard type with real machine and may not be effective in simulators.

Slots

| | | | --- | --- | --- | | **slotname** | **Description** | **Required** | | extra | Used to render the description right to input-item. | No |

Common Questions

When the `setData` data is empty, the breakpoint money value is set to empty, but why 0 is still shown in the input box?

When `this.setData` sets data as empty, it does not render the page. It is recommended to use the component clear.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_form_inputitem

---

Introduction {#introduction}

Last updated: 2021-05-09

Path: miniprogram_gcash

Introduction

2021-05-09 18:43

By calling my.canIUse('component'), it is possible to judge whether the custom component function is usable in the current version. For the usage of my.canIUse(), see canIUse.

The custom component function can abstract the functions to be reused into custom component so that it can be reused on different pages. It also allows publishing custom component on npm, so that it can be reused in different Mini Programs. For the usage of npm, see npm.

The my.canIUse('component2') can be used to check if the current version supports the new custom component function.

To use the related functions of component2, in the **Details > Project configuration** of the developer tool, check component2:

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_custom-component_introduction

---

Introduction {#introduction}

Last updated: 2022-07-04

Path: miniprogram_gcash

Introduction

2022-07-04 03:44

By calling my.canIUse('component'), it is possible to judge whether the custom component function is usable in the current version. For the usage of my.canIUse(), see canIUse.

The custom component function can abstract the functions to be reused into custom component so that it can be reused on different pages. It also allows publishing custom component on npm, so that it can be reused in different Mini Programs. For the usage of npm, see npm.

The my.canIUse('component2') can be used to check if the current version supports the new custom component function.

To use the related functions of component2, in the **Details > Project configuration** of the developer tool, check component2:

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_custom-component_introduction

---

JSAPI {#jsapi}

Last updated: 2022-07-07

Path: miniprogram_gcash

JSAPI

2022-07-07 17:08

You can create JSAPIs as the metadata that can be used to construct the content of a feature. You can select multiple JSAPIs for a feature and assign this feature to a mini program. For more information, see Feature.

Features

You can perform the following activities:

- Create new JSAPIs with the following details:
- `JSAPI Name` (Mandatory)
- `JSAPI Usage Scenarios` (Mandatory)
- `App`: You can choose the native app to which the mini program with this JSAPI is released.
- Request and response parameters (callback and return parameters)
- `Error Code`
- `Sample Code`

Note:

You can refer to the JSAPI details in your IDE and enter the details to these above accordingly. For more information, see JSAPI Reference.

- See JSAPIs in a list

You can see the name, description, status, and change history of the JSAPIs in a list.

More information

Overview

Manage Mini Programs

Feature

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/jsapi

---

JSAPI Reference {#jsapi-reference}

Last updated: 2023-01-29

Path: miniprogram_gcash

JSAPI Reference

2023-01-29 20:55

Basic

| | | --- | --- | | **API Name** | **Function Description** | | my.canIUse | Judge whether the current Mini Program API, incoming parameter or return value, component, attribute, etc. are supported in the current version. | | my.SDKVersion | Get the SDK version. | | my.getAppIdSync | Obtain the Mini Program App ID synchronously. | | my.getRunScene | Obtain the running version of the current Mini Program. |

In-App Event

| | | --- | --- | | **API Name** | **Function Description** | | my.offAppHide | Unlisten for the event that the mini program is switched to background from foreground. | | my.offAppShow | Unlisten for the event that the mini program is switched to foreground from background. | | my.onAppHide | Listen for the event that the mini program is switched to background from foreground. | | my.onAppShow | Listen for the event that the mini program is switched to foreground from background. | | my.offError | Unlisten for the event that JS errors occur in the mini program. | | my.onError | Listen for the event that errors occur in the mini program. | | my.offUnhandledRejection | Unlisten for the *unhandledrejection* event. | | my.onUnhandledRejection | Listen for the *unhandledrejection* event. |

UI

NavigationBar

| | | | --- | --- | | **API Name** | **Function Description** | | my.hideBackHome | Hide back to home button. | | my.showNavigationBarLoading | Show navigation bar loading. | | my.hideNavigationBarLoading | Hide navigation bar loading. | | my.setNavigationBar | Set the navigation bar text and style. |

TabBar

| | | | --- | --- | | **API Name** | **Function Description** | | my.hideTabBar | Hide the tabBar. | | my.setTabBarStyle | Set the style of tabBar. |

Note: For the FAQs of tab bar, refer to TabBar FAQ.

Route

| | | | --- | --- | | **API Name** | **Function Description** | | my.switchTab | Jump to the specified tabBar page and close all other pages that are not tabBar. | | my.navigateTo | Maintain the current page and jump to the specified page within the application.
Use `my.navigateBack` to return to the original page. | | my.navigateBack | Close the current page and return to the previous one or more pages. | | my.redirectTo | Close the current page and jump to the specified page within the application. | | my.reLaunch | Close all current pages and jump to the specified page within the application. |

FeedBack

| | | | --- | --- | | **API Name** | **Function Description** | | my.alert | Alert box. | | my.confirm | Confirm box. | | my.prompt | Prompt box. | | my.showLoading | Show the loading hint. | | my.hideLoading | Hide the loading hint. | | my.showToast | Show a weak hint, which disappears in the specified seconds. | | my.hideToast | Hide the weak hint. | | my.showActionSheet | Display an operation menu. |

Pulldown

| | | | --- | --- | | **API Name** | **Function Description** | | my.startPullDownRefresh | Start the pull-to-refresh function. | | onPullDownRefresh | On the Page, customize the onPullDownRefresh function to listen to the pull-to-refresh event of user. | | my.stopPullDownRefresh | Stop the pull-to-refresh for the current page. |

Contact

| | | | --- | --- | | **API Name** | **Function Description** | | my.choosePhoneContact | Choose contact from system contact. |

Choose Data

| | | | --- | --- | | **API Name** | **Function Description** | | my.datePicker | Open a date selection list. |

Animation

| | | | --- | --- | | **API Name** | **Function Description** | | my.createAnimation | Create an instance of animation. |

Canvas

| | | | --- | --- | | **API Name** | **Function Description** | | my.createCanvasContext | Create the context of the canvas. |

Keyboard

| | | | --- | --- | | **API Name** | **Function Description** | | my.hideKeyboard | Hide keyboard. |

Scroll

| | | | --- | --- | | **API Name** | **Function Description** | | my.pageScrollTo | Scroll to the destination location of the page. |

SelectorQuery

| | | | --- | --- | | **API Name** | **Function Description** | | my.createSelectorQuery | Create an instance of SelectorQuery. |

Multiple Level Select

| | | | --- | --- | | **API Name** | **Function Description** | | my.multiLevelSelect | Multiple level selector, used for associated select with multiple levels. |

Set Background

| | | | --- | --- | | **API Name** | **Function Description** | | my.setBackgroundColor | Set background color. |

Set Page Pulldown

| | | | --- | --- | | **API Name** | **Function Description** | | my.setCanPullDown | Set whether page can support pulldown. |

Media

Image

| | | | --- | --- | | **API Name** | **Function Description** | | my.chooseImage | Choose image from camera or album of cellphone. | | my.previewImage | Preview image. | | my.saveImage | Save the online images to cellphone album. | | my.getImageInfo | Get the information of the image. | | my.compressImage | Compress large images to fulfill the size restrictions from some mini programs. |

Storage

| | | | --- | --- | | **API Name** | **Function Description** | | my.getStorage | Get cached data. | | my.getStorageSync | Get cached data synchronously. | | my.setStorage | Store the data in the specified key in the local cache, which overlaps the original data corresponding to the key. | | my.setStorageSync | Store synchronously the data in the specified key in the local cache. | | my.removeStorage | Remove cached data. | | my.removeStorageSync | Remove cached data synchronously. | | my.clearStorage | Clear local data cache. | | my.clearStorageSync | Clear local data cache synchronously. |

File

| | | | --- | --- | | **API Name** | **Function Description** | | my.saveFile | Save file to the device. | | my.getFileInfo | Get the information of the file. | | my.getSavedFileInfo | Get the information of the saved file. | | my.getSavedFileList | Get the information of all the saved file. | | my.removeSavedFile | Remove the saved file. | | my.chooseFileFromDisk | Choose a file to upload. | | my.openDocument | Enable users to preview a PDF file within the mini program. |

Location

| | | | --- | --- | | **API Name** | **Function Description** | | my.getLocation | Get the current geographical location of the user. | | my.openLocation | View the location on the built-in map. | | my.chooseLocation | Open the built-in map to choose a location. |

Map

| | | | --- | --- | | **API Name** | **Function Description** | | my.createMapContext | Create and return a map context object mapContext. |

Network

| | | | --- | --- | | **API Name** | **Function Description** | | my.request | Network request. | | my.uploadFile | Upload the local resource to the developer server. | | my.downloadFile | Download file resource to local location. | | my.connectSocket | Create a WebSocket

connection. | | my.onSocketOpen | Listen to the event of enabling the WebSocket connection. | | my.offSocketOpen | Unlisten to the event of enabling the WebSocket connection. | | my.onSocketError | Listen to WebSocket error events. | | my.offSocketError | Unlisten to WebSocket error events. | | my.sendSocketMessage | Send data over WebSocket connection. | | my.onSocketMessage | Listen to the event of receiving server messages by WebSocket. | | my.offSocketMessage | Unlisten to the event of receiving server messages by WebSocket. | | my.closeSocket | Close the WebSocket connection. | | my.onSocketClose | Listen to the event of closing the WebSocket connection. | | my.offSocketClose | Unlisten to the event of closing the WebSocket connection. |

Device

System Information

| | | --- | --- | | **API Name** | **Function Description** | | my.getSystemInfo | Get system information. |

Network Status

| | | --- | --- | | **API Name** | **Function Description** | | my.getNetworkType | Get the current network status. |

Clipboard

| | | --- | --- | | **API Name** | **Function Description** | | my.getClipboard | Get the clipboard data. | | my.setClipboard | Set the clipboard data. |

Watch Shake

| | | --- | --- | | **API Name** | **Function Description** | | my.watchShake | The watchshake function. |

Accelerometer

| | | --- | --- | | **API Name** | **Function Description** | | my.onAccelerometerChange | Listen to the acceleration data event. | | my.offAccelerometerChange | Unlisten to the acceleration data event. |

Compass

| | | --- | --- | | **API Name** | **Function Description** | | my.onCompassChange | Listen to the compass data change event. | | my.offCompassChange | Unlisten to the compass data change event. |

Vibrate

| | | --- | --- | | **API Name** | **Function Description** | | my.vibrate | Invoke the vibrate ability of device. |

Make Phone Call

| | | | --- | --- | | **API Name** | **Function Description** | | my.makePhoneCall | Make a phone call. |

Get Server Time

| | | | --- | --- | | **API Name** | **Function Description** | | my.getServerTime | Get the server time. |

Capture Screen

| | | | --- | --- | | **API Name** | **Function Description** | | my.onUserCaptureScreen | Listen to the capture screen event by users. | | my.offUserCaptureScreen | Cancel the listen to the capture screen event by users. |

Screen Brightness

| | | | --- | --- | | **API Name** | **Function Description** | | my.setKeepScreenOn | Set whether screen keeps awake. | | my.getScreenBrightness | Get the screen brightness. | | my.setScreenBrightness | Set the screen brightness. |

Screen Orientation

| | | | --- | --- | | **API Name** | **Function Description** | | my.getScreenOrientation | Get screen orientation | | my.setScreenOrientation | Set screen orientation |

Setting

| | | | --- | --- | | **API Name** | **Function Description** | | my.openSetting | Open the mini program settings page. | | my.getSetting | Obtain the user's current settings. |

Add Phone Contact

| | | | --- | --- | | **API Name** | **Function Description** | | my.addPhoneContact | Add contact to phone contact. |

Permission Guide

| | | | --- | --- | | **API Name** | **Function Description** | | my.showAuthGuide | Guide user to grant the authorization when the permission needed. |

Scan

| | | | --- | --- | | **API Name** | **Function Description** | | my.scan | Call the scanning QR code function. |

Memory Warning

| | | --- | --- | | **API Name** | **Function Description** | | my.onMemoryWarning | Listen to the insufficient memory alarm event. | | my.offMemoryWarning | Unlisten to the insufficient memory alarm event. |

Battery Information

| | | --- | --- | | **API Name** | **Function Description** | | my.getBatteryInfo | Obtain the battery level and the charging state of the current device asynchronously. | | my.getBatteryInfoSync | Obtain the battery level and the charging state of the current device synchronously. |

Bluetooth

| | | --- | --- | | **Reference** | **Description** | | Bluetooth API Overview | Provide the system requirement and the process flow on how the Bluetooth APIs work. | | Bluetooth API Error Code Table | List the normal error code and relative solutions. | | Bluetooth API FAQ | FAQs about using Bluetooth API. |

BLE

| | | --- | --- | | **API Name** | **Function Description** | | my.connectBLEDevice | Connect to low energy Bluetooth devices. | | my.disconnectBLEDevice | Disconnect to low energy Bluetooth devices. | | my.getBLEDeviceCharacteristics | Get the characteristics of low energy Bluetooth devices. | | my.getBLEDeviceServices | Get all the low energy Bluetooth devices that are discovered, including the connected devices. | | my.notifyBLECharacteristicValueChange | Enable the function to notify changes to the characteristic value. | | my.offBLECharacteristicValueChange | Disable the function to notify changes to the characteristic value. | | my.offBLEConnectionStateChanged | Disable the the event listener for the connection status. | | my.onBLECharacteristicValueChange | Enable the event listener for changes to the characteristic value. | | my.onBLEConnectionStateChanged | Enable the event listener for changes to the connection status, such as device lost and device disconnected. | | my.readBLECharacteristicValue | Read the characteristic value. | | my.writeBLECharacteristicValue | Write data to the characteristic value. |

Bluetooth

| | | --- | --- | | **API Name** | **Function Description** | | my.closeBluetoothAdapter | Close the Bluetooth module in the mini program. | | my.openBluetoothAdapter | Initialize the Bluetooth module in the mini program. | | my.getBluetoothAdapterState | Check the Bluetooth adapter status in the mini program. | | my.startBluetoothDevicesDiscovery | Start discovering Bluetooth devices. | | my.stopBluetoothDevicesDiscovery | Stop discovering Bluetooth devices. | | my.getBluetoothDevices | Get all the Bluetooth devices that are discovered, including those that are connected to the current device. | | my.getConnectedBluetoothDevices | Get the Bluetooth devices that are connected. | | my.onBluetoothDeviceFound | Use this API when a new Bluetooth device is found. | | my.offBluetoothDeviceFound | Remove the Bluetooth devices that are found. | | my.onBluetoothAdapterStateChange | Monitor the Bluetooth adapter state changes. | | my.offBluetoothAdapterStateChange | Remove the Bluetooth adapter with a state change. |

Sharing

| | | | --- | --- | | **API Name** | **Function Description** | | my.showSharePanel | Trigger the sharing. |

Update

| | | | --- | --- | | **API Name** | **Function Description** | | my.getUpdateManager | Create an *UpdateManager* object. |

web-view

| | | | --- | --- | | **API Name** | **Function Description** | | my.createWebViewContext | By creating `webViewContext`, mini program can send message to `web-view`. This API will create a `webViewContext` instance. |

Site Info

| | | | --- | --- | | **API Name** | **Function Description** | | my.getSiteInfo | Get the site information. |

Open Capabilities

| | | | --- | --- | | **API Name** | **Function Description** | | my.navigateToMiniProgram | Jump to other mini program. | | my.navigateBackMiniProgram | Return to the previous mini program, only when another mini program jumps to the current mini program. | | my.getAuthCode | Get auth code. | | my.getOpenUserInfo | Get basic information about user. | | my.tradePay | Start to pay. | | my.signContract | Sign the withholding contract. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_api-reference

---

JSAPI Reference {#jsapi-reference}

Last updated: 2022-07-24

Path: miniprogram_gcash

JSAPI Reference

2022-07-24 23:36

Basic

| | | | --- | --- | | **API Name** | **Function Description** | | my.canIUse | Judge whether the current Mini Program API, incoming parameter or return value, component, attribute, etc. are supported in the current version. | | my.SDKVersion | Get the SDK version. | | my.getAppIdSync | Obtain the Mini Program App ID synchronously. | | my.getRunScene | Obtain the running version of the current Mini Program. |

UI

NavigationBar

| | | | --- | --- | | **API Name** | **Function Description** | | my.hideBackHome | Hide back to home button. | | my.hideNavigationBarLoading | Hide navigation bar loading. | | my.setNavigationBar | Set the navigation bar text and style. | | my.showNavigationBarLoading | Show navigation bar loading. |

TabBar

| | | | --- | --- | | **API Name** | **Function Description** | | my.hideTabBar | Hide the tabBar. | | my.setTabBarStyle | Set the style of tabBar. | | TabBar FAQ | The FAQ of TabBar. |

Route

| | | | --- | --- | | **API Name** | **Function Description** | | my.switchTab | Jump to the specified tabBar page and close all other pages that are not tabBar. | | my.navigateTo | Maintain the current page and jump to the specified page within the application.
Use `my.navigateBack` to return to the original page. | | my.reLaunch | Close all current pages and jump to the specified page within the application. | | my.navigateBack | Close the current page and return to the previous one or more pages. | | my.redirectTo | Close the current page and jump to the specified page within the application. |

FeedBack

| | | | --- | --- | | **API Name** | **Function Description** | | my.alert | Alert box. | | my.confirm | Confirm box. | | my.prompt | Prompt box. | | my.showToast | Show a weak hint, which disappears in the specified seconds. | | my.hideToast | Hide the weak hint. | | my.showLoading | Show the loading hint. | | my.hideLoading | Hide the loading hint. | | my.showActionSheet | Display an operation menu. |

Pulldown

| | | | --- | --- | | **API Name** | **Function Description** | | onPullDownRefresh | On the Page, customize the onPullDownRefresh function to listen to the pull-to-refresh event of user. | | my.stopPullDownRefresh | Stop the pull-to-refresh for the current page. | | my.startPullDownRefresh | Start the pull-to-refresh function. |

Contact

| | | | --- | --- | | **API Name** | **Function Description** | | my.choosePhoneContact | Choose contact from system contact. |

Choose Data

| | | | --- | --- | | **API Name** | **Function Description** | | my.datePicker | Open a date selection list. |

Animation

| | | | --- | --- | | **API Name** | **Function Description** | | my.createAnimation | Create an instance of animation. |

Canvas

| | | | --- | --- | | **API Name** | **Function Description** | | my.createCanvasContext | Create the context of the canvas. |

Keyboard

| | | | --- | --- | | **API Name** | **Function Description** | | my.hideKeyboard | Hide keyboard. |

Scroll

| | | | --- | --- | | **API Name** | **Function Description** | | my.pageScrollTo | Scroll to the destination location of the page. |

SelectorQuery

| | | | --- | --- | | **API Name** | **Function Description** | | my.createSelectorQuery | Create an instance of SelectorQuery. |

Multiple Level Select

| | | | --- | --- | | **API Name** | **Function Description** | | my.multiLevelSelect | Multiple level selector, used for associated select with multiple levels. |

Set Background

| | | | --- | --- | | **API Name** | **Function Description** | | my.setBackgroundColor | Set background color. |

Set Page Pulldown

| | | | --- | --- | | **API Name** | **Function Description** | | my.setCanPullDown | Set whether page can support pulldown. |

Media

Image

| | | | --- | --- | | **API Name** | **Function Description** | | my.chooseImage | Choose image from camera or album of cellphone. | | my.previewImage | Preview image. | | my.saveImage | Save the online images to cellphone album. | | my.getImageInfo | Get the information of the image. |

Storage

| | | | --- | --- | | **API Name** | **Function Description** | | my.setStorage | Store the data in the specified key in the local cache, which overlaps the original data corresponding to the key. | | my.setStorageSync | Store synchronously the data in the specified key in the local cache. | | my.getStorage | Get cached data. | | my.getStorageSync | Get cached data synchronously. | | my.removeStorage | Remove cached data. | | my.removeStorageSync | Remove cached data synchronously. | | my.clearStorage | Clear local data cache. | | my.clearStorageSync | Clear local data cache synchronously. |

File

| | | | --- | --- | | **API Name** | **Function Description** | | my.saveFile | Save file to the device. | | my.getFileInfo | Get the information of the file. | | my.getSavedFileInfo | Get the information of the saved file. | | my.getSavedFileList | Get the information of all the saved file. | | my.removeSavedFile | Remove the saved file. |

Location

| | | | --- | --- | | **API Name** | **Function Description** | | my.getLocation | get the current geographical location of the user. |

Network

| | | | --- | --- | | **API Name** | **Function Description** | | my.request | Network request. | | my.uploadFile | Upload the local resource to the developer server. | | my.downloadFile | Download file resource to local location. | | my.connectSocket | Create a WebSocket connection. | | my.onSocketOpen | Listen to the event of enabling the WebSocket connection. | | my.offSocketOpen | Unlisten to the event of enabling the WebSocket connection. | | my.onSocketError | Listen to WebSocket error events. | | my.offSocketError | Unlisten to WebSocket error events. | | my.sendSocketMessage | Send data over WebSocket connection. | | my.onSocketMessage | Listen to the event of receiving server

messages by WebSocket. | | my.offSocketMessage | Unlisten to the event of receiving server messages by WebSocket. | | my.closeSocket | Close the WebSocket connection. | | my.onSocketClose | Listen to the event of closing the WebSocket connection. | | my.offSocketClose | Unlisten to the event of closing the WebSocket connection. |

Device

System Information

| | | | --- | --- | | **API Name** | **Function Description** | | my.getSystemInfo | Get system information. |

Network Status

| | | | --- | --- | | **API Name** | **Function Description** | | my.getNetworkType | Get the current network status. |

Clipboard

| | | | --- | --- | | **API Name** | **Function Description** | | my.getClipboard | Get the clipboard data. | | my.setClipboard | Set the clipboard data. |

Watch Shake

| | | | --- | --- | | **API Name** | **Function Description** | | my.watchShake | The watchshake function. |

Accelerometer

| | | | --- | --- | | **API Name** | **Function Description** | | my.onAccelerometerChange | Listen to the acceleration data event. | | my.offAccelerometerChange | Unlisten to the acceleration data event. |

Compass

| | | | --- | --- | | **API Name** | **Function Description** | | my.onCompassChange | Listen to the compass data change event. | | my.offCompassChange | Unlisten to the compass data change event. |

Vibrate

| | | | --- | --- | | **API Name** | **Function Description** | | my.vibrate | Invoke the vibrate ability of device. |

Make Phone Call

| | | | --- | --- | | **API Name** | **Function Description** | | my.makePhoneCall | Make a phone call. |

Get Server Time

| | | | --- | --- | | **API Name** | **Function Description** | | my.getServerTime | Get the server time. |

Capture Screen

| | | | --- | --- | | **API Name** | **Function Description** | | my.onUserCaptureScreen | Listen to the capture screen event by users. | | my.offUserCaptureScreen | Cancel the listen to the capture screen event by users. |

Screen Brightness

| | | | --- | --- | | **API Name** | **Function Description** | | my.setKeepScreenOn | Set whether screen keeps awake. | | my.getScreenBrightness | Get the screen brightness. | | my.setScreenBrightness | Set the screen brightness. |

Setting

| | | | --- | --- | | **API Name** | **Function Description** | | my.openSetting | Open the Mini Program settings page. | | my.getSetting | Obtain the user's current settings. |

Add Phone Contact

| | | | --- | --- | | **API Name** | **Function Description** | | my.addPhoneContact | Add contact to phone contact. |

Permission Guide

| | | | --- | --- | | **API Name** | **Function Description** | | my.showAuthGuide | Guide user to grant the authorization when the permission needed. |

Scan

| | | | --- | --- | | **API Name** | **Function Description** | | my.scan | Call the scanning QR code function. |

Memory Warning

| | | | --- | --- | | **API Name** | **Function Description** | | my.onMemoryWarning | Listen to the insufficient memory alarm event. | | my.offMemoryWarning | Unlisten to the insufficient memory alarm event. |

Battery Information

| | | --- | --- | | **API Name** | **Function Description** | | my.getBatteryInfo | Obtain the battery level and the charging state of the current device asynchronously. | | my.getBatteryInfoSync | Obtain the battery level and the charging state of the current device synchronously. |

Bluetooth

| | | --- | --- | | **API Name** | **Function Description** | | Bluetooth API Error Code Table | List the normal error code and relative solutions. | | Bluetooth API FAQ | FAQ about using Bluetooth API. |

BLE

| | | --- | --- | | **API Name** | **Function Description** | | my.connectBLEDevice | Connect to low energy bluetooth devices. | | my.disconnectBLEDevice | Disconnect to low energy bluetooth devices. | | my.getBLEDeviceCharacteristics | Get the characteristics of low energy bluetooth devices. | | my.getBLEDeviceServices | Get all the low energy bluetooth devices that are discovered, including the connected devices. | | my.notifyBLECharacteristicValueChange | Enable the function to notify changes to the characteristic value. | | my.offBLECharacteristicValueChange | Disable the function to notify changes to the characteristic value. | | my.offBLEConnectionStateChanged | Disable the the event listener for the connection status. | | my.onBLECharacteristicValueChange | Enable the event listener for changes to the characteristic value. | | my.onBLEConnectionStateChanged | Enable the the event listener for changes to the connection status , such as device lost and device disconnected. | | my.readBLECharacteristicValue | Read the characteristic value. | | my.writeBLECharacteristicValue | Write data to the characteristic value. |

Bluetooth

| | | --- | --- | | **API Name** | **Function Description** | | my.openBluetoothAdapter | Use this API to initialize the Bluetooth module in the mini program. | | my.closeBluetoothAdapter | Use this API to close the Bluetooth module in the mini program. | | my.getBluetoothAdapterState | Use this API to check the Bluetooth adapter status in the mini program. | | my.startBluetoothDevicesDiscovery | Use this API to start discovering bluetooth devices. | | my.stopBluetoothDevicesDiscovery | Use this API to stop discovering bluetooth devices. | | my.getBluetoothDevices | Use this API to get all the bluetooth devices that are discovered, including those that are connected to the current device. | | my.getConnectedBluetoothDevices | Use this API to get the bluetooth devices that are connected. | | my.onBluetoothDeviceFound | Use this API when a new Bluetooth device is found. | | my.offBluetoothDeviceFound | Use this API to remove the bluetooth devices that are found. | | my.onBluetoothAdapterStateChange | Use this API to monitor the bluetooth adapter state changes. | | my.offBluetoothAdapterStateChange | Use this API to remove the bluetooth adapter with a state change. |

web-view

| | | --- | --- | | **API Name** | **Function Description** | | my.createWebViewContext | By creating `webViewContext`, Mini Program can send message to `web-view`. This API will create a webViewContext instance |

Site Info

| | | --- | --- | | **API Name** | **Function Description** | | my.getSiteInfo | Get the site information. |

Open Capabilities

| | | --- | --- | | **API Name** | **Function Description** | | my.getAuthCode | Get auth code. | | my.getOpenUserInfo | Get basic information about user. | | my.tradePay | Start to pay. | | my.navigateToMiniProgram | Jump to other Mini Program. | | my.navigateBackMiniProgram | Return to the previous Mini Program, only when another Mini Program jumps to the current Mini Program. | | my.signContract | Sign the withholding contract. | | my.getPhoneNumber | Get the phone number. |

Experience Mini Program

Developers can use the Android Demo App to scan the QR code shown in the API documents.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/api_api-reference

---

Launching the User Interface {#launching-the-user-interface}

Last updated: 2022-07-03

Path: miniprogram_gcash

Launching the User Interface

2022-07-03 18:44

Prerequisites

Make sure you have created a developer account. For more information, see Apply for an Account.

Features

When you launch the Mini Program Studio, you can do the following tasks in the user interface.

- Login
- Joined workspaces
- Create a new Mini Program
- Open an existing Mini Program
- Delete a Mini Program

For more information, see the demos below:

Login

The first time opening the Mini Program Studio, login is required. You can use the account that applied from Mini Program Developer Portal to login the Mini Program.

If password is forgot, try to retrieve the password in the Mini Program Developer Portal.

Joined Workspaces

After login, you can see which workspaces you have joined in the left side of the launch interface.

A developer can join multiple workspaces by invitation of workspace admins, and Mini Programs are separated in different workspaces. So here you can develop different Mini Program project for different workspaces.

Create a New Mini Program

The first time opening the Mini Program, you can create a new Mini Program project. The Mini Program Studio has provided two ways to create a new Mini Program:

- Creating a Mini Program scaffold project, it contains basic files of a Mini Program.
- Creating a Mini Program from the provided templates, which provides the template codes and help you to complete the Mini Program quickly.

Creating Scaffold Project

By clicking the `Add card` in the launch interface, you can create a scaffold project. In the project setup page, you need to set up following properties:

- Project Name: Project name to be displayed in the Mini Program Studio. Space is not allowed in the name. We recommend you to use letters to set the a meaning name.
- Project Path: Location for saving the project in the disk.

After the setup of the properties, click the complete button to complete the creating of the Mini Program.

Creating Template Project

In the launch interface, there is a `templates` button. Click it, you can see there are three templates and there will be more templates in the future.

Choose a template then click `Next`. Then similar with creating a new Mini Program, set the name and path for the project and click `Complete` button to complete the creation process.

Open an Existing Mini Program

If you have created Mini Program project before, or you have a local Mini Program project which may from a git source, you can open it with the `Open Project` function.

Open Local Project

After clicking the `Open Project` button, a local selector will pop up, you need to select the root path of your local Mini Program project and then confirm the selection.

Open Recent Opened Project

If you have opened a Mini Program project before, it will show in the launch interface. You can click the card to open the project directly.

Delete Project

If you do not want the launch interface to show too many opened projects, you can right click the project card and then a delete popup will show, then you can remove it from the launch interface to make the interface clean.

> Note: the delete function will not remove the local project files, only remove the entries in the launch interface.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/mini-program-studio_interface_launch-interface

---

Launching the User Interface {#launching-the-user-interface}

Last updated: 2021-05-10

Path: miniprogram_gcash

Launching the User Interface

2021-05-10 03:43

Prerequisites

Make sure you have created a developer account. For more information, see Apply for an Account.

Features

When you launch the Mini Program Studio, you can do the following tasks in the user interface.

- Login
- Joined workspaces
- Create a new Mini Program
- Open an existing Mini Program
- Delete a Mini Program

For more information, see the demos below:

Login

The first time opening the Mini Program Studio, login is required. You can use the account that applied from Mini Program Developer Portal to login the Mini Program.

If password is forgot, try to retrieve the password in the Mini Program Developer Portal.

Joined Workspaces

After login, you can see which workspaces you have joined in the left side of the launch interface.

A developer can join multiple workspaces by invitation of workspace admins, and Mini Programs are separated in different workspaces. So here you can develop different Mini Program project for different workspaces.

Create a New Mini Program

The first time opening the Mini Program, you can create a new Mini Program project. The Mini Program Studio has provided two ways to create a new Mini Program:

- Creating a Mini Program scaffold project, it contains basic files of a Mini Program.
- Creating a Mini Program from the provided templates, which provides the template codes and help you to complete the Mini Program quickly.

Creating Scaffold Project

By clicking the `Add card` in the launch interface, you can create a scaffold project. In the project setup page, you need to set up following properties:

- Project Name: Project name to be displayed in the Mini Program Studio. Space is not allowed in the name. We recommend you to use letters to set the a meaning name.
- Project Path: Location for saving the project in the disk.

After the setup of the properties, click the complete button to complete the creating of the Mini Program.

Creating Template Project

In the launch interface, there is a `templates` button. Click it, you can see there are three templates and there will be more templates in the future.

Choose a template then click `Next`. Then similar with creating a new Mini Program, set the name and path for the project and click `Complete` button to complete the creation process.

Open an Existing Mini Program

If you have created Mini Program project before, or you have a local Mini Program project which may from a git source, you can open it with the `Open Project` function.

Open Local Project

After clicking the `Open Project` button, a local selector will pop up, you need to select the root path of your local Mini Program project and then confirm the selection.

Open Recent Opened Project

If you have opened a Mini Program project before, it will show in the launch interface. You can click the card to open the project directly.

Delete Project

If you do not want the launch interface to show too many opened projects, you can right click the project card and then a delete popup will show, then you can remove it from the launch interface to make the interface clean.

> Note: the delete function will not remove the local project files, only remove the entries in the launch interface.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/mini-program-studio_interface_launch-interface

---

Learn More About Todo App Demo {#learn-more-about-todo-app-demo}

Last updated: 2022-07-03

Path: miniprogram_gcash

Learn More About Todo App Demo

2022-07-03 18:44

Overview

We will introduce the development of a Mini Program in detail through the Todo App Demo.

Global Configuration

The `app.js` is the entry point to a Mini Program, you can configure the lifecycle of a Mini Program, declare global variables, and perform app initialization in this file. The following code snippet shows an example of calling APIs for storage and getting user

information. For more APIs, see API document.

copy

```
// Call the storage API to get the stored data
const todos = my.getStorageSync({key:'todos'}).data || [\
    { text: 'Learning Javascript', completed: true },\
    { text: 'Learning ES2016', completed: true },\
    { text: 'Learning Mini Program ', completed: false },\
  ];

App({
  // Declare global data
  todos,

  userInfo: null,

  // Declare global method
  setTodos(todos) {
    this.todos = todos;
    // Call storage API to store data
    my.setStorageSync({key:'todos', data:todos});
  },

  getUserInfo() {
    return new Promise((resolve, reject) => {
      if (this.userInfo) resolve(this.userInfo);
      // Call user authorization API to get user info
      my.getAuthCode({
        success: (authcode) => {
          console.info(authcode);

          my.getAuthUserInfo({
            scopes: ['auth_user'],
            success: (res) => {
              this.userInfo = res;
              resolve(this.userInfo);
            },
            fail: () => {
              reject({});
            },
          });
        },
        fail: () => {
          reject({});
        },
      });
    });
  },
});
```

The `app.json` is the global configuration file of the Mini Program, where it is possible to configure the general navigation bar title, window background color and other configurations of the Mini Program. For more configurations, see global configuration documentation.

copy

```
{
  "pages": [\
    "pages/todos/todos",\
    "pages/add-todo/add-todo"\
  ],
  "window": {
    "defaultTitle": "Todo App"
  }
}
```

The `app.acss` is the global style of a Mini Program. The selectors defined in the app.acss can be applied to all pages in the Mini Program project.

copy

```
page {
  flex: 1;
  display: flex;
}
```

The page selector is a special selector supported by the framework, which works with the page root node container available in the framework.

Mini Program Page

We have two pages in this demo project: Todo List page and Add Todo page, both reside in the `pages` directory. All page paths of the Mini Program must be declared in the `app.json`, and a page path starts from the project root directory and should omit the filename extension. The very first path declared `app.json` is the home page of a Mini Program.

Each Mini Program page consists of four types of files under the same directory:

- JS logic script file with the `.js` extension
- Configuration file with the `.json` extension
- Style file with the `.acss` extension
- UI Layout file with the `.axml` extension.

Todo List Page

The todos.axml is the structure template file of the page:

copy

```
<view class="page-todos">
  <view class="user">
    <image class="avatar" src="{{user.avatar}}" background-
size="cover"></image>
    <view class="nickname">{{user.nickName}}'s Todo List</view>
  </view>
  <view class="todo-items">
    <checkbox-group class="todo-items-group" onChange="onTodoChanged">
      <label class="todo-item" a:for="{{todos}}">
        <checkbox value="{{item.text}}" checked="{{item.completed}}"
/>
        <text class="{{item.completed ? 'checked' : ''}}">
{{item.text}}</text>
      </label>
    </checkbox-group>
    <view class="todo-item">
      <button onTap="addTodo">Add Todo</button>
    </view>
  </view>
</view>
```

This UI layout have built-in UI components such as `<view/>`, `<image/>`, `<text/>`, `<button/>`, `<label/>` and `<checkbox/>` to build up the page structure, and we use `a:for` attribute to perform list rendering which iterates the todos list and render a `label` for each todo data respectively.

For binding data, see Data binding document. For binding event, see event handling document.

The `todos.js` is the logic script file of the page:

copy

```
// Get global app instance
const app = getApp();

Page({
  data: {},
  onLoad() {
    // Get user information and render
    app.getUserInfo().then(
      user => this.setData({
        user,
      }),
    );
  },
  // Listen to lifecycle
  onShow() {
    // Render global data to current page
    this.setData({ todos: app.todos });
  },
```

```
  // Event handler
  onTodoChanged(e) {
    // Modify global data and re-render
    const checkedTodos = e.detail.value;
    app.setTodos(app.todos.map(todo => ({
      ...todo,
      completed: checkedTodos.indexOf(todo.text) > -1,
    })));
    this.setData({ todos: app.todos });
  },
  addTodo() {
    // Call page jump API for page jump
    my.navigateTo({ url: '../add-todo/add-todo' });
  },
});
```

In this file, we have:

- Listen to and process the lifecycle function of the page (`onHide`, `onShow`, `onLoad`, `onUnload`, `onReady`).
- Get Mini Program app instance and other page instances (`getApp`, `getCurrentPages`).
- Declare and process data
- Respond to page interaction events, call APIs, etc.
- Attention here: the `app.todos` object is the global variable defined in `app.js`.

The `todos.acss` is the style file of the page:

copy

```
.page-todos {
  flex: 1;
  display: flex;
  flex-direction: column;
}

.user {
  display: flex;
  padding: 30px 30px 0 30px;
}

.avatar {
  width: 128rpx;
  height: 128rpx;
  margin-right: 40rpx;
  border-radius: 50%;
}

.nickname {
```

```
  display: flex;
  flex-direction: column;
  justify-content: center;
  font-size: 40rpx;
}

.todo-items {
  padding: 80rpx;
}

.todo-items-group {
  display: flex;
  flex-direction: column;
}

.checked {
  color: #d9d9d9;
  text-decoration: line-through;
}

.todo-item {
  margin-bottom: 15px;
}
```

The `acss` styling file is not mandatory. See Style document for `acss` file syntax. When a page has the style sheet, the style rule in the page style sheet overrides the style rules in the `app.acss`. if the style sheet is not specified for the page, it is also possible to directly use the style rules specified in `app.acss`.

The `todos.json` is the configuration file of the page. Here it is an empty file.

The configuration file is not mandatory. When a page has the configuration file, the configuration item overwrites the same configuration items in the window of app.json. If no page configuration file is specified, the page directly uses the default configuration in app.json. Therefore, the index page title is the Todo App specified for app.json.

Add Todo Page

The `add-todo.axml` is the structure template file of the page:

copy

```
<view class="page-add-todo">
  <view class="add-todo">
    <input
      class="add-todo-input"
      placeholder="What needs to be done?"
      onBlur="onBlur"
      value="{{inputValue}}"
    />
  </view>
  <view class="todo-footer">
```

```
    <add-button text="Add Todo" onClickMe="add" ></add-button>
  </view>
</view>
```

There are two core functions in the page:

1. Use the <input/> component to accept user input.
2. The <add-button> is a custom component. We can wrap the whole codes of some function into a custom component for easy reuse elsewhere.

add-todo.js page logic code:

copy

```
const app = getApp();

Page({
  data: {
    inputValue: '',
  },
  onBlur(e) {
    this.setData({
      inputValue: e.detail.value,
    });
  },
  add() {
    app.todos = app.todos.concat([\
      {\
        text: this.data.inputValue,\
        compeleted: false,\
      },\
    ]);
    my.navigateBack();
  },
});
```

add-todo.acss is consistent with todos.acss usage and will not be described again.

Since the add-todo.json refers to a custom component, it should be declared in json, otherwise error will be reported:

copy

```
"usingComponents": {
    "add-button": "/components/add-button/add-button"
}
```

We will learn how to publish a Mini Program in the next tutorial.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/quick-start_learn-more-about-todo-app-demo

---

Learn More About Todo App Demo {#learn-more-about-todo-app-demo}

Last updated: 2021-05-09

Path: miniprogram_gcash

Learn More About Todo App Demo

2021-05-09 18:43

Overview

We will introduce the development of a Mini Program in detail through the Todo App Demo.

Global Configuration

The `app.js` is the entry point to a Mini Program, you can configure the lifecycle of a Mini Program, declare global variables, and perform app initialization in this file. The following code snippet shows an example of calling APIs for storage and getting user information. For more APIs, see API document.

copy

```
// Call the storage API to get the stored data
const todos = my.getStorageSync({key:'todos'}).data || [\
    { text: 'Learning Javascript', completed: true },\
    { text: 'Learning ES2016', completed: true },\
    { text: 'Learning Mini Program ', completed: false },\
  ];

App({
  // Declare global data
  todos,

  userInfo: null,

  // Declare global method
  setTodos(todos) {
    this.todos = todos;
    // Call storage API to store data
    my.setStorageSync({key:'todos', data:todos});
  },

  getUserInfo() {
```

```
    return new Promise((resolve, reject) => {
      if (this.userInfo) resolve(this.userInfo);
      // Call user authorization API to get user info
      my.getAuthCode({
        success: (authcode) => {
          console.info(authcode);

          my.getAuthUserInfo({
            scopes: ['auth_user'],
            success: (res) => {
              this.userInfo = res;
              resolve(this.userInfo);
            },
            fail: () => {
              reject({});
            },
          });
        },
        fail: () => {
          reject({});
        },
      });
    });
  },
});
```

The `app.json` is the global configuration file of the Mini Program, where it is possible to configure the general navigation bar title, window background color and other configurations of the Mini Program. For more configurations, see global configuration documentation.

copy

```
{
  "pages": [\
    "pages/todos/todos",\
    "pages/add-todo/add-todo"\
  ],
  "window": {
    "defaultTitle": "Todo App"
  }
}
```

The `app.acss` is the global style of a Mini Program. The selectors defined in the app.acss can be applied to all pages in the Mini Program project.

copy

```
page {
  flex: 1;
  display: flex;
}
```

The page selector is a special selector supported by the framework, which works with the page root node container available in the framework.

Mini Program Page

We have two pages in this demo project: Todo List page and Add Todo page, both reside in the `pages` directory. All page paths of the Mini Program must be declared in the `app.json`, and a page path starts from the project root directory and should omit the filename extension. The very first path declared `app.json` is the home page of a Mini Program.

Each Mini Program page consists of four types of files under the same directory:

- JS logic script file with the `.js` extension
- Configuration file with the `.json` extension
- Style file with the `.acss` extension
- UI Layout file with the `.axml` extension.

Todo List Page

The todos.axml is the structure template file of the page:

copy

```
<view class="page-todos">
  <view class="user">
    <image class="avatar" src="{{user.avatar}}" background-
size="cover"></image>
    <view class="nickname">{{user.nickName}}'s Todo List</view>
  </view>
  <view class="todo-items">
    <checkbox-group class="todo-items-group" onChange="onTodoChanged">
      <label class="todo-item" a:for="{{todos}}">
        <checkbox value="{{item.text}}" checked="{{item.completed}}"
/>
        <text class="{{item.completed ? 'checked' : ''}}">
{{item.text}}</text>
      </label>
    </checkbox-group>
    <view class="todo-item">
      <button onTap="addTodo">Add Todo</button>
    </view>
  </view>
</view>
```

This UI layout have built-in UI components such as `<view/>`, `<image/>`, `<text/>`, `<button/>`, `<label/>` and `<checkbox/>` to build up the page structure, and we use `a:for` attribute to perform list rendering which iterates the todos list and render a `label` for each todo data respectively.

For binding data, see Data binding document. For binding event, see event handling document.

The `todos.js` is the logic script file of the page:

copy

```
// Get global app instance
const app = getApp();

Page({
  data: {},
  onLoad() {
    // Get user information and render
    app.getUserInfo().then(
      user => this.setData({
        user,
      }),
    );
  },
  // Listen to lifecycle
  onShow() {
    // Render global data to current page
    this.setData({ todos: app.todos });
  },
  // Event handler
  onTodoChanged(e) {
    // Modify global data and re-render
    const checkedTodos = e.detail.value;
    app.setTodos(app.todos.map(todo => ({
      ...todo,
      completed: checkedTodos.indexOf(todo.text) > -1,
    })));
    this.setData({ todos: app.todos });
  },
  addTodo() {
    // Call page jump API for page jump
    my.navigateTo({ url: '../add-todo/add-todo' });
  },
});
```

In this file, we have:

- Listen to and process the lifecycle function of the page (`onHide`, `onShow`, `onLoad`, `onUnload`, `onReady`).
- Get Mini Program app instance and other page instances (`getApp`, `getCurrentPages`).
- Declare and process data
- Respond to page interaction events, call APIs, etc.

- Attention here: the `app.todos` object is the global variable defined in `app.js`.

The `todos.acss` is the style file of the page:

copy

```
.page-todos {
  flex: 1;
  display: flex;
  flex-direction: column;
}

.user {
  display: flex;
  padding: 30px 30px 0 30px;
}

.avatar {
  width: 128rpx;
  height: 128rpx;
  margin-right: 40rpx;
  border-radius: 50%;
}

.nickname {
  display: flex;
  flex-direction: column;
  justify-content: center;
  font-size: 40rpx;
}

.todo-items {
  padding: 80rpx;
}

.todo-items-group {
  display: flex;
  flex-direction: column;
}

.checked {
  color: #d9d9d9;
  text-decoration: line-through;
}

.todo-item {
  margin-bottom: 15px;
}
```

The `acss` styling file is not mandatory. See Style document for `acss` file syntax. When a page has the style sheet, the style rule in the page style sheet overrides the style rules in the `app.acss`. if the style sheet is not specified for the page, it is also possible to directly

use the style rules specified in `app.acss`.

The `todos.json` is the configuration file of the page. Here it is an empty file.

The configuration file is not mandatory. When a page has the configuration file, the configuration item overwrites the same configuration items in the window of app.json. If no page configuration file is specified, the page directly uses the default configuration in app.json. Therefore, the index page title is the Todo App specified for app.json.

Add Todo Page

The `add-todo.axml` is the structure template file of the page:

copy

```
<view class="page-add-todo">
  <view class="add-todo">
    <input
      class="add-todo-input"
      placeholder="What needs to be done?"
      onBlur="onBlur"
      value="{{inputValue}}"
    />
  </view>
  <view class="todo-footer">
    <add-button text="Add Todo" onClickMe="add" ></add-button>
  </view>
</view>
```

There are two core functions in the page:

1. Use the `<input/>` component to accept user input.
2. The `<add-button>` is a custom component. We can wrap the whole codes of some function into a custom component for easy reuse elsewhere.

`add-todo.js` page logic code:

copy

```
const app = getApp();

Page({
  data: {
    inputValue: '',
  },
  onBlur(e) {
    this.setData({
      inputValue: e.detail.value,
    });
  },
  add() {
    app.todos = app.todos.concat([\
```

```
      {\
        text: this.data.inputValue,\
        compeleted: false,\
      },\
    ]);
    my.navigateBack();
  },
});
```

`add-todo.acss` is consistent with `todos.acss` usage and will not be described again.

Since the `add-todo.json` refers to a custom component, it should be declared in json, otherwise error will be reported:

copy

```
"usingComponents": {
    "add-button": "/components/add-button/add-button"
}
```

We will learn how to publish a Mini Program in the next tutorial.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/quick-start_learn-more-about-todo-app-demo

---

Lifecycle {#lifecycle}

Last updated: 2021-05-09

Path: miniprogram_gcash

Lifecycle

2021-05-09 18:43

Lifecycle Function

The lifecycle function of component is triggered by framework at special timing. Its detailed information is described in table below.

| | | | --- | --- | --- | | **Lifecycle** | **Parameter** | **Description** | | onInit | No | Trigger on component creation. | | deriveDataFromProps | nextProps | Trigger on component creation and before update. | | didMount | No | Trigger on component creation completion. | | didUpdate | (prevProps,prevData) | Trigger on component update completion. | | didUnmount | No | Trigger on component deletion. |

onInit

The onInit is triggered on component creation. In onInit, it is possible to:

- Access such attributes as this.is, this.$id and this.$page
- Access this.data and this.props
- Access the custom attribute in component “methods”
- Call this.setData and this.$spliceData to modify data

Example 1:

copy

```
// /components/counter/index.js
Component({
  data: {
    counter: 0,
  },
  onInit() {
    this.setData({
      counter: 1,
      is: this.is,
    });
  },
})
```

copy

```
<!-- /components/counter/index.axml -->
<view>{{counter}}</view>
<view>{{is}}</view>
```

When the component is rendered on the page, the page output is as below:

copy

```
1
/components/counter/index
```

Example 2:

copy

```
// /components/counter/index.js
Component({
  onInit() {
    this.xxx = 2;
    this.data = { counter: 0 };
  },
})
```

copy

```
<!-- /components/counter/index.axml -->
<view>{{counter}}</view>
```

When the component is rendered on the page, the page output is as below:

copy

```
0
```

deriveDataFromProps

The deriveDataFromProps is triggered on component creation and update. In the deriveDataFromProps, it is possible to:

- Access such attributes as this.is, this.$id and this.$page
- Access this.data and this.props
- Access the custom attribute in component “methods”
- Call this.setData and this.$spliceData to modify data
- Use the nextProps parameter to get the props parameter to be updated

Sample code:

Note

In this example, click the + button, and the counter on the page remains unchanged till the pCounter value is greater than 5.

copy

```
// /components/counter/index.js
Component({
  data: {
    counter: 5,
  },
  deriveDataFromProps(nextProps) {
    if (this.data.counter < nextProps.pCounter) {
      this.setData({
        counter: nextProps.pCounter,
      });
    }
  },
})
```

copy

```
<!-- /components/counter/index.axml -->
<view>{{counter}}</view>
```

copy

```
// /pages/index/index.js
Page({
  data: {
    counter: 1,
  },
  plus() {
    this.setData({ counter: this.data.counter + 1 })
  },
})
```

copy

```
<!-- /pages/index/index.axml -->
<counter pCounter="{{counter}}" />
<button onTap="plus">+</button>
```

didMount

The didMount is the callback after the initial renderof the custom component. Now the page has been rendered, and usually server end data is requested.

Sample code:

copy

```
Component({
  data: {},
  didMount() {
    let that = this;
    my.httpRequest({
      url: 'http://httpbin.org/post',
      success: function(res) {
        console.log(res);
        that.setData({name: 'Name Example'});
      }
    });
  },
});
```

didUpdate

The didUpdate is the callback after the update of custom component. It is called whenever the component data changes.

Sample code:

copy

```
Component({
  data: {},
```

```
  didUpdate(prevProps, prevData) {
    console.log(prevProps, this.props, prevData, this.data);
  },
});
```

Note:

- Internal call of **this.setData** in the component triggers didUpdate
- External call of **this.setData** triggers didUpdate, too

didUnmount

The didUnmount is the callback after the custom component deletion. It is called whenever the component instance is unloaded from the page.

Sample code:

copy

```
Component({
  data: {},
  didUnmount() {
    console.log(this);
  },
});
```

Component Lifecycle Illustration

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_custom-component_create-custom-component_lifecycle

---

Lifecycle {#lifecycle}

Last updated: 2022-07-03

Path: miniprogram_gcash

Lifecycle

2022-07-03 18:44

Lifecycle Function

The lifecycle function of component is triggered by framework at special timing. Its detailed information is described in table below.

| | | | --- | --- | --- | | **Lifecycle** | **Parameter** | **Description** | | onInit | No | Trigger on component creation. | | deriveDataFromProps | nextProps | Trigger on component creation and before update. | | didMount | No | Trigger on component creation completion. | | didUpdate | (prevProps,prevData) | Trigger on component update completion. | | didUnmount | No | Trigger on component deletion. |

onInit

The onInit is triggered on component creation. In onInit, it is possible to:

- Access such attributes as this.is, this.$id and this.$page
- Access this.data and this.props
- Access the custom attribute in component “methods”
- Call this.setData and this.$spliceData to modify data

Example 1:

copy

```
// /components/counter/index.js
Component({
  data: {
    counter: 0,
  },
  onInit() {
    this.setData({
      counter: 1,
      is: this.is,
    });
  },
})
```

copy

```
<!-- /components/counter/index.axml -->
<view>{{counter}}</view>
<view>{{is}}</view>
```

When the component is rendered on the page, the page output is as below:

copy

```
1
/components/counter/index
```

Example 2:

copy

```
// /components/counter/index.js
Component({
  onInit() {
    this.xxx = 2;
    this.data = { counter: 0 };
  },
})
```

copy

```
<!-- /components/counter/index.axml -->
<view>{{counter}}</view>
```

When the component is rendered on the page, the page output is as below:

copy

```
0
```

deriveDataFromProps

The deriveDataFromProps is triggered on component creation and update. In the deriveDataFromProps, it is possible to:

- Access such attributes as this.is, this.$id and this.$page
- Access this.data and this.props
- Access the custom attribute in component “methods”
- Call this.setData and this.$spliceData to modify data
- Use the nextProps parameter to get the props parameter to be updated

Sample code:

Note

In this example, click the + button, and the counter on the page remains unchanged till the pCounter value is greater than 5.

copy

```
// /components/counter/index.js
Component({
  data: {
    counter: 5,
  },
  deriveDataFromProps(nextProps) {
    if (this.data.counter < nextProps.pCounter) {
```

```
      this.setData({
        counter: nextProps.pCounter,
      });
    }
  },
})
```

copy

```
<!-- /components/counter/index.axml -->
<view>{{counter}}</view>
```

copy

```
// /pages/index/index.js
Page({
  data: {
    counter: 1,
  },
  plus() {
    this.setData({ counter: this.data.counter + 1 })
  },
})
```

copy

```
<!-- /pages/index/index.axml -->
<counter pCounter="{{counter}}" />
<button onTap="plus">+</button>
```

didMount

The didMount is the callback after the initial renderof the custom component. Now the page has been rendered, and usually server end data is requested.

Sample code:

copy

```
Component({
  data: {},
  didMount() {
    let that = this;
    my.httpRequest({
      url: 'http://httpbin.org/post',
      success: function(res) {
        console.log(res);
        that.setData({name: 'Name Example'});
      }
    });
  },
});
```

didUpdate

The didUpdate is the callback after the update of custom component. It is called whenever the component data changes.

Sample code:

copy

```
Component({
  data: {},
  didUpdate(prevProps, prevData) {
    console.log(prevProps, this.props, prevData, this.data);
  },
});
```

Note:

- Internal call of **this.setData** in the component triggers didUpdate
- External call of **this.setData** triggers didUpdate, too

didUnmount

The didUnmount is the callback after the custom component deletion. It is called whenever the component instance is unloaded from the page.

Sample code:

copy

```
Component({
  data: {},
  didUnmount() {
    console.log(this);
  },
});
```

Component Lifecycle Illustration

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_custom-component_create-custom-component_lifecycle

---

List {#list}

Last updated: 2022-07-03

Path: miniprogram_gcash

List

2022-07-03 18:44

List

List

| | | | | --- | --- | --- | | **Property** | **Description** | **Type** | | className | Custom class. | String |

Slots

| | | | --- | --- | | **slotName** | **Description** | | header | Optional, list header. | | footer | Optional, used to render list footer. |

List-item

| | | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | | className | Custom class. | String | | | thumb | Thumbnail, picture address. | String | | | arrow | With arrow or not. | Boolean | false | | align | Vertical alignment of sub-elements, choices: `top`, `middle`, `bottom` | String | middle | | index | Unique index of list item. | String | | | onClick | Call this function when clicking list-item. | ({index, target}) => void | | | last | If it is the last list item. | Boolean | false | | disabled | Not clickable, no hover effect. | Boolean | false | | multipleLine | Multiple lines. | Boolean | false | | wrap | Wrap or not. By default, excessive text length is hidden. | Boolean | false |

Slots

| | | | --- | --- | | **slotname** | **Description** | | extra | Optional, used to render right-hand notes of list item. | | prefix | Optional, used to render left-hand notes of list item. |

Example

copy

```
{
  "defaultTitle": "AntUI Component Library",
  "usingComponents": {
    "list": "mini-antui/es/list/index",
    "list-item": "mini-antui/es/list/list-item/index"
  }
}
```

copy

```
<view>
  <list>
```

```
    <view slot="header">
      List Header
    </view>
    <block a:for="{{items}}">
      <list-item
        thumb="{{item.thumb}}"
        arrow="{{item.arrow}}"
        align="{{item.align}}"
        index="{{index}}"
        onClick="onItemClick"
        key="items-{{index}}"
        last="{{index === (items.length - 1)}}"
      >
      {{item.title}}
        <view class="am-list-brief">{{item.brief}}</view>
        <view slot="extra">
          {{item.extra}}
        </view>
      </list-item>
    </block>
    <view slot="footer">
      List footer
    </view>
  </list>
  <list>
    <view slot="header">
      List Header
    </view>
    <block a:for="{{items2}}">
      <list-item
        thumb="{{item.thumb}}"
        arrow="{{item.arrow}}"
        onClick="onItemClick"
        index="items2-{{index}}"
        key="items2-{{index}}"
        last="{{index === (items2.length - 1)}}"
      >
       {{item.title}}
        <view class="am-list-brief">{{item.brief}}</view>
        <view a:if="{{item.extra}}" slot="extra">
          {{item.extra}}
        </view>
      </list-item>
    </block>
    <view slot="footer">
      List footer
    </view>
  </list>
</view>
```

copy

```
Page({
  data: {
    items: [\
      {\
        title: 'Simple List',\
        extra: 'Details',\
      },\
    ],
    items2: [\
      {\
        title: 'Complex List',\
        arrow: true,\
      },\
      {\
        title: 'Complex List',\
        arrow: 'up',\
      },\
      {\
        title: 'Complex List',\
        arrow: 'down',\
      },\
      {\
        title: 'Complex List',\
        arrow: 'empty',\
      },\
      {\
        title: 'Complex List',\
      },\
    ],
  },
  onItemClick(ev) {
    my.alert({
      content: `Click the ${ev.index} row`,
    });
  },
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_layout-navigation_list

---

List Rendering {#list-rendering}

Last updated: 2021-05-09

Path: miniprogram_gcash

List Rendering

2021-05-09 18:43

a:for

Using `a:for` attribute on component can bind an array, and then the data in the array can be used to render the component repeatedly.

The current item in the array has a default subscript variable name `index`. The current item of the array has a default variable name `item`.

copy

```
<view a:for="{{array}}">
  {{index}}: {{item.message}}
</view>
```

copy

```
Page({
  data: {
    array: [{\
      message: 'foo',\
    }, {\
      message: 'bar',\
    }],
  },
});
```

Use `a:for-item` to specify the variable name for the current element of the array. Use `a:for-index` to specify the current subscript variable name of the array.

copy

```
<view a:for="{{array}}" a:for-index="idx" a:for-item="itemName">
  {{idx}}: {{itemName.message}}
</view>
```

`A:for` supports nesting.

Below are the sample codes for the Multiplication Table nesting.

copy

```
<view a:for="{{[1, 2, 3, 4, 5, 6, 7, 8, 9]}}" a:for-item="i">
  <view a:for="{{[1, 2, 3, 4, 5, 6, 7, 8, 9]}}" a:for-item="j">
    <view a:if="{{i <= j}}">
      {{i}} * {{j}} = {{i * j}}
    </view>
  </view>
</view>
```

block a:for

Similar to block `a:if`, `a:for` can be used on the `<block/>` tag to render a structural block with multiple nodes.

copy

```
<block a:for="{{[1, 2, 3]}}">
  <view> {{index}}: </view>
  <view> {{item}} </view>
</block>
```

a:key

If the list item may change position dynamically or new item will be added into the list, and it is expected to maintain the features and state of the list item (such as the entry contents of `<input/>` and the checked status of `<switch/>`), the `a:key` should be used to specify the unique identifier of the list item.

The `a:key` value is provided in one of the two modes:

- String: Representing an attribute of the list item. The attribute value needs to be a unique string or number in the list, such as ID, and cannot change dynamically.
- Reserved keyword `*this`: Representing the list item itself. Moreover, it is the unique string or number. For example, when the change of the current data triggers re-rendering, the component with `key` will be rectified. The framework ensures they are reordered, but not recreated. In this way, the component can maintain its status, increasing the list rendering efficiency.

Note:

- If the `a:key` is not provided, it reports a warning.
- This can be ignored if it is known the list is static or the order is not concerned.

Below are the sample codes:

copy

```
<view class="container">
  <view a:for="{{list}}" a:key="*this">
    <view onTap="bringToFront" data-value="{{item}}">
    {{item}}: click to bring to front
    </view>
  </view>
</view>
```

copy

```
Page({
  data:{
    list:['1', '2', '3', '4'],
```

```
  },
  bringToFront(e) {
    const { value } = e.target.dataset;
    const list = this.data.list.concat();
    const index = list.indexOf(value);
    if (index !== -1) {
      list.splice(index, 1);
      list.unshift(value);
      this.setData({ list });
    }
  },
});
```

key

The key is a more popular writing style than a:key, where any expression and string can be filled.

Note: The key can not be set on block.

Below are the sample codes:

copy

```
<view class="container">
  <view a:for="{{list}}" key="{{item}}">
    <view onTap="bringToFront" data-value="{{item}}">
    {{item}}: click to bring to front
    </view>
  </view>
</view>
```

copy

```
Page({
  data:{
    list:['1', '2', '3', '4'],
  },
  bringToFront(e) {
    const { value } = e.target.dataset;
    const list = this.data.list.concat();
    const index = list.indexOf(value);
    if (index !== -1) {
      list.splice(index, 1);
      list.unshift(value);
      this.setData({ list });
    }
  },
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_axml-reference_list-rendering

---

List Rendering {#list-rendering}

Last updated: 2022-07-03

Path: miniprogram_gcash

List Rendering

2022-07-03 18:44

a:for

Using `a:for` attribute on component can bind an array, and then the data in the array can be used to render the component repeatedly.

The current item in the array has a default subscript variable name `index`. The current item of the array has a default variable name `item`.

copy

```
<view a:for="{{array}}">
  {{index}}: {{item.message}}
</view>
```

copy

```
Page({
  data: {
    array: [{\
      message: 'foo',\
    }, {\
      message: 'bar',\
    }],
  },
});
```

Use `a:for-item` to specify the variable name for the current element of the array. Use `a:for-index` to specify the current subscript variable name of the array.

copy

```
<view a:for="{{array}}" a:for-index="idx" a:for-item="itemName">
  {{idx}}: {{itemName.message}}
</view>
```

`A:for` supports nesting.

Below are the sample codes for the Multiplication Table nesting.

copy

```
<view a:for="{{[1, 2, 3, 4, 5, 6, 7, 8, 9]}}" a:for-item="i">
  <view a:for="{{[1, 2, 3, 4, 5, 6, 7, 8, 9]}}" a:for-item="j">
    <view a:if="{{i <= j}}">
      {{i}} * {{j}} = {{i * j}}
    </view>
  </view>
</view>
```

block a:for

Similar to block `a:if`, `a:for` can be used on the `<block/>` tag to render a structural block with multiple nodes.

copy

```
<block a:for="{{[1, 2, 3]}}">
  <view> {{index}}: </view>
  <view> {{item}} </view>
</block>
```

a:key

If the list item may change position dynamically or new item will be added into the list, and it is expected to maintain the features and state of the list item (such as the entry contents of `<input/>` and the checked status of `<switch/>`), the `a:key` should be used to specify the unique identifier of the list item.

The `a:key` value is provided in one of the two modes:

- String: Representing an attribute of the list item. The attribute value needs to be a unique string or number in the list, such as ID, and cannot change dynamically.
- Reserved keyword `*this`: Representing the list item itself. Moreover, it is the unique string or number. For example, when the change of the current data triggers re-rendering, the component with `key` will be rectified. The framework ensures they are reordered, but not recreated. In this way, the component can maintain its status, increasing the list rendering efficiency.

Note:

- If the `a:key` is not provided, it reports a warning.
- This can be ignored if it is known the list is static or the order is not concerned.

Below are the sample codes:

copy

```
<view class="container">
  <view a:for="{{list}}" a:key="*this">
    <view onTap="bringToFront" data-value="{{item}}">
    {{item}}: click to bring to front
    </view>
  </view>
</view>
```

copy

```
Page({
  data:{
    list:['1', '2', '3', '4'],
  },
  bringToFront(e) {
    const { value } = e.target.dataset;
    const list = this.data.list.concat();
    const index = list.indexOf(value);
    if (index !== -1) {
      list.splice(index, 1);
      list.unshift(value);
      this.setData({ list });
    }
  },
});
```

key

The `key` is a more popular writing style than `a:key`, where any expression and string can be filled.

Note: The `key` can not be set on block.

Below are the sample codes:

copy

```
<view class="container">
  <view a:for="{{list}}" key="{{item}}">
    <view onTap="bringToFront" data-value="{{item}}">
    {{item}}: click to bring to front
    </view>
  </view>
</view>
```

copy

```
Page({
  data:{
    list:['1', '2', '3', '4'],
```

```
  },
  bringToFront(e) {
    const { value } = e.target.dataset;
    const list = this.data.list.concat();
    const index = list.indexOf(value);
    if (index !== -1) {
      list.splice(index, 1);
      list.unshift(value);
      this.setData({ list });
    }
  },
});
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_axml-reference_list-rendering

---

List-secondary {#list-secondary}

Last updated: 2022-07-07

Path: miniprogram_gcash

List-secondary

2022-07-07 17:08

You can use the list-secondary component to display the additional information on the right side of the list item. The list-secondary component is placed in the **extra** slot. See list-item for details.

Sample code

See the sample codes in different languages:

.json

copy

```
{
 "defaultTitle": "List",
 "usingComponents":{
   "list": "mini-ali-ui/es/list/index",
   "list-item": "mini-ali-ui/es/list/list-item/index",
   "list-secondary": "mini-ali-ui/es/list/list-secondary/index"
```

```
 }
}
```

.axml

copy

```
<list>
 <view slot="header">
   list header
 </view>
 <list-item thumb="http://thumb.link.png"
   arrow="{{true}}"
   onClick="onItemClick"
   upperSubtitle="upper subtitle"
   lowerSubtitle="lower subtitle" >
   main title
   <list-secondary
     title="secondary title"
     subtitle="secondary subtitle"
     thumb="http://thumb.url.jpg"
     thumbSize="20"
     slot="extra" />
 </list-item>
 <view slot="footer">
   list footer
 </view>
</list>
```

.js

copy

```
Page({
 onItemClick() {
   my.alert({
     content: 'click the event on list item'
   })
 }
})
```

Parameters

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | thumb | String | URL of thumbnail image. | | title | String | Title. | | subtitle | String | Subtitle. | | thumbSize | String | Size of thumbnail image, which is required when *thumb* is specified. It is recommended to manually set the size. Otherwise, the image height is adjusted automatically, but might not be consistent with the text height. |

slots

Six slots are available for one list item. The following figure illustrates the name and position of each slot:

| | | --- | --- | | **Slot Name** | **Description** | | supporting | Header slot on the left side of the list item. | | default | Default slot, which is used to display the title. | | afterTitle | Slot on the right of the title, which is used to display lable or icon. | | afterUpperSubtitle | Slot on the right of the upper subtitle, which is used to display lable or icon. | | afterLowerSubtitle | Slot on the right of the lower subtitle, which is used to display lable or icon. | | extra | Slot on the right side of the list item, which is used to display additional information. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_layout_secondlist

---

List-term {#list-term}

Last updated: 2022-07-03

Path: miniprogram_gcash

List-term

2022-07-03 18:44

You can use the list-item component to customize items in a list.

Sample code

See the sample codes in different languages:

.json

copy

```
{
  "defaultTitle": "List",
  "usingComponents":{
    "list": "mini-ali-ui/es/list/index",
    "list-item": "mini-ali-ui/es/list/list-item/index"
  }}
```

.axml

copy

```
<list>
 <view slot="header">
   list header
 </view>
 <list-item thumb="http://thumb.link.png"
   arrow="{{true}}"
   onClick="onItemClick"
   upperSubtitle="upper subtitle"
   lowerSubtitle="lower subtitle" >
   main title
   <view slot="extra">
     additional information
   </view>
 </list-item>
 <view slot="footer">
   list footer
 </view></list>
```

.js

copy

```
Page({
 onItemClick() {
   my.alert({
     content: 'click the event on list item'
   })
 }})
```

Parameters

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | arrow | Boolean | An indicator of whether to use an arrow. The default value is `true`. | | thumb | String | URL of the thumbnail image. | | index | String | The index that is used to record the position, which is returned in the event callback. | | borderRadius | Boolean | An indicator of whether the list item is rounded. The default value is `false`. | | upperSubtitle | String | Upper subtitle. | | lowerSubtitle | String | Lower subtitle. | | titlePosition | String | Title position. Valid values are:
- `top`
- `middle`
- `bottom`
The default value is `top`. | | thumbSize | String | Size of thumbnail image, which is required if *thumb* is specified. The default value is `40 px`. | | onClick | Function | The event that is triggered when users tap the list item. | | last | Boolean | An indicator of whether to display a line under the list item. The default value is `false`. |

slot

Six slots are available for one list item. The following figure illustrates the name and position of each slot:

| | | --- | --- | | **Slot Name** | **Description** | | supporting | Header slot on the left side of the list item. | | default | Default slot, which is used to display the title. | | afterTitle | Slot on the right of the title, which is used to display lable or icon. | | afterUpperSubtitle | Slot on the right of the upper subtitle, which is used to display lable or icon. | | afterLowerSubtitle | Slot on the right of the lower subtitle, which is used to display lable or icon. | | extra | Slot on the right side of the list item, which is used to display additional information. |

FAQ

How do I remove the line under the last list item?

If you want to remove the line under the last list item, set the value of *last* as `true`.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_layout_list-term

---

Long Password {#long-password}

Last updated: 2022-07-03

Path: miniprogram_gcash

Long Password

2022-07-03 18:44

You can use the long-password component to display the input box for the password.

Note:

The long-password is a controlled component. The component value needs to be obtained by the *onInput* event.

Sample code

See the sample codes in different languages:

.json

copy

```
{
  "defaultTitle": "verify-code",
  "usingComponents":{
    "long-password": "mini-ali-ui/es/long-password/index"
```

```
  }
}
```

.axml

copy

```
<view>
  <view style="margin-top: 10px;" />
  <view style="padding: 0 10px;">Long password box</view>
  <view style="margin-top: 10px;" />
  <long-password
    placeholder=""
    value="{{longPassword}}"
    clear="{{true}}"
    onInput="onInput"
    onClear="onClear" />
</view>
```

.js

copy

```
Page({
  data: {
    longPassword: '',
  },
  onInput(e) {
    this.setData({
      longPassword: e.detail.value,
    });
  },
  onClear() {
    this.setData({
      longPassword: '',
    });
  },
});
```

Parameters

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | className | String | Customized class. | | inputCls | String | Customized class for the input box that uses the input component. | | last | Boolean | An indicator of whether the input box is the last one. The default value is `false`. | | value | String | Initial content. | | name | String | Component name, which is used to obtain data by submitting the form. | | placeholder | String | Placehoder. | | placeholderStyle | String | Style of the placeholder. | | placeholderClass | String | Style class of the placeholder. | | disabled | Boolean | An indicator of whether to disable the function of clearing the password. The default value is `false`. | | maxlength | Number | Maximum length of the password. The default value is `140`. | | focus | Boolean |

An indicator of whether to get focus. The default value is `false`. | | clear | Boolean | An indicator of whether to clear the input. The default value is `true`, and takes effect only when the value of *disabled* is `false`. | | onInput | (e: Object) => void | The event that is triggered when users tap the keyboard. | | onConfirm | (e: Object) => void | The event that is triggered when users tap the **Done** button on the keyboard. | | onFocus | (e: Object) => void | The event that is triggered when an element gets the focus. | | onBlur | (e: Object) => void | The event that is triggered when an element loses the focus. | | onClear | (e: Object) => void | The event that is triggered when users tap the **Clear** button. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_form_password

---

Manage Mini Programs {#manage-mini-programs}

Last updated: 2022-07-07

Path: miniprogram_gcash

Manage Mini Programs

2022-07-07 17:08

The Mini Program Manage functionality is available for admins and developers to manage mini programs. Depending on your roles, you can perform different actions to manage your mini programs. Check this video to quickly get an overview of how to manage mini programs.

Admins, Reviewers, and Developers can manage mini programs with the highest authorization, while **Developers** have the authorization for their own mini programs. For more information about the roles, see Member Role.

Features

Depending on your roles, you can benefit from all or some of the following features. Admins can manage all the mini programs with all the features, while reviewers, developer admins, and developers can view their mini programs by either adding members (as developers or admins) or done in the mini program development.

To be specific, you can benefit from the following features:

Create a new mini program

You can choose the mini program type and create a new mini program after you provide the required information. This feature is available for both admins and developers.

See Create Mini Programs for detailed steps.

Note:

See Mini Program Types to check the difference between the two types of mini programs.

To create a HTML5 mini program, see How to transform an HTML 5 mobile app to an HTML 5 mini program.

Check mini programs in the list

You can check the mini programs in a list.

- Admins can check all the mini programs in the space.
- Other roles can check the authorized mini programs.

To be specific, you can:

- Check the basic information, such as name, type, ID, creator and last modified time of a mini program.
- View the performance and quality of each mini program and navigate to these pages:
- Performance
- Real-time analysis
- Quality

Navigate to the detail page of the mini program

You can search for the mini program by mini program name, mini program ID, or creator. You click one specific mini program in the list and navigate to the detail page to check the following features:

- `Information`
- `Versions`
- `Members`
- `Features`
- `QR Code`
- `Configuration`

Manage a selected mini program in the detail page

To be specific, you can:

- Information
- admins can modify or update the mini program information that is provided when you create the mini program. In addition, you can use multi-languages to describe your mini program information to localize the mini program.
- Other roles can view the information.
- Versions

You can check the version list of the mini program and filter mini programs by status. By selecting one specific version, you can check the detailed version information, JSAPIs that are used in the mini program, and the release progress. admins can also add release notes under `Version Info` > `Release Note` and submit an approval request to publish the version.

- Members
- admins can add members to join the mini program project and remove members from the project to develop, debug, release and manage mini programs.
- Set the member to either of the following roles:
- Developer Admin
- Developer
- Other roles can view the member information.
- Features

The feature is a package that contains JSAPIs that are used in the mini program. You can view the feature list. By selecting a specific feature, you can view all the JSAPIs contained in the feature.

In addition, developer admins can also add or delete features.

- QR Code

This feature is available for admins. You can take the following actions to the QR code:

- Display a default QR code for testing, which will disappear after a period of time as shown below the QR code.
- Define a new QR code with the URL and Page Parameters.
- Delete the default QR codes.
- Check the details of a QR code.
- Download a QR code.
- Configuration
- admins can configure the client ID, the merchant ID, and whitelists.
- Admins can view the configured information.

Remove the mini program

You can send your removal request of the mini program. After the removal request is approved, you can remove the mini program.

Note:

This feature is only available for admins.

More information

Overview

How-to videos

How to manage Mini Programs

How to transform an HTML 5 mobile app to an HTML 5 mini program

How to customize your analysis

Settings

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/mini-program

---

Manage apps {#manage-apps}

Last updated: 2022-07-07

Path: miniprogram_gcash

Manage apps

2022-07-07 17:08

With the App Manage functionality, you can view app information and manage apps. This functionality is available for workspace admins.

Features

You can benefit from the following features:

- Create a new app

You can create a new app after you provide the required information.

- Check app information

You can check the information of one specific app, which includes:

- `Mini Program Accessing Key (RSA-2048)`: A unique key that allows container SDKs to access mini program data through SaaS.
- `Offline Package RSA Public Key`: The public key that is used by the app to verify the signature after downloading the offline package.

- `Android App Package Name`: The Android app package name. This field is required for Android app.
- `Android App Minimum Support OS Version`: The minimum OS version that the Android app supports. This field is required for Android app.
- `iOS App Bundle ID`: The iOS app bundle ID. This field is required for iOS app.
- `iOS App Minimum Support OS Version`: The minimum OS version that the iOS app supports. This field is required for iOS app.
- Manage the app

You can modify the app information and delete the app.

More information

Overview

Member Role

Workflow Procedures

Manage Mini Program

Manage Workspace

Authorization

Approvals

Settings

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/apps

---

Manage workspace {#manage-workspace}

Last updated: 2022-07-07

Path: miniprogram_gcash

Manage workspace

2022-07-07 17:08

The Workspace Manage functionality is available for workspace admins and developer admins. Depending on your roles, you can perform different actions to manage the workspace.

Features

The following features are available for workspace admins under `Workspace`:

- Check workspace admins

You can check all other workspace admins.

- Manage the tenant workspace

You can check the following information about the tenant workspace under `Tenant Workspace` and change the information by sending us an email.

- `Workspace ID`: The unique ID that is assigned by the Mini Program Development Platform to identify a workspace.
- `Workspace Name`: Name of the tenant workspace.
- `Status`: Workspace status that identifies whether the workspace is verified by AIMPDP.
- `Business Address`: The company address that is used for the registration.
- `Business representative`: Name of legal representative.
- `Representative email`: Email address of legal representative.

Figure 1. Workspace admins manage the tenant workspace

- Manage the developer workspace

Workspace admins can check the list of developer workspaces under `Developer Workspaces`, which includes the following information:

- `Workspace Name`: Name of the developer workspace.
- `Company Name`: Name of the merchant company.
- `Number of Mini Programs`: Number of mini programs in the developer workspace.
- `Action`: The action that is taken to the developer workspace.
- Workspace admins can remove the developer workspace only when there's neither published mini program, nor any upload of mini programs into the workspace in the past 30 days.
- Workspace admins can view details of the developer workspace.

Figure 2. Workspace admins manage developer workspaces

The following features are available for developer admins under `Developer Workspace`:

- Check the workspace admins
- Check the workspace details, such as creation time, workspace ID, workspace name and status.
- Check the details of your business information, such as business address, scope of business, and contact email address.

Figure 3. Developer admins check the developer workspace info

More information

Overview

Member Role

Workflow Procedures

Manage Mini Program

Settings

Authorization

Approvals

Manage Apps

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/workspace

---

MapContext Overview {#mapcontext-overview}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext Overview

2022-07-03 18:44

Call my.createMapContext to obtain a MapContext instance. A MapConext instance is bound with a map component through its ID. Manipulate the corresponding map component through MapContext.

Methods

| | | --- | --- | | **Option** | **Description** | | MapContext.clearRoute | Clear the walking route on the map. | | MapContext.gestureEnable | Enable or disable all gestures. | | MapContext.getCenterLocation | Get the center location of the current map. | | MapContext.moveToLocation | Display the pinned location at the center of the map and restore the default zoom level. Use this method together with the *show-location* method of the map component. | | MapContext.showRoute | Show the default walking route planned. Only one route is displayed. | | MapContext.showsCompass | Set the visibility of the compass. | | MapContext.updateComponents | Update the map API incrementally. |

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_mapcontext_overview

---

MapContext.clearRoute {#mapcontextclearroute}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext.clearRoute

2022-07-03 18:44

Clear the walking route on the map.

Sample code

copy

```
this.mapCtx = my.createMapContext('map');
this.mapCtx.clearRoute();
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_clearroute

---

MapContext.gestureEnable {#mapcontextgestureenable}

Last updated: 2022-07-04

Path: miniprogram_gcash

MapContext.gestureEnable

2022-07-04 03:44

Enable or disable all gestures.

- 1: enable all gestures.
- 0: disable all gestures.

Parameter~~~~

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | isGestureEnable | Int | Yes | An indicator of whether the gestures are enabled. |

Sample code

copy

```
this.mapCtx = my.createMapContext('map');
this.mapCtx.gestureEnable({isGestureEnable:1});
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_gestureenable

---

MapContext.getCenterLocation {#mapcontextgetcenterlocation}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext.getCenterLocation

2022-07-03 18:44

Get the center location of the current map.

Parameters~~~~

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | success | Function | No | The callback method that indicates a successful call. | | fail | Function | No | The callback method that indicates a failed call. | | complete | Function | No | The callback method that indicates the call is completed (this will be executed regardless of whether the call succeeds or fails). |

Success callback function

The parameters are in object type and have the following properties:

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | longitude | Number | Longitude. | | latitude | Number | Latitude. |

Sample code

copy

```
// .js
this.mapCtx = my.createMapContext('map');
this.mapCtx.getCenterLocation({
success: res => {
my.alert({
    content: 'longitude:' + res.longitude + '\nlatitude:' +
res.latitude,
});
console.log(res.longitude);
console.log(res.latitude);
}
});
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_getcenterlocation

---

MapContext.moveToLocation {#mapcontextmovetolocation}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext.moveToLocation

2022-07-03 18:44

Display the pinned location at the center of the map and restore the default zoom level. Use this API together with the `show-location` method of the map component.

Sample code

copy

```
this.mapCtx = my.createMapContext('map');
this.mapCtx.moveToLocation();
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_movetolocation

---

MapContext.showRoute {#mapcontextshowroute}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext.showRoute

2022-07-03 18:44

Get the default walking route planned. Only one route is displayed.

Note:

The IDE simulator cannot obtain the return values. Use the real development environment to obtain the return values.

Parameters ~~~~

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | searchType | String | No | Valid values are:
- walk: walking
- bus: public transit
- drive: driving
- ride: bicycling
The default value is **walk**. | | startLat | Number | Yes | The latitude of the start point. | | startLng | Number | Yes | The longitude of the start point. | | endLat | Number | Yes | The latitude of the end point. | | endLng | Number | Yes | The longitude of the end point. | | throughPoints | Array | No | A set of points on the route. It is only available for driving guidance, that is, available when the value of `searchType` is **drive**. | | routeColor | HexColor | No | The color of the route. This parameter takes effect only in 2D maps. | | iconPath | String | No | The texture of the route. In base library v1.11.0 and earlier versions, this parameter takes precedence over `routeColor` in 3D maps. Routes are covered by texture instead of a solid color. It's suggested that this parameter is not specified in base library v1.13.0 and later versions, because a default texture pattern is provided in 3D maps. It's also suggested to set the size of the image to an integer power of 2, such as 64*64. | | iconWidth | Int | No | The width of the texture. This parameter takes effect only in base library v1.11.0 and earlier versions. It's suggested that this parameter is not specified in base library v1.13.0 and later versions, because a default texture width is set in 3D maps. | | routeWidth | Int | No | The width of the route. This parameter takes effect only when texture is not used. It's suggested that this parameter is not specified in base library v1.13.0 and later versions, because a default value is set in 2D maps and it is no longer required in 3D maps.ww | | zIndex | Int | No | The Z-axis index of the overlay. | | mode | Int | No | Only supported for driving and public transit modes. See Mode values for details. | | city | String | Yes | Required in the public transit mode. | | destinationCity | String | Yes | Required in the cross-city public transit mode. |

Mode values

| | | | --- | --- | --- | | **Mode** | **Bus** | **Drive** | | 0 | Fastest route | Speed first (time). | | 1 | Most economical route | Least cost (Choose the fastest toll-free route.) | | 2 | Minimum transfers | Shortest distance. | | 3 | Shortest walking distance | Avoid expressways. | | 4 | Coziest route | Real-time route planning (to avoid traffic jams). | | 5 | Avoid subway | Multiple strategies (comprehensively considering the speed first, least cost, and shortest distance strategies). | | 6 | - | Avoid highways. | | 7 | - | Avoid highway and toll roads. | | 8 | - | Avoid toll roads and traffic jams. | | 9 | - | Avoid highways, toll roads, and traffic jams. |

Success callback function

The type of the incoming parameter is Object with the following attributes:

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | success | Boolean | Indicates whether the API operation is successful. | | distance | Number | Distance. | | duration | Number | Time in seconds. |

Sample code

copy

```
//.js
onReady() {
// Call my.createMapContext to obtain the map context.
this.mapCtx.showRoute({
    searchType: "walk",         // searchType: "walk", "bus", "drive",
"ride". Default value: walk. Added in v10.1.50.
    startLat: 1.339712,        // The latitude of the start point.
    startLng: 103.855457,       // The longitude of the start point.
    endLat: 1.342983,         //  The latitude of the end point.
    endLng: 103.867935,        // The longitude of the end point.
    throughPoints: [{ lat: 1.343573, lng: 103.861916 }],// A set of
points on the route. It is only available for driving planning, that
is, available when searchType="drive". Added in v10.1.50.
    routeColor: '#FFB90F',       // The color of the route. This
parameter takes effect only in 2D maps in versions later than 10.1.50.
    iconWidth: 10,           // The width of the texture. In v10.1.35,
this parameter takes effect only when iconPath is specified. We
recommend that you do not specify this parameter in v10.1.50. A
default texture width is set in 3D maps.
    routeWidth: 10,          // The width of the route. This parameter
takes effect when texture is not used. We recommend that you do not
specify this parameter in v10.1.50, because a default value is set in
2D maps and it is no longer required in 3D maps.
    zIndex: 4,             // The z-axis index of the overlay 10.1.35
    mode: 0,             // Only supported in driving and public
transit modes. For more information about its values, see the mode
values list below.
```

```
    city: 'Singapore',         // Required in public transit mode.
    destinationCity: 'Singapore',    // Required in cross-city public
transit mode.
    success: method(res) {
     console.log(res, 2323)
    }
});
console.log(1121)
},
onLoad() {
//  this.mapCtx = my.createMapContext('map');
this.mapCtx = my.createMapContext('map');
this.setData({
    includePoints:[ {\
     latitude: 1.347016,\
    longitude: 103.860167,\
    },{\
     latitude: 1.351628,\
    longitude: 103.863718,\
    }]
})
}
```

copy

```
//.axml
<map
     id="map"
     customMapStyle="light"
     longitude="{{longitude}}"
     latitude="{{latitude}}"
     scale="{{scale}}"
     controls="{{controls}}"
     onControlTap="controltap"
     markers="{{markers}}"
     onMarkerTap="markertap"
     polyline="{{polyline}}"
     polygon="{{polygon}}"
     circles="{{circles}}"
     onRegionChange="regionchange"
     onTap="tap"
     onCalloutTap="callouttap"
     show-location style="width: 100%; height: 200px;"
     include-points="{{includePoints}}"
     ground-overlays="{{ground-overlays}}">
```

Sample of the success callback function

copy

```
{
    distance:328,
    duration:262,
    success:true
}
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_showroute

---

MapContext.showsCompass {#mapcontextshowscompass}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext.showsCompass

2022-07-03 18:44

Set the visibility of the compass.

- 1: visible.
- 0: invisible.

Parameter

The parameter is in object type and has the following property:

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | isShowsCompass | Int | Yes | An indicator of whether the compass is visible. |

Sample code

copy

```
this.mapCtx = my.createMapContext('map');
this.mapCtx.showsCompass({isShowsCompass:1});
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_showcompass

---

MapContext.updateComponents {#mapcontextupdatecomponents}

Last updated: 2022-07-03

Path: miniprogram_gcash

MapContext.updateComponents

2022-07-03 18:44

Update the map API incrementally.

Parameters

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | latitude | Number | The latitude of the center location. | | longitude | Number | The longitude of the center location. | | scale | Number | Zoom level. The value ranges from 5 to 18. The default value is 16. | | markers | Array | Point markers that overlay the map. | | polyline | Array | A set of consecutive points (a route) that overlays the map. | | include-points | Array | Slightly zoom out the map to include these points. | | include-padding | Object | Show the map inside the area that includes paddings. | | settings | Object | The settings. | | command | Object | The command to update the marker animation. |

Sample code

copy

```
this.mapCtx = my.createMapContext('map');
this.mapCtx.updateComponents({
scale: 14,
longitude: 103.863718,
latitude: 1.351628,
command:{
     // Marker animation
     markerAnim:[\
       {\
         type:0      // Jumping animation\
         markerId:xxx,\
       }\
     ],
},
setting:{
     // Gesture
     gestureEnable:0/1,
     // Scale
```

```
      showScale:0/1,
      // Compass
      showCompass:0/1,
      // Tilt gestures with both hands
      tiltGesturesEnabled:0/1,
      // Show or hide traffic
      trafficEnabled:0/1,
      // Points of interest on the map
      showMapText:0/1,
      // Location of Amap logo
      logoPosition:{centerX:150, centerY:90},
},
markers:[{},{}],
polyline:[{},{}],
include-points:[{},{}],
include-padding:{left:0, right:0, top:0, bottom:0},
});
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_map_updatecomponents

---

Mask {#mask}

Last updated: 2022-07-03

Path: miniprogram_gcash

Mask

2022-07-03 18:44

You can use the mask component to display the pop-up element with a mask.

Sample code

See the sample codes in different languages:

.json

copy

```
{
  "defaultTitle": "Mask",
  "usingComponents": {
    "mask": "mini-ali-ui/es/mask/index"
```

```
  }
}
```

.axml

copy

```
<mask type="{{type}}" show="{{show}}" maskZindex="{{maskZindex}}"
onMaskTap="maskClick"></mask>
```

.js

copy

```
Page({
 data: {
   type: 'market',
   maskZindex: 10,
 },
 maskClick() {
   if (this.data.type === 'market') {
     this.setData({
       type: 'product',
     });
   } else {
     this.setData({
       type: 'market',
       show: false,
     });
   }
 },
});
```

Parameters

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | type | String | The mask with different opacity. Valid values are:
- `product`: the pop-up element to display information about the product, with the opacity value of 0.55.
- `market`: the pop-up element to display information about the marketing, with the opacity value of 0.75.
The default value is `product`. | | maskZindex | Number | The z-index property of the customized mask. | | show | Boolean | An indicator of whether to display the mask. | | onMaskTap | EventHandle | The event that is triggered when users tap the mask. The default value is `() => { }`. | | fixMaskFull | Boolean | An indicator of whether to solve the incomplete display of the mask affected by transformation. The default value is `false`. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_prompt-guide_mask

Member role authorization in developer workspace {#member-role-authorization-in-developer-workspace}

Last updated: 2022-07-07

Path: miniprogram_gcash

Member role authorization in developer workspace

2022-07-07 17:08

This topic provides you with access permissions for each member role in the developer workspace. You can have an overview of how members collaborate to manage the whole life cycle of mini programs.

Developer Admin, Developer, and Operator

You can view the authorization of the developer admin, developer, and operator in the following table:

| | | | | | --- | --- | --- | --- | --- | | **Menu In the Mini Program Console** | **Operations** | **Developer Admin** | **Developer** | **Operator** | | Mini Program | View mini programs. | ✓ | Mini programs of the merchant | ✓ | Mini programs assigned to the role | ✓ | Mini programs assigned to the role | | Create mini programs. | ✓ | × | × | | View mini program list. | ✓ | ✓ | Mini programs assigned to the role | × | | Edit mini programs' information. | ✓ | ✓ | Mini programs assigned to the role | × | | Delete mini programs. | ✓ | ✓ | Mini programs assigned to the role | × | | Manage members. | ✓ | ✓ | Mini programs assigned to the role | × | | Upload, preview, and debug IDE features. | ✓ | ✓ | Mini programs assigned to the role | × | | Edit Client ID. | ✓ | ✓ | Mini programs assigned to the role | × | | Edit Merchant ID. | ✓ | ✓ | Mini programs assigned to the role | × | | Server Domain Whitelist | ✓ | ✓ | Mini programs assigned to the role | × | | Managewhitelist. | ✓ | ✓ | Mini programs assigned to the role | × | | Release mini programs. | ✓ | ✓ | Mini programs assigned to the role | × | | Remove mini programs. | ✓ | ✓ | Mini programs assigned to the role | × | | Manage mini program versions. | ✓ | ✓ | Mini programs assigned to the role | × | | Withdraw mini program grayscale release. | ✓ | ✓ | Mini programs assigned to the role | × | | Features | View feature list. | ✓ | × | × | | Add features. | ✓ | × | × | | View features' details. | ✓ | × | × | | View added features. | ✓ | × | × | | Edit features. | ✓ | × | × | | JSAPIs | View JSAPIs' list. | ✓ | × | × | | Add JSAPIs. | ✓ | × | × | | View JSAPIs' details. | ✓ | × | × | | Edit JSAPIs. | ✓ | × | × | | Analytics | View performance | ✓

Mini programs of the merchant | ✓
Mini programs assigned to the role | × | | View user behaviors. | ✓
Mini programs of the merchant | ✓
Mini programs assigned to the role | ✓
Mini programs of the merchant | | View transaction record. | ✓
Mini programs of the merchant | × | ✓
Mini programs of the merchant | | View daily metrics. | ✓
Mini programs of the merchant | × | ✓
Mini programs of the merchant | | Quality | Search mini programs. | ✓
Mini programs of the merchant | ✓
Mini programs assigned to the role | × | | Crash and abnormal tracking | ✓
Mini programs of the merchant | ✓
Mini programs assigned to the role | × | | Workspace | Manage workspace information. | × | × | × | | View details of developer workspaces. | × | × | × | | Delete developer workspaces. | × | × | × | | APP | View Apps list. | × | × | × | | View Apps details. | × | × | × | | Edit Apps' Information. | × | × | × | | Add Apps and maintain keys' information. | × | × | × | | Members | Invite members and withdraw member invitations. | ✓ | × | × | | Set members as:
- Developer Admin
- Developer
- Operator | ✓ | × | × | | Block members. | ✓ | × | × | | Delete members. | ✓ | × | × | | Approvals | Release mini programs. | ✓
All work orders of merchants | × | × | | Remove mini programs. | ✓
All work orders of merchants | × | × | | Add features. | ✓
All work orders of merchants | × | × | | Add related features. | ✓
All work orders of merchants | × | × | | Release mini programs in target apps. | ✓
All work orders of merchants | × | × | | Task running of notification delivery | × | × | × | | Task running of advertising | × | × | × | | My approvals | ✓ | ✓ | ✓ | | Feedback | View feedback list. | × | × | × | | View details of feedback. | × | × | × | | Response to feedback. | × | × | × | | Provide solutions for feedback. | × | × | × | | Audience | Create and manage audiences. | ✓ | × | ✓ | | Notification delivery | Create and manage campaigns. | ✓ | × | ✓ | | Create and manage tasks. | ✓ | × | ✓ | | Configure fatigue settings. | × | × | × | | View templates. | ✓ | ✓ | ✓ | | Manage templates | × | × | × | | View notification analytics. | ✓ | × | ✓ | | Advertising | Create and manage tasks. | ✓ | × | ✓ | | View ad analytics. | ✓ | × | ✓ |

More information

Members

Add members

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/developer-member-authorization

---

Member role authorization in tenant workspace {#member-role-authorization-in-tenant-workspace}

Last updated: 2022-07-07

Path: miniprogram_gcash

Member role authorization in tenant workspace

2022-07-07 17:08

This topic provides you with access permissions for each member role in the tenant workspace. You can have an overview of how members collaborate to manage the whole life cycle of mini programs.

Workspace admin and reviewer

You can view the authorization of the workspace admin and workspace reviewer in the following tables:

| | | | | --- | --- | --- | --- | | **Menu In the Mini Program Console** | **Operations** | **Workspace Admin** | **Workspace Reviewer** | | Mini Program | View mini programs. | ✓ Mini programs of the native app and its merchants | ✓
Mini programs assigned to the role | | Create mini programs. | ✓ | × | | View mini program list. | ✓ | × | | Edit mini programs' information. | ✓ | × | | Delete mini programs. | ✓ | × | | Manage members. | ✓ | × | | Upload, preview, and debug IDE features. | ✓ | × | | Edit Client ID. | ✓ | × | | Edit Merchant ID. | ✓ | × | | Server domain whitelist. | ✓ | × | | Managewhitelist. | ✓ | × | | Release mini programs. | ✓ | × | | Remove mini programs. | ✓ | × | | Manage mini program versions. | ✓ | × | | Withdraw mini program grayscale release. | ✓ | × | | Features | View feature list. | ✓ | × | | Add features. | ✓ | × | | View features' details. | ✓ | × | | View added features. | ✓ | × | | Edit features. | ✓ | × | | JSAPIs | View JSAPIs' list. | ✓ | × | | Add JSAPIs. | ✓ | × | | View JSAPIs' details. | ✓ | × | | Edit JSAPIs. | ✓ | × | | Analytics | View performance | ✓ | × | | View user behaviors. | ✓
Mini programs of the native app and its merchants | × | | View transaction record. | ✓
Mini programs of the native app and its merchants | × | | View daily metrics. | ✓
Mini programs of the native app and its merchants | × | | Quality | Search mini programs. | ✓ | × | | Crash and abnormal tracking | ✓ | × | | Workspace | Manage workspace information. | ✓ | × | | View details of developer workspaces. | ✓ | × | | Delete developer workspaces. | ✓ | × | | App | View App list. | ✓ | × | | View App details. | ✓ | × | | Edit Apps' Information. | | × | | Add Apps and maintain keys' information. | ✓ | × | | Members | Add members and set members as:
- Workspace Admin
- Workspace Reviewer
- Workspace Developer
- Workspace Operator | ✓ | × | | Block members. | ✓ | × | | Delete members. | ✓ | × | | Approvals | Release mini programs. | ✓
Work orders of the native app and its merchants | ✓
Work orders submitted to the role | | Remove mini programs. | ✓
Work orders of the native app and its merchants | ✓ | | Add features. | ✓
Work orders of the native app and its merchants | ✓ | | Add related features. | ✓
Work orders of the native app and its merchants | ✓ | | Release mini programs in target apps. | ✓
Work orders of the native app and its merchants | × | | Task running of notification delivery | ✓
Work orders of the native app and its merchants | ✓ | | Task running of advertising | ✓
Work orders of the native app and its merchants | ✓ | | My approvals | ✓

Work orders of the native app and its merchants | × | | Feedback | View feedback list. | ✓
Mini programs of the native app and its merchants | × | | View details of feedback. | ✓
Mini programs of the native app and its merchants | × | | Response to feedback. | ✓
Mini programs of the native app and its merchants | × | | Provide solutions for feedback. | ✓
Mini programs of the native app and its merchants | × | | Audience | Create and manage audiences. | ✓ | × | | Mini service | Set mini service. | ✓ | × | | View mini service analytics. | ✓ | × | | Notification delivery | Create and manage campaigns. | ✓ | × | | Create and manage tasks. | ✓ | × | | Configure fatigue settings. | ✓
Set notification delivery frequency for both the native app and merchants. | × | | View templates. | ✓ | × | | Manage templates | ✓ | × | | View notification analytics. | ✓ | × | | Advertising | Create and manage tasks. | ✓ | × | | View ad analytics. | ✓ | × |

Workspace developer and operator

You can view the authorization of the workspace developer and workspace operator in the following tables:

| | | | | | --- | --- | --- | --- | | **Menu In the Mini Program Console** | **Operations** | **Workspace Developer** | **Workspace Operator** | | Mini Program | View mini programs. | ✓
Mini programs assigned to the role. | ✓
Mini programs assigned to the role. | | Create mini programs. | ✓ | × | | View mini program list. | ✓ | × | | Edit mini programs' information. | ✓ | × | | Delete mini programs. | ✓ | × | | Manage members. | × | × | | Upload, preview, and debug IDE features. | ✓ | × | | Edit Client ID. | ✓ | × | | Edit Merchant ID. | ✓ | × | | Server domain whitelist. | ✓ | × | | Managewhitelist. | ✓ | × | | Release mini programs. | ✓ | × | | Remove mini programs. | ✓ | × | | Manage mini program versions. | ✓ | × | | Withdraw mini program grayscale release. | ✓ | × | | Features | View feature list. | ✓ | × | | Add features. | ✓ | × | | View features' details. | ✓ | × | | View added features. | ✓ | × | | Edit features. | ✓ | × | | JSAPIs | View JSAPIs' list. | ✓ | × | | Add JSAPIs. | ✓ | × | | View JSAPIs' details. | ✓ | × | | Edit JSAPIs. | ✓ | × | | Analytics | View performance | ✓
Mini programs assigned to the role | × | | View user behaviors. | ✓ | ✓
Mini programs of the native app and its merchants | | View transaction record. | × | ✓
Mini programs of the native app and its merchants | | View daily metrics. | × | ✓
Mini programs of the native app and its merchants | | Quality | Search mini programs. | ✓
Mini programs assigned to the role | × | | Crash and abnormal Tracking | ✓
Mini programs assigned to the role | × | | Workspace | Manage workspace information. | × | × | | View details of developer workspaces. | × | × | | Delete developer workspaces. | × | × | | App | View App list. | × | × | | View App details. | × | × | | Edit Apps' Information. | × | × | | Add Apps and maintain keys' information. | × | × | | Members | Add members and set members as:
- Workspace Admin
- Workspace Reviewer
- Workspace Developer
- Workspace Operator | × | × | | Block members. | × | × | | Delete members. | × | × | | Approvals | Release mini programs. | × | × | | Remove mini programs. | × | × | | Add features. | × | × | | Add related features. | × | × | | Release mini programs in target apps. | × | × | | Task running of notification delivery | × | × | | Task running of advertising | × | × | | My approvals | ✓
Approvals submitted by the role | ✓
Approvals submitted by the role | | Feedback | View feedback list. | × | ✓
Mini programs of the native app and its merchants | | View details of feedback. | × | ✓

Mini programs of the native app and its merchants | | Response to feedback. | × | ✓
Mini programs of the native app and its merchants | | Provide solutions for feedback. | × | ✓
Mini programs of the native app and its merchants | | Audience | Create and manage audiences. | × | ✓ | | Mini service | Set mini service. | × | ✓ | | View mini service analytics. | × | ✓ | | Notification delivery | Create and manage campaigns. | × | ✓ | | Create and manage tasks. | × | ✓ | | Configure fatigue settings. | × | ✓
Set notification delivery frequency for both the native app and merchants. | | View templates. | ✓ | ✓ | | Manage templates. | × | ✓ | | View notification analytics. | × | ✓
All mini programs' data of the native app and its merchants | | Advertising | Create and manage tasks. | × | ✓ | | View ad analytics. | × | ✓
All mini programs' data of the native app and its merchants |

More information

Members

Add members

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/tenant-member-authorization

---

Members {#members}

Last updated: 2024-12-24

Path: miniprogram_gcash

Members

2024-12-24 21:43

You can use this feature to manage members in a workspace. You can invite members and assign different roles to these members to perform different activities in the workspace.

This feature is available for workspace admins and developer admins.

Features

You can perform the following actions:

- Add new members by sending invitation with the following details:
- Name

- Email address
- Role: You can set the following roles:
- Workspace Developer
- Workspace Admin
- Workspace Reviewer
- Check the members in a list with the following details
- User
- Role
- Developer Workspace
- Status
- Start Date
- Action: You can take the following actions to a member:
- Resend the invitation
- Withdraw the invitation
- Delete members

Workspace admins can delete a member from the workspace and the member has no access to the workspace.

- Block the role of a member and unblock the role

Figure 1. Workspace admins manage members

The following features are located in the mini program detail page:

- Add a member by choosing a member from the list, which is defined in the Member page.
- Change the role of a member

You can set the role as either developer or admin.

- Delete a member.

For more information, see Member in Manage Mini Programs.

Figure 2. Developer admins manage members

More Information

Overview

Manage Mini Programs

Member Role

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/member

---

Merchant onboarding {#merchant-onboarding}

Last updated: 2023-01-30

Path: miniprogram_gcash

Merchant onboarding

2023-01-30 16:02

This topic is intended for merchants/ISVs to know how to complete the onboarding process to the Mini Program Platform. See the product guide Using Mini Program Platform for more information.

Quick overview

As an Merchant, you can the following onboarding process as illustrated below:

The onboarding process consists of the following procedures:

1. Apply for an account

Go to the Mini Program portal and enter your basic contact information to apply for an account.

2. Know Your Business (KYB)

Submit your business information for the wallet to conduct the Know Your Business (KYB) project.

3. Approval process

Submit your registration and wait for your wallet's approval email notification.

4. Log in to the platform

Log into the Mini Program Platform to manage mini programs.

Prerequisites

The merchant must become a partner of the wallet first. After that, the merchant contacts the wallet for the mini program platform address and starts to sign up.

Also, make sure the wallet has finished the onboarding process. See Wallet onboarding for details.

Procedures

The merchant/ISV onboarding process consists of the following steps.

1. Apply for an account

Contact the wallet for the link to the Mini Program Platform. By clicking the link within the invitation email you received, create your account as below:

Enter your contact information, and the account creation process is completed.

2. KYB

After you register your account, you'll then be redirected to the Know Your Business page.

Enter the required information, and click **Submit**.

3. Apply for the approval

After you submit the approval request, the wallet admin will review and process your request. When the request is approved, you will receive a notification email.

4. Log in to the platform

Use your account to sign in to the wallet's mini program platform and start to create your mini program.

Now your onboarding process is finished. You can start to create mini programs.

Next steps

Check this Video Tutorial for the Mini Program Platform to get started with mini programs.

Or go to the product guide Using Mini Program Platform to explore the functionalities.

More information

About Mini Program

Developing Mini Program

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/getting-started/merchant-on-boarding

---

Message {#message}

Last updated: 2022-07-03

Path: miniprogram_gcash

Message

2022-07-03 18:44

Result page.

Sample Code

copy

```
// API-DEMO page/component/message/message.json
{
  "defaultTitle": "Mini Program AntUI component library",
  "usingComponents": {
    "message": "mini-antui/es/message/index"
  }
}
```

copy

```
<!-- API-DEMO page/component/message/message.axml -->
<view>
  <message
    title="{{title}}"
    subTitle="{{subTitle}}"
    type="success"
    mainButton="{{messageButton.mainButton}}"
    subButton="{{messageButton.subButton}}"
    onTapMain="goBack">
  </message>
</view>
```

copy

```
    // API-DEMO page/component/message/message.js
    Page({
      data: {
        title: "Operation succeeded",
        subTitle: "Content details can be wrapped. Up to two lines are
recommended",
        messageButton: {
          mainButton: {
            buttonText: "Main operation"
          },
          subButton: {
            buttonText: "Auxiliary operation"
          }
        }
      },
      goBack() {
        my.navigateBack();
      }
    });
```

Attributes

| | | | | | --- | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | **Required** | | className | Customized class. | String | - | No | | type | Five status types include success, fail, info, warn and waiting, success by default. | String | success | No | | title | Main title. | String | - | Yes | | subTitle | Sub title. | String | - | No | | mainButton | Text of the main button is related with the availability. | Object<buttonText, disabled> | - | No | | subButton | Text of the auxiliary button is related with the availability. | Object<buttonText, disabled> | - | No | | onTapMain | Click function of the main button. | () => {} | - | No | | onTapSub | Click function of the auxiliary button. | () => {} | - | No |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_result_message

---

Message encoding {#message-encoding}

Last updated: 2022-07-03

Path: miniprogram_gcash

Message encoding

2022-07-03 18:44

To prevent errors or ambiguity caused by special charaters enclosed in a message, need to properly encode the message before message is transmitted.

| | | --- | --- | | **Encoding scenarios** | **Encoding method** | | For the byte data, such as the signature and the encrypted content, encode the data with the `base64` algorithm before transmitting. | Use the `base64UrlEncode` function to encode data when calculating to generate a signature. | | For the HTTPS URL data, perform URL encoding first before transmitting.
For example:
- Original URL: https://www.merchant.com/authorizationResult
- Encoded URL: `https%3A%2F%2Fwww.merchant.com%2FauthorizationResult` |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/encode

---

Message encoding {#message-encoding}

Last updated: 2021-05-09

Path: miniprogram_gcash

Message encoding

2021-05-09 18:43

To prevent errors or ambiguity caused by special charaters enclosed in a message, need to properly encode the message before message is transmitted.

| | | --- | --- | | **Encoding scenarios** | **Encoding method** | | For the byte data, such as the signature and the encrypted content, encode the data with the `base64` algorithm before transmitting. | Use the `base64UrlEncode` function to encode data when calculating to generate a signature. | | For the HTTP/HTTPS URL data, perform URL encoding first before transmitting.
For example:
- Original URL: https://www.merchant.com/authorizationResult
- Encoded URL: `https%3A%2F%2Fwww.merchant.com%2FauthorizationResult` |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/encode

---

Message transmission security {#message-transmission-security}

Last updated: 2021-05-09

Path: miniprogram_gcash

Message transmission security

2021-05-09 18:43

The message transmission security is guaranteed with signing a request and validating the signature of the response.

Message signing and signature validation is mandatory for all requests and response.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/msg_transmission_security

---

Message transmission security {#message-transmission-security}

Last updated: 2022-07-03

Path: miniprogram_gcash

Message transmission security

2022-07-03 18:44

The message transmission security is guaranteed with signing a request and validating the signature of the response.

Message signing and signature validation is mandatory for all requests and response.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/msg_transmission_security

---

Method and Attribute {#method-and-attribute}

Last updated: 2021-05-09

Path: miniprogram_gcash

Method and Attribute

2021-05-09 18:43

Component Methods

The custom component can not only render static data but also respond to user click event, so as to handle and trigger custom component re-render. In the methods, it is possible to define any customized method.

Note:

Different from Page, the custom component needs to define the event handler in the methods.

Modify component axml:

copy

```
// /components/counter/index.axml
<view>{{counter}}</view>
<button onTap="plusOne">+1</button>
```

Handle event in component js:

copy

```
// /components/counter/index.js
Component({
  data: { counter: 0 },
  methods: {
    plusOne(e) {
      console.log(e);
      this.setData({ counter: this.data.counter + 1 });
    },
  },
});
```

Now the page renders an additional button. Each click on it will increase the page number by 1.

Props

Custom component is not isolated from the outside. By now, the example is a standalone module. To make it interact with the outside, the custom component can accept external input. After processing is done, it can notify the outside with “Done”. All those can be implemented with props.

Example:

Note:

- The props is the attribute transferred from outside. It is possible to specify default attribute, and cannot modify in the internal codes of the custom component.
- In the custom component axml, it is possible to refer to the props attribute directly.

- For the event in the custom component axml, only the method in the “methods” of the js of the custom component can respond. If it is required to call the function transferred from the parent component, it is possible to call it via this.props in the methods.

copy

```
// /components/counter/index.js
Component({
  data: { counter: 0 },
  props: {
    onCounterPlusOne: (data) => console.log(data),
    extra: 'default extra',
  },
  methods: {
    plusOne(e) {
      console.log(e);
      const counter = this.data.counter + 1;
      this.setData({ counter });
      this.props.onCounterPlusOne(counter); // Response to the event
in axml can be through the method in "methods" only
    },
  },
});
```

The above codes set default attributes for props, and then the event handler get those attributes via **this.props**.

copy

```
// /components/counter/index.axml
<view>{{counter}}</view>
<view>extra: {{extra}}</view>
<button onTap="plusOne">+1</button>
```

External use: do not transfer props

copy

```
// /pages/index/index.axml
<my-component />
```

Page output:

copy

```
0
extra: default extra
+1
```

Now no parameter is transferred, so the page shows the default configured for props in the component js.

External use: transfer props

Note:

When custom component is used externally, if the transfer parameter is a function, the “on” suffix is necessary; otherwise it will be processed as a string.

copy

```
// /pages/index/index.js
Page({
  onCounterPlusOne(data) {
    console.log(data);
  }
});
```

copy

```
// /pages/index/index.axml
<my-component extra="external extra"
onCounterPlusOne="onCounterPlusOne" />
```

Page output:

copy

```
0
extra: external extra
+1
```

Here parameter is transferred, so the page shows the extra value transferred externally “external extra”.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_custom-component_create-custom-component_method-and-attribute

---

Method and Attribute {#method-and-attribute}

Last updated: 2022-07-03

Path: miniprogram_gcash

Method and Attribute

2022-07-03 18:44

Component Methods

The custom component can not only render static data but also respond to user click event, so as to handle and trigger custom component re-render. In the methods, it is possible to define any customized method.

Note:

Different from Page, the custom component needs to define the event handler in the methods.

Modify component axml:

copy

```
// /components/counter/index.axml
<view>{{counter}}</view>
<button onTap="plusOne">+1</button>
```

Handle event in component js:

copy

```
// /components/counter/index.js
Component({
  data: { counter: 0 },
  methods: {
    plusOne(e) {
      console.log(e);
      this.setData({ counter: this.data.counter + 1 });
    },
  },
});
```

Now the page renders an additional button. Each click on it will increase the page number by 1.

Props

Custom component is not isolated from the outside. By now, the example is a standalone module. To make it interact with the outside, the custom component can accept external input. After processing is done, it can notify the outside with “Done”. All those can be implemented with props.

Example:

Note:

- The props is the attribute transferred from outside. It is possible to specify default attribute, and cannot modify in the internal codes of the custom component.
- In the custom component axml, it is possible to refer to the props attribute directly.

- For the event in the custom component axml, only the method in the “methods” of the js of the custom component can respond. If it is required to call the function transferred from the parent component, it is possible to call it via this.props in the methods.

copy

```
// /components/counter/index.js
Component({
  data: { counter: 0 },
  props: {
    onCounterPlusOne: (data) => console.log(data),
    extra: 'default extra',
  },
  methods: {
    plusOne(e) {
      console.log(e);
      const counter = this.data.counter + 1;
      this.setData({ counter });
      this.props.onCounterPlusOne(counter); // Response to the event
in axml can be through the method in "methods" only
    },
  },
});
```

The above codes set default attributes for props, and then the event handler get those attributes via **this.props**.

copy

```
// /components/counter/index.axml
<view>{{counter}}</view>
<view>extra: {{extra}}</view>
<button onTap="plusOne">+1</button>
```

External use: do not transfer props

copy

```
// /pages/index/index.axml
<my-component />
```

Page output:

copy

```
0
extra: default extra
+1
```

Now no parameter is transferred, so the page shows the default configured for props in the component js.

External use: transfer props

Note:

When custom component is used externally, if the transfer parameter is a function, the “on” suffix is necessary; otherwise it will be processed as a string.

copy

```
// /pages/index/index.js
Page({
  onCounterPlusOne(data) {
    console.log(data);
  }
});
```

copy

```
// /pages/index/index.axml
<my-component extra="external extra"
onCounterPlusOne="onCounterPlusOne" />
```

Page output:

copy

```
0
extra: external extra
+1
```

Here parameter is transferred, so the page shows the extra value transferred externally “external extra”.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_custom-component_create-custom-component_method-and-attribute

---

Mini Program Javascript Engine {#mini-program-javascript-engine}

Last updated: 2022-07-03

Path: miniprogram_gcash

Mini Program Javascript Engine

2022-07-03 18:44

Running Engine

Mini program JavaScript codes fall into logic layer script and sjs script, which run on different threads in the the same JavaScript engine. Mini Program JavaScript Engine is different on different OSs. On the iOS platform, the script runs on the JavaScriptCore provided by the operating system. On the Android platform, the script runs on the V8 engine .

Mini program performs the babel conversion for the codes uploaded by the developers, so that the JavaScript engine supports most of the ES6 new features. For the internal object of the ES6 extension, Mini Program does not provide polyfill on the JavaScript engine, which causes difference on the supports for different ES6 extension internal objects for the JavaScript on different platforms. the developers need to avoid using the internal objects unsupported by JavaScript engine, and can provide polyfill for the internal objects. (Polyfill means the unsupported raw API codes that are used to implement browsers or other JavaScript engines, such as babel-polyfill)

Supports of Client OSs for ES6 Extension Internal Objects

The table below lists the OS supports for ES6 extension internal objects.

| | | | | --- | --- | --- | --- | | **Object** | **iOS 9** | **iOS 10 and above** | **Android** | | Object.is | Support | Support | Support | | Object.assign | Support | Support | Support | | Object.keys | Support | Support | Support | | Object.getOwnPropertyDescriptor | Support | Support | Support | | Object.getOwnPropertyNames | Support | Support | Support | | Object.getOwnPropertySymbols | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **String** | **iOS 9** | **iOS 10 and above** | **Android** | | String.prototype.codePointAt | Support | Support | Support | | String.prototype.normalize | No support | Support | Support | | String.prototype.includes | Support | Support | Support | | String.prototype.startsWith | Support | Support | Support | | String.prototype.endsWith | Support | Support | Support | | String.prototype.repeat | Support | Support | Support | | String.fromCodePoint | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Array** | **iOS 9** | **iOS 10 and above** | **Android** | | Array.prototype.copyWithin | Support | Support | Support | | Array.prototype.find | Support | Support | Support | | Array.prototype.findIndex | Support | Support | Support | | Array.prototype.find | Support | Support | Support | | Array.prototype.entries | Support | Support | Support | | Array.prototype.keys | Support | Support | Support | | Array.prototype.values | Support | Support | No support | | Array.prototype.includes | Support | Support | Support | | Array.from | Support | Support | Support | | Array.of | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Number** | **iOS 9** | **iOS 10 and above** | **Android** | | Number.isFinite | Support | Support | Support | | Number.isNaN | Support | Support | Support | | Number.parseInt | Support | Support | Support | | Number.parseFloat | Support | Support | Support | | Number.isInteger | Support | Support | Support | | Number.EPSILON | Support | Support | Support | | Number.isSafeInteger | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Math** | **iOS 9** | **iOS 10 and above** | **Android** | | Math.trunc | Support | Support | Support | | Math.sign | Support | Support | Support | | Math.cbrt | Support | Support | Support | | Math.clz32 | Support | Support | Support | | Math.imul | Support | Support | Support | | Math.fround | Support | Support | Support | | Math.hypot | Support | Support | Support | | Math.expm1 | Support | Support | Support | | Math.log1p | Support | Support | Support | | Math.log10 | Support | Support | Support | | Math.log2 | Support | Support | Support | | Math.sinh | Support | Support | Support | | Math.cosh | Support | Support | Support | | Math.tanh | Support | Support | Support | | Math.asinh | Support | Support | Support | | Math.acosh | Support | Support | Support | | Math.atanh | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Internal objects** | **iOS 9** | **iOS 10 and above** | **Android** | | Set | Support | Support | Support | | Map | Support | Support | Support | | Proxy | No support | Support | Support | | Reflect | Support | Support | Support | | Promise | Support | Support | Support |

Limitations on Dynamic Execution Script

For sake of security, Mini Program limits some of the syntax and APIs of ES.

- It does not support eval using
- setTimeout and setInterval functions, and supports only the function for callback parameters. Dynamic execution of codes is not supported.
- It does not support using new Function to create a function.

Reserved Words of Module Name

The logic layer of Mini Program supports the ES2015 module syntax but regards some internal object names (such as window and document) of browser as the reserved words for emergency needs in the future. Those reserved words cannot be used as the module name. The reserved words include `globalThis`, `global`, `fetch`, `self`, `window`, `document`, `location` and `XMLHttpRequest`. For more details, see the descriptions of module name reserved words in framework description.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_mini-program-javascript-engine

---

Mini Program Javascript Engine {#mini-program-javascript-engine}

Last updated: 2021-05-09

Path: miniprogram_gcash

Mini Program Javascript Engine

2021-05-09 18:43

Running Engine

Mini program JavaScript codes fall into logic layer script and sjs script, which run on different threads in the the same JavaScript engine. Mini Program JavaScript Engine is different on different OSs. On the iOS platform, the script runs on the JavaScriptCore provided by the operating system. On the Android platform, the script runs on the V8 engine .

Mini program performs the babel conversion for the codes uploaded by the developers, so that the JavaScript engine supports most of the ES6 new features. For the internal object of the ES6 extension, Mini Program does not provide polyfill on the JavaScript engine, which causes difference on the supports for different ES6 extension internal objects for the JavaScript on different platforms. the developers need to avoid using the internal objects unsupported by JavaScript engine, and can provide polyfill for the internal objects. (Polyfill means the unsupported raw API codes that are used to implement browsers or other JavaScript engines, such as babel-polyfill)

Supports of Client OSs for ES6 Extension Internal Objects

The table below lists the OS supports for ES6 extension internal objects.

| | | | | --- | --- | --- | --- | | **Object** | **iOS 9** | **iOS 10 and above** | **Android** | | Object.is | Support | Support | Support | | Object.assign | Support | Support | Support | | Object.keys | Support | Support | Support | | Object.getOwnPropertyDescriptor | Support | Support | Support | | Object.getOwnPropertyNames | Support | Support | Support | | Object.getOwnPropertySymbols | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **String** | **iOS 9** | **iOS 10 and above** | **Android** | | String.prototype.codePointAt | Support | Support | Support | | String.prototype.normalize | No support | Support | Support | | String.prototype.includes | Support | Support | Support | | String.prototype.startsWith | Support | Support | Support | | String.prototype.endsWith | Support | Support | Support | | String.prototype.repeat | Support | Support | Support | | String.fromCodePoint | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Array** | **iOS 9** | **iOS 10 and above** | **Android** | | Array.prototype.copyWithin | Support | Support | Support | | Array.prototype.find | Support | Support | Support | | Array.prototype.findIndex | Support | Support | Support | | Array.prototype.find | Support | Support | Support | | Array.prototype.entries | Support | Support | Support | | Array.prototype.keys | Support | Support | Support | | Array.prototype.values | Support | Support | No support | | Array.prototype.includes | Support | Support | Support | | Array.from | Support | Support | Support | | Array.of | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Number** | **iOS 9** | **iOS 10 and above** | **Android** | | Number.isFinite | Support | Support | Support | | Number.isNaN | Support | Support | Support | | Number.parseInt | Support | Support | Support | | Number.parseFloat | Support | Support | Support | | Number.isInteger | Support | Support | Support | | Number.EPSILON | Support | Support | Support | | Number.isSafeInteger | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Math** | **iOS 9** | **iOS 10 and above** | **Android** | | Math.trunc | Support | Support | Support | | Math.sign | Support | Support | Support | | Math.cbrt | Support | Support | Support | | Math.clz32 | Support | Support | Support | | Math.imul | Support | Support | Support | | Math.fround | Support | Support | Support | | Math.hypot | Support | Support | Support | | Math.expm1 | Support | Support | Support | | Math.log1p | Support | Support | Support | | Math.log10 | Support | Support | Support | | Math.log2 | Support | Support | Support | | Math.sinh | Support | Support | Support | | Math.cosh | Support | Support | Support | | Math.tanh | Support | Support | Support | | Math.asinh | Support | Support | Support | | Math.acosh | Support | Support | Support | | Math.atanh | Support | Support | Support |

| | | | | --- | --- | --- | --- | | **Internal objects** | **iOS 9** | **iOS 10 and above** | **Android** | | Set | Support | Support | Support | | Map | Support | Support | Support | | Proxy | No support | Support | Support | | Reflect | Support | Support | Support | | Promise | Support | Support | Support |

Limitations on Dynamic Execution Script

For sake of security, Mini Program limits some of the syntax and APIs of ES.

- It does not support eval using
- setTimeout and setInterval functions, and supports only the function for callback parameters. Dynamic execution of codes is not supported.
- It does not support using new Function to create a function.

Reserved Words of Module Name

The logic layer of Mini Program supports the ES2015 module syntax but regards some internal object names (such as window and document) of browser as the reserved words for emergency needs in the future. Those reserved words cannot be used as the module name. The reserved words include `globalThis`, `global`, `fetch`, `self`, `window`, `document`, `location` and `XMLHttpRequest`. For more details, see the descriptions of module name reserved words in framework description.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_mini-program-javascript-engine

---

Mini Program Operation Mechanism {#mini-program-operation-mechanism}

Last updated: 2021-05-09

Path: miniprogram_gcash

Mini Program Operation Mechanism

2021-05-09 18:43

Download

Mini program does not need installation. When it is used for the first time by the user, AppContainer downloads resources of the Mini Program from the server. The downloaded Mini Program resources are cached in the mobile app client for some time. When the Mini Program with cached resources is opened again, the download process is skipped so the Mini Program is opened faster.

Hot Startup and Cold Startup

- **Cold startup:** When the user opens a Mini Program that has not been started or has been destroyed, this is called a cold startup. Here, the Mini Program starts the initialization process. When the process is completed, it triggers the `onLaunch` callback function.
- **Hot startup:** When the user opens a Mini Program that has been opened but is running in background, this is called the hot startup. Here, the Mini Program is not destroyed and restarted. Instead, it is switched from background to foreground. The `onLaunch` callback function is not triggered.

Foreground/Background Running

- **Foreground running:** When the user opens a Mini Program for the first time, the Mini Program runs in the foreground.
- **Background running:** When the user clicks the close button in the top-right corner to close the Mini Program or press the Home button to leave mobile App, the Mini Program is not destroyed but switched to background.
- **Switching from background to foreground:** When the Mini Program that has not been destroyed by the system is reopened or reactivated, it is switched from background to foreground.

The callback function for foreground/background switching can be registered in app.js. When the Mini Program switches from background to foreground, it triggers `onShow`; conversely it triggers `onHide`.

Cache

The local cache system can provide store, get and remove ability to control cache. A Mini Program can store at most 10 MB data. There are two types of API: sync API and async API. The sync API will block current task until the method ends and sync API will not block current task.

| | | | | --- | --- | --- | --- | | **Operation** | **Sync API** | **Async API** | **Description** | | Store | my.setStorageSync | my.setStorage | Store the data according to the `key`, the original data will be overwrite if the `key` is the same. | | Get | my.getStorageSync | my.getStorage | Read cache data. | | Remove | my.removeStorageSync | my.removeStorage | Remove specific data. | | Clear | my.clearStorageSync | my.clearStorage | Clear all the cache data. |

Destroy

When the user clicks the close button in the top-right corner to close the Mini Program, the Mini Program is not destroyed but switched to background. Only when the Mini Program stays in background for some time or the system resource consumption is too high, it is really destroyed.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_mini-program-operation-mechanism

---

Mini Program Operation Mechanism {#mini-program-operation-mechanism}

Last updated: 2022-07-03

Path: miniprogram_gcash

Mini Program Operation Mechanism

2022-07-03 18:44

Download

Mini program does not need installation. When it is used for the first time by the user, AppContainer downloads resources of the Mini Program from the server. The downloaded Mini Program resources are cached in the mobile app client for some time. When the Mini Program with cached resources is opened again, the download process is skipped so the Mini Program is opened faster.

Hot Startup and Cold Startup

- **Cold startup:** When the user opens a Mini Program that has not been started or has been destroyed, this is called a cold startup. Here, the Mini Program starts the initialization process. When the process is completed, it triggers the `onLaunch` callback function.
- **Hot startup:** When the user opens a Mini Program that has been opened but is running in background, this is called the hot startup. Here, the Mini Program is not destroyed and restarted. Instead, it is switched from background to foreground. The `onLaunch` callback function is not triggered.

Foreground/Background Running

- **Foreground running:** When the user opens a Mini Program for the first time, the Mini Program runs in the foreground.
- **Background running:** When the user clicks the close button in the top-right corner to close the Mini Program or press the Home button to leave mobile App, the Mini Program is not destroyed but switched to background.
- **Switching from background to foreground:** When the Mini Program that has not been destroyed by the system is reopened or reactivated, it is switched from background to foreground.

The callback function for foreground/background switching can be registered in app.js. When the Mini Program switches from background to foreground, it triggers `onShow`; conversely it triggers `onHide`.

Cache

The local cache system can provide store, get and remove ability to control cache. A Mini Program can store at most 10 MB data. There are two types of API: sync API and async API. The sync API will block current task until the method ends and sync API will not block current task.

| | | | | --- | --- | --- | --- | | **Operation** | **Sync API** | **Async API** | **Description** | | Store | my.setStorageSync | my.setStorage | Store the data according to the `key`, the original data will be overwrite if the `key` is the same. | | Get | my.getStorageSync | my.getStorage | Read cache data. | | Remove | my.removeStorageSync | my.removeStorage | Remove specific data. | | Clear | my.clearStorageSync | my.clearStorage | Clear all the cache data. |

Destroy

When the user clicks the close button in the top-right corner to close the Mini Program, the Mini Program is not destroyed but switched to background. Only when the Mini Program stays in background for some time or the system resource consumption is too high, it is really destroyed.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_mini-program-operation-mechanism

---

Mini Program Studio {#mini-program-studio}

Last updated: 2022-07-03

Path: miniprogram_gcash

Mini Program Studio

2022-07-03 18:44

The Mini Program Studio is a one-stop development tool that helps you to quickly write, deploy and debug mini programs. With the Integrated Development Environment (IDE), you can build mini programs across all languages. manage the project. In addition, you can use the studio for project management that helps to deal with collaboration across your teams.

Note:

Make sure you have implemented the IDE. If you have not implemented the IDE, choose RESOURCES in the Mini Program Portal to download.

Features

You can use this studio to do the following:

- Create or open a mini program
- Set up the project with a name and location
- Associate with a mini program to upload code packages
- Customize the compiling mode to pass parameters to the simulator
- Develop and build a mini program
- Manage the structure of mini program
- Develop a mini program
- Debug with a local simulator
- Preview and debug with a real machine
- Upload the mini program

- Check the related information of the current development environment

In addition to the basic functions, you can also benefit from the following functions that are specific to mini programs:

- Realtime preview
- Autocomplete
- Syntax prompt/suggestions

More Information

The Main Interface

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/mini-program-studio_overview

---

Mini Program Studio {#mini-program-studio}

Last updated: 2021-05-09

Path: miniprogram_gcash

Mini Program Studio

2021-05-09 18:43

The Mini Program Studio is a one-stop development tool that helps you to quickly write, deploy and debug mini programs. With the Integrated Development Environment (IDE), you can build mini programs across all languages. manage the project. In addition, you can use the studio for project management that helps to deal with collaboration across your teams.

Note:

Make sure you have implemented the IDE. If you have not implemented the IDE, choose Mini Program Studio in the Mini Program Portal to download.

Features

You can use this studio to do the following:

- Create or open a mini program
- Set up the project with a name and location

- Associate with a mini program to upload code packages
- Customize the compiling mode to pass parameters to the simulator
- Develop and build a mini program
- Manage the structure of mini program
- Develop a mini program
- Debug with a local simulator
- Preview and debug with a real machine
- Upload the mini program
- Check the related information of the current development environment

In addition to the basic functions, you can also benefit from the following functions that are specific to mini programs:

- Realtime preview
- Autocomplete
- Syntax prompt/suggestions

More Information

The Main Interface

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/mini-program-studio_overview

---

Mini Program general design guidelines {#mini-program-general-design-guidelines}

Last updated: 2021-05-10

Path: miniprogram_gcash

Mini Program general design guidelines

2021-05-10 04:11

User experience is important to deliver services with high quality for merchants in the Mini Program platform. Based on the fast speed and simplicity of the Mini Program and on the principle of fully respecting users' right to know and right to operate, these design guidelines and recommendations are intended to create a friendly, efficient, and

consistent user experience within the wallet ecosystem, while fulfilling a variety of requirements, and achieving a mutually beneficial situation for both users and the Mini Program service providers.

Mini Program Design Guidelines_V1.0.sketch

Mini Program Design Guidelines_V1.0.pdf

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/design/guideline

---

Mini Program types {#mini-program-types}

Last updated: 2022-07-07

Path: miniprogram_gcash

Mini Program types

2022-07-07 17:08

When you create a new mini program, you can choose the mini program type, which can be:

- `DSL( Default)`: Native mini program
- `HTML5`: HTML5 mini program

For more information about how to create HTML5 mini programs, see How to transform an HTML 5 mobile app to an HTML 5 mini program?

Native mini program

Native mini programs call native APIs provided by the wallet app directly. For the functionality introduction about native mini programs, see Manage Mini Programs.

HTML5 mini program

HTML 5 mini programs are embedded with web pages and call native APIs provided by the wallet app through bridges. The overall Mini Program Manage functionalities are the same as that of native mini programs, but have a slight difference between two mini program types. The difference exists in the following aspects:

| | | --- | --- | | **Category** | **Difference description** | | Create the mini program | The `Entrance URL` field is required. Entrance URL is the entrance of the HTML 5 mini program. You can modify the entrance URL after the HTML 5 mini program is created. | | Develop the mini program | The IDE is not required for the HTML 5 mini program development. | | Publish the mini program | - The publishment steps are simpler than that of the native mini program. The package build process is not required.

- When you submit the publishment application, you need to review the mini program information. If all required information is filled in, submit the application. If some required information is missed, fill out the information under `Mini Programs` > `Info`.|| Approve the application | When workspace admins approve the HTML 5 mini program publishment application, the step of mini program quality review is not supported. |

Table 1. Difference for HTML 5 mini program

Note:

After you update information, such as client ID, under the `Configuration` tab, you must save the changes that you made:

- If an unpublished HTML 5 mini program version exists, all changes can be saved for the current unpublished version.
- If no unpublished HTML 5 mini program version exists, all changes can be saved for a new version.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/miniprogramtype

---

Mini program member roles {#mini-program-member-roles}

Last updated: 2022-07-07

Path: miniprogram_gcash

Mini program member roles

2022-07-07 17:08

This topic provides you with the main duties of mini program member roles and the relations between mini program member roles and workspace member roles.

Overview

Mini program member roles are responsible for managing mini programs from uploading versions to deleting mini programs (all versions). After creating mini programs, mini program members are generated from workspace members. The workspace member who creates a mini program will be set as an admin of the mini program automatically.

Whether for the wallet or merchants, there are two mini program member roles:

- **Admin**: The role can edit mini programs, add mini program members, upload mini programs, set configurations, and release and delete mini programs. A mini program can have multiple admins.

- **Developer**:The role can upload versions from Mini Program Studio to workspaces. A mini program can have multiple developers.

Relations illustration

Mini program member roles are generated from workspace member roles, you can see relations between them in the following figures:

- Wallet

For the wallet, all workspace members can be set as mini program admins or developers. Only workspace admins and workspace developers can create mini programs. After creating a mini program, workspace admins and workspace developers will be set as admins of the mini program automatically. Then admins can invite workspace admins, workspace developers, workspace reviewers, or workspace operators as admins or developers to manage the mini program together.

- Merchant

For merchants, all workspace members can be set as mini program admins or developers. Only developer admins can create mini programs. After creating a mini program, developer admins will be set as admins of the mini program automatically. Then admins can invite developers (workspace) or operators as admins or developers (mini program) to manage the mini program together.

Mini program member authorization

Mini program members of the wallet and merchants only participate in the workflow from add members to delete mini programs. You can see their authorization in the following table:

| | | | --- | --- | --- | | **Operations** | **Admin** | **Developer** | | Create mini programs | Mini program members generate after creating mini programs. This step is intended for workspace members such as workspace admins, workspace developers, and developer admins to complete. | | View mini programs | ✓ | ✓ | | Edit mini programs | ✓ | × | | Add mini program members | ✓ | × | | Upload mini programs | ✓ | ✓ | | Configure mini programs | ✓ | × | | Release mini programs | ✓ | × | | Remove mini programs | ✓ | × | | Delete mini programs | ✓ | × |

More information

Workspace member roles

Member role authorization in tenant workspace

Member role authorization in developer workspace

How to add members to workspaces

Add mini program members

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/platform/miniprogram-member-role

---

Mini program project configurations {#mini-program-project-configurations}

Last updated: 2022-07-03

Path: miniprogram_gcash

Mini program project configurations

2022-07-03 18:44

With this topic, you can learn to use the `mini.project.json` file in the root directory to configure the mini program project.

Overview

In addition to `app` and `page`, the mini program framework also includes `project` to describe the whole mini program project. For more information, refer to Framework overview.

The `project` refers to the `mini.project.json` file that must be placed in the root directory of the mini program project. You can use the `mini.project.json` file to configure advanced features, as listed in the table below.

Features

The following table lists the configuration features of the mini program project:

| | | | --- | --- | --- | | **Field Name** | **Data Type** | **Description** | | miniprogramRoot | Path String | Specifies the relative path of the mini program source code (the directory where the `app.json` file is located). | | component2 | Boolean | Whether to enable custom components.
To enable custom components, check **Enable component2 compile** in the **Details > Project configuration** of the IDE. For more information, refer to Custom component introduction. | | axmlStrictCheck | Boolean | Whether to enable axml strict syntax check. Once enabled, it can detect the error of unclosed tags and more.
To enable axml strict syntax check, check **Enable Strick Axml Check** in the **Details > Project configuration** of the IDE. | | enableHMR | Boolean | Whether to enable simulator hot update. | | enableNodeModuleBabelTransform | Boolean | Whether to enable es6 syntax transformation. | | exclude | Array[String] | The file or file folder to exclude when building package. It follows Glob syntax. For more information, see exclude package blacklist. |

`exclude` package blacklist

When you upload a mini program to Mini Program Platform, the local source codes will be packaged and uploaded to the cloud for building. In addition to the above files, the source code package also includes the following contents:

- Mini program source code files:
- `.acss`
- `.axml`
- `.js`
- `.json`
- `.sjs`
- Dependency packages in the `node_modules` directory

When uploading, if the source code package is still too large after compression, network timeout may be triggered. To resolve this error, you can add unnecessary files that are built on the cloud to the `exclude` blacklist. For example, if you precompile the code through `src` -> `dist`, you need to exclude the files in the `src` directory and the `devDependencies` tool in the `node_modules` directory when you upload the mini program.

The following sample code indicates that the source code package excludes the files in the `src` and `node_modules` directories under the root directory of the project:

copy

```
"exclude": [\
  "src/**",\
  "node_modules/**"\
]
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_project

---

Mixins {#mixins}

Last updated: 2021-05-09

Path: miniprogram_gcash

Mixins

2021-05-09 18:43

The developer may implement multiple custom components which may have common logic to be processed. The Mini Program provides the mixins to meet the requirement.

Sample code:

copy

```
// /minxins/lifecylce.js
export default {
  onInit(){},
  deriveDataFromProps(nextProps){},
  didMount(){},
  didUpdate(prevProps,prevData){},
  didUnmount(){},
};
```

copy

```
// /pages/components/xx/index.js
import lifecylce from '../../minxins/lifecylce';
const initialState = {
  data: {
    y: 2
  },
};
const defaultProps = {
  props: {
    a: 3,
  },
};
const methods = {
  methods: {
    onTapHandler() {},
  },
}
Component({
  mixins: [\
    lifecylce,\
    initialState,\
    defaultProps,\
    methods\
  ],
  data: {
    x: 1,
  },
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_custom-component_create-custom-component_mixins

---

Mixins {#mixins}

Last updated: 2022-07-03

Path: miniprogram_gcash

Mixins

2022-07-03 18:44

The developer may implement multiple custom components which may have common logic to be processed. The Mini Program provides the mixins to meet the requirement.

Sample code:

copy

```
// /minxins/lifecylce.js
export default {
  onInit(){},
  deriveDataFromProps(nextProps){},
  didMount(){},
  didUpdate(prevProps,prevData){},
  didUnmount(){},
};
```

copy

```
// /pages/components/xx/index.js
import lifecylce from '../../minxins/lifecylce';
const initialState = {
  data: {
    y: 2
  },
};
const defaultProps = {
  props: {
    a: 3,
  },
};
const methods = {
  methods: {
    onTapHandler() {},
  },
}
Component({
  mixins: [\
    lifecylce,\
    initialState,\
    defaultProps,\
```

```
    methods\
  ],
  data: {
    x: 1,
  },
});
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_custom-component_create-custom-component_mixins

---

Modal {#modal}

Last updated: 2022-07-03

Path: miniprogram_gcash

Modal

2022-07-03 18:44

Dialog box.

| | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | | className | Custom class. | String | | | show | Show `modal` or not. | Boolean | false | | showClose | `Turn off` render or not. | Boolean | true | | closeType | Close chart type 0: gray icon 1: white icon. | String | 0 | | onModalClick | Callback on clicking `footer`. | () => void | | | onModalClose | Callback on clicking `close`, not required when `showClose` is false. | () => void | | | topImage | Top image. | String | | | topImageSize | Top image rule, options including `lg`, `md` and `sm`. | String | md | | advice | Is operation popup or not. | Boolean | false |

Slots

| | | --- | --- | | **slotName** | **Description** | | header | Optional, modal header. | | footer | Optional, modal footer. |

Example

copy

```
{
  "defaultTitle": "AntUI Component Library",
  "usingComponents": {
    "modal": "mini-antui/es/modal/index"
  }
}
```

copy

```
<view>
  <button onTap="openModal">Show Modal</button>
  <modal
    show="{{modalOpened}}"
    onModalClick="onModalClick"
    onModalClose="onModalClose"
    topImage="https://img.example.com/example.png"
  >
    <view slot="header">Title</view>
    Explain the current status, prompt the user solution, preferably
no more than two lines
    <view slot="footer">Confirm</view>
  </modal>
</view>
```

copy

```
Page({
  data: {
    modalOpened: false,
  },
  openModal() {
    this.setData({
      modalOpened: true,
    });
  },
  onModalClick() {
    this.setData({
      modalOpened: false,
    });
  },
  onModalClose() {
    this.setData({
      modalOpened: false,
    });
  }
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_floating-layer_modal

---

Multi Liner {#multi-liner}

Last updated: 2022-07-03

Path: miniprogram_gcash

Multi Liner

2022-07-03 18:44

You can use the multi-liner component to allow users to enter multiple lines of content in an input box.

Note:

The features of the multi-liner component are mainly on the basis of the textarea component. You can see textarea for reference.

Sample code

See the sample codes in different languages:

.json

copy

```
{
  "defaultTitle": "Multi-liner",
  "usingComponents": {
    "multi-liner": "mini-ali-ui/es/multi-liner/index"
  }
}
```

.axml

copy

```
<view>
  <view style="margin-top: 10px;" />
  <view class="title">multi-liner</view>
  <multi-liner
    placeholder="word count↘"
    value="{{value}}"
    onInput="onInput"
    last="{{true}}"
    auto-height="{{true}}"
    controlled="{{controlled}}"/>
  <view style="margin: 10px;" />
</view>
```

.js

copy

```
Page({
  data: {
    value: 'content',
    controlled: true,
  },
  onInput(e) {
    this.setData({
      value: e.detail.value,
    });
  },
});
```

Parameters

| | | | --- | --- | --- | | **Property** | **Type** | **Description** | | className | String | Customized class. | | inputCls | String | Customized class for the input box that uses the input component. | | last | Boolean | An indicator of whether the input box is the last one. The default value is `false`. | | value | String | Initial content in the input box. | | name | String | Component name, which is used to obtain data by submitting the form. | | placeholder | String | Placehoder. | | placeholderStyle | String | Style of the placeholder. | | placeholderClass | String | Style class of the placeholder. | | disabled | Boolean | An indicator of whether to disable the function of clearing the entered content. The default value is `false`. | | maxlength | Number | Maximum length of the verification code. The default value is `140`. | | focus | Boolean | An indicator of whether to get focus. The default value is `false`. | | auto-height | Boolean | An indicator of whether to heighten the input box automatically with the lines increased. The default value is `false`. | | show-count | Boolean | An indicator of whether to display the number of the entered words. The default value is `true`. | | controlled | Boolean | An indicator of whether the multi-liner component is a controlled component. If the value is `true`, the content in the input box is controlled by *setData*. | | onInput | (e: Object) => void | The event that is triggered when users tap the keyboard. | | onConfirm | (e: Object) => void | The event that is triggered when users tap the **Done** button on the keyboard. | | onFocus | (e: Object) => void | The event that is triggered when an element gets the focus. | | onBlur | (e: Object) => void | The event that is triggered when an element loses the focus. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_form_multi-liner

---

NPM Package Management {#npm-package-management}

Last updated: 2021-05-09

Path: miniprogram_gcash

NPM Package Management

2021-05-09 18:43

The developer can manage easily all NPM dependencies in the current project within the Mini Program Studio. The entry is shown in the figure below.

Configuration Workspace

Default Workspace

The Mini Program Studio automatically creates default workspace on basis of the path of the package.json file under the current directory. If the `package.json` is found in the current project, it will show the default workspace as shown above. If the `package.json` is not found in the root path of the project, it will show an empty workspace. Then you need to click the `Add Folder` to add the root path manually. Then the Mini Program Studio will create a `package.json` file in the root path automatically.

Dependency Management

In the npm package management interface, you can manage the dependencies.

- Install specific dependency: input the dependency name in the input box and press `enter` to install the dependency. There are two options to install dependencies. The npm packages installed in the `Dependencies` area will go to production and the npm packages installed in the `DEV DEPENDENCIES` only work for development environment.
- Install all dependencies: click the install all dependencies button to install all dependencies.
- Delete dependency: click the delete button to delete specific dependency.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/mini-program-studio_function-panel_npm-package-management

---

NPM Package Management {#npm-package-management}

Last updated: 2022-07-03

Path: miniprogram_gcash

NPM Package Management

2022-07-03 18:44

The developer can manage easily all NPM dependencies in the current project within the Mini Program Studio. The entry is shown in the figure below.

Configuration Workspace

Default Workspace

The Mini Program Studio automatically creates default workspace on basis of the path of the package.json file under the current directory. If the `package.json` is found in the current project, it will show the default workspace as shown above. If the `package.json` is not found in the root path of the project, it will show an empty workspace. Then you need to click the `Add Folder` to add the root path manually. Then the Mini Program Studio will create a `package.json` file in the root path automatically.

Dependency Management

In the npm package management interface, you can manage the dependencies.

- Install specific dependency: input the dependency name in the input box and press `enter` to install the dependency. There are two options to install dependencies. The npm packages installed in the `Dependencies` area will go to production and the npm packages installed in the `DEV DEPENDENCIES` only work for development environment.
- Install all dependencies: click the install all dependencies button to install all dependencies.
- Delete dependency: click the delete button to delete specific dependency.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/mini-program-studio_function-panel_npm-package-management

---

Notice {#notice}

Last updated: 2022-07-03

Path: miniprogram_gcash

Notice

2022-07-03 18:44

Guide.

| | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Default** | | mode | Prompt option type: `link`, `closable`. | String | '' | | action | Prompt showing text. | String | '' | | actionCls | Prompt showing text custom class. | String | '' | | show | Show the notice bar or not. | Boolean | true | | onClick | Click button callback. | () => void | | | enableMarquee | Enable animation or not. | Boolean | false | | marqueeProps | Marquee parameter, `loop` for loop, `leading` for a pause before animation, `training` for pause between animations when the `loop` is true, `fps` for frame rate. | `Object<loop, leading, trailing, fps>` | {loop: false, leading: 500, trailing: 800, fps: 40 } |

Example

copy

```
{
  "defaultTitle": "AntUI Component Library",
  "usingComponents": {
    "notice": "mini-antui/es/notice/index"
  }
}
```

copy

```
<view class="demo-title">NoticeBar</view>
<view class="demo-item">
  <notice>Due to the upgrade of the national citizenship system, add
bank card </notice>
</view>
<view class="demo-item">
  <notice mode="link" onClick="linkClick">Due to the upgrade of the
national citizenship system, add bank card</notice>
</view>
<view class="demo-item">
  <notice mode="closable" onClick="closableClick" show="
{{closeShow}}">Due to the upgrade of the national citizenship system,
add bank card</notice>
</view>
<view class="demo-item">
  <notice mode="link" action="See details"
onClick="linkActionClick">Due to the upgrade of the national
citizenship system, add bank card</notice>
</view>
<view class="demo-item">
  <notice mode="closable" action="Do not remind again"
onClick="closableActionClick" show="{{closeActionShow}}">Due to the
upgrade of the national citizenship system, add bank card</notice>
</view>
```

copy

```
Page({
  data:{
    closeShow:true,
    closeActionShow:true
  },
  linkClick() {
    my.showToast({
      content: 'Click the icon',
      duration: 3000
    });
  },
  closableClick() {
```

```
    this.setData({
      closeShow:false
    })
    my.showToast({
      content: 'Click the icon',
      duration: 3000
    });
  },
  linkActionClick() {
    my.showToast({
      content: 'Click the text',
      duration: 3000
    });
  },
  closableActionClick() {
    this.setData({
      closeActionShow:false
    })
    my.showToast({
      content: 'Click the text',
      duration: 3000
    });
  }
})
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/extended-component_prompt-guide_notice

---

Obtain basic user information {#obtain-basic-user-information}

Last updated: 2021-05-09

Path: miniprogram_gcash

Obtain basic user information

2021-05-09 18:43

The Mini Program is allowed to obtain basic information of the wallet users. This is an open service which, after a user's authorization, captures basic user information such as avatar, nickname, gender, and region.

Requirements

When obtaining basic user information, mini program developers must meet the following requirements:

- Do not guide users to grant authorization at the launch of the mini program. Users have the right to fully understand the mini program and its operations before giving any authorization.
- Do not obtain the user ID and the user's real name. The information to obtain can only include basic user information, such as user avatar, nickname, gender, and location.
- As the basic user information and the user's mobile phone number are obtained by two JSAPIs, these two kinds of information cannot be requested in the same modal.
- Do not obtain user information that is not related with the business. If the user does not grant authorization on the first request, display the modal to allow the user to reverse the decision when the business requires the authorization again.

Procedures

To obtain the user's basic information in the mini program, mini program developers must complete the following steps:

Step 1: Create a mini program

Apply for an account and create a mini program on the Mini Program platform.

Step 2: Add the feature

In the created mini program, add the feature of obtaining basic user information under the **Features** tab.

Step 3: Call the JSAPI

Call the my.getOpenUserInfo JSAPI to obtain the user's basic information.

Note:

Developers must consider the possibility of users rejecting to grant the mini program authorization to collect their user information. For such cases, developers must have corresponding solutions, such as guiding the user to manually fill in or upload the user information.

Display the authorization modal

In the button component, set the value of *open-type* as `getAuthorize` and set the value of *scope* as `userInfo`.

Sample code:

copy

```
<!-- .axml -->
<button
    open-type="getAuthorize"
    onGetAuthorize="onGetAuthorize"
    onError="onAuthError"
    scope='userInfo'>
</button>
```

Button properties

| | | --- | --- | | **Name** | **Description** | | open-type | The value is `getAuthorize`. | | scope | The value is `userInfo`. | | onGetAuthorize | Authorization success callback. The Mini Program can call my.getOpenUserInfo to get information in this callback. | | onError | Authorization failure callback, including user rejection and system exceptions. |

Call the my.getOpenUserInfo JSAPI

After the user grants the authorization, the user basic information can be obtained by calling the my.getOpenUserInfo JSAPI.

Sample code:

copy

```
// .js
onGetAuthorize(res) {
    my.getOpenUserInfo({
        fail: (res) => {
        },
        success: (res) => {
            let userInfo = JSON.parse(res.response).response
        }
    });
}
```

Sample of a successfully returned message format:

copy

```
{
    "response":{
        "response":{
            "code":"10000",
            "msg":"Success",
            "countryCode":"code",
            "gender":"f",
            "nickName":"XXX",
            "avatar":"https://image_domain/images/partner/XXXXXXXX",
            "city":"city",
            "province":"province"
        }
    }
}
```

FAQs

Can the function of obtaining basic user information obtain the wallet userId?

No. If mini programs need to obtain the user ID, see user authorization for details and call my.getAuthCode to get the user ID.

Can the mini program obtain the user's public information such as mobile phone number, avatar, and nickname at the same time?

Mini programs cannot obtain user's mobile phone number, avatar, nickname at the same time in the same modal.

The following public user information can be obtained with my.getOpenUserInfo:

- Avatar
- Nickname
- Gender
- Country

The following private user information can be obtained with my.getAuthCode:

- User ID
- Phone number

Can mini programs get user ID, real name and private user information through the function of obtaining basic user information?

No. Through the function of obtaining basic user information, mini programs can only get user avatar, nickname, gender, location, and other public information.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/api_openapi_obtainbasicmemberinformation

---

Obtain basic user information {#obtain-basic-user-information}

Last updated: 2022-07-03

Path: miniprogram_gcash

Obtain basic user information

2022-07-03 18:44

The mini program is allowed to obtain basic information of the wallet users. This is an open service that captures basic user information such as avatar, nickname, gender, and region after obtaining the user's authorization.

Requirements

When obtaining basic user information, mini program developers must meet the following requirements:

- Do not guide users to grant authorization at the launch of the mini program. Users have the right to fully understand the mini program and its operations before giving any authorization.
- Do not obtain the user ID and the user's real name. The information to obtain can only include basic user information, such as user avatar, nickname, gender, and location.
- As the basic user information and the user's mobile phone number are obtained by two JSAPIs, these two kinds of information cannot be requested in the same modal.
- Do not obtain user information that is not related to the business. If the user does not grant authorization at the first request, display the modal to allow the user to reverse the decision when the business requires the authorization again.

Procedures

To obtain the user's basic information in the mini program, mini program developers must complete the following steps:

Step 1: Create a mini program

Apply for an account and create a mini program on the Mini Program Platform.

Step 2: Add the feature

Enter the mini program you just created. Click the " **Features**" tab, then " **Add Feature**" to pop up the feature list. Tick "obtain basic member information" and click the "Confirm" button to activate the feature.

Step 3: Call the JSAPI

Call the my.getOpenUserInfo JSAPI to obtain the user's basic information.

Note:

Developers must consider the possibility that users reject to authorize the mini program to collect their information. In such cases, developers must have corresponding solutions, such as guiding users to manually fill in or upload their information.

Display the authorization modal

In the button component, set the value of `open-type` as `getAuthorize` and set the value of `scope` as `userInfo`.

Sample code:

copy

```
<!-- .axml -->
<button
    open-type="getAuthorize"
    onGetAuthorize="onGetAuthorize"
    onError="onAuthError"
    scope='userInfo'>
</button>
```

Button properties

| | | --- | --- | | **Name** | **Description** | | open-type | The value is `getAuthorize`. | | scope | The value is `userInfo`. | | onGetAuthorize | Authorization success callback. The mini program can call `my.getOpenUserInfo` to get information in this callback. | | onError | Authorization failure callback, including user rejection and system errors. |

Call the my.getOpenUserInfo JSAPI

After the user grants the authorization, the mini program can call the my.getOpenUserInfo JSAPI to obtain basic user information.

Sample code:

copy

```
// .js
onGetAuthorize(res) {
    my.getOpenUserInfo({
        fail: (res) => {
        },
        success: (res) => {
            let userInfo = JSON.parse(res.response).response
        }
    });
}
```

Sample of a successfully returned message format:

copy

```
{
    "response":{
        "response":{
            "code":"10000",
            "msg":"Success",
```

```
            "countryCode":"code",
            "gender":"f",
            "nickName":"XXX",
            "avatar":"https://image_domain/images/partner/XXXXXXXX",
            "city":"city",
            "province":"province"
        }
    }
}
```

API list

| | | --- | --- | | **JSAPI** | **Description** | | my.getAuthCode | Gets user's authorization code. | | my.getOpenUserInfo | Gets basic user information. |

FAQs

1. Can the function of obtaining basic user information be used to obtain the wallet `userId` ?

No. If mini programs need to obtain the user ID, see user authorization for details and call my.getAuthCode to get the user ID.

2. Can mini programs obtain the user's mobile phone number, avatar, nickname, and other public information at the same time?

No. Mini programs cannot obtain the user's mobile phone number, avatar, nickname at the same time in the same modal.

The following public user information can be obtained with my.getOpenUserInfo:

- Avatar
- Nickname
- Gender
- Country

The following private user information can be obtained with my.getAuthCode:

- User ID
- Phone number

3. Can mini programs get user ID, real name, and private information through the function of obtaining basic user information?

No. Through the function of obtaining basic user information, mini programs can only get user avatar, nickname, gender, location, and other public information.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_openapi_obtainbasicmemberinformation

---

Overview {#overview}

Last updated: 2022-07-03

Path: miniprogram_gcash

Overview

2022-07-03 18:44

File Structure

The Mini Program is divided into two layers: `app` and `page`. The app describes the whole program; the page describes the individual pages.

The app consists of three files and must be placed in the root directory of the project.

| | | | --- | --- | --- | | **File** | **Mandatory** | **Function** | | app.js | Yes | Mini Program logic. | | app.json | Yes | Mini Program global configuration. | | app.acss | No | Mini Program global style sheet. |

The page consists of four file types:

| | | | --- | --- | --- | | **File type** | **Mandatory** | **Function** | | js | Yes | Page logic. | | axml | Yes | Page structure. | | acss | No | Page style sheet. | | json | No | Page configuration. |

Note: For the convenience of developers, we specify these four files must have the same path and filename. All the codes written by the developer will eventually be packaged into a JavaScript script which runs when Mini Program start and is destroyed when Mini Program finish running.

Logic Structure

The core of Mini Program is a responsive data binding system, composed of the view layer and logic layer. The two layers keep always synchronous. Whenever the data is modified in logic layer, the view layer is updated accordingly.

See the following simple example.

copy

```
<!-- View layer -->
<view> Hello {{name}}! </view>
<button onTap="changeName"> Click me! </button>
```

copy

```
// Logic layer
var initialData = {
```

```
  name: 'AppContainer'
};
// Register a Page.
Page({
  data: initialData,
  changeName(e) {
    // sent data change to view
    this.setData({
      name: 'Mini Program'
    })
  }
});
```

In the above codes, the framework automatically binds the `name` in the logic layer to the `name` in the view layer, so whenever the page is opened, it displays `Hello AppContainer!`

When the user presses the button, the view layer sends the `changeName` event to the logic layer. The logic layer finds the corresponding event handler. The logic layer executes the `setData` operation, changing the `name` from `AppContainer` to `Mini Program`. Since the logic layer and view layer are already bound, the displaying of the view layer automatically changes to `Hello Mini Program!`.

Note: Since the framework does not work in the browser, some web capabilities of JavaScript cannot be used, such as the `document` and `window` objects.

For the logic layer js, the codes can be organized through the es2015 modular syntax:

copy

```
import util from './util'; // Loading relative path
import absolute from '/absolute'; // Loading project root directory
path
```

Reserved Names for Module

Mini Program regards some object names in browser such as `window`, `document` as reserved names for future use. The reserved names include **globalThis**, **global**, **fetch**, **self**, **window**, **document**, **location**, **XMLHttpRequest**. Please do not use these names for module name, or the module can not be used normally. For example:

copy

```
import { window } from './myWindow'
console.log(window) // undefined
```

The above codes show that if using the reserved name as the module name, the imported module will be `undefined`. So you should not use these reserved names or rename the module name by using `as` when importing the module. For example:

copy

```
import { window as myWindow } from './myWindow'
console.log(myWindow)
```

Third-party NPM Module

The Mini Program supports introduction of the third-party module. It is required to firstly run the following command to install the module in the Mini Program root directory:

copy

```
$ npm install lodash --save
```

After the installation, it can be used directly in the logic layer:

copy

```
import lodash from 'lodash'; // Loading the third-party npm module
```

Note: Since the third-party module in the node_modules does not go through the converter, for the compatibility in various terminals, the codes in the node_modules should be converted into the es5 format before using. For the module format, it is recommended to use the import/export of es2015. Meanwhile, the browser related capabilities of the browser cannot be used either.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/framework_overview

---

Overview {#overview}

Last updated: 2021-05-09

Path: miniprogram_gcash

Overview

2021-05-09 18:43

Similar to Page, the customized components consist of four parts: axml, js, json and acss.

There are two steps to create a customized component:

1. Declare the customized component in json. If it is dependent on other components, it is required to declare additionally the dependent customized components.

Sample code:

copy

```
{
  "component": true, // mandate, the value for customized component
must be true
  "usingComponents": {
```

```
    "c1":"../x/index"
  }//Dependent component
}
```

Parameter details:

| | | | | --- | --- | --- | --- | | **Parameter** | **Type** | **Required** | **Description** | | component | Boolean | Yes | Declare customized component. | | usingComponents | Object | No | Path of the customized component in the dependence declaration Absolute project path starts with “/”, and relative path starts with “./” or “../” The npm path does not start with “/”. |

1. Use the Component function to register the customized component. See Component constructor 。

Component parameter description:

| | | | | --- | --- | --- | | **Parameter** | **Description** | **Document** | | onInit | Callback on creation | Component lifecycle. | | deriveDataFromProps | Callback on creation and update | Component lifecycle. | | data | local status | Same as Page (can be modified via setData and $spliceData). | | props | Attribute transferred from outside | Component method and external attribute-props. | | methods | Customized method | Component method and external attribute - methods. |

Sample code:

copy

```
// /components/customer/index.js
Component({
  mixins: [], // minxin easy reuse code
  data: { x: 1 }, // internal data of component
  props: { y: 1 }, // Can add default to attribute transferred from
outside
  onInit() {}, // trigger on component creation, added in version
2.0.0
  deriveDataFromProps(nextProps) {}, // trigger on component creation
and before update, added in version 2.0.0
  didMount(){}, // Lifecycle function
  didUpdate(){},
  didUnmount(){},
  methods: { // customized method
    handleTap() {
      this.setData({ x: this.data.x + 1}); // Can use setData to
change internal attribute
    },
  },
})
```

in addition, the customized component supports slot and can build flexible page structure. See component template and style .

Sample code:

copy

```
<!-- // /components/customer/index.axml -->
<view>
  <view>x: {{x}}</view>
  <button onTap="handleTap">plusOne</button>
  <slot>
    <view>default slot & default value</view>
  </slot>
</view>
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_custom-component_create-custom-component_overview

---

Overview {#overview}

Last updated: 2022-07-05

Path: miniprogram_gcash

Overview

2022-07-05 23:31

Mini programs offer a set of OpenAPIs to achieve various capabilities, for example, the payment capability. You can use the POST method to send HTTPS requests and receive responses accordingly.

The following section mainly introduces the message structure and the end-to-end message transmission workflow.

Versioning

The current API version is `v1`. The version is specified in the URL.For example:

`https://{host}/miniprogram/api/v1/payments/pay`.

Message structure

Before you make any request, it is important to understand how OpenAPI works and how requests and responses are structured. This section presents general information (such as message structure, message fields, and message transmission) of online messages between your system and wallet backend.

Request structure

The following figure illustrates the request structure.

Figure 1. Request structure

Request URL

The request URL is: `https://{host}/api/{restful_path}`, which has the following structure:

- **host**: includes the host that is the standard domain name assigned by the wallet backend.
- **resful_path :** is the path to the interface, for example, `/{version}/payments/pay`
- **version**: is the version of OpenAPIs, for example, `v1` or `v2`.

An interface can be uniquely identified by `restful_path`.For example, the `/v1/payments/pay` is different from `/v2/payments/pay`.

Request method

POST method is used to make an HTTP request.

Request header

The request header mainly contains the following fields.

Note: Field names are case-insensitive.

| | | | --- | --- | --- | | **Header** | **Required** | **Code sample** | | Signature | Yes | `Signature: algorithm=RSA256, keyVersion=1, signature=****` | | Content-Type | No | `Content-Type: application/json; charset=UTF-8` | | Client-Id | Yes | `Client-Id: ****` | | Request-Time | Yes | `Request-Time: 2019-04-04T12:08:56.253+05:30` |

Table. Request header

For details of each header field, see the following description.

Signature

Signature contains key-value pairs that are separated by comma (,). Each key-value pair is an equation, which is a key joined with its value with an equal sign (=).

The following keys can be configured:

- **algorithm**: Specify the digital signature algorithm that is used to generate the signature. The value is not case-sensitive. RSA256 and ECC224 are supported, and RSA256 is by default.
- **keyVersion**: Specify the key version that is used to generate or validate the signature. By default, the value is the latest version of the key associated with **Client-Id**.

- **signature**: Contain the signature value of the request. For details about how to generate a signature, see the Generate a signature section.

Example:

copy

```
Signature: algorithm=RSA256, keyVersion=1,
signature=KEhXthj4bJ801Hqw8kaLvEKc0Rii8KsNUazw7kZgjxyGSPuOZ48058UVJUkkF
rWiHPae8ZRPuBagh2H3qu7fxY5GxVDWayJUhUYkr9m%2FOW4UQVmXaQ9yn%2Fw2dCtzwAWC
pMk%2BfDDmRflA%2FAMJhQ71yeyhufIA2PCJV8%2FCMOa46303A0WHhH0YPJ9%2FI0UeLVM
bRFvcowQwt0lP1XkoPmSLGpBevDE8%2FQ9WnxjPNDfrHnKgV2fp0hpMKVXNM%2BrLHNyMv3
2FFYDAwSd%2B6%2FEOFo9UbdlKcmodJwjKlQoxZZIzmF8w%3D%3Dxxxx
```

Content-Type (Optional)

Content-Type indicates the media type of the body of the request, as defined by RFC2616. In which, **charset** is used for generating/validating the signature.

For example:

copy

```
Content-Type: application/json; charset=UTF-8
```

Client-Id

Client-Id is used to identify a client and is associated with the keys that are used for the signature.

Also in Mini Program OAuth scenario, here the **Client-Id** is also the client ID of OAuth, which is filled in the Mini Program SAAS Platform as follow:

Request-Time

Specify the time when the request is sent, as defined by ISO8601.

Note: This field must be accurate to milliseconds.

copy

```
Request-Time: 2019-04-04T12:08:56.253+05:30
```

Request body

The request body contains the detailed request information in JSON format. Fields enclosed in the request body vary depending on services. For more information, see the specific API specification document.

Response structure

The following figures illustrate the response structure:

Figure 2. Response structure

Response header

The response header carries the information about the response, mainly containing the following fields.

Note: Field names are case-insensitive.

| | | | | --- | --- | --- | | **Header** | **Required** | **Code sample** | | Signature | Yes | `Signature: algorithm=RSA256, keyVersion=1, signature=****` | | Content-Type | No | `Content-Type: application/json; charset=UTF-8` | | Client-Id | Yes | `Client-Id: ****` | | Response-Time | Yes | `Response-Time: 2019-04-04T12:08:56.253+05:30` |

Table. Response header

For details of each header field, see the following description.

Signature

Signature contains key-value pairs that are separated by comma (,). Each key-value pair is an equation, which is a key joined with its value with an equal sign (=).

The following keys can be configured:

- **algorithm**: Specifies the digital signature algorithm that is used to generate the signature. The value is not case-sensitive. RSA256 and ECC224 are supported, and RSA256 is by default.
- **keyVersion**: Specifies the key version that is used to generate or validate the signature. By default, the value is the latest version of the key associated with **Client-Id**.
- **signature**: Contains the signature value of the response.

Example:

copy

```
Signature: algorithm=RSA256, keyVersion=1,
signature=KEhXthj4bxxxJ801Hqw8kaLvEKc0Rii8KsNUazw7kZgjxyGSPuOZ48058UVJ
rWiHPae8ZRPuBagh2H3qu7fxY5GxVDWayJUhUYkr9m%2FOW4UQVmXaQ9yn%2Fw2dCtzwAW
pMk%2BfDDmRflA%2FAMJhQ71yeyhufIA2PCJV8%2FCMOa46303A0WHhH0YPJ9%2FI0UeLV
bRFvcowQwt0lP1XkoPmSLGpBevxxxDE8%2FQ9WnxjPNDfrHnKgV2fp0hpMKVXNM%2BrLHN
2FFYDAwSd%2B6%xxxx
```

Content-Type (Optional)

Content-Type indicates the media type of the body of the response, as defined by RFC2616. In which, **charset** is used for generating/validating the signature. For example:

copy

```
Content-Type: application/json; charset=UTF-8
```

Client-Id

Client-Id is used to identify a client and is associated with the keys that are used for signature.

Response-Time

Specifies the time when the response is returned, as defined by ISO8601.

Note: This field must be accurate to milliseconds.

copy

```
Response-Time: 2019-04-04T14:08:56.253+05:30
```

Response body

The response body contains the information responding to the client. Fields in this section vary depending on services. However, the `result` object, which indicates the result of an API call, is always contained.

When the result status (`resultStatus`) is failed, unknown, or accepted, the result code (`resultCode`) means an error code and the result message (`resultMessage`) means an error message, which is used for error handling. For more information about error codes, see the Error codes chapter.

| | | | | --- | --- | --- | --- | | **Field** | **Data Type** | **Required** | **Description** | | resultStatus | String | No | Result status. Valid values are:
- `S` : Successful
- `F` : Failed
- `U` : Unknown
- `A` : Accepted, not yet succeeded, but can proceed with some actions. | | resultCode | String | No | Result code.
Maximum length: 64 characters | | resultMessage | String | No | Result message that describes the result code in detail.
Maximum length: 256 characters |

Message transmission workflow

The following figure is an example that illustrates the message transmission workflow in Mini Program.

Figure 1. Message transmission workflow

Overall procedure

Follow the overall procedure to call an API.

Preparations

To prevent some potential errors that you might get in the response, consider the following factors:

- Understand API idempontency

1. Construct a request

Construct a request by complying with the request structure, including the request header and body.

To ensure message transmission security, perform the following security measures when constructing a request. For details, see the Message transmission security chapter.

1. Must sign a request. Message signing and signature validation are mandatory for all requests and responses.
2. Encode a request to prevent errors or ambiguity that might be caused by special characters enclosed in a request. For more information, see the Message encoding chapter.

2. Send a request

You can send a request with your preferred platforms or tools, for example, Postman or cURL command.

3. Check the response

The response is usually returned in JSON or XML format. For details about the response, see the response structure section. After you receive the response, validate the signature of the response.

4. Check the status code

The response data can vary depending on the services. However, the `result` field, which indicates the result of an API call, is always included. If an error occurs when you call an API, an error response is returned, where the `result` object indicates the error code and error message for you to troubleshoot issues. For more information, see the Error code chapter.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/openapi_overview

---

Overview {#overview}

Last updated: 2021-05-09

Path: miniprogram_gcash

Overview

2021-05-09 18:43

File Structure

The Mini Program is divided into two layers: `app` and `page`. The app describes the whole program; the page describes the individual pages.

The app consists of three files and must be placed in the root directory of the project.

| | | | --- | --- | --- | | **File** | **Mandatory** | **Function** | | app.js | Yes | Mini Program logic. | | app.json | Yes | Mini Program global configuration. | | app.acss | No | Mini Program global style sheet. |

The page consists of four file types:

| | | | --- | --- | --- | | **File type** | **Mandatory** | **Function** | | js | Yes | Page logic. | | axml | Yes | Page structure. | | acss | No | Page style sheet. | | json | No | Page configuration. |

Note: For the convenience of developers, we specify these four files must have the same path and filename. All the codes written by the developer will eventually be packaged into a JavaScript script which runs when Mini Program start and is destroyed when Mini Program finish running.

Logic Structure

The core of Mini Program is a responsive data binding system, composed of the view layer and logic layer. The two layers keep always synchronous. Whenever the data is modified in logic layer, the view layer is updated accordingly.

See the following simple example.

copy

```
<!-- View layer -->
<view> Hello {{name}}! </view>
<button onTap="changeName"> Click me! </button>
```

copy

```
// Logic layer
var initialData = {
  name: 'AppContainer'
};
// Register a Page.
Page({
  data: initialData,
  changeName(e) {
    // sent data change to view
    this.setData({
      name: 'Mini Program'
    })
  }
});
```

In the above codes, the framework automatically binds the `name` in the logic layer to the `name` in the view layer, so whenever the page is opened, it displays `Hello AppContainer!`

When the user presses the button, the view layer sends the `changeName` event to the logic layer. The logic layer finds the corresponding event handler. The logic layer executes the `setData` operation, changing the `name` from `AppContainer` to `Mini Program`. Since the logic layer and view layer are already bound, the displaying of the view layer automatically changes to `Hello Mini Program!`.

Note: Since the framework does not work in the browser, some web capabilities of JavaScript cannot be used, such as the `document` and `window` objects.

For the logic layer js, the codes can be organized through the es2015 modular syntax:

copy

```
import util from './util'; // Loading relative path
import absolute from '/absolute'; // Loading project root directory
path
```

Reserved Names for Module

Mini Program regards some object names in browser such as `window`, `document` as reserved names for future use. The reserved names include **globalThis**, **global**, **fetch**, **self**, **window**, **document**, **location**, **XMLHttpRequest**. Please do not use these names for module name, or the module can not be used normally. For example:

copy

```
import { window } from './myWindow'
console.log(window) // undefined
```

The above codes show that if using the reserved name as the module name, the imported module will be `undefined`. So you should not use these reserved names or rename the module name by using `as` when importing the module. For example:

copy

```
import { window as myWindow } from './myWindow'
console.log(myWindow)
```

Third-party NPM Module

The Mini Program supports introduction of the third-party module. It is required to firstly run the following command to install the module in the Mini Program root directory:

copy

```
$ npm install lodash --save
```

After the installation, it can be used directly in the logic layer:

copy

```
import lodash from 'lodash'; // Loading the third-party npm module
```

Note: Since the third-party module in the node_modules does not go through the converter, for the compatibility in various terminals, the codes in the node_modules should be converted into the es5 format before using. For the module format, it is recommended to use the import/export of es2015. Meanwhile, the browser related capabilities of the browser cannot be used either.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/framework_overview

---

Overview {#overview}

Last updated: 2021-05-09

Path: miniprogram_gcash

Overview

2021-05-09 18:43

Mini Programs offers a set of OpenAPIs to achieve various capabilities, for example, the payment capability. You can use POST method to send HTTPS requests and receive response accordingly.

Message structure

Before you make any request, it is important to understand how OpenAPI works and how requests and responses are structured. This section presents general information (such as message structure, message fields, and message transmission) of online message between your system and wallet backend.

Request structure

The following figure illustrates the request structure.

Figure 1. Request structure

Request URL

The request URL is: `https://{host}/api/{restful_path}`

where,

- **host:** includes the host that is the standard domain name assigned by the wallet backend.
- **resful_path :** is the path to the interface, for example, `/{version}/payments/pay`
- **version**: is the version of OpenAPIs, for example, `v1` or `v2`.

An interface can be uniquely identified by `restful_path` .For example, the `/v1/payments/pay` is different from `/v2/payments/pay`.

Request method

POST method is used to make an HTTP request.

Request header

The request header mainly contains the following fields.

Tip: Field names are case-insensitive.

| | | | --- | --- | --- | | **Header field** | **Required** | **Code sample** | | Signature | Yes | `Signature: algorithm=RSA256, keyVersion=1, signature=****` | | Encrypt | No (It is used only when data includes sensitive information) | `Encrypt: algorithm=RSA_AES, keyVersion=1, symmetricKey=****` | | Content-Type | No | `Content-Type: application/json; charset=UTF-8` | | Client-Id | Yes | `Client-Id: ****` | | Request-Time | Yes | `Request-Time: 2019-04-04T12:08:56.253+05:30` |

Table. Request header

For details of each header field, see the following description.

Signature

Signature contains key-value pairs that are separated by comma (,). Each key-value pair is an equation, which is a key joined with its value with an equal sign (=).

The following keys can be configured:

- **algorithm**: Specifies the digital signature algorithm that is used to generate the signature. The value is not case-sensitive. RSA256 and ECC224 are supported, and RSA256 by default.
- **keyVersion**: Specifies the key version that is used to generate or validate the signature. By default, the value is the latest version of the key associated with **Client-Id**.
- **signature**: Contains the signature value of the request. For details about how to generate a signature, see the **Generate a signature** section.

Example:

copy

```
Signature: algorithm=RSA256, keyVersion=1,
signature=KEhXthj4bJ801Hqw8kaLvEKc0Rii8KsNUazw7kZgjxyGSPuOZ48058UVJUkkF
rWiHPae8ZRPuBagh2H3qu7fxY5GxVDWayJUhUYkr9m%2FOW4UQVmXaQ9yn%2Fw2dCtzwAW(
pMk%2BfDDmRflA%2FAMJhQ71yeyhufIA2PCJV8%2FCMOa46303A0WHhH0YPJ9%2FI0UeLVM
bRFvcowQwt0lP1XkoPmSLGpBevDE8%2FQ9WnxjPNDfrHnKgV2fp0hpMKVXNM%2BrLHNyMv3
2FFYDAwSd%2B6%2FEOFo9UbdlKcmodJwjKlQoxZZIzmF8w%3D%3Dxxxx
```

Encrypt

This field is required when a message need to be encrypted, especially when sensitive information is included in the message. **Encrypt** contains key-value pairs that are separated by comma (,). Each key-value pair is an equation, which is a key joined with its value with an equal sign (=).

The following keys can be configured:

• **algorithm**: Specifies the symmetric key algorithm that is used to encrypt message. The value is not case-sensitive, and currently only RSA_AES is supported.

• **keyVersion**: Specifies the symmetric key version that is used to encrypt message. By default, the value is the latest version of the key associated with clientId.

• **symmetricKey**: Contains the encrypted symmetric key.

For example:

copy

```
Encrypt: algorithm=RSA_AES, keyVersion=1,
symmetricKey=bqS8HSmdaRrpKSuPy7CqUlyd8lJurG93xxxx
```

Content-Type

Optional. **Content-Type** indicates the media type of the body of the request, as defined by RFC2616. In which, **charset** is used for generating/validating signature and encrypting/decrypting content.

For example:

copy

```
Content-Type: application/json; charset=UTF-8
```

Client-Id

Client-Id is used to identify a client, and is associated with the keys that are used for signature and encryption.

Also in Mini Program OAuth scenario, here the**Client-Id** is also the client id of OAuth, which is filled in Mini Program SAAS platform as follow:

Request-Time

Specifies the time when the request is sent, as defined by RFC3339. Note: This field must be accurate to milliseconds.

copy

```
Request-Time: 2019-04-04T12:08:56.253+05:30
```

Request body

The request body contains the detailed request information in a JSON format. Fields enclosed in the request body vary depending on services. For more information, see the specific API specification.

Response structure

The following figures illustrate the response structure:

Figure 2. Response structure

Response header

The response header carries the information about the response, mainly containing the following fields.

Tip: Field names are case-insensitive.

| | | | --- | --- | --- | | **Header field** | **Required** | **Code sample** | | Signature | Yes | `Signature: algorithm=RSA256, keyVersion=1, signature=****` | | Encrypt | No (It is used only when data includes sensitive information) | `Encrypt: algorithm=RSA_AES, keyVersion=1, symmetricKey=****` | | Content-Type | No | `Content-Type: application/json; charset=UTF-8` | | traceId | No | `traceId: 0be9923****` | | Client-Id | Yes | `Client-Id: ****` | | Response-Time | Yes | `Response-Time: 2019-04-04T12:08:56.253+05:30` |

Table. Response header

For details of each header field, see the following description.

Signature

Signature contains key-value pairs that are separated by comma (,). Each key-value pair is an equation, which is a key joined with its value with an equal sign (=).

The following keys can be configured:

- **algorithm**: Specifies the digital signature algorithm that is used to generate the signature. The value is not case-sensitive. RSA256 and ECC224 are supported, and RSA256 by default.
- **keyVersion**: Specifies the key version that is used to generate or validate the signature. By default, the value is the latest version of the key associated with **Client-Id**.
- **signature**: Contains the signature value of the response.

Example:

copy

```
Signature: algorithm=RSA256, keyVersion=1,
signature=KEhXthj4bxxxJ801Hqw8kaLvEKc0Rii8KsNUazw7kZgjxyGSPuOZ48058UVJ
rWiHPae8ZRPuBagh2H3qu7fxY5GxVDWayJUhUYkr9m%2FOW4UQVmXaQ9yn%2Fw2dCtzwAW
pMk%2BfDDmRflA%2FAMJhQ71yeyhufIA2PCJV8%2FCMOa46303A0WHhH0YPJ9%2FI0UeLV
bRFvcowQwt0lP1XkoPmSLGpBevxxxDE8%2FQ9WnxjPNDfrHnKgV2fp0hpMKVXNM%2BrLHN
2FFYDAwSd%2B6%xxxx
```

Encrypt

This field is required when a response needs to be encrypted. **Encrypt** contains key-value pairs that are separated by comma (,). Each key-value pair is an equation, which is a key joined with its value with an equal sign (=).

The following keys can be configured:

• **algorithm**: Specifies the symmetric key algorithm that is used to encrypt a message. The value is not case-sensitive, and currently only RSA_AES is supported.

• **keyVersion**: Specifies the symmetric key version that is used to encrypt a message. By default, the value is the latest version of the key associated with clientId.

• **symmetricKey**: Contains the encrypted symmetric key.

For example:

copy

```
Encrypt: algorithm=RSA_AES, keyVersion=1,
symmetricKey=bqS8HSmdaRrpKSuPy7CqUlyd8lJurG93xxxx
```

Content-Type

Optional. **Content-Type** indicates the media type of the body of the response, as defined by RFC2616. In which, **charset** is used for generating/validating the signature and encrypting/decrypting content.

For example:

copy

```
Content-Type: application/json; charset=UTF-8
```

traceId

The **traceId** field is used for troubleshooting when there is something wrong with a request processing. For example, use **traceId** to identify the specific request that has issues.

Client-Id

Client-Id is used to identify a client, and is associated with the keys that are used for signature and encryption.

Response**-Time**

Specifies the time when the response is returned, as defined by RFC3339. Note: This field must be accurate to milliseconds.

copy

```
Response-Time: 2019-04-04T14:08:56.253+05:30
```

Response body

Response body contains the information responding to the client. Fields in this section vary depending on services. However, the `result` object, which indicates the result of an API call, is always contained.

When the result status (`resultStatus`) is failed, unknown, or accepted, the result code (`resultCode`) means an error code and the result message (`resultMessage`) means an error message, which is used for error handling. For more information about error codes, see the Error codes chapter.

| | | | | --- | --- | --- | --- | | **Field** | **Data type** | **Required** | **Description** | | resultStatus | String | No | Result status. Valid values are:
- `S` : Successful
- `F` : Failed
- `U` : Unknown
- `A` : accepted, not yet succeed, but can proceed with some actions. | | resultCode | String | No | Result code.

Max. length: 64 characters | | resultMessage | String | No | Result message that describes the result code in details.
Max. length: 256 characters |

Message transmission workflow

The following figure is an example that illustrates the message transmission workflow in Mini Program.

Figure 1. Message transmission workflow

Overall procedure

Follow the overall procedure to call an API.

Preparations

To prevent some potential errors that you might get in the response, consider the following factors:

- Understand API idempontency

1. Construct a request

Construct a request by complying with therequest structure, including the request header and body.

To ensure the message transmission security, perform the following security measures when constructing a request. For details, see the Message transmission security chapter.

1. Encrypt a request when the data includes sensitive information or it is required by clients. If encryption is required, the message body should be encrypted before it is signed.
2. Must sign a request. Message signing and signature validation is mandatory for all requests and responses.
3. Encode a request to prevent errors or ambiguity that might be caused by special charaters enclosed in a request. For more information, see the Message encoding chapter.

2. Send a request

You can send a request for example via Postman or cURL command.

3. Check the response

The response is returned usually in JSON or XML format. For details about the response, see the Response structure section.

After you receive the response, perform the following actions:

1. Validate the signature of the response.
2. Decrypt the response if the request is encrypted.

4. Check the status code

If an error occurs when you call an API, an error response is returned, where the `result` object indicates the error code and error message for you to troubleshoot issues.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/xxpbkg

---

Overview {#overview}

Last updated: 2022-07-03

Path: miniprogram_gcash

Overview

2022-07-03 18:44

Basic Component

The Mini Program provides the developers with a series of basic components so that the developers can combine them for service development.

Attribute Type

The component provides a series of attribute configuration. Each attribute value has the requirement for type:

| | | | --- | --- | --- | | **Type** | **Description** | **Notes** | | Boolean | Boolean | | | Number | Number | | | String | String | | | Array | Array | | | Object | Object | | | EventHandle | Event handler | Need to define the implementation for the event handler in Page . | | any | Any type | |

Common Component Attribute

All components include the following attributes:

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Description** | **Notes** | | id | String | Unique component identifier. | | | class | String | Style class. | | | style | String | Inline style. | | | data-* | Any | Custom attributes. | When the event is triggered, the custom attribute is transferred to the event handler. | | on *catch* | EventHandle | Event binding, following the ump nomenclature specifications, such as onTap. | Refer to event
. |

Tips

The {{}} is required to transfer inside the specified attribute type data. For example

copy

```
<view disable-scroll="false"> <!-- Error is a string, not a boolean, equivalent to boolean type true -->
<view disable-scroll="{{false}}"> <!--right, or empty attribute, meaning false-->
```

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/component_overview

---

Overview {#overview}

Last updated: 2021-05-09

Path: miniprogram_gcash

Overview

2021-05-09 18:43

API

The framework provides the developers with more JSAPI and OpenAPI capabilities so that they can launch diversified convenient services to the users.

Notes:

The APIs started with my.on are used to listen to the system events and accept one callback function as the parameter. When the event is triggered, it calls the callback function, which will transfer to the related API started with my.off to cancel the listening