In the codes shown above, both the `a.js` and `b.js` have declared the variable localValue, but they will not affect each other, because the variable and function of each script take effect only in their own scope.

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

---

getApp {#getapp}

Last updated: 2021-05-09

Path: miniprogram_gcash

getApp

2021-05-09 18:43

A global `getApp()` function is available for obtaining the instance of currently running Mini Program. This is generally used in `page` to get the top-level `app`.

copy

```
var app = getApp()
console.log(app.globalData) // Get globalData
```

Note:

- Do not call `getApp()` in `App()`. Instead, use `this` to get the app instance.
- After the instance is obtained with getApp(), do not call the lifecycle function of `App`.
- Please distinguish `App` global data and `Page` global data.

The global data can be set in App(). The individual sub-pages can get the global application instance through the global function `getApp()`. Here is an example.

copy

```
// app.js
App({
  globalData: 1
})
```

copy

```
// a.js
// localValue effective only in a.js
var localValue = 'a'
// generating app instance
var app = getApp()
```

```
// get global data and change it
app.globalData++
```

copy

```
// b.js
// localValue effective only in b.js
var localValue = 'b'
// if a.js runs first, the globalData returns 2
console.log(getApp().globalData)
```

In the codes shown above, both the `a.js` and `b.js` have declared the variable localValue, but they will not affect each other, because the variable and function of each script take effect only in their own scope.

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

---

getCurrentPages {#getcurrentpages}

Last updated: 2021-05-10

Path: miniprogram_gcash

getCurrentPages

2021-05-10 03:43

`getCurrentPages()` is used to get the instance of the page stack, it will return an array of page. The first element is the home page, and the last element is the current page.

The framework maintains the current pages by stack. And the relationship of routing switch and page stack is shown in the following table.

| | | | | --- | --- | | **Routing** | **Page Stack Behavior** | | Initialization | Push new page into stack. | | Open a new page | Push new page into stack. | | Redirect of page | Pop current page from stack and push new page into stack. | | Page returned | Pop current page from stack. | | Tab switch | Pop all pages from stack except the new tab page. |

Following codes can help to detect whether current stack reaches 5 layer of pages.

copy

```
if (getCurrentPages().length === 5) {
    my.redirectTo({
        url: '/pages/logs/logs'
    });
} else {
    my.navigateTo({
        url: '/pages/index/index'
    });
}
```

Note: do not try to modify the page stack, or error about page routing and page status may happen.

FAQ

Q: How to get the path of current page by `getCurrentPages()`?

A: `JSON.stringify(getCurrentPages()[0].__proto__.route)` can get the path of current page.

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

---

getCurrentPages {#getcurrentpages}

Last updated: 2022-07-03

Path: miniprogram_gcash

getCurrentPages

2022-07-03 18:44

`getCurrentPages()` is used to get the instance of the page stack, it will return an array of page. The first element is the home page, and the last element is the current page.

The framework maintains the current pages by stack. And the relationship of routing switch and page stack is shown in the following table.

| | | | --- | --- | | **Routing** | **Page Stack Behavior** | | Initialization | Push new page into stack. | | Open a new page | Push new page into stack. | | Redirect of page | Pop current page from stack and push new page into stack. | | Page returned | Pop current page from stack. | | Tab switch | Pop all pages from stack except the new tab page. |

Following codes can help to detect whether current stack reaches 5 layer of pages.

copy

```
if (getCurrentPages().length === 5) {
    my.redirectTo({
        url: '/pages/logs/logs'
    });
} else {
    my.navigateTo({
        url: '/pages/index/index'
    });
}
```

Note: do not try to modify the page stack, or error about page routing and page status may happen.

FAQ

Q: How to get the path of current page by `getCurrentPages()`?

A: `JSON.stringify(getCurrentPages()[0].__proto__.route)` can get the path of current page.

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

---

icon {#icon}

Last updated: 2021-05-09

Path: miniprogram_gcash

icon

2021-05-09 18:43

Icon.

Scan QR code to try:

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Default** | **Description** | | type | String | | Icon type, effective value: info, warn, waiting, cancel, download, search, clear, success, success_no_circle,loading. | | size | Number | 23 | Icon size, in px. | | color | Color | | Icon color, same as css color. |

Screenshot

Sample Code

copy

```
<block a:for="{{iconType}}">
  <view class="item">
    <icon type="{{item}}" aria-label="{{item}}" size="45"/>
    <text>{{item}}</text>
  </view>
</block>
<block a:for="{{iconSize}}">
  <view class="item">
    <icon type="success" size="{{item}}"/>
    <text>{{item}}</text>
  </view>
</block>
<block a:for="{{iconColor}}">
  <view class="item">
    <icon type="success" size="45" color="{{item}}"/>
    <text style="color:{{item}}">{{item}}</text>
  </view>
</block>
```

copy

```
Page({
  data: {
    iconSize: [20, 30, 40, 50, 60],
    iconColor: [\
      'red', 'yellow', 'blue', 'green'\
    ],
    iconType: [\
      'success',\
      'info',\
      'warn',\
      'waiting',\
      'clear',\
      'success_no_circle',\
      'download',\
      'cancel',\
      'search',\
    ]
  }
})
```

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

---

icon {#icon}

Last updated: 2022-07-03

Path: miniprogram_gcash

icon

2022-07-03 18:44

Icon.

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Default** | **Description** | | type | String | | Icon type, effective value: info, warn, waiting, cancel, download, search, clear, success, success_no_circle,loading. | | size | Number | 23 | Icon size, in px. | | color | Color | | Icon color, same as css color. |

Screenshot

Sample Code

copy

```
<block a:for="{{iconType}}">
  <view class="item">
    <icon type="{{item}}" aria-label="{{item}}" size="45"/>
    <text>{{item}}</text>
  </view>
</block>
<block a:for="{{iconSize}}">
  <view class="item">
    <icon type="success" size="{{item}}"/>
    <text>{{item}}</text>
  </view>
</block>
<block a:for="{{iconColor}}">
  <view class="item">
    <icon type="success" size="45" color="{{item}}"/>
    <text style="color:{{item}}">{{item}}</text>
  </view>
</block>
```

copy

```
Page({
  data: {
    iconSize: [20, 30, 40, 50, 60],
    iconColor: [\
      'red', 'yellow', 'blue', 'green'\
    ],
    iconType: [\
      'success',\
      'info',\
      'warn',\
      'waiting',\
      'clear',\
      'success_no_circle',\
      'download',\
      'cancel',\
      'search',\
    ]
  }
})
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/component_basic-content_icon

---

image {#image}

Last updated: 2022-07-03

Path: miniprogram_gcash

image

2022-07-03 18:44

Image

Property	**Type**	**Default**	**Description**
src	String		Image address.
mode	String	scaleToFill	Image mode.
class	String	external style	
style	String	Inline style	
onLoad	EventHandle		Trigger upon image loading completion, event object event.detail = {height:'image height px', width:'image width px'}.
onError	EventHandle		Trigger on image loading error, event object event.detail = {errMsg: 'something wrong'}.
onTap	EventHandle		Triggered when clicking on an image, and pass click events to the parent component.
catchTap	EventHandle		Triggered when clicking on an image， and do not pass click events to the parent component.

Note: The image component has default width 300px and height 225px

Mode

There are 13 modes, 4 of which are scaling mode and 9 are cropping mode.

Scaling Mode

Property	**Description**
scaleToFill	Scale without aspect ratio and stretch image width to fill the image element.
aspectFit	Scale with aspect ratio and show fully long side In other words, the whole image is displayed in full.
aspectFill	Scale with aspect ratio and ensure short side to be displayed fully. In other words, the image is complete in horizontal or vertical direction, and the other direction is cropped.
widthFix	Width not changed and height changed automatically with aspect ratio unchanged.

Cropping Mode

| | | | --- | --- | | **Property** | **Description** | | top | Not scaling image, showing only top area. | | bottom | Not scaling image, showing only bottom area. | | center | Not scaling image, showing only central area. | | left | Not scaling image, showing only left area. | | right | Not scaling image, showing only right area. | | top left | Not scaling image, showing only top left area. | | top right | Not scaling image, showing only top right area. | | bottom left | Not scaling image, showing only bottom left area. | | bottom right | Not scaling image, showing only bottom right area. |

Note: The image height cannot be set as auto. If the image height has to be auto, just set mode as widthFix.

Sceenshot

Original Image

scaleToFill

Fit image completely without maintaining aspect ratio

aspectFit

Scale with aspect ratio and show fully long side

aspectFill

Scale with aspect ratio and ensure short side to be displayed fully.

widthFix

Width not changed and height changed automatically with aspect ratio unchanged

top

Not scaling image and showing only top area

bottom

Not scaling image and showing only bottom area

center

Not scaling image and showing only central area

left

Not scaling image and showing only left area

right

Not scaling image and showing only right area

top left

Not scaling image and showing only top left area

top right

Not scaling image and showing only right top area

bottom left

Not scaling image and showing only bottom left area

bottom right

Not scaling image and showing only bottom right area

Sample Code

copy

```
<view class="section" a:for="{{array}}" a:for-item="item">
  <view class="title">{{item.text}}</view>
  <image style="background-color: #eeeeee; width: 300px;
height:300px;" mode="{{item.mode}}" src="{{src}}" onError="imageError"
onLoad="imageLoad" />
</view>
```

copy

```
Page({
  data: {
    array: [{\
      mode: 'scaleToFill',\
      text: 'scaleToFill: scale without aspect ratio and fit image
completely’\
    }, {\
      mode: 'aspectFit',\
      text: 'aspectFit: scale with aspect ratio and show fully long
side’\
    }, {\
      mode: 'aspectFill',\
      text: 'aspectFill: scale with aspect ratio and ensure short side
to be displayed fully.’\
    }, {\
      mode: 'top',\
      text: 'top: Not scaling image, showing only top area’\
    }, {\
      mode: 'bottom',\
      text: 'bottom: Not scaling image, showing only bottom area’\
    }, {\
      mode: 'center',\
      text: 'center: Not scaling image, showing only central area’\
    }, {\
      mode: 'left',\
      text: 'left: Not scaling image, showing only left area’\
    }, {\
      mode: 'right',\
      text: ‘right: Not scaling image, showing only right area’\
    }, {\
      mode: 'top left',\
      text: ‘top left: Not scaling image, showing only top left area’\
    }, {\
      mode: 'top right',\
      text: ‘top right: Not scaling image, showing only top right
area’\
    }, {\
      mode: 'bottom left',\
      text: ‘bottom left: Not scaling image, showing only bottom left
area’\
    }, {\
      mode: 'bottom right',\
      text: ‘bottom right: Not scaling image, showing only bottom
right area’\
    }],
    src: './2.png'
  },
  imageError: function (e) {
    console.log('image3 error happened', e.detail.errMsg)
  },
  imageLoad: function (e) {
    console.log('image loaded successfully', e);
  }
})
```

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

---

image {#image}

Last updated: 2021-05-09

Path: miniprogram_gcash

image

2021-05-09 18:43

Image

Scan QR code to try:

Property	**Type**	**Default**	**Description**
src	String		Image address.
mode	String	scaleToFill	Image mode.
class	String	external style	
style	String	Inline style	
onLoad	EventHandle		Trigger upon image loading completion, event object event.detail = {height:'image height px', width:'image width px'}.
onError	EventHandle		Trigger on image loading error, event object event.detail = {errMsg: 'something wrong'}.
onTap	EventHandle		Triggered when clicking on an image, and pass click events to the parent component.
catchTap	EventHandle		Triggered when clicking on an image， and do not pass click events to the parent component.

Note: The image component has default width 300px and height 225px

Mode

There are 13 modes, 4 of which are scaling mode and 9 are cropping mode.

Scaling Mode

Property	**Description**
scaleToFill	Scale without aspect ratio and stretch image width to fill the image element.
aspectFit	Scale with aspect ratio and show fully long side In other words, the whole image is displayed in full.
aspectFill	Scale with aspect ratio and ensure short side to be displayed fully. In other words, the image is complete in horizontal or vertical direction, and the other direction is cropped.
widthFix	Width not changed and height changed automatically with aspect ratio unchanged.

Cropping Mode

Property	**Description**
top	Not scaling image, showing only top area.
bottom	Not scaling image, showing only bottom area.
center	Not scaling image, showing only central area.
left	Not scaling image, showing only left area.
right	Not scaling image, showing only right area.
top left	Not scaling image, showing only top left area.
top right	Not scaling image, showing only top right area.
bottom left	Not scaling image, showing only bottom left area.
bottom right	Not scaling image, showing only bottom right area.

Note: The image height cannot be set as auto. If the image height has to be auto, just set mode as widthFix.

Sceenshot

Original Image

scaleToFill

Fit image completely without maintaining aspect ratio

aspectFit

Scale with aspect ratio and show fully long side

aspectFill

Scale with aspect ratio and ensure short side to be displayed fully.

widthFix

Width not changed and height changed automatically with aspect ratio unchanged

top

Not scaling image and showing only top area

bottom

Not scaling image and showing only bottom area

center

Not scaling image and showing only central area

left

Not scaling image and showing only left area

right

Not scaling image and showing only right area

top left

Not scaling image and showing only top left area

top right

Not scaling image and showing only right top area

bottom left

Not scaling image and showing only bottom left area

bottom right

Not scaling image and showing only bottom right area

Sample Code

copy

```
<view class="section" a:for="{{array}}" a:for-item="item">
  <view class="title">{{item.text}}</view>
  <image style="background-color: #eeeeee; width: 300px;
height:300px;" mode="{{item.mode}}" src="{{src}}" onError="imageError"
onLoad="imageLoad" />
</view>
```

copy

```
Page({
  data: {
    array: [{\
      mode: 'scaleToFill',\
      text: 'scaleToFill: scale without aspect ratio and fit image
completely’\
    }, {\
      mode: 'aspectFit',\
      text: 'aspectFit: scale with aspect ratio and show fully long
side’\
    }, {\
      mode: 'aspectFill',\
      text: 'aspectFill: scale with aspect ratio and ensure short side
to be displayed fully.’\
    }, {\
      mode: 'top',\
      text: 'top: Not scaling image, showing only top area’\
    }, {\
      mode: 'bottom',\
      text: 'bottom: Not scaling image, showing only bottom area’\
    }, {\
      mode: 'center',\
      text: 'center: Not scaling image, showing only central area’\
    }, {\
      mode: 'left',\
      text: 'left: Not scaling image, showing only left area’\
    }, {\
      mode: 'right',\
      text: ‘right: Not scaling image, showing only right area’\
    }, {\
      mode: 'top left',\
      text: ‘top left: Not scaling image, showing only top left area’\
    }, {\
      mode: 'top right',\
      text: ‘top right: Not scaling image, showing only top right
area’\
    }, {\
      mode: 'bottom left',\
      text: ‘bottom left: Not scaling image, showing only bottom left
area’\
    }, {\
      mode: 'bottom right',\
      text: ‘bottom right: Not scaling image, showing only bottom
right area’\
    }],
    src: './2.png'
  },
  imageError: function (e) {
    console.log('image3 error happened', e.detail.errMsg)
  },
  imageLoad: function (e) {
    console.log('image loaded successfully', e);
  }
})
```

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

---

input {#input}

Last updated: 2021-05-09

Path: miniprogram_gcash

input

2021-05-09 18:43

Input box

Scan QR code to try:

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Default** | **Description** | | value | String | | Initial contents. | | name | String | | Component name, used for the form submission of obtained data. | | type | String | text | Input type, effective value: text, number, digit. | | password | Boolean | false | Is password type or not. | | placeholder | String | | Placeholder . | | placeholder-style | String | | Specify placeholder style. | | placeholder-class | String | | Specify placeholder style class. | | disabled | Boolean | false | Disable or not. | | maxlength

| Number | 140 | Maximum length. | | cursor | Number | | Cursor location when specifying focus. | | onInput | EventHandle | | Trigger input event on keyboard entry, event.detail = {value: value}. | | onConfirm | EventHandle | | Trigger on clicking keyboard completion, event.detail = {value: value}. | | onFocus | EventHandle | | Trigger on getting focus, event.detail = {value: value}. | | onBlur | EventHandle | | Trigger on losing focus, event.detail = {value: value}. |

Note (For iOS):

Due to iOS system restrictions, the input component has the following known issues:

- The cursor of input might be misaligned with the input element.
- The keyboard might be hidden with long press onthe input.

To solve these issues, add **enableNative={{false}}** to the input element of your MiniProgram code to downgrade to pure HTML5 elements.

Now the **enableNative** propertyis set to false. In this case,the number type is no longer supported, and only text type input is supported for inputs.

Screenshot

Sample Code

copy

```
<input maxlength="10" placeholder="maximum entered length 10" />
<input onInput="bindKeyInput" placeholder="entry synchronized to
view"/>
<input type="number" placeholder="This is a numeral entry box" />
<input password type="text" placeholder="This is a password entry box"
/>
<input type="digit" placeholder="numeral keyboard with decimal"/>
```

copy

```
Page({
  data: {
    inputValue: '',
  },
  bindKeyInput(e) {
    this.setData({
      inputValue: e.detail.value,
    });
  },
});
```

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

---

input {#input}

Last updated: 2022-07-03

Path: miniprogram_gcash

input

2022-07-03 18:44

Input box

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Default** | **Description** | | value | String | | Initial contents. | | name | String | | Component name, used for the form submission of obtained data. | | type | String | text | Input type, effective value: text, number, digit. | | password | Boolean | false | Is password type or not. | | placeholder | String | | Placeholder . | | placeholder-style | String | | Specify placeholder style. | | placeholder-class | String | | Specify placeholder style class. | | disabled | Boolean | false | Disable or not. | | maxlength | Number | 140 | Maximum length. | | cursor | Number | | Cursor location when specifying focus. | | onInput | EventHandle | | Trigger input event on keyboard entry, event.detail = {value: value}. | | onConfirm | EventHandle | | Trigger on clicking keyboard completion, event.detail = {value: value}. | | onFocus | EventHandle | | Trigger on getting focus, event.detail = {value: value}. | | onBlur | EventHandle | | Trigger on losing focus, event.detail = {value: value}. |

Note (For iOS):

Due to iOS system restrictions, the input component has the following known issues:

- The cursor of input might be misaligned with the input element.
- The keyboard might be hidden with long press onthe input.

To solve these issues, add **enableNative={{false}}** to the input element of your MiniProgram code to downgrade to pure HTML5 elements.

Now the **enableNative** propertyis set to false. In this case,the number type is no longer supported, and only text type input is supported for inputs.

Screenshot

Sample Code

copy

```
<input maxlength="10" placeholder="maximum entered length 10" />
<input onInput="bindKeyInput" placeholder="entry synchronized to
view"/>
<input type="number" placeholder="This is a numeral entry box" />
<input password type="text" placeholder="This is a password entry box"
```

```
/>
<input type="digit" placeholder="numeral keyboard with decimal"/>
```

copy

```
Page({
  data: {
    inputValue: '',
  },
  bindKeyInput(e) {
    this.setData({
      inputValue: e.detail.value,
    });
  },
});
```

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/component_form-component_input

---

label {#label}

Last updated: 2021-05-09

Path: miniprogram_gcash

label

2021-05-09 18:43

The label can be used to improve the form component availability. Use the ‘for’ attribute to find the ‘id’ of the related component or place the component under the label. When it is clicked, the focus is on the related component.

The priority of ‘for’ is higher than that of internal component. When there are multiple components internally, the first component is triggered by default.

Currently the following controls can be bound: checkbox, radio, input, textarea.

Scan QR code to try:

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | for | String | Id of the bound component. |

Screenshot

Sample Code

copy

```
<view class="section">
  <view class="title">Checkbox  label cover checkbox</view>
    <checkbox-group>
      <view>
        <label>
          <checkbox value="aaa" />
          <text>aaa</text>
        </label>
      </view>
      <view>
        <label>
          <checkbox value="bbb" />
          <text>bbb</text>
        </label>
      </view>
    </checkbox-group>
  </view>
</view>
<view class="section">
  <view class="title">Radio, associating with the ‘for’
attribute</view>
    <radio-group>
      <view>
        <radio id="aaa" value="aaa" />
        <label for="aaa">aaa</label>
      </view>
      <view>
        <radio id="bbb" value="bbb" />
        <label for="bbb">bbb</label>
      </view>
    </radio-group>
  </view>
</view>
<view class="section">
  <view class="title">select only one when multiple checkboxes are
clicked</view>
    <label>
      <checkbox>Check me</checkbox>
      <checkbox>Not checked</checkbox>
      <checkbox>Not checked</checkbox>
      <checkbox>Not checked</checkbox>
      <view>
        <text>Click Me</text>
      </view>
    </label>
  </view>
</view>
```

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

---

map {#map}

Last updated: 2022-07-07

Path: miniprogram_gcash

map

2022-07-07 17:08

This topic introduces the map components. If multiple map components are displayed on the same page, different IDs are required. The map components are the native components in the application, which have the highest level of the hierarchy. No matter what the value of `zIndex` is, the level of other components on the page cannot be higher than that of the map components.

Related API: my.createMapContext(mapId)

Prerequisites

- Currently the map components support the Google Maps only.
- Do not use the map components in scroll-view.
- The map components do not support CSS animation.
- If the user zooms in or zooms out the map, reset the value of `scale` in the `onRegionChange` function before setting the latitude and longitude of the location. Otherwise, the map restores to the original size. See sample codes of `regionchange` for details.
- The mini program does not support obtaining the latitude and longitude of the current map.

Sample codes

copy

```
<view>
 <map id="map" longitude="103.855457" latitude="1.339712" scale="
{{scale}}" controls="{{controls}}"
```

```
 onControlTap="controltap" markers="{{markers}}"
 onMarkerTap="markertap"
 polyline="{{polyline}}"
 circles="{{circles}}"
 onRegionChange="regionchange"
 onTap="tap"
 show-location style="width: 100%; height: 300px;"
 include-points="{{includePoints}}"></map>
 <button onTap="changeScale">changeScale</button>
 <button onTap="getCenterLocation">getCenterLocation</button>
 <button onTap="moveToLocation">moveToLocation</button>
 <button onTap="changeCenter">changeCenter</button>
 <button onTap="changeMarkers">changeMarkers</button>
</view>
```

copy

```
Page({
  data: {
    scale: 14,
    longitude: 103.855457,
    latitude: 1.339712,
    markers: [{\
      iconPath: "/image/green_tri.png",\
      id: 10,\
      latitude:  1.342983,\
      longitude: 103.867935,\
      width: 50,\
      height: 50\
   },{\
      iconPath: "/image/green_tri.png",\
      id: 11,\
      latitude: 1.343573,\
      longitude: 103.861916,\
      width: 50,\
      height: 50,\
      customCallout: {\
        type: 1,\
        time: '1',\
     },\
      fixedPoint:{\
        originX: 400,\
        originY: 400,\
     },\
      iconAppendStr: 'Map icon'\
   }],
    includePoints: [{\
      latitude: 1.347016,\
      longitude: 103.860167,\
   }],
    polyline: [{\
      points: [{\
        longitude: 103.863218,\
        latitude: 1.351628\
     }, {\
        longitude: 103.862718,\
        latitude: 1.351428\
     }, {\
        longitude: 103.862218,\
        latitude: 1.350828\
     }, {\
        longitude: 103.861718,\
        latitude: 1.350428\
     }, {\
        longitude: 103.861018,\
        latitude: 1.351028\
     }],\
      color: "#FF0000DD",\
      width: 5,\
      dottedLine: false\
   }],
    circles: [{\
      latitude: 1.351628,\
      longitude: 103.863718,\
      color: "#000000AA",\
      fillColor: "#000000AA",\
      radius: 80,\
      strokeWidth: 5,\
   }],
    controls: [{\
      id: 5,\
      iconPath: '../../resources/pic/2.jpg',\
      position: {\
        left: 0,\
        top: 300 - 50,\
        width: 50,\
        height: 50\
     },\
      clickable: true\
   }]
 },

 onReady(e) {
   // Use my.createMapContext to obtain the map context.
   this.mapCtx = my.createMapContext('map')
 },

 getCenterLocation() {
   this.mapCtx.getCenterLocation(function (res) {
      console.log(res.longitude)
      console.log(res.latitude)
   })
 },
 moveToLocation() {
   this.mapCtx.moveToLocation()
 },

 regionchange(e) {
    console.log('regionchange', e);
// Note: If the user zooms in or zooms out the map, reset the value of
scale of the onRegionChange function before setting the latitude and
longitude of the location. Otherwise the map restores to the original
size.
if (e.type === 'end') {
     this.setData({
        scale: e.scale
     });
   }
 },

 markertap(e) {
    console.log('marker tap', e);
 },

 controltap(e) {
    console.log('control tap', e);
 },

 tap() {
    console.log('tap:');
 },

 changeScale() {
   this.setData({
      scale: 8,
   });
 },

 changeCenter() {
   this.setData({
      longitude: 103.867935,
      latitude: 1.343573,
      includePoints: [{\
        latitude: 1.351028,\
        longitude: 103.861018,\
     }],
   });
 },
 //An indicator of whether to support gesture events. When
isGestureEnable is 1, gesture events are supported. Otherwise gesture
events are not supported.
 gestureEnable() {
   this.mapCtx.gestureEnable({isGestureEnable:1});
 },
 //An indicator of whether to show the compass. When isShowCompass is
1, display the compass. Otherwise the compass is not displayed.
 showsCompass() {
   this.mapCtx.showsCompass({isShowsCompass:1});
 },
 changeMarkers() {
   this.setData({
      markers: [{\
        iconPath: "/image/green_tri.png",\
        id: 10,\
        latitude: 1.351028,\
        longitude: 103.861018,\
        width: 50,\
        height: 50\
     }],
      includePoints: [{\
        latitude: 1.350428,\
        longitude: 103.861718,\
     }],
   });
 },
})
```

Parameters

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | style | String | Inline style. | | class | String | Style name. | | latitude | Number | The latitude of the central point. | | longitude | Number | The longitude of the central point. | | scale | Number | The zoom level. The value ranges from 5 to 18 and is 16 by default. | | markers | Array | The location marker. See markers for details. | | polyline | Array | The polyline. See polyline for details. | | circles | Array | The circle. See circles for details. | | polygon | Array | The polygon. See polygon for details. | | show-location | Boolean | An indicator of whether to display the current location with directions. | | include-points | Array | The view is extended in a small scale with the passed coordinates.
`Example:`
`[{`
`latitude: 1.350428,`
`longitude: 103.861718,`
`}]` | | include-padding | Object | The view is displayed within the map padding.
```
Example:
[{
   left:0, right:0,
   top:0, bottom:0
```
`}]` | | setting | Object | Settings.
```
Example:
{
gestureEnable: 1, <br>
showScale: 1, <br>
showCompass: 1,   <br>
```

```
tiltGesturesEnabled: 1,
trafficEnabled: 0,    <br>
showMapText: 0,
logoPosition: {
centerX: 150,
centerY: 90
}
```
} | | onMarkerTap | EventHandle | Call this function when clicking on Marker.
```
Example:
{
markerId,
latitude,
longitude,
```
} | | onCalloutTap | EventHandle | Call this function when clicking on the tooltip of Marker.
```
Example:
{
markerId,
latitude,
longitude,
```
} | | onControlTap | EventHandle | Call this function when clicking on control.
```
Example:
{
controlId
```
} | | onRegionChange | EventHandle | Call this function when the view is changed.
```
Example:
{
type: "begin/end",
latitude,
longitude,
scale
```
} | | onTap | EventHandle | Call this function when clicking on the map.
```
Example:
{
latitude,
longitude,
```
} |

markers

The location marker.

Notes:

- With the `markers` parameter, multiple location markers can be displayed.
- The description of the location marker does not support English.

polygon

Specify a series of coordinates, which form a closed polygon based on the `points`.

| | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | points | An array of the latitude and longitude.
`Example:`
`[{`
`latitude: 0,`
`longitude: 0`
`}]` | Array | Yes | | color | The stroke color. Use hexadecimal numbers to set colors.
`Example: #eeeeeeAA` | String | No | | fillColor | The fill color. Use hexadecimal numbers to set colors.
`Example: #eeeeeeAA` | String | No | | width | The stroke width. | Number | No |

polyline

Specify a series of coordinates, which are connected from the first item to the last item in an array.

| | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | points | An array of the latitude and longitude.
`Example:`
`[{`
`latitude: 0,`
`longitude: 0`
`}]` | Array | Yes | | color | The stroke color. Use hexadecimal numbers to set colors.
`Example: #eeeeeeAA` | String | No | | width | The stroke width. | Number | No | | iconWidth | The icon width. | Number | No | | zIndex | The zIndex compared to other polys. | Number | - | | iconPath | Displayed icon.
Note:
The image path in the project directory. The path can be written as a relative path prefixed with a forward slash (/), which indicates a relative root directory of a mini program. If this parameter is specified, ignore the value of `color`. To create the multiple-color polyline, use `iconPath` and `colorList` jointly. Set the background of the icon to be transparent so that the multiple-color polyline is not covered. | String | - |

circles

Display a circle on the map.

| | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | latitude | The latitude. The value ranges from -90 to 90. | Float | Yes | | longitude | The longitude. The value ranges from -180 to 180. | Float | Yes | | color | The stroke color. Use hexadecimal numbers to set colors.
`Example: #eeeeeeAA` | String | No | | fillColor | The fill color. Use hexadecimal numbers to set colors.
`Example: #eeeeeeAA` | String | No | | radius | The radius in meters. | Number | Yes | | strokeWidth | The stroke width. | Number | No |

callout Deprecated

The customs tooltip over the location marker.

| | | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | content | The content in the tooltip, which is empty by default. | String | No |

customCallout Deprecated

The customs background of the tooltip.

| | | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | type | The style of the background. Valid values are:
- 0: Black background
- 1: White background
- 2:Background and text | Number | Yes | | time | The time. | String | Yes | | descList | The description array.
```
Example:
{
"type": 0,
"time": "3",
"descList": [{
"desc": "Click to take a taxi",
"descColor": "#ffffff"      }],
"isShow": 1
```
} | Array | Yes |

fixedPoint

The fixed point based on the screen.

| | | | | | --- | --- | --- | --- | | **Property** | **Description** | **Type** | **Required** | | originX | The number of pixels in the horizontal direction from the upper-left corner of the map. | Number | Yes | | originY | The number of pixels in the vertical direction from the upper-left corner of the map. | Number | Yes |

The latitude and longitude must be set for the map components. Otherwise the default coordinate is Beijing's latitude and longitude.

Location marker design

Priority

- The customCallout, callout, and label are excluded with each other. The priority order is label > customCallout > callout.
- The style and icon are excluded with each other. The priority order is:
- style > iconAppendStr
- style > icon

style

| | | | --- | --- | | **Sample code** | **Sample legend** | | copy
`<br>{<br>    type:1,<br>    text1:"Style1",<br>    icon1:'xxx',<br>    icon2:'xxx'<br>}<br>` | | | copy
`<br>{<br>    type:2,<br>    text1:"Style2",<br>    icon1:'xxx',<br>    icon2:'xxx'<br>}<br>` | | | copy
`<br>{<br>   type:3,<br>   icon:xxx, //Optional<br>   text:xxx,   //Mandatory<br>   color:xxx, //#33B276 by default<br>   bgColor:xxx, //#FFFFFF by default <br>   gravity:"left/center/right", //center by default<br>   fontType:"small/standard/large" //standard by default<br>}<br>` | |

label

| | | | | --- | --- | --- | | **Property** | **Required** | **Remark** | | content | Yes | - | | color | No | The default value is #000000. | | fontsize | No | The default value is 14. | | borderRadius | No | The default value is 20. | | bgColor | No | The default value is #FFFFFF. | | padding | No | The default value is 10. |

FAQs

How do the map components redirect the mini program to the Google Maps for the navigation?

Use the my.openLocation API.

How to obtain the value of `scale` when the `optimize` property of the map components is true?

Use the `onRegionChange` function.

How to create the polygon area on the map manually?

Use the `polygon` property.

Does the text in iconAppendStr support line breaks?

No, the text in iconAppendStr does not support line breaks.

How to modify icons of the first item and the last item in the map components after `iconPath` is set?

Currently the modification is not supported.

More information

my.createMapContext

MapContext Overview

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

---

movable-area {#movable-area}

Last updated: 2021-05-09

Path: miniprogram_gcash

movable-area

2021-05-09 18:43

A movable area of movable-view component. movable-area must set the width and height properties. Otherwise, the default 10px is used.

Sample Code

index.axml

copy

```
<!-- API-DEMO page/component/movable-view.axml -->
<view class="page">
  <view class="page-description">movable-view</view>
  <view class="page-section">
    <view class="page-section-title">movable-view is less than
movable-area</view>
    <view class="page-section-demo">
      <movable-area>
        <movable-view x="{{x}}" y="{{y}}" direction="all">movable-
view</movable-view>
      </movable-area>
    </view>
    <button style="margin-left: 10px; margin-right: 10px;"
type="primary" onTap="onButtonTap">Click Me to Move to (30px, 30px)
</button>
  </view>
  <view class="page-section">
    <view class="page-section-title">movable-view is greater than
movable-area</view>
    <view class="page-section-demo">
      <movable-area>
```

```
        <movable-view class="max" direction="all">movable-
view</movable-view>
      </movable-area>
    </view>
  </view>
  <view class="page-section">
    <view class="page-section-title">Can only be moved
laterally</view>
    <view class="page-section-demo">
     <movable-area>
        <movable-view direction="horizontal">
          movable-view
        </movable-view>
      </movable-area>
    </view>
  </view>
  <view class="page-section">
    <view class="page-section-title">Can only be moved
vertically</view>
    <view class="page-section-demo">
     <movable-area>
        <movable-view direction="vertical">
          movable-view
        </movable-view>
      </movable-area>
    </view>
  </view>
</view>
```

index.js

copy

```
// API-DEMO page/component/movable-view.js
Page({
  data: {
    x: 0,
    y: 0,
  },
  onButtonTap() {
    const { x, y } = this.data;
    if (x === 30) {
      this.setData({
        x: x + 1,
        y: y + 1,
      });
    } else {
      this.setData({
        x: 30,
        y: 30
      });
    }
  },
});
```

index.json

copy

```
// API-DEMO page/component/movable-view.json
{
  "allowsBounceVertical": "NO"
}
```

index.acss

copy

```
/* API-DEMO page/component/movable-view.acss */
movable-area {
  height: 400rpx;
  width: 400rpx;
  margin: 50rpx 0rpx 0 50rpx;
  background-color: #ccc;
  overflow: hidden;
}

movable-view {
  display: flex;
  align-items: center;
  justify-content: center;
  height: 200rpx;
  width: 200rpx;
  background: #108ee9;
  color: #fff;
}

.max {
  width: 600rpx;
  height: 600rpx;
}
```

Parameters

| | | | | | | --- | --- | --- | --- | --- | | **Property** | **Type** | **Default** | **Required** | **Description** | | scale-area | Boolean | false | No | When the movable-view component is set to support two-finger scaling, this component allows you to modify the effective area for scaling gestures to the entire movable area. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/component_view-container_movable-area

---

movable-area {#movable-area}

Last updated: 2022-07-03

Path: miniprogram_gcash

movable-area

2022-07-03 18:44

A movable area of movable-view component. movable-area must set the width and height properties. Otherwise, the default 10px is used.

Sample Code

index.axml

copy

```
<!-- API-DEMO page/component/movable-view.axml -->
<view class="page">
  <view class="page-description">movable-view</view>
  <view class="page-section">
    <view class="page-section-title">movable-view is less than
movable-area</view>
    <view class="page-section-demo">
      <movable-area>
        <movable-view x="{{x}}" y="{{y}}" direction="all">movable-
view</movable-view>
      </movable-area>
    </view>
    <button style="margin-left: 10px; margin-right: 10px;"
type="primary" onTap="onButtonTap">Click Me to Move to (30px, 30px)
</button>
  </view>
  <view class="page-section">
    <view class="page-section-title">movable-view is greater than
movable-area</view>
    <view class="page-section-demo">
      <movable-area>
        <movable-view class="max" direction="all">movable-
view</movable-view>
      </movable-area>
    </view>
  </view>
  <view class="page-section">
    <view class="page-section-title">Can only be moved
laterally</view>
```

```
    <view class="page-section-demo">
     <movable-area>
        <movable-view direction="horizontal">
          movable-view
        </movable-view>
      </movable-area>
    </view>
  </view>
  <view class="page-section">
    <view class="page-section-title">Can only be moved
vertically</view>
    <view class="page-section-demo">
     <movable-area>
        <movable-view direction="vertical">
          movable-view
        </movable-view>
      </movable-area>
    </view>
  </view>
</view>
```

index.js

copy

```
// API-DEMO page/component/movable-view.js
Page({
  data: {
    x: 0,
    y: 0,
  },
  onButtonTap() {
    const { x, y } = this.data;
    if (x === 30) {
      this.setData({
        x: x + 1,
        y: y + 1,
      });
    } else {
      this.setData({
        x: 30,
        y: 30
      });
    }
  },
});
```

index.json

copy

```
// API-DEMO page/component/movable-view.json
{
  "allowsBounceVertical": "NO"
}
```

index.acss

copy

```
/* API-DEMO page/component/movable-view.acss */
movable-area {
  height: 400rpx;
  width: 400rpx;
  margin: 50rpx 0rpx 0 50rpx;
  background-color: #ccc;
  overflow: hidden;
}

movable-view {
  display: flex;
  align-items: center;
  justify-content: center;
  height: 200rpx;
  width: 200rpx;
  background: #108ee9;
  color: #fff;
}

.max {
  width: 600rpx;
  height: 600rpx;
}
```

Parameters

| | | | | | | --- | --- | --- | --- | --- | | **Property** | **Type** | **Default** | **Required** | **Description** | | scale-area | Boolean | false | No | When the movable-view component is set to support two-finger scaling, this component allows you to modify the effective area for scaling gestures to the entire movable area. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/component_view-container_movable-area

---

movable-view {#movable-view}

Path: miniprogram_gcash

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev-old/component_view-container_movable-view

movable-view {#movable-view}

Last updated: 2022-07-03

Path: miniprogram_gcash

movable-view

2022-07-03 18:44

movable-view can be dragged and slid in the page. The movable-view component must be in the movable-area component and must be a direct child node. Otherwise, the component can't move.

Notes:

- movable-view must set the width and height properties. Otherwise, the default value 10px is used.
- By default, movable-view uses absolute positioning, which can't be changed. The values of the top and left properties are 0 px.
- When movable-view is smaller than movable-area, the moving range of movable-view is within the movable-area. When movable-view is larger than movable-area, the moving range of movable-view must cover the movable-area. The x-axis direction and the y-axis direction are separately considered.

Sample Code

index.axml

copy

```
<!-- API-DEMO page/component/movable-view.axml -->
<view class="page">
  <view class="page-description">movable-view</view>
  <view class="page-section">
    <view class="page-section-title">movable-view is less than
movable-area</view>
    <view class="page-section-demo">
      <movable-area>
        <movable-view x="{{x}}" y="{{y}}" direction="all">movable-
view</movable-view>
      </movable-area>
    </view>
    <button style="margin-left: 10px; margin-right: 10px;"
type="primary" onTap="onButtonTap">Click Me to Move to (30px, 30px)
</button>
  </view>
  <view class="page-section">
    <view class="page-section-title">movable-view is greater than
movable-area</view>
    <view class="page-section-demo">
      <movable-area>
        <movable-view class="max" direction="all">movable-
view</movable-view>
      </movable-area>
    </view>
  </view>
  <view class="page-section">
    <view class="page-section-title">Can only be moved
laterally</view>
    <view class="page-section-demo">
     <movable-area>
        <movable-view direction="horizontal">
          movable-view
        </movable-view>
      </movable-area>
    </view>
  </view>
  <view class="page-section">
    <view class="page-section-title">Can only be moved
vertically</view>
    <view class="page-section-demo">
     <movable-area>
        <movable-view direction="vertical">
          movable-view
        </movable-view>
      </movable-area>
    </view>
  </view>
</view>
```

index.js

copy

```
// API-DEMO page/component/movable-view.js
Page({
  data: {
    x: 0,
    y: 0,
  },
  onButtonTap() {
    const { x, y } = this.data;
    if (x === 30) {
      this.setData({
        x: x + 1,
        y: y + 1,
      });
    } else {
      this.setData({
        x: 30,
        y: 30
      });
    }
  },
});
```

index.json

copy

```
// API-DEMO page/component/movable-view.json
{
  "allowsBounceVertical": "NO"
}
```

index.acss

copy

```
/* API-DEMO page/component/movable-view.acss */
movable-area {
  height: 400rpx;
  width: 400rpx;
  margin: 50rpx 0rpx 0 50rpx;
  background-color: #ccc;
  overflow: hidden;
}

movable-view {
  display: flex;
  align-items: center;
  justify-content: center;
  height: 200rpx;
  width: 200rpx;
  background: #108ee9;
  color: #fff;
}

.max {
  width: 600rpx;
  height: 600rpx;
}
```

Parameters

Property	**Type**	**Default**	**Description**
direction	String	none	The moving direction of movable-view. Valid values are "all", "vertical", "horizontal", and "none".
inertia	Boolean	false	This field specifies whether movable-view has inertia.
out-of-bounds	Boolean	false	This field specifies whether movable-view can move after the view container is out of the movable area.
x	Number	0	This field defines the offset in the direction of the x axis, which is converted to the left property of the component. If the value of x is not within the movable range, the component is automatically moved to the movable range.
y	Number	0	This field defines the offset in the direction of the Y-axis, which is direction is converted to the top property. If the value of Y is not within the movable range, it will be automatically moved to the movable range.
damping	Number	20	The damping coefficient, which is used to control the animation triggered when the value of x or y changes and the animation that is pulled back when the component exceeds the range. A higher value leads to faster movement.
friction	Number	2	The friction coefficient, which is used to control the animation that moves due to inertia. A higher value leads to higher friction and indicates that the movement stops earlier. Must be greater than **0**. Otherwise the default value is used.
disabled	Boolean	false	This field specifies whether to disable the component.
scale	Boolean	false	This field specifies whether to support two-finger scaling. The effective area for scaling gestures falls within the movable-view by default.
scale-min	Number	0.5	The minimum value of the scaling level.
scale-max	Number	10	The maximum value of the scaling level.
scale-value	Number	1	The scale level. Can range from 0.5 to 10.
animation	Boolean	false	This field specifies whether to use animations.
onTouchStart	EventHandle	-	Finger touch starts and this event is passed to the parent node.
catchTouchStart	EventHandle	-	Finger touch starts and this event only acts on the component and is not passed to the parent node.
onTouchMove	EventHandle	-	Finger moves after touch, the event is passed to the parent node.
catchTouchMove	EventHandle	-	Finger moves after touch, the event only acts on the component and is not passed to the parent node.
onTouchEnd	EventHandle	-	The touch action ends, the event is passed to the parent node.
catchTouchEnd	EventHandle	-	The touch action ends, the event only acts on the component and is not passed to the parent node.
onTouchCancel	EventHandle	-	The touch action is interrupted, such as call reminding and popups.
onChange	EventHandle	-	The event triggered during dragging, `event. detail = {x: x, y: y, source: touch}`, where `source` shows the reason of the movement, for example, the value is `touch`.
onChangeEnd	EventHandle	-	The event triggered after dragging, `event.detail = {x: x, y: y}`.
onScale	EventHandle	-	The event triggered during zooming, `event.detail = {x, y, scale}`.

onChange return value detail.source

The source field shows the reason of the movement.

Value	**Description**
touch	Dragging.
touch-out-of-bounds	The movable range is exceeded.
out-of-bounds	Pullback after the movable range is exceeded.
friction	Inertia.
Empty string	`setData`.

Instruction: Please check the **event type** in the event introduction for bubbling event.

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/component_view-container_movable-view

my.SDKVersion {#mysdkversion}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.SDKVersion

2022-07-03 18:44

Get the version of basic library (for reference only). **Do not rely on this value for code logic.**

Sample Code

copy

```
<!-- API-DEMO page/API/sdk-version/sdk-version.axml-->
<view class="page">
  <view class="page-description">Get version of basic library
API</view>
  <view class="page-section">
    <view class="page-section-title">my.SDKVersion</view>
    <view class="page-section-demo">
      <button type="primary" onTap="getSDKVersion">Get version of
basic library</button>
    </view>
  </view>
</view>
```

copy

```
// API-DEMO page/API/sdk-version/sdk-version.js
Page({
  getSDKVersion() {
    my.alert({
      content: my.SDKVersion,
    });
  },
});
```

Return Value

String, version of basic library

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

---

my.SDKVersion {#mysdkversion}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.SDKVersion

2021-05-09 18:43

Get the version of basic library (for reference only). **Do not rely on this value for code logic.**

Sample Code

copy

```
<!-- API-DEMO page/API/sdk-version/sdk-version.axml-->
<view class="page">
  <view class="page-description">Get version of basic library
API</view>
  <view class="page-section">
    <view class="page-section-title">my.SDKVersion</view>
    <view class="page-section-demo">
      <button type="primary" onTap="getSDKVersion">Get version of
basic library</button>
    </view>
  </view>
</view>
```

copy

```
// API-DEMO page/API/sdk-version/sdk-version.js
Page({
  getSDKVersion() {
    my.alert({
      content: my.SDKVersion,
    });
  },
});
```

Return Value

String, version of basic library

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

---

my.addPhoneContact {#myaddphonecontact}

Last updated: 2022-07-04

Path: miniprogram_gcash

my.addPhoneContact

2022-07-04 03:44

Version requirements: Basic library 1.10.0 or higher version. If the version is low, suggest Compatible treatment

This form enables the user to write the form into phone contacts via create contacts or add to existing contacts.

Sample Code

copy

```
// API-DEMO page/API/contact/contact.json
{
   "defaultTitle": "Contact"
}
```

copy

```
<!-- API-DEMO page/API/contact/contact.axml-->
<view class="page">

  <view class="page-description">Contact API</view>
  <view class="page-section">
    <view class="page-section-title">my.addPhoneContact</view>
    <view class="page-section-demo">

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Basic information</text>
      </view>
      <view class="form-row">
        <view class="form-row-label">Nickname</view>
        <view class="form-row-content">
          <input id="nickName" onInput="onInput" class="input"
value="Baking July" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Last name</view>
        <view class="form-row-content">
          <input id="lastName" onInput="onInput" class="input"
value="Last" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Middle name</view>
        <view class="form-row-content">
          <input id="middleName" onInput="onInput" class="input"
value="Middle" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">First name</view>
        <view class="form-row-content">
          <input id="firstName" onInput="onInput" class="input"
value="First" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Remarks</view>
        <view class="form-row-content">
          <input id="remark" onInput="onInput" class="input"
value="This is the remarks" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Phone number</view>
        <view class="form-row-content">
          <input id="mobilePhoneNumber" onInput="onInput"
class="input" value="13800000000" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
```

```
bottom:18px">Contact address</text>
      </view>

      <view class="form-row">
        <view class="form-row-label">Country</view>
        <view class="form-row-content">
          <input id="addressCountry" onInput="onInput" class="input"
value="address country" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Province</view>
        <view class="form-row-content">
          <input id="addressState" onInput="onInput" class="input"
value="address state" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">City</view>
        <view class="form-row-content">
          <input id="addressCity" onInput="onInput" class="input"
value="address city" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Street</view>
        <view class="form-row-content">
          <input id="addressStreet" onInput="onInput" class="input"
value="address street" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Postcode</view>
        <view class="form-row-content">
          <input id="addressPostalCode" onInput="onInput"
class="input" value="94016" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Work</text>
      </view>

      <view class="form-row">
        <view class="form-row-label">Company</view>
        <view class="form-row-content">
```

```
          <input id="organization" onInput="onInput" class="input"
value="organization" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Title</view>
        <view class="form-row-content">
          <input id="title" onInput="onInput" class="input"
value="Developer" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Work fax</view>
        <view class="form-row-content">
          <input id="workFaxNumber" onInput="onInput" class="input"
value="11111111" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Work phone</view>
        <view class="form-row-content">
          <input id="workPhoneNumber" onInput="onInput" class="input"
value="11111112" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Company phone</view>
        <view class="form-row-content">
          <input id="hostNumber" onInput="onInput" class="input"
value="11111113" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Email</view>
        <view class="form-row-content">
          <input id="email" onInput="onInput" class="input"
value="liuhuo01@miniprogram.com" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Website</view>
        <view class="form-row-content">
          <input id="url" onInput="onInput" class="input"
value="www.miniprogram.com" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Company address</text>
      </view>

      <view class="form-row">
        <view class="form-row-label">Country</view>
        <view class="form-row-content">
          <input id="workAddressCountry" onInput="onInput"
class="input" value="work country" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Province</view>
        <view class="form-row-content">
          <input id="workAddressState" onInput="onInput" class="input"
value="work state" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">City</view>
        <view class="form-row-content">
          <input id="workAddressCity" onInput="onInput" class="input"
value="work city" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Street</view>
        <view class="form-row-content">
          <input id="workAddressStreet" onInput="onInput"
class="input" value="work street" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Postcode</view>
        <view class="form-row-content">
          <input id="workAddressPostalCode" onInput="onInput"
class="input" value="111111" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Home</text>
      </view>
      <view class="form-row">
        <view class="form-row-label">Fax</view>
        <view class="form-row-content">
          <input id="homeFaxNumber" onInput="onInput" class="input"
value="11111114" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Phone</view>
        <view class="form-row-content">
          <input id="homePhoneNumber" onInput="onInput" class="input"
value="11111115" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Country</view>
        <view class="form-row-content">
          <input id="homeAddressCountry" onInput="onInput"
class="input" value="home country" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Province</view>
        <view class="form-row-content">
          <input id="homeAddressState" onInput="onInput" class="input"
value="home state" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">City</view>
        <view class="form-row-content">
          <input id="homeAddressCity" onInput="onInput" class="input"
value="home city" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Street</view>
        <view class="form-row-content">
          <input id="homeAddressStreet" onInput="onInput"
class="input" value="home street" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Postcode</view>
        <view class="form-row-content">
          <input id="homeAddressPostalCode" onInput="onInput"
class="input" value="123456" />
        </view>
      </view>

      <button type="primary" onTap="addPhoneContact">Add to phone
contact</button>

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

copy

```
// API-DEMO page/API/contact/contact.js
Page({
  data:{
      "photoFilePath": "/sdcard/DCIM/Camera/a.jpg",
      "nickName": "Baking July",
      "lastName": "Last",
      "middleName": "Middle",
      "firstName": "First",
      "remark": "This is remarks",
      "mobilePhoneNumber": "13800000000",
      "homePhoneNumber": "11111115",
      "workPhoneNumber": "11111112",
      "homeFaxNumber": "11111114",
      "workFaxNumber": "11111111",
      "hostNumber": "11111113",
      "addressCountry": "address country",
      "addressState": "address state",
      "addressCity": "address city",
      "addressStreet": "address street",
      "addressPostalCode": "94016",
      "workAddressCountry": "work country",
      "workAddressState": "work state",
      "workAddressCity": "work city",
      "workAddressStreet": "work street",
      "workAddressPostalCode": "111111",
      "homeAddressCountry": "home country",
      "homeAddressState": "home state",
      "homeAddressCity": "home city",
      "homeAddressStreet": "home street",
      "homeAddressPostalCode": "123456",
      "organization": "organization",
      "title": "Developer",
      "email": "liuhuo01@miniprogram.com",
      "url": "www.miniprogram.com",
      success: (res) => {
        my.alert({
          content: 'addPhoneContact response: ' + JSON.stringify(res)
        });
      },
      fail: (res) => {
        my.alert({
          content: 'addPhoneContact response: ' + JSON.stringify(res)
        });
      }
  },
  onInput(e) {
    this.data[e.currentTarget.id] = e.detail.value;
  },
  addPhoneContact() {
    if (my.canIUse('addPhoneContact')) {
      my.addPhoneContact(this.data);
    } else {
      my.alert({
        title: 'Client version too low',
        content: 'my.addPhoneContact() needs higher version'
      });
    }
  }
});
```

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
photoFilePath	String	No	Local file path of avatar.
nickName	String	No	Nickname.
lastName	String	No	Surname.
middleName	String	No	Middle name.
firstName	String	No	First name.
remark	String	No	Remarks.
mobilePhoneNumber	String	No	Cell number.
addressCountry	String	No	Country in contact address.
addressState	String	No	Province in contact address.
addressCity	String	No	City in contact address.
addressStreet	String	No	Street in contact address.
addressPostalCode	String	No	Postcode in contact address.
organization	String	No	Company.
title	String	No	Title.
workFaxNumber	String	No	Work fax.
workPhoneNumber	String	No	Work phone number.
hostNumber	String	No	Company phone number.
email	String	No	Email.
url	String	No	Website.
workAddressCountry	String	No	Country in work address.
workAddressState	String	No	Province in work address.
workAddressCity	String	No	City in work address.
workAddressStreet	String	No	Street in work address.
workAddressPostalCode	String	No	Postcode in work address.
homeFaxNumber	String	No	Home fax.
homePhoneNumber	String	No	Home phone.
homeAddressCountry	String	No	Country in home address.
homeAddressState	String	No	Province in home address.
homeAddressCity	String	No	City in home address.
homeAddressStreet	String	No	Street in home address.
homeAddressPostalCode	String	No	Postcode in home address.
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Depending on the support of contact App built in different ROM, the above fields may not support emoji or kaomoji. In such cases, this option is ignored.

Return Value

Success:

success = true

Failure:

| | | | | --- | --- | --- | | **Error** | **ErrorMessage** | **Description** | | 11 | fail cancel | The user cancels the operation. | | 3 | fail ${detail} | Call failure, detail includes the detailed information. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/API_Device_Add-Contact_addPhoneContact

---

my.addPhoneContact {#myaddphonecontact}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.addPhoneContact

2021-05-09 18:43

Version requirements: Basic library 1.10.0 or higher version. If the version is low, suggest Compatible treatment

This form enables the user to write the form into phone contacts via create contacts or add to existing contacts.

Sample Code

copy

```
// API-DEMO page/API/contact/contact.json
{
   "defaultTitle": "Contact"
}
```

copy

```
<!-- API-DEMO page/API/contact/contact.axml-->
<view class="page">

  <view class="page-description">Contact API</view>
  <view class="page-section">
```

```
    <view class="page-section-title">my.addPhoneContact</view>
    <view class="page-section-demo">

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Basic information</text>
      </view>

      <view class="form-row">
        <view class="form-row-label">Nickname</view>
        <view class="form-row-content">
          <input id="nickName" onInput="onInput" class="input"
value="Baking July" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Last name</view>
        <view class="form-row-content">
          <input id="lastName" onInput="onInput" class="input"
value="Last" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Middle name</view>
        <view class="form-row-content">
          <input id="middleName" onInput="onInput" class="input"
value="Middle" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">First name</view>
        <view class="form-row-content">
          <input id="firstName" onInput="onInput" class="input"
value="First" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Remarks</view>
        <view class="form-row-content">
          <input id="remark" onInput="onInput" class="input"
value="This is the remarks" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Phone number</view>
        <view class="form-row-content">
```

```
          <input id="mobilePhoneNumber" onInput="onInput"
class="input" value="13800000000" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Contact address</text>
      </view>

      <view class="form-row">
        <view class="form-row-label">Country</view>
        <view class="form-row-content">
          <input id="addressCountry" onInput="onInput" class="input"
value="address country" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Province</view>
        <view class="form-row-content">
          <input id="addressState" onInput="onInput" class="input"
value="address state" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">City</view>
        <view class="form-row-content">
          <input id="addressCity" onInput="onInput" class="input"
value="address city" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Street</view>
        <view class="form-row-content">
          <input id="addressStreet" onInput="onInput" class="input"
value="address street" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Postcode</view>
        <view class="form-row-content">
          <input id="addressPostalCode" onInput="onInput"
class="input" value="94016" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
```

```
          <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Work</text>
        </view>

        <view class="form-row">
          <view class="form-row-label">Company</view>
          <view class="form-row-content">
            <input id="organization" onInput="onInput" class="input"
value="organization" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Title</view>
          <view class="form-row-content">
            <input id="title" onInput="onInput" class="input"
value="Developer" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Work fax</view>
          <view class="form-row-content">
            <input id="workFaxNumber" onInput="onInput" class="input"
value="11111111" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Work phone</view>
          <view class="form-row-content">
            <input id="workPhoneNumber" onInput="onInput" class="input"
value="11111112" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Company phone</view>
          <view class="form-row-content">
            <input id="hostNumber" onInput="onInput" class="input"
value="11111113" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Email</view>
          <view class="form-row-content">
            <input id="email" onInput="onInput" class="input"
value="liuhuo01@miniprogram.com" />
          </view>
        </view>
```

```
      <view class="form-row">
        <view class="form-row-label">Website</view>
        <view class="form-row-content">
          <input id="url" onInput="onInput" class="input"
value="www.miniprogram.com" />
        </view>
      </view>

      <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
        <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Company address</text>
      </view>

      <view class="form-row">
        <view class="form-row-label">Country</view>
        <view class="form-row-content">
          <input id="workAddressCountry" onInput="onInput"
class="input" value="work country" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Province</view>
        <view class="form-row-content">
          <input id="workAddressState" onInput="onInput" class="input"
value="work state" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">City</view>
        <view class="form-row-content">
          <input id="workAddressCity" onInput="onInput" class="input"
value="work city" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Street</view>
        <view class="form-row-content">
          <input id="workAddressStreet" onInput="onInput"
class="input" value="work street" />
        </view>
      </view>

      <view class="form-row">
        <view class="form-row-label">Postcode</view>
        <view class="form-row-content">
          <input id="workAddressPostalCode" onInput="onInput"
class="input" value="111111" />
```

```
          </view>
        </view>

        <view style="font-size:18px;margin-top:18px;margin-bottom:18px">
          <text style="font-size:18px;margin-top:18px;margin-
bottom:18px">Home</text>
        </view>

        <view class="form-row">
          <view class="form-row-label">Fax</view>
          <view class="form-row-content">
            <input id="homeFaxNumber" onInput="onInput" class="input"
value="11111114" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Phone</view>
          <view class="form-row-content">
            <input id="homePhoneNumber" onInput="onInput" class="input"
value="11111115" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Country</view>
          <view class="form-row-content">
            <input id="homeAddressCountry" onInput="onInput"
class="input" value="home country" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Province</view>
          <view class="form-row-content">
            <input id="homeAddressState" onInput="onInput" class="input"
value="home state" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">City</view>
          <view class="form-row-content">
            <input id="homeAddressCity" onInput="onInput" class="input"
value="home city" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Street</view>
          <view class="form-row-content">
            <input id="homeAddressStreet" onInput="onInput"
  class="input" value="home street" />
          </view>
        </view>

        <view class="form-row">
          <view class="form-row-label">Postcode</view>
          <view class="form-row-content">
            <input id="homeAddressPostalCode" onInput="onInput"
  class="input" value="123456" />
          </view>
        </view>

        <button type="primary" onTap="addPhoneContact">Add to phone
  contact</button>

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

copy

```
// API-DEMO page/API/contact/contact.js
Page({
  data:{
      "photoFilePath": "/sdcard/DCIM/Camera/a.jpg",
      "nickName": "Baking July",
      "lastName": "Last",
      "middleName": "Middle",
      "firstName": "First",
      "remark": "This is remarks",
      "mobilePhoneNumber": "13800000000",
      "homePhoneNumber": "11111115",
      "workPhoneNumber": "11111112",
      "homeFaxNumber": "11111114",
      "workFaxNumber": "11111111",
      "hostNumber": "11111113",
      "addressCountry": "address country",
      "addressState": "address state",
      "addressCity": "address city",
      "addressStreet": "address street",
      "addressPostalCode": "94016",
      "workAddressCountry": "work country",
      "workAddressState": "work state",
      "workAddressCity": "work city",
      "workAddressStreet": "work street",
      "workAddressPostalCode": "111111",
      "homeAddressCountry": "home country",
      "homeAddressState": "home state",
      "homeAddressCity": "home city",
      "homeAddressStreet": "home street",
      "homeAddressPostalCode": "123456",
      "organization": "organization",
      "title": "Developer",
      "email": "liuhuo01@miniprogram.com",
      "url": "www.miniprogram.com",
      success: (res) => {
        my.alert({
          content: 'addPhoneContact response: ' + JSON.stringify(res)
        });
      },
      fail: (res) => {
        my.alert({
          content: 'addPhoneContact response: ' + JSON.stringify(res)
        });
      }
  },
  onInput(e) {
    this.data[e.currentTarget.id] = e.detail.value;
  },
  addPhoneContact() {
    if (my.canIUse('addPhoneContact')) {
      my.addPhoneContact(this.data);
    } else {
      my.alert({
        title: 'Client version too low',
        content: 'my.addPhoneContact() needs higher version'
      });
    }
  }
});
```

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
photoFilePath	String	No	Local file path of avatar.
nickName	String	No	Nickname.
lastName	String	No	Surname.
middleName	String	No	Middle name.
firstName	String	No	First name.
remark	String	No	Remarks.
mobilePhoneNumber	String	No	Cell number.
addressCountry	String	No	Country in contact address.
addressState	String	No	Province in contact address.
addressCity	String	No	City in contact address.
addressStreet	String	No	Street in contact address.
addressPostalCode	String	No	Postcode in contact address.
organization	String	No	Company.
title	String	No	Title.
workFaxNumber	String	No	Work fax.
workPhoneNumber	String	No	Work phone number.
hostNumber	String	No	Company phone number.
email	String	No	Email.
url	String	No	Website.
workAddressCountry	String	No	Country in work address.
workAddressState	String	No	Province in work address.
workAddressCity	String	No	City in work address.
workAddressStreet	String	No	Street in work address.
workAddressPostalCode	String	No	Postcode in work address.
homeFaxNumber	String	No	Home fax.
homePhoneNumber	String	No	Home phone.
homeAddressCountry	String	No	Country in home address.
homeAddressState	String	No	Province in home address.
homeAddressCity	String		

No | City in home address. | | homeAddressStreet | String | No | Street in home address. | | homeAddressPostalCode | String | No | Postcode in home address. | | success | Function | No | Callback function upon call success. | | fail | Function | No | Callback function upon call failure. | | complete | Function | No | Callback function upon call completion (to be executed upon either call success or failure). |

Depending on the support of contact App built in different ROM, the above fields may not support emoji or kaomoji. In such cases, this option is ignored.

Return Value

Success:

success = true

Failure:

| | | | | --- | --- | --- | | **Error** | **ErrorMessage** | **Description** | | 11 | fail cancel | The user cancels the operation. | | 3 | fail ${detail} | Call failure, detail includes the detailed information. |

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

---

my.alert {#myalert}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.alert

2021-05-09 18:43

Alert box

Sample Code

copy

```
my.alert({
  title: 'Tips',
  content: 'Your bill for this month has been released',
  buttonText: 'Show',
  success: () => {
    my.alert({
      title: 'Click  Show  ',
    });
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | title | String | No | Title of the alert box. | | content | String | No | Contents of the alert box. | | buttonText | String | No | Button text, which is OK by default. | | success | Function | No | Callback function upon call success. | | fail | Function | No | Callback function upon call failure. | | complete | Function | No | Callback function upon call completion (to be executed upon either call success or failure). |

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

---

my.alert {#myalert}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.alert

2022-07-03 18:44

Alert box

Sample Code

copy

```
my.alert({
  title: 'Tips',
  content: 'Your bill for this month has been released',
  buttonText: 'Show',
  success: () => {
    my.alert({
      title: 'Click  Show  ',
    });
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | title | String | No | Title of the alert box. | | content | String | No | Contents of the alert box. | | buttonText | String | No | Button text, which is OK by default. | | success | Function | No | Callback function upon call success. | | fail | Function | No | Callback function upon call failure. | | complete | Function | No | Callback function upon call completion (to be executed upon either call success or failure). |

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

---

my.canIUse {#mycaniuse}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.canIUse

2022-07-03 18:44

Check whether the current Mini Program API, incoming parameter or return value, component, attribute, etc. are supported in the current version.

The parameters are called via `${API}.${type}.${param}.${option}` or `${component}.${attribute}.${option}` mode.

- API is the name of the API
- type value object/return/callback, indicating the api judgment type
- param indicates an attribute name of the parameter
- option indicates the detailed attribute value of the parameter attribute
- component indicates the name of the component
- attribute indicates the name of the component attribute
- option indicates the value of the component attribute

Sample Code

copy

```
// check whether newly added API is available
my.canIUse('getFileInfo')
```

```
// check whether newly added property of API is available
my.canIUse('getLocation.object.type')
// check whether newly added returned property of API is available
my.canIUse('getSystemInfo.return.brand')
// check whether newly added property of component is available
my.canIUse('button.open-type.getAuthorize')
```

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

---

my.canIUse {#mycaniuse}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.canIUse

2021-05-09 18:43

Check whether the current Mini Program API, incoming parameter or return value, component, attribute, etc. are supported in the current version.

The parameters are called via `${API}.${type}.${param}.${option}` or `${component}.${attribute}.${option}` mode.

- API is the name of the API
- type value object/return/callback, indicating the api judgment type
- param indicates an attribute name of the parameter
- option indicates the detailed attribute value of the parameter attribute
- component indicates the name of the component
- attribute indicates the name of the component attribute
- option indicates the value of the component attribute

Sample Code

copy

```
// check whether newly added API is available
my.canIUse('getFileInfo')
// check whether newly added property of API is available
my.canIUse('getLocation.object.type')
```

```
// check whether newly added returned property of API is available
my.canIUse('getSystemInfo.return.brand')
// check whether newly added property of component is available
my.canIUse('button.open-type.getAuthorize')
```

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

---

my.chooseFileFromDisk {#mychoosefilefromdisk}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.chooseFileFromDisk

2022-07-03 18:44

You can choose a file to upload. You can also view the details of the file or delete the file you have added.

Sample code

copy

```
<!--.axml-->
<view class="page">
  <view class="page-description">file API</view>
    <view class="page-section-demo">
      <button class="page-body-button"
onTap="chooseFileFromDisk">Choose file from disk</button>
    </view>
  </view>
</view>
```

copy

```
// .js
Page({
  chooseFileFromDisk(){
    my.chooseFileFromDisk({
      success: (res) => {
        my.alert({
          content: JSON.stringify(res),
        });
      },
      fail: (res) => {
        my.alert({
          content: JSON.stringify(res),
        });
      }
    });
  },
});
```

Parameters

Property	**Type**	**Description**	**Required**
success	Function	The callback function for a successful API call.	NO
fail	Function	The callback function for a failed API call.	NO
complete	Function	The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails.	NO

Success callback function

Property	**Type**	**Description**
apFilePath	string	The temporary local file path.

Failure callback function

Property	**Type**	**Description**
error	number	Error code.
errorMessage	string	Error message.

Error codes

Error code	**Error message**	**Remark**
4	The JSAPI call is denied.	The mini program that is developed by the merchant/ISV has no right to call the JSAPI.
12	Errors occur when copying the file.	
15	The user cancels the file selection.	
16	The user denies the permission to read the storage.	Android Only.
17	No permission to read the storage.	Android Only.
18	Choosing large file is not supported.	

Note:

The following list describes the difference between the error code 16 and 17:

- 16: The user denies the permission to read the storage for the current request.
- 17: The user denied the permission last time and chose to never asking the permission again. In this case, the dialog box for the permission cannot pop up. The error code of 17 is returned directly. The mini program developer requires to call the showAuthGuide to guide the user to grant the related permission.

File size limit

If you choose a file that is too large, the mini program or the wallet app cannot run properly. For this reason, the file size is limited to 50MB. When the file size exceeds 50MB, the error code of 18 is returned and the corresponding error message is displayed.

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

---

my.chooseFileFromDisk {#mychoosefilefromdisk}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.chooseFileFromDisk

2021-05-09 18:43

You can choose a file to upload. You can also view the details of the file or delete the file you have added.

Sample code

copy

```
<!--.axml-->
<view class="page">
  <view class="page-description">file API</view>
    <view class="page-section-demo">
      <button class="page-body-button"
onTap="chooseFileFromDisk">Choose file from disk</button>
    </view>
  </view>
</view>
```

copy

```
// .js
Page({
  chooseFileFromDisk(){
    my.chooseFileFromDisk({
      success: (res) => {
        my.alert({
          content: JSON.stringify(res),
        });
      },
      fail: (res) => {
        my.alert({
          content: JSON.stringify(res),
        });
      }
    });
  },
});
```

Parameters

Property	**Type**	**Description**	**Required**
success	Function	The callback function for a successful API call.	NO
fail	Function	The callback function for a failed API call.	NO
complete	Function	The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails.	NO

Success callback function

Property	**Type**	**Description**
apFilePath	string	The temporary local file path.

Failure callback function

Property	**Type**	**Description**
error	number	Error code.
errorMessage	string	Error message.

Error codes

Error code	**Error message**	**Remark**
4	The JSAPI call is denied.	The mini program that is developed by the merchant/ISV has no right to call the JSAPI.
12	Errors occur when copying the file.	
15	The user cancels the file selection.	
16	The user denies the permission to read the storage.	Android Only.
17	No permission to read the storage.	Android Only.
18	Choosing large file is not supported.	

Note:

The following list describes the difference between the error code 16 and 17:

- 16: The user denies the permission to read the storage for the current request.
- 17: The user denied the permission last time and chose to never asking the permission again. In this case, the dialog box for the permission cannot pop up. The error code of 17 is returned directly. The mini program developer requires to call the showAuthGuide to guide the user to grant the related permission.

File size limit

If you choose a file that is too large, the mini program or the wallet app cannot run properly. For this reason, the file size is limited to 50MB. When the file size exceeds 50MB, the error code of 18 is returned and the corresponding error message is displayed.

九色鹿

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

---

my.chooseImage {#mychooseimage}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.chooseImage

2021-05-09 18:43

Choose an image from the camera or gallery of a device.

Sample Code

copy

```
my.chooseImage({
  success: (res) => {
    img.src = res.apFilePaths[0];
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | sizeType | StringArray | No | Original image, compressed image, both by default. | | sourceType | String Array | No | Camera or album, [‘camera’,‘album’] by default. | | success | Function | No | Callback function upon call success. | | fail | Function | No | Callback function upon call failure. | | complete | Function | No | Callback function upon call completion (to be executed upon either call success or failure). |

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

Property	**Type**	**Description**
apFilePaths	String Array	Image file description.

Error Code

Error	**Description**
11	User cancels operation.

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

---

my.chooseImage {#mychooseimage}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.chooseImage

2022-07-03 18:44

Choose an image from the camera or gallery of a device.

Sample Code

copy

```
my.chooseImage({
  success: (res) => {
    img.src = res.apFilePaths[0];
  },
});
```

Parameters

Property	**Type**	**Required**	**Description**
sizeType	StringArray	No	Original image, compressed image, both by default.
sourceType	String Array	No	Camera or album, [‘camera’,‘album’] by default.
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | apFilePaths | String Array | Image file description. |

Error Code

| | | | --- | --- | | **Error** | **Description** | | 11 | User cancels operation. |

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

---

my.chooseLocation {#mychooselocation}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.chooseLocation

2022-07-03 18:44

Open the built-in map to choose a location.

Sample codes

JSON

copy

```
// API-DEMO page/API/choose-location/choose-location.json
{
    "defaultTitle": "Choose Location"
}
```

AXML

copy

```
<!-- API-DEMO page/API/choose-location/choose-location.axml-->
<view class="page">
  <view class="page-section">
```

```
    <view class="page-section-demo">
      <text>Longitude:</text>
      <input value="{{longitude}}"></input>
    </view>
    <view class="page-section-demo">
      <text>Latitude:</text>
      <input value="{{latitude}}"></input>
    </view>
    <view class="page-section-demo">
      <text>Name:</text>
      <input value="{{name}}"></input>
    </view>
    <view class="page-section-demo">
      <text>Address:</text>
      <input value="{{address}}"></input>
    </view>
    <view class="page-section-btns">
      <view onTap="chooseLocation">choose Location</view>
    </view>
  </view>
</view>
```

JavaScript

copy

```
// API-DEMO page/API/choose-location/choose-location.js
Page({
  data: {
    longitude: '103.873834',
    latitude: '1.355572',
    name: 'Serangoon Stadium',
    address: '33 Yio Chu Kang Rd, Singapore',
  },
  chooseLocation() {
    var that = this
    my.chooseLocation({
         success:(res)=>{
          console.log(JSON.stringify(res))
          that.setData({
            longitude:res.longitude,
            latitude:res.latitude,
            name:res.name,
            address:res.address
          })
        },
        fail:(error)=>{
          my.alert({content: 'failed  '+JSON.stringify(error)});
        },
    });
    },
})
```

ACSS

copy

```
/* API-DEMO page/API/choose-location/choose-location.acss */
.page-body-info {
  height: 250rpx;
}
.page-body-text-location {
  display: flex;
  font-size: 50rpx;
}
.page-body-text-location text {
  margin: 10rpx;
}
.page-section-location-text{
    color: #49a9ee;
}
```

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

Property	**Type**	**Description**
name	String	The location.
address	String	The detailed address of the location.
latitude	Number	The latitude that is expressed by a floating-point number. The value ranges from -90 to +90, and the negative number means south latitude.
longitude	Number	The longitude that is expressed by a floating-point number. The value ranges from -180 to +180, and the negative number means west longitude.
provinceName	String	The province.
cityName	String	The city.

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

---

my.chooseLocation {#mychooselocation}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.chooseLocation

2021-05-09 18:43

Open the built-in map to choose a location.

Sample codes

JSON

copy

```
// API-DEMO page/API/choose-location/choose-location.json
{
    "defaultTitle": "Choose Location"
}
```

AXML

copy

```
<!-- API-DEMO page/API/choose-location/choose-location.axml-->
<view class="page">
  <view class="page-section">
    <view class="page-section-demo">
      <text>Longitude:</text>
      <input value="{{longitude}}"></input>
    </view>
    <view class="page-section-demo">
      <text>Latitude:</text>
      <input value="{{latitude}}"></input>
    </view>
    <view class="page-section-demo">
      <text>Name:</text>
      <input value="{{name}}"></input>
    </view>
    <view class="page-section-demo">
      <text>Address:</text>
      <input value="{{address}}"></input>
    </view>
    <view class="page-section-btns">
      <view onTap="chooseLocation">choose Location</view>
    </view>
  </view>
</view>
```

JavaScript

copy

```
// API-DEMO page/API/choose-location/choose-location.js
Page({
  data: {
    longitude: '103.873834',
    latitude: '1.355572',
    name: 'Serangoon Stadium',
    address: '33 Yio Chu Kang Rd, Singapore',
  },
  chooseLocation() {
    var that = this
    my.chooseLocation({
         success:(res)=>{
          console.log(JSON.stringify(res))
          that.setData({
            longitude:res.longitude,
            latitude:res.latitude,
            name:res.name,
            address:res.address
          })
        },
        fail:(error)=>{
          my.alert({content: 'failed  '+JSON.stringify(error)});
        },
    });
    },
})
```

ACSS

copy

```
/* API-DEMO page/API/choose-location/choose-location.acss */
.page-body-info {
  height: 250rpx;
}
.page-body-text-location {
  display: flex;
  font-size: 50rpx;
}
.page-body-text-location text {
  margin: 10rpx;
}
.page-section-location-text{
    color: #49a9ee;
}
```

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

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | name | String | The location. | | address | String | The detailed address of the location. | | latitude | Number | The latitude that is expressed by a floating-point number. The value ranges from -90 to +90, and the negative number means south latitude. | | longitude | Number | The longitude that is expressed by a floating-point number. The value ranges from -180 to +180, and the negative number means west longitude. | | provinceName | String | The province. | | cityName | String | The city. |

九色鹿

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

---

my.choosePhoneContact {#mychoosephonecontact}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.choosePhoneContact

2021-05-09 18:43

Select the phone number of a contact in the local system directory.

Sample Code

copy

```
//.json
{
  "defaultTitle": "Contact"
}
```

copy

```
<!-- .axml -->
<view class="page">
```

```
  <view class="page-description">Contact API</view>

  <view class="page-section">
    <view class="page-section-title">my.choosePhoneContact</view>
    <view class="page-section-demo">
      <button type="primary" onTap="choosePhoneContact">Evoke local
directory</button>
    </view>
  </view>

  </view>

</view>
```

copy

```
//.js
Page({
  choosePhoneContact() {
    my.choosePhoneContact({
      success: (res) => {
        my.alert({
          content: 'choosePhoneContact response: ' +
JSON.stringify(res)
        });
      },
      fail: (res) => {
        my.alert({
          content: 'choosePhoneContact response: ' +
JSON.stringify(res)
        });
      },
    });
  }
});
```

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | name | String | Selected contact name. | | mobile | String | Selected contact phone. |

Error Code

| | | | --- | --- | | **Error Code** | **Description** | | 10 | No permission. | | 11 | The user cancels the operation (or the device does not authorize use of the directory). |

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

---

my.choosePhoneContact {#mychoosephonecontact}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.choosePhoneContact

2022-07-03 18:44

Select the phone number of a contact in the local system directory.

Sample Code

copy

```
//.json
{
   "defaultTitle": "Contact"
}
```

copy

```
<!-- .axml -->
<view class="page">

  <view class="page-description">Contact API</view>

  <view class="page-section">
    <view class="page-section-title">my.choosePhoneContact</view>
    <view class="page-section-demo">
      <button type="primary" onTap="choosePhoneContact">Evoke local
directory</button>
    </view>
  </view>
  </view>

</view>
```

copy

```
//.js
Page({
  choosePhoneContact() {
    my.choosePhoneContact({
      success: (res) => {
        my.alert({
          content: 'choosePhoneContact response: ' +
JSON.stringify(res)
        });
      },
      fail: (res) => {
        my.alert({
          content: 'choosePhoneContact response: ' +
JSON.stringify(res)
        });
      },
    });
  }
});
```

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

Property	**Type**	**Description**
name	String	Selected contact name.
mobile	String	Selected contact phone.

Error Code

Error Code	**Description**
10	No permission.
11	The user cancels the operation (or the device does not authorize use of the directory).

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

---

my.clearStorage {#myclearstorage}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.clearStorage

2022-07-03 18:44

Clear local data cache.

> This is an asynchronous interface.
>
> Clearing the webview embedded storage data will not clear the storage data of the Mini Program.

Sample Code

copy

```
my.clearStorage();
```

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

---

my.clearStorage {#myclearstorage}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.clearStorage

2021-05-09 18:43

Clear local data cache.

> This is an asynchronous interface.
>
> Clearing the webview embedded storage data will not clear the storage data of the Mini Program.

Sample Code

copy

```
my.clearStorage();
```

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

---

my.clearStorageSync {#myclearstoragesync}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.clearStorageSync

2022-07-03 18:44

Clear local data cache synchronously.

> This is a synchronous interface.

Sample Code

copy

```
my.clearStorageSync();
```

九色鹿

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

---

my.clearStorageSync {#myclearstoragesync}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.clearStorageSync

2021-05-09 18:43

Clear local data cache synchronously.

> This is a synchronous interface.

Sample Code

copy

```
my.clearStorageSync();
```

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

---

my.closeBluetoothAdapter {#myclosebluetoothadapter}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.closeBluetoothAdapter

2022-07-03 18:44

Use this API to close the Bluetooth module in the mini program. You can call the following mini program API and receive event callbacks that are related to the Bluetooth module in the effective period when you the API `my.openBluetoothAdapter` is called. The effective period is ended when the API `my.closeBluetoothAdapter` is called.

Instructions :

- Call this API to disconnect all the established Bluetooth connections and releases system resources.
- It's recommend to call this API after you end the Bluetooth process of the Mini Program, which is in pair with the API my.openBluetoothAdapter.
- Calling the API `my.closeBluetoothAdapter` is asynchronous to releasing resources. It's recommend to call the API `my.closeBluetoothAdapter` and `my.openBluetoothAdapter` as part of the exception handling process. Synchronization errors might error if you reinitialize the process.

Note:

Currently simulation in IDE is not supported. Please debug in the production environment.

Code Sample

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
```

```
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Unlistens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Unlistens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary"
onTap="offBLEConnectionStateChanged">Unlistens to Bluetooth connection
state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
        devid = res.devices[0].deviceId;
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain all searched devices
  getBluetoothDevices() {
    my.getBluetoothDevices({
      success: res => {
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  bindKeyInput(e) {
    this.setData({
      devid: e.detail.value,
    });
  },

  //Connect the device
  connectBLEDevice() {
    my.connectBLEDevice({
      deviceId: this.data.devid,
      success: res => {
        my.alert({ content: 'Succeeded to connect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Disconnect the device
  disconnectBLEDevice() {
    my.disconnectBLEDevice({
      deviceId: this.data.devid,
      success: () => {
        my.alert({ content: 'Succeeded to disconnect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Obtain the char ID of the connected device, read and write
characteristics are respectively screened out.
  getBLEDeviceCharacteristics() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.getBLEDeviceCharacteristics({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            //See the related document for more information of the
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
```

```
              this.setData({
                charid: res.characteristics[0].characteristicId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Read and write data
    readBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.readBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            //1  Android reading service
            // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
            success: res => {
              my.alert({ content: JSON.stringify(res) });
            },
            fail: error => {
              my.alert({ content: 'Failed to read' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    writeBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.writeBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.charid,
            //Android writing service
            //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
            value: 'ABCD',
            success: res => {
              my.alert({ content: 'Succeeded to write data!' });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },
    notifyBLECharacteristicValueChange() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.notifyBLECharacteristicValueChange({
            state: true,
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            success: () => {
              //Listens to characteristic change events
              my.onBLECharacteristicValueChange({
                success: res => {
                  //  my.alert({content: 'Changes of
characteristics  '+JSON.stringify(res)});
                  my.alert({ content: 'Obtain the response data = ' +
res.value });
                },
              });
              my.alert({ content: 'Succeeded to listen' });
            },
            fail: error => {
              my.alert({ content: 'Failed to listen' +
JSON.stringify(error) });
            },
          });
        },
      });
    },
    offBLECharacteristicValueChange() {
      my.offBLECharacteristicValueChange();
    },

    //Other events
    bluetoothAdapterStateChange() {

  my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
    },
    onBluetoothAdapterStateChange() {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of the Bluetooth state  ' +
  JSON.stringify(res) });
      }
    },
    offBluetoothAdapterStateChange() {

  my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStat
    },
    getBind(name) {
      if (!this[`bind${name}`]) {
        this[`bind${name}`] = this[name].bind(this);
      }
      return this[`bind${name}`];
    },
    BLEConnectionStateChanged() {

  my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
    },
    onBLEConnectionStateChanged(res) {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of connection state  ' +
  JSON.stringify(res) });
      }
    },
    offBLEConnectionStateChanged() {

  my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChan
    },
    onUnload() {
      this.offBLEConnectionStateChanged();
      this.offBLECharacteristicValueChange();
      this.offBluetoothAdapterStateChange();
      this.closeBluetoothAdapter();
  },
});
```

Parameters

The input parameters are displayed in the following table:

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function for a completed API call (Regardless of whether the call is successful or not). |

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

---

my.closeBluetoothAdapter {#myclosebluetoothadapter}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.closeBluetoothAdapter

2021-05-09 18:43

Use this API to close the Bluetooth module in the mini program. You can call the following mini program API and receive event callbacks that are related to the Bluetooth module in the effective period when you the API `my.openBluetoothAdapter` is called. The effective period is ended when the API `my.closeBluetoothAdapter` is called.

Instructions :

- Call this API to disconnect all the established Bluetooth connections and releases system resources.
- It's recommend to call this API after you end the Bluetooth process of the Mini Program, which is in pair with the API my.openBluetoothAdapter.
- Calling the API `my.closeBluetoothAdapter` is asynchronous to releasing resources. It's recommend to call the API `my.closeBluetoothAdapter` and `my.openBluetoothAdapter` as part of the exception handling process. Synchronization errors might error if you reinitialize the process.

Note:

Currently simulation in IDE is not supported. Please debug in the production environment.

Code Sample

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
```

```
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Unlistens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Unlistens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary"
onTap="offBLEConnectionStateChanged">Unlistens to Bluetooth connection
state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {
```

```
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
```

```
          my.alert({ content: JSON.stringify(res) });
          devid = res.devices[0].deviceId;
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain all searched devices
    getBluetoothDevices() {
      my.getBluetoothDevices({
        success: res => {
          my.alert({ content: JSON.stringify(res) });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },
    bindKeyInput(e) {
      this.setData({
        devid: e.detail.value,
      });
    },

    //Connect the device
    connectBLEDevice() {
      my.connectBLEDevice({
        deviceId: this.data.devid,
        success: res => {
          my.alert({ content: 'Succeeded to connect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Disconnect the device
    disconnectBLEDevice() {
      my.disconnectBLEDevice({
        deviceId: this.data.devid,
        success: () => {
          my.alert({ content: 'Succeeded to disconnect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },
    //Obtain the services of the connected device
    getBLEDeviceServices() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          my.getBLEDeviceServices({
            deviceId: this.data.devid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              this.setData({
                serid: res.services[0].serviceId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Obtain the char ID of the connected device, read and write
  characteristics are respectively screened out.
    getBLEDeviceCharacteristics() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.getBLEDeviceCharacteristics({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              //See the related document for more information of the
  properties of the characteristics. Pair the characteristics according
  to the properties and record the value for later use.
              this.setData({
                charid: res.characteristics[0].characteristicId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Read and write data
    readBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.readBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            //1  Android reading service
            // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
            success: res => {
              my.alert({ content: JSON.stringify(res) });
            },
            fail: error => {
              my.alert({ content: 'Failed to read' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    writeBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.writeBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.charid,
            //Android writing service
            //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
            value: 'ABCD',
            success: res => {
              my.alert({ content: 'Succeeded to write data!' });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },
    notifyBLECharacteristicValueChange() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.notifyBLECharacteristicValueChange({
            state: true,
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            success: () => {
              //Listens to characteristic change events
              my.onBLECharacteristicValueChange({
                success: res => {
                  //  my.alert({content: 'Changes of 
  characteristics  '+JSON.stringify(res)});
                  my.alert({ content: 'Obtain the response data = ' + 
  res.value });
                },
              });
              my.alert({ content: 'Succeeded to listen' });
            },
            fail: error => {
              my.alert({ content: 'Failed to listen' + 
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    offBLECharacteristicValueChange() {
      my.offBLECharacteristicValueChange();
    },
    //Other events
    bluetoothAdapterStateChange() {

  my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStateC
    },
    onBluetoothAdapterStateChange() {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of the Bluetooth state  ' +
  JSON.stringify(res) });
      }
    },
    offBluetoothAdapterStateChange() {

  my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
    },
    getBind(name) {
      if (!this[`bind${name}`]) {
        this[`bind${name}`] = this[name].bind(this);
      }
      return this[`bind${name}`];
    },
    BLEConnectionStateChanged() {

  my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChange
    },
    onBLEConnectionStateChanged(res) {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of connection state  ' +
  JSON.stringify(res) });
      }
    },
    offBLEConnectionStateChanged() {

  my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
    },
    onUnload() {
      this.offBLEConnectionStateChanged();
      this.offBLECharacteristicValueChange();
      this.offBluetoothAdapterStateChange();
      this.closeBluetoothAdapter();
    },
  });
```

Parameters

The input parameters are displayed in the following table:

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function for a completed API call (Regardless of whether the call is successful or not). |

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

---

my.closeSocket {#myclosesocket}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.closeSocket

2022-07-03 18:44

Use this API to disable the WebSocket connection.

Sample Code

copy

```
my.onSocketOpen(function() {
  my.closeSocket()
})

my.onSocketClose(function(res) {
  console.log('The WebSocket is closed!')
})
```

Note: The case is only for reference. Please use your own URL to test.

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

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

---

my.closeSocket {#myclosesocket}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.closeSocket

2021-05-09 18:43

Use this API to disable the WebSocket connection.

Sample Code

copy

```
my.onSocketOpen(function() {
  my.closeSocket()
})

my.onSocketClose(function(res) {
  console.log('The WebSocket is closed!')
})
```

Note: The case is only for reference. Please use your own URL to test.

Parameters

| | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

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

---

my.compressImage {#mycompressimage}

Last updated: 2021-05-10

Path: miniprogram_gcash

my.compressImage

2021-05-10 03:43

Compress large images to fulfill the size restrictions from some mini programs.

Parameters

Property	**Type**	**Required**	**Description**
apFilePaths	Array	Yes	An array of the image paths to be compressed.
compressLevel	int	No	The image quality after the compress. Valid values are: - *0*: Low quality - *1*: Medium quality - *2:* High quality - *4*: The quality depends on the network. When the network is Wi-Fi, the image is compressed to the high-quality image. Otherwise, the image is compressed to the medium-quality image. The default value is *4*.
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

Property	**Type**	**Required**	**Description**
apFilePaths	Array	Yes	The local file path of the compressed image.

Error code

Error Code	**Error Message**	**Description**
2	Compress failed	Failed to compress all images.
Invalid parameter	Failed to compress one of all images.	

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

---

my.compressImage {#mycompressimage}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.compressImage

2022-07-03 18:44

Compress large images to fulfill the size restrictions from some mini programs.

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | apFilePaths | Array | Yes | An array of the image paths to be compressed. | | compressLevel | int | No | The image quality after the compress. Valid values are:
- *0*: Low quality
- *1*: Medium quality
- *2:* High quality
- *4*: The quality depends on the network. When the network is Wi-Fi, the image is compressed to the high-quality image. Otherwise, the image is compressed to the medium-quality image.
The default value is *4*. | | 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

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | apFilePaths | Array | Yes | The local file path of the compressed image. |

Error code

| | | | | --- | --- | --- | | **Error Code** | **Error Message** | **Description** | | 2 | Compress failed | Failed to compress all images. | | Invalid parameter | Failed to compress one of all images. |

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

---

my.confirm {#myconfirm}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.confirm

2021-05-09 18:43

Confirm box.

Sample Code

copy

```
my.confirm({
  title: 'Tips',
  content: 'Do you want to check the courier number: 1234567890?',
  confirmButtonText: 'Inquire now',
  cancelButtonText: 'Not needed',
  success: (result) => {
    my.alert({
      title: `${result.confirm}`,
    });
  },
});
```

Parameters

Property	**Type**	**Required**	**Description**
title	String	No	Title of the confirm box.
content	String	No	Content of the confirm box.
confirmButtonText	String	No	OK button text, which is “OK” by default.
cancelButtonText	String	No	OK button text, which is “Cancel” by default.
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

Property	**Type**	**Description**
confirm	Boolean	Click Confirm to return true; click Cancel to return false.

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

---

my.connectBLEDevice {#myconnectbledevice}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.connectBLEDevice

2021-05-09 18:43

Use this API to connect to a Bluetooth Low Energy (BLE) device.

Instructions:

- If the Mini Program has previously discovered a Bluetooth device and successfully connected to it, you can directly pass the device ID obtained previously to connect to the device, without the need to perform a search operation.
- If the specified Bluetooth device is already connected, repeated connection request will return success directly.

Note: Currently simulation in IDE is not supported. Please debug in production environment.

Sample Code

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>
```

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {

            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];

              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
```

```
          my.alert({ content: 'Succeeded!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain the connected device
    getConnectedBluetoothDevices() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connecting devices!' });
            return;
          }
          my.alert({ content: JSON.stringify(res) });
          devid = res.devices[0].deviceId;
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain all searched devices
    getBluetoothDevices() {
      my.getBluetoothDevices({
        success: res => {
          my.alert({ content: JSON.stringify(res) });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },
    bindKeyInput(e) {
      this.setData({
        devid: e.detail.value,
      });
    },

    //Connect the device
    connectBLEDevice() {
      my.connectBLEDevice({
        deviceId: this.data.devid,
        success: res => {
          my.alert({ content: 'Succeeded to connect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Disconnect the device
    disconnectBLEDevice() {
      my.disconnectBLEDevice({
        deviceId: this.data.devid,
        success: () => {
          my.alert({ content: 'Succeeded to disconnect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain the services of the connected device
    getBLEDeviceServices() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          my.getBLEDeviceServices({
            deviceId: this.data.devid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              this.setData({
                serid: res.services[0].serviceId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Obtain the char ID of the connected device, read and write
  characteristics are respectively screened out.
    getBLEDeviceCharacteristics() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
```

```
          devid: res.devices[0].deviceId,
        });
        my.getBLEDeviceCharacteristics({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });

            //See the related document for more information of the
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
            this.setData({
              charid: res.characteristics[0].characteristicId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Read and write data
  readBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.readBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,

          //1  Android reading service
          // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
          success: res => {
            my.alert({ content: JSON.stringify(res) });
          },
          fail: error => {
            my.alert({ content: 'Failed to read' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  writeBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.writeBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.charid,

          //Android writing service
          //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
          value: 'ABCD',
          success: res => {
            my.alert({ content: 'Succeeded to write data!' });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },
  notifyBLECharacteristicValueChange() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.notifyBLECharacteristicValueChange({
          state: true,
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          success: () => {

            //Listens to characteristic change events
            my.onBLECharacteristicValueChange({
              success: res => {
```

```
                //  my.alert({content: 'Changes of 
characteristics  '+JSON.stringify(res)});
                my.alert({ content: 'Obtain the response data = ' + 
res.value });
              },
            });
            my.alert({ content: 'Succeeded to listen' });
          },
          fail: error => {
            my.alert({ content: 'Failed to listen' + 
JSON.stringify(error) });
          },
        });
      },
    });
  },
  offBLECharacteristicValueChange() {
    my.offBLECharacteristicValueChange();
  },

  //Other events
  bluetoothAdapterStateChange() {
    
my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  onBluetoothAdapterStateChange() {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of the Bluetooth state  ' + 
JSON.stringify(res) });
    }
  },
  offBluetoothAdapterStateChange() {
    
my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  getBind(name) {
    if (!this[`bind${name}`]) {
      this[`bind${name}`] = this[name].bind(this);
    }
    return this[`bind${name}`];
  },
  BLEConnectionStateChanged() {
    
my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChange
  },
  onBLEConnectionStateChanged(res) {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of connection state  ' +
JSON.stringify(res) });
    }
  },
  offBLEConnectionStateChanged() {

my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChan
  },
  onUnload() {
    this.offBLEConnectionStateChanged();
    this.offBLECharacteristicValueChange();
    this.offBluetoothAdapterStateChange();
    this.closeBluetoothAdapter();
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | deviceId | String | Yes | The Bluetooth device ID. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

九色鹿

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

---

my.connectBLEDevice {#myconnectbledevice}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.connectBLEDevice

2022-07-03 18:44

Use this API to connect to a Bluetooth Low Energy (BLE) device.

Instructions:

- If the Mini Program has previously discovered a Bluetooth device and successfully connected to it, you can directly pass the device ID obtained previously to connect to the device, without the need to perform a search operation.

- If the specified Bluetooth device is already connected, repeated connection request will return success directly.

Note: Currently simulation in IDE is not supported. Please debug in production environment.

Sample Code

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
```

```
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>
    </view>
  </view>
</view>
```

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {

            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];

              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
```

```
getConnectedBluetoothDevices() {
  my.getConnectedBluetoothDevices({
    success: res => {
      if (res.devices.length === 0) {
        my.alert({ content: 'No connecting devices!' });
        return;
      }
      my.alert({ content: JSON.stringify(res) });
      devid = res.devices[0].deviceId;
    },
    fail: error => {
      my.alert({ content: JSON.stringify(error) });
    },
  });
},

//Obtain all searched devices
getBluetoothDevices() {
  my.getBluetoothDevices({
    success: res => {
      my.alert({ content: JSON.stringify(res) });
    },
    fail: error => {
      my.alert({ content: JSON.stringify(error) });
    },
  });
},
bindKeyInput(e) {
  this.setData({
    devid: e.detail.value,
  });
},

//Connect the device
connectBLEDevice() {
  my.connectBLEDevice({
    deviceId: this.data.devid,
    success: res => {
      my.alert({ content: 'Succeeded to connect!' });
    },
    fail: error => {
      my.alert({ content: JSON.stringify(error) });
    },
  });
},

//Disconnect the device
disconnectBLEDevice() {
  my.disconnectBLEDevice({
    deviceId: this.data.devid,
    success: () => {
          my.alert({ content: 'Succeeded to disconnect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain the services of the connected device
    getBLEDeviceServices() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          my.getBLEDeviceServices({
            deviceId: this.data.devid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              this.setData({
                serid: res.services[0].serviceId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Obtain the char ID of the connected device, read and write
  characteristics are respectively screened out.
    getBLEDeviceCharacteristics() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.getBLEDeviceCharacteristics({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });

              //See the related document for more information of the
```

```
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
            this.setData({
              charid: res.characteristics[0].characteristicId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Read and write data
  readBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.readBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,

          //1  Android reading service
          // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
          success: res => {
            my.alert({ content: JSON.stringify(res) });
          },
          fail: error => {
            my.alert({ content: 'Failed to read' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  writeBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
```

```
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.writeBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.charid,

            //Android writing service
            //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
            value: 'ABCD',
            success: res => {
              my.alert({ content: 'Succeeded to write data!' });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },
    notifyBLECharacteristicValueChange() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.notifyBLECharacteristicValueChange({
            state: true,
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            success: () => {

              //Listens to characteristic change events
              my.onBLECharacteristicValueChange({
                success: res => {

                  //  my.alert({content: 'Changes of
  characteristics  '+JSON.stringify(res)});
                  my.alert({ content: 'Obtain the response data = ' +
  res.value });
                },
              });
              my.alert({ content: 'Succeeded to listen' });
            },
          fail: error => {
            my.alert({ content: 'Failed to listen' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  offBLECharacteristicValueChange() {
    my.offBLECharacteristicValueChange();
  },

  //Other events
  bluetoothAdapterStateChange() {

my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  onBluetoothAdapterStateChange() {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of the Bluetooth state  ' +
JSON.stringify(res) });
    }
  },
  offBluetoothAdapterStateChange() {

my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStat
  },
  getBind(name) {
    if (!this[`bind${name}`]) {
      this[`bind${name}`] = this[name].bind(this);
    }
    return this[`bind${name}`];
  },
  BLEConnectionStateChanged() {

my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
  },
  onBLEConnectionStateChanged(res) {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of connection state  ' +
JSON.stringify(res) });
    }
  },
  offBLEConnectionStateChanged() {

my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChan
  },
  onUnload() {
    this.offBLEConnectionStateChanged();
    this.offBLECharacteristicValueChange();
    this.offBluetoothAdapterStateChange();
    this.closeBluetoothAdapter();
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | deviceId | String | Yes | The Bluetooth device ID. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

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

---

my.connectSocket {#myconnectsocket}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.connectSocket

2021-05-09 18:43

Use this API to create a WebSocket connection. An Mini Program can only have one WebSocket connection at a time. If a WebSocket connection already exists when a new one is created, the existing one will be automatically disabled.

Sample Code

copy

```
my.connectSocket({
  url: 'test.php',
  data: {},
  header:{
    'content-type': 'application/json'
  },
});
```

Note: The case is only for reference. Please use your own URL to test.

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | url | String | Yes | The address of target server interface.
Note:
Some newly released Mini Programs only support WSS protocol. | | data | Object | No | The request parameters. | | header | Object | No | Header of the request. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

Error Code

| | | | | --- | --- | --- | | **Error Code** | **Description** | **Solution** | | 1 | An unknown error. | - | | 2 | A network connection already exists. | An Mini Program can only keep one WebSocket connection for a period of time. If a WebSocket connection already exists when a new one is created, the existing one will be automatically disabled. | | 3 | The URL parameter is null. | Replace the URL link. | | 4 | An unrecognized URL format. | Replace the URL link. | | 5 | The URL must start with WS or WSS. | Replace the URL link. | | 6 | Connection timed out. | Try again later. | | 7 | The HTTPS certificate returned by the server is invalid. | The Mini Program must start a network request using HTTPS/WSS. When a request is sent, the HTTPS certificate of the server domain name is checked. If the check fails, the request cannot be successfully initiated. Due to system limitations, different platforms have different requirements for certificates. To ensure the compatibility of Mini Programs, developers are recommended to configure certificates according to the highest standards and use relevant tools to check existing certificates to ensure that the certificates are valid. | | 8 | The protocol header returned by the server is invalid. | Starting from May 2019, newly created Mini Programs must use HTTPS and WSS protocols by default and HTTP and WS protocols are not supported. | | 9 | The Sec-WebSocket-Protocol request header is not specified for the WebSocket request. | Please specify the Sec-WebSocket-Protocol request header. | | 10 | The network is not available and the message cannot be sent. | Please call my.sendSocketMessage to send a data message after connecting to the server normally. Use my.onSocketOpen to check if the connection with the server is correct.
Note:
To send data through a WebSocket connection, you need to use `my.connectSocket` to start the connection first, and then call my.sendSocketMessage to send data after the my.onSocketOpen callback. | | 11 | Failed to send message. | Try again later. | | 12 | Unable to request more memory to read network data. | Please check the memory. |

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

---

my.connectSocket {#myconnectsocket}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.connectSocket

2022-07-03 18:44

Use this API to create a WebSocket connection. An Mini Program can only have one WebSocket connection at a time. If a WebSocket connection already exists when a new one is created, the existing one will be automatically disabled.

Sample Code

copy

```
my.connectSocket({
  url: 'test.php',
  data: {},
  header:{
    'content-type': 'application/json'
  },
});
```

Note: The case is only for reference. Please use your own URL to test.

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | url | String | Yes | The address of target server interface.
Note:
Some newly released Mini Programs only support WSS protocol. | | data | Object | No | The request parameters. | | header | Object | No | Header of the request. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

Error Code

| | | | | --- | --- | --- | | **Error Code** | **Description** | **Solution** | | 1 | An unknown error. | - | | 2 | A network connection already exists. | An Mini Program can only keep one WebSocket connection for a period of time. If a WebSocket connection already exists when a new one is created, the existing one will be automatically disabled. | | 3 | The URL parameter is null. | Replace the URL link. | | 4 | An unrecognized URL format. | Replace the URL link. | | 5 | The URL must start with WS or WSS. | Replace the URL link. | | 6 | Connection timed out. | Try again later. | | 7 | The HTTPS certificate returned by the server is invalid. | The Mini Program must start a network request using HTTPS/WSS. When a request is sent, the HTTPS certificate of the server domain name is checked. If

the check fails, the request cannot be successfully initiated. Due to system limitations, different platforms have different requirements for certificates. To ensure the compatibility of Mini Programs, developers are recommended to configure certificates according to the highest standards and use relevant tools to check existing certificates to ensure that the certificates are valid. | | 8 | The protocol header returned by the server is invalid. | Starting from May 2019, newly created Mini Programs must use HTTPS and WSS protocols by default and HTTP and WS protocols are not supported. | | 9 | The Sec-WebSocket-Protocol request header is not specified for the WebSocket request. | Please specify the Sec-WebSocket-Protocol request header. | | 10 | The network is not available and the message cannot be sent. | Please call my.sendSocketMessage to send a data message after connecting to the server normally. Use my.onSocketOpen to check if the connection with the server is correct.
Note:
To send data through a WebSocket connection, you need to use `my.connectSocket` to start the connection first, and then call my.sendSocketMessage to send data after the my.onSocketOpen callback. | | 11 | Failed to send message. | Try again later. | | 12 | Unable to request more memory to read network data. | Please check the memory. |

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

---

my.createAnimation {#mycreateanimation}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.createAnimation

2022-07-03 18:44

Create an animation instance. Call the instance method to describe animation, and then use the `export` method of animation instance to export the animation data and transfer to the component `animation` attribute.

Note: After the `export` method is called, the previous animation operation will be cleared.

Sample Code

copy

```
//.json
{
    "defaultTitle": "Animation"
}
```

copy

```
<!-- .axml -->
<view class="page">
  <view class="page-description">Animation API</view>
  <view class="page-section">
    <view class="page-section-title">my.createAnimation</view>
    <view class="page-section-demo">
      <view class="animation-element" animation="{{animation}}">
</view>
    </view>
    <view class="page-section-btns">
      <view type="primary" onTap="rotate">Rotate</view>
      <view type="primary" onTap="scale"> Scale</view>
      <view type="primary" onTap="translate">Translate</view>
    </view>
    <view class="page-section-btns">
      <view type="primary" onTap="skew">Skew</view>
      <view type="primary" onTap="rotateAndScale">Rotate and
scale</view>
      <view type="primary" onTap="rotateThenScale">Rotate and then
scale</view>
    </view>
    <view class="page-section-btns">
      <view type="primary" onTap="all">Expand all simultaneously
</view>
      <view type="primary" onTap="allInQueue">Expand all in
order</view>
      <view type="primary" onTap="reset">Reset</view>
    </view>
  </view>
</view>
```

copy

```
//.js
Page({
  onReady() {
    this.animation = my.createAnimation()
  },
  rotate() {
    this.animation.rotate(Math.random() * 720 - 360).step()
    this.setData({ animation: this.animation.export() })
  },
  scale() {
    this.animation.scale(Math.random() * 2).step()
    this.setData({ animation: this.animation.export() })
  },
  translate() {
    this.animation.translate(Math.random() * 100 - 50, Math.random() *
100 - 50).step()
    this.setData({ animation: this.animation.export() })
  },
  skew() {
    this.animation.skew(Math.random() * 90, Math.random() * 90).step()
    this.setData({ animation: this.animation.export() })
  },
  rotateAndScale() {
    this.animation.rotate(Math.random() * 720 - 360)
      .scale(Math.random() * 2)
      .step()
    this.setData({ animation: this.animation.export() })
  },
  rotateThenScale() {
    this.animation.rotate(Math.random() * 720 - 360).step()
      .scale(Math.random() * 2).step()
    this.setData({ animation: this.animation.export() })
  },
  all() {
    this.animation.rotate(Math.random() * 720 - 360)
      .scale(Math.random() * 2)
      .translate(Math.random() * 100 - 50, Math.random() * 100 - 50)
      .skew(Math.random() * 90, Math.random() * 90)
      .step()
    this.setData({ animation: this.animation.export() })
  },
  allInQueue() {
    this.animation.rotate(Math.random() * 720 - 360).step()
      .scale(Math.random() * 2).step()
      .translate(Math.random() * 100 - 50, Math.random() * 100 -
50).step()
      .skew(Math.random() * 90, Math.random() * 90).step()
    this.setData({ animation: this.animation.export() })
  },
  reset() {
    this.animation.rotate3d(0, 0, 0, 0)
      .rotateX(0)
      .rotateY(0)
      .rotateZ(0)
      .scale(1)
      .translate(0, 0)
      .skew(0, 0)
      .step({ duration: 0 })
    this.setData({ animation: this.animation.export() })
  }
})
```

copy

```
.animation-element {
  width: 200rpx;
  height: 200rpx;
  background-color: #108ee9;
  transform: scaleX(1) scaleY(1);
}
```

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
duration	Integer	No	Animation duration, in ms, 400 by default.
timeFunction	String	No	Define animation effect, linear by default, effective values including linear, ease, ease-in, ease-in-out, ease-out, step-start and step-end .
delay	Integer	No	Animation delay, in ms, 0 by default.
transformOrigin	String	No	Set transform-origin, 50% 50% 0 by default.

Sample Code

copy

```
//.js
const animation = my.createAnimation({
  transformOrigin: "top right",
  duration: 3000,
  timeFunction: "ease-in-out",
  delay: 100,
})
```

Animation

The animation instance may call the following method to describe the animation. At the end of the call, the instance itself is returned. The chain call style is supported. When the view animation attribute is initialized as `{}`, error may appears on basic library 1.11.0 (not including 1.11.0) and lower version. It is recommended to initialize as `null`.

Style

Method	**Parameters**	**Description**
opacity	value	Transparency, range 0~1.
backgroundColor	color	Color value.
width	length	Set the width:length values, in px, such as 300 px.
height	length	Set the height:length values, in px, such as 300 px.
top	length	Set the top:length values, in px, such as 300 px.
left	length	Set the left:length values, in px, such as 300 px.
bottom	length	Set the bottom:length values, in px, such as 300 px.
right	length	Set the right:length values, in px, such as 300 px.

Rotation

Method	**Parameters**	**Description**
rotate	deg	Deg range -180 ~ 180, rotate by deg degrees clockwise from origin.
rotateX	deg	Deg range -180 ~ 180, rotate by deg degrees on X axis.
rotateY	deg	Deg range -180 ~ 180, rotate by deg

degrees on Y axis. | | rotateZ | deg | Deg range -180 ~ 180, rotate by deg degrees on Z axis. | | rotate3d | (x, y , z, deg) | Same as transform-function rotate3d. |

Scale

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | scale | sx,[sy] | When there is only one parameter, it indicates scaling sx times on X and Y axises at the same time. When there are two parameters, it indicates scaling sx times on X axis and sy times on Y axis. | | scaleX | sx | Scale sx times on X axis. | | scaleY | sy | Scale sy times on Y axis. | | scaleZ | sz | Scale sz times on Z axis. | | scale3d | (sx,sy,sz) | Scale sx times on X axis, sy times on Y axis and sz times on Z axis. |

Translate

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | translate | tx,[ty] | When there is only one parameter, it indicates translating by tx on X axis. When there are two parameters, it indicates translating by tx on X axis and ty on Y axis. | | translateX | tx | Translate by tx on X axis, in px. | | translateY | ty | Translate by ty on Y axis, in px. | | translateZ | tz | Translate by tz on Z axis, in px. | | translate3d | (tx,ty,tz) | Translate by tx on X axis, ty on Y axis and tz on Z axis, in px. |

Skew

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | skew | ax,[ay] | Range -180~180 When there is only one parameter, Y stays unchanged and X skews by ax degrees clockwise. When there are two parameters, X skews by ax degrees and Y skews by ay degrees. | | skewX | ax | Range -180~180 Y stays unchanged and X skews by ax degrees clockwise. Degree. | | skewY | ay | Range -180~180 X stays unchanged and Y skews by ay degrees clockwise. |

Matrix transformation

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | matrix | (a,b,c,d,tx,ty) | Same as transform-function. | | matrix3d | (a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, d3, a4, b4, c4, d4) | Same as transform-function matrix3d. |

Animation Queue

• When the animation operation method is called, it is required to call `step()` to indicates the completion of a group of animations. Within a group of animation, it is possible to call any number of animation methods. All animations in the group start at the same time. It does not enter into the next group until the current animation group ends.

• The `step()` can transfer a configuration parameter that is the same as `my.createAnimation()`, which is used to specify the configuration of the current animation group.

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_ui_animation_creat eanimation

my.createAnimation {#mycreateanimation}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.createAnimation

2021-05-09 18:43

Create an animation instance. Call the instance method to describe animation, and then use the `export` method of animation instance to export the animation data and transfer to the component `animation` attribute.

Note: After the `export` method is called, the previous animation operation will be cleared.

Sample Code

copy

```
//.json
{
    "defaultTitle": "Animation"
}
```

copy

```
<!-- .axml -->
<view class="page">
  <view class="page-description">Animation API</view>
  <view class="page-section">
    <view class="page-section-title">my.createAnimation</view>
    <view class="page-section-demo">
      <view class="animation-element" animation="{{animation}}">
</view>
    </view>
    <view class="page-section-btns">
      <view type="primary" onTap="rotate">Rotate</view>
      <view type="primary" onTap="scale"> Scale</view>
      <view type="primary" onTap="translate">Translate</view>
    </view>
    <view class="page-section-btns">
      <view type="primary" onTap="skew">Skew</view>
      <view type="primary" onTap="rotateAndScale">Rotate and
scale</view>
      <view type="primary" onTap="rotateThenScale">Rotate and then
```

```
scale</view>
    </view>
    <view class="page-section-btns">
      <view type="primary" onTap="all">Expand all simultaneously
</view>
      <view type="primary" onTap="allInQueue">Expand all in
order</view>
      <view type="primary" onTap="reset">Reset</view>
    </view>
  </view>
</view>
```

copy

```
//.js
Page({
  onReady() {
    this.animation = my.createAnimation()
  },
  rotate() {
    this.animation.rotate(Math.random() * 720 - 360).step()
    this.setData({ animation: this.animation.export() })
  },
  scale() {
    this.animation.scale(Math.random() * 2).step()
    this.setData({ animation: this.animation.export() })
  },
  translate() {
    this.animation.translate(Math.random() * 100 - 50, Math.random() *
100 - 50).step()
    this.setData({ animation: this.animation.export() })
  },
  skew() {
    this.animation.skew(Math.random() * 90, Math.random() * 90).step()
    this.setData({ animation: this.animation.export() })
  },
  rotateAndScale() {
    this.animation.rotate(Math.random() * 720 - 360)
      .scale(Math.random() * 2)
      .step()
    this.setData({ animation: this.animation.export() })
  },
  rotateThenScale() {
    this.animation.rotate(Math.random() * 720 - 360).step()
      .scale(Math.random() * 2).step()
    this.setData({ animation: this.animation.export() })
  },
  all() {
    this.animation.rotate(Math.random() * 720 - 360)
      .scale(Math.random() * 2)
      .translate(Math.random() * 100 - 50, Math.random() * 100 - 50)
      .skew(Math.random() * 90, Math.random() * 90)
      .step()
    this.setData({ animation: this.animation.export() })
  },
  allInQueue() {
    this.animation.rotate(Math.random() * 720 - 360).step()
      .scale(Math.random() * 2).step()
      .translate(Math.random() * 100 - 50, Math.random() * 100 -
50).step()
      .skew(Math.random() * 90, Math.random() * 90).step()
    this.setData({ animation: this.animation.export() })
  },
  reset() {
    this.animation.rotate3d(0, 0, 0, 0)
      .rotateX(0)
      .rotateY(0)
      .rotateZ(0)
      .scale(1)
      .translate(0, 0)
      .skew(0, 0)
      .step({ duration: 0 })
    this.setData({ animation: this.animation.export() })
  }
})
```

copy

```
.animation-element {
  width: 200rpx;
  height: 200rpx;
  background-color: #108ee9;
  transform: scaleX(1) scaleY(1);
}
```

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
duration	Integer	No	Animation duration, in ms, 400 by default.
timeFunction	String	No	Define animation effect, linear by default, effective values including linear, ease, ease-in, ease-in-out, ease-out, step-start and step-end .
delay	Integer	No	Animation delay, in ms, 0 by default.
transformOrigin	String	No	Set transform-origin, 50% 50% 0 by default.

Sample Code

copy

```
//.js
const animation = my.createAnimation({
  transformOrigin: "top right",
  duration: 3000,
  timeFunction: "ease-in-out",
  delay: 100,
})
```

Animation

The animation instance may call the following method to describe the animation. At the end of the call, the instance itself is returned. The chain call style is supported. When the view animation attribute is initialized as `{}`, error may appears on basic library 1.11.0 (not including 1.11.0) and lower version. It is recommended to initialize as `null`.

Style

Method	**Parameters**	**Description**
opacity	value	Transparency, range 0~1.
backgroundColor	color	Color value.
width	length	Set the width:length values, in px, such as 300 px.
height	length	Set the height:length values, in px, such as 300 px.
top	length	Set the top:length values, in px, such as 300 px.
left	length	Set the left:length values, in px, such as 300 px.
bottom	length	Set the bottom:length values, in px, such as 300 px.
right	length	Set the right:length values, in px, such as 300 px.

Rotation

Method	**Parameters**	**Description**
rotate	deg	Deg range -180 ~ 180, rotate by deg degrees clockwise from origin.
rotateX	deg	Deg range -180 ~ 180, rotate by deg degrees on X axis.
rotateY	deg	Deg range -180 ~ 180, rotate by deg degrees on Y axis.
rotateZ	deg	Deg range -180 ~ 180, rotate by deg degrees on Z axis.
rotate3d	(x, y , z, deg)	Same as transform-function rotate3d.

Scale

Method	**Parameters**	**Description**
scale	sx,[sy]	When there is only one parameter, it indicates scaling sx times on X and Y axises at the same time. When there are two parameters, it indicates scaling sx times on X axis and sy times on Y axis.
scaleX	sx	Scale sx times on X axis.
scaleY	sy	Scale sy times on Y axis.
scaleZ	sz	Scale sz times on Z axis.
scale3d	(sx,sy,sz)	Scale sx times on X axis, sy times on Y axis and sz times on Z axis.

Translate

Method	**Parameters**	**Description**
translate	tx,[ty]	When there is only one parameter, it indicates translating by tx on X axis. When there are two parameters, it indicates translating by tx on X axis and ty on Y axis.
translateX	tx	Translate by tx on X axis, in px.
translateY	ty	Translate by ty on Y axis, in px.
translateZ	tz	Translate by tz on Z axis, in px.
translate3d	(tx,ty,tz)	Translate by tx on X axis, ty on Y axis and tz on Z axis, in px.

Skew

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | skew | ax,[ay] | Range -180~180 When there is only one parameter, Y stays unchanged and X skews by ax degrees clockwise. When there are two parameters, X skews by ax degrees and Y skews by ay degrees. | | skewX | ax | Range -180~180 Y stays unchanged and X skews by ax degrees clockwise. Degree. | | skewY | ay | Range -180~180 X stays unchanged and Y skews by ay degrees clockwise. |

Matrix transformation

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | matrix | (a,b,c,d,tx,ty) | Same as transform-function. | | matrix3d | (a1, b1, c1, d1, a2, b2, c2, d2, a3, b3, c3, d3, a4, b4, c4, d4) | Same as transform-function matrix3d. |

Animation Queue

• When the animation operation method is called, it is required to call `step()` to indicates the completion of a group of animations. Within a group of animation, it is possible to call any number of animation methods. All animations in the group start at the same time. It does not enter into the next group until the current animation group ends.

• The `step()` can transfer a configuration parameter that is the same as `my.createAnimation()`, which is used to specify the configuration of the current animation group.

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

---

my.createCanvasContext {#mycreatecanvascontext}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.createCanvasContext

2021-05-09 18:43

Create canvas context This canvas context works on the `<canvas/>` of the corresponding `canvasId` only.

Parameters

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | canvasId | String | ID defined on the `<canvas/>`. |

Return Value

`CanvasContext`

Sample Code

copy

```
//.js
const ctx = my.createCanvasContext('myCanvas')

const grd = ctx.createLinearGradient(30, 10, 120, 10)
grd.addColorStop(0, 'red')
grd.addColorStop(0.16, 'orange')
grd.addColorStop(0.33, 'yellow')
grd.addColorStop(0.5, 'green')
grd.addColorStop(0.66, 'cyan')
grd.addColorStop(0.83, 'blue')
grd.addColorStop(1, 'purple')

// Fill color
ctx.setFillStyle(grd)
ctx.fillRect(10, 10, 150, 80)
ctx.draw()
```

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

---

my.createCanvasContext {#mycreatecanvascontext}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.createCanvasContext

2022-07-03 18:44

Create canvas context This canvas context works on the `<canvas/>` of the corresponding `canvasId` only.

Parameters

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | canvasId | String | ID defined on the `<canvas/>`. |

Return Value

`CanvasContext`

Sample Code

copy

```
//.js
const ctx = my.createCanvasContext('myCanvas')

const grd = ctx.createLinearGradient(30, 10, 120, 10)
grd.addColorStop(0, 'red')
grd.addColorStop(0.16, 'orange')
grd.addColorStop(0.33, 'yellow')
grd.addColorStop(0.5, 'green')
grd.addColorStop(0.66, 'cyan')
grd.addColorStop(0.83, 'blue')
grd.addColorStop(1, 'purple')

// Fill color
ctx.setFillStyle(grd)
ctx.fillRect(10, 10, 150, 80)
ctx.draw()
```

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

---

my.createMapContext {#mycreatemapcontext}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.createMapContext

2022-07-03 18:44

Call this API to create and return a map context object mapContext.

For more information about the related components, see map.

Parameter~~~~

| | | | | --- | --- | --- | | **Property** | **Required** | **Description** | | mapId | Yes | The ID of the map component. |

Return value

The return value is MapContext.

Sample code

copy

```
//.axml
<view class="page-section">
      <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}}">
      </map>
  </view>
```

copy

```
//.js
Page({
  // ... ...
  onReady() {
    //Call my.createMapContext to obtain the map context.
    this.mapCtx = my.createMapContext('map');
  },
```

```
  // ... ...
}
```

PageContext.setData(Object)

Call this operation to initialize or reset map data. The parameters are optional.

Sample code

copy

```
//  .js

this.setData({
    scale: 14,
    longitude: 120.131441,
    latitude: 30.279383,
    'show-location':true,
    // Add a ground overlay. Feature added in v10.1.35.
    'ground-overlays':[{\
        'include-points':[{// Upper-right\
            latitude: 39.935029,\
            longitude: 116.384377,\
          },{// Lower-left\
            latitude: 39.939577,\
            longitude: 116.388331,\
          }],\
        image:'/image/groundoverlay.png',\
        alpha:0.75,\
        zIndex:0,\
    }],
    // Add a tile overlay. It is a feature added in v10.1.35.
    'tile-overlay':{
      url:'http://xixi.fullspeed.cn/public/map',
      type:0,
      tileWidth:256,
      tileHeight:256,
      zIndex:1,
    },
    markers:[{},{}],
    'include-points':[{},{}],
    // New overview logic added in v10.1.35.
    'include-padding':{left:0, right:0, top:0, bottom:0},
    polyline: [{},{}],
    circles: [{},{}],
    controls: [{},{}],
    polygon: [{},{}],
    'include-padding':{},
    // Support settings at map initialization. It is a feature added
in v10.1.50.
```

```
    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,
        // Map POI
        showMapText:0/1,
        // Map logo position
        logoPosition:{centerX:150, centerY:90},
    },
});
```

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

---

my.createSelectorQuery {#mycreateselectorquery}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.createSelectorQuery

2021-05-09 18:43

Return a SelectorQuery object instance.

Return Value

SelectorQuery

Sample Code

copy

```
<!-- .axml -->
<view class="page">
  <view class="page-description">Node query API</view>
  <view class="page-section">
    <view className="all">Node all1</view>
    <view className="all">Node all2</view>
    <view id="one">Node one</view>
    <view id="scroll" style="height:200px;overflow: auto">
      <view style="height:400px">Independent scroll region</view>
    </view>
    <button type="primary" onTap="createSelectorQuery">Node
query</button>
  </view>
</view>
```

copy

```
//.js
Page({
  createSelectorQuery() {
    my.createSelectorQuery()
      .select('#non-exists').boundingClientRect()
      .select('#one').boundingClientRect()
      .selectAll('.all').boundingClientRect()
      .select('#scroll').scrollOffset()
      .selectViewport().boundingClientRect()
      .selectViewport().scrollOffset().exec((ret) => {
      console.log(ret);
      my.alert({
        content: JSON.stringify(ret, null, 2),
      });
    })
  },
});
```

ret Structure

copy

```
[\
  null,\
  {\
    "x": 1,\
    "y": 2,\
    "width": 1367,\
    "height": 18,\
    "top": 2,\
    "right": 1368,\
    "bottom": 20,\
    "left": 1\
  },\
  [\
    {\
      "x": 1,\
      "y": -34,\
      "width": 1367,\
      "height": 18,\
      "top": -34,\
      "right": 1368,\
      "bottom": -16,\
      "left": 1\
    },\
    {\
      "x": 1,\
      "y": -16,\
      "width": 1367,\
      "height": 18,\
      "top": -16,\
      "right": 1368,\
      "bottom": 2,\
      "left": 1\
    }\
  ],\
  {\
    "scrollTop": 0,\
    "scrollLeft": 0\
  },\
  {\
    "width": 1384,\
    "height": 360\
  },\
  {\
    "scrollTop": 35,\
    "scrollLeft": 0\
  }\
]
```

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

---

my.createSelectorQuery (UI/Selector-Query) {#mycreateselectorquery-ui/selector-query}

Path: miniprogram_gcash

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/API_UI_Selector-Query_createSelectorQuery

---

my.createWebViewContext {#mycreatewebviewcontext}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.createWebViewContext

2022-07-03 18:44

By creating `webviewContext`, it creates the capability to send messages from Mini Program to `web-view`. Create and return `web-view` context `webViewContext` object.

Sample Code

copy

```
<!-- .axml -->
<view>
  <web-view id="web-view-1" src="..." onMessage="onMessage"></web-
view>
</view>
```

copy

```
// .js
Page({
  onLoad() {
    this.webViewContext = my.createWebViewContext('web-view-1');
  },
  // Receive message from HTML5
  onMessage(e) {
    console.log(e); //{'sendToMiniProgram': '0'}
    // Send message to HTML5
  this.webViewContext.postMessage({'sendToWebView': '1'});
  }
})
```

copy

```
// .js
// my.onMessage needs to be defined in HTML5 js code at first to
receive the message from Mini Program.
my.onMessage = function(e) {
  console.log(e); //{'sendToWebView': '1'}
}
// HTML5 sends message to Mini Program
my.postMessage({'sendToMiniProgram': '0'});
```

Note: The workflow of the above two-way communication capability is that HTML5 sends message to Mini Program at first. After the Mini Program receives the message, it sends message to HTML5.

Parameters

Object type with the following attributes:

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | webviewId | String | Yes | ID attribute corresponding to the web-view to be created. |

Return Value

Create a webViewContext object

`webViewContext` is bound with a `web-view` component via webviewId to implement some functions. List of `webViewContext` object methods:

| | | | | --- | --- | --- | | **Method** | **Parameters** | **Description** | | postMessage | Object | The Mini Program sends message to the web-view component, and works with the my.postMessage provided by web-view.js to implement the two-way communication between Mini Program and web-view page. |

Source: https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/api_web-view_createwebviewcontext

---

my.createWebViewContext {#mycreatewebviewcontext}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.createWebViewContext

2021-05-09 18:43

By creating `webviewContext`, it creates the capability to send messages from Mini Program to `web-view`. Create and return `web-view` context `webViewContext` object.

Sample Code

copy

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

```
  <web-view id="web-view-1" src="..." onMessage="onMessage"></web-
view>
</view>
```

copy

```
// .js
Page({
  onLoad() {
    this.webViewContext = my.createWebViewContext('web-view-1');
  },
  // Receive message from HTML5
  onMessage(e) {
    console.log(e); //{'sendToMiniProgram': '0'}
    // Send message to HTML5
  this.webViewContext.postMessage({'sendToWebView': '1'});
  }
})
```

copy

```
// .js
// my.onMessage needs to be defined in HTML5 js code at first to
receive the message from Mini Program.
my.onMessage = function(e) {
  console.log(e); //{'sendToWebView': '1'}
}
// HTML5 sends message to Mini Program
my.postMessage({'sendToMiniProgram': '0'});
```

Note: The workflow of the above two-way communication capability is that HTML5 sends message to Mini Program at first. After the Mini Program receives the message, it sends message to HTML5.

Parameters

Object type with the following attributes:

Property	**Type**	**Required**	**Description**
webviewId	String	Yes	ID attribute corresponding to the web-view to be created.

Return Value

Create a webViewContext object

`webViewContext` is bound with a `web-view` component via webviewId to implement some functions. List of `webViewContext` object methods:

Method	**Parameters**	**Description**
postMessage	Object	The Mini Program sends message to the web-view component, and works with the my.postMessage provided by web-view.js to implement the two-way communication

between Mini Program and web-view page. |

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

---

my.datePicker {#mydatepicker}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.datePicker

2022-07-03 18:44

Use this API to open the date selection list.

Sample code

index.json

copy

```
// API-DEMO page/API/date-picker/date-picker.json
{
    "defaultTitle": "Date Picker"
}
```

index.axml

copy

```
<!-- API-DEMO page/API/date-picker/date-picker.axml -->
<view class="page">
  <view class="page-description">Date picker API</view>
  <view class="page-section">
    <view class="page-section-title">my.datePicker</view>
    <view class="page-section-demo">
      <button class="page-body-button" type="primary"
onTap="datePicker">Pick Date-1</button>
      <button class="page-body-button" type="primary"
onTap="datePickerHMS">Pick Date-2</button>
      <button class="page-body-button" type="primary"
onTap="datePickerYMDHMS">Pick Date-3</button>
    </view>
  </view>
</view>
```

index.js

copy

```
// API-DEMO page/API/date-picker/date-picker.js
Page({
  datePicker() {
    my.datePicker({
      currentDate: '2016-10-10',
      startDate: '2016-10-9',
      endDate: '2017-10-9',
      success: (res) => {
        my.alert({
          title: 'datePicker response: ' + JSON.stringify(res)
        });
      },
    });
  },
  datePickerHMS() {
    my.datePicker({
      format: 'HH:mm',
      currentDate: '12:12',
      startDate: '11:11',
      endDate: '13:13',
      success: (res) => {
        my.alert({
          title: 'datePicker response: ' + JSON.stringify(res)
        });
      },
    });
  },
  datePickerYMDHMS() {
    my.datePicker({
      format: 'yyyy-MM-dd HH:mm',
      currentDate: '2012-01-09 11:11',
      startDate: '2012-01-01 11:11',
      endDate: '2012-01-10 11:11',
      success: (res) => {
        my.alert({
          title: 'datePicker response: ' + JSON.stringify(res)
        });
      },
    });
  },
});
```

index.acss

copy

```
/* API-DEMO page/API/date-picker/date-picker.acss */
button + button {
  margin-top: 20rpx;
}
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | format | String | No | The returned date format. | | currentDate | String | No | The date and time initially selected. By default, the current time date and time are used. | | startDate | String | No | Minimum date and time. | | endDate | | No | Maximum date and time. | | success | Function | Yes | The callback function for a successful API call. | | fail | Function | Yes | The callback function for a failed API call. | | complete | Function | Yes | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

The returned date formats include:

- yyyy-MM-dd (default)
- HH:mm
- yyyy-MM-dd HH:mm
- yyyy-MM. Pass in `canIUse('datePicker.object.format.yyyy-MM')` to my.canIUse to query if the current version can be used.
- yyyy. Pass in `canIUse('datePicker.object.format.yyyy')` to my.canIUse to query if the current version can be used.

Success callback function

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | date | String | The selected date. |

Error Code

| | | | | --- | --- | --- | | **Error Code** | **Description** | **Solution** | | 11 | The user cancelled the operation. | The user cancelled the operation and no action is required. |

Source:
https://miniprogram.gcash.com/docs/miniprogram_gcash/mpdev/API_UI_Choose-Date_datePicker

---

my.datePicker {#mydatepicker}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.datePicker

2021-05-09 18:43

Use this API to open the date selection list.

Sample code

index.json

copy

```
// API-DEMO page/API/date-picker/date-picker.json
{
    "defaultTitle": "Date Picker"
}
```

index.axml

copy

```
<!-- API-DEMO page/API/date-picker/date-picker.axml -->
<view class="page">
  <view class="page-description">Date picker API</view>
  <view class="page-section">
    <view class="page-section-title">my.datePicker</view>
    <view class="page-section-demo">
      <button class="page-body-button" type="primary"
onTap="datePicker">Pick Date-1</button>
      <button class="page-body-button" type="primary"
onTap="datePickerHMS">Pick Date-2</button>
      <button class="page-body-button" type="primary"
onTap="datePickerYMDHMS">Pick Date-3</button>
    </view>
  </view>
</view>
```

index.js

copy

```
// API-DEMO page/API/date-picker/date-picker.js
Page({
  datePicker() {
    my.datePicker({
      currentDate: '2016-10-10',
      startDate: '2016-10-9',
      endDate: '2017-10-9',
      success: (res) => {
        my.alert({
          title: 'datePicker response: ' + JSON.stringify(res)
        });
      },
    });
  },
  datePickerHMS() {
    my.datePicker({
      format: 'HH:mm',
      currentDate: '12:12',
      startDate: '11:11',
      endDate: '13:13',
      success: (res) => {
        my.alert({
          title: 'datePicker response: ' + JSON.stringify(res)
        });
      },
    });
  },
  datePickerYMDHMS() {
    my.datePicker({
      format: 'yyyy-MM-dd HH:mm',
      currentDate: '2012-01-09 11:11',
      startDate: '2012-01-01 11:11',
      endDate: '2012-01-10 11:11',
      success: (res) => {
        my.alert({
          title: 'datePicker response: ' + JSON.stringify(res)
        });
      },
    });
  },
});
```

index.acss

copy

```
/* API-DEMO page/API/date-picker/date-picker.acss */
button + button {
  margin-top: 20rpx;
}
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | format | String | No | The returned date format. | | currentDate | String | No | The date and time initially selected. By default, the current time date and time are used. | | startDate | String | No | Minimum date and time. | | endDate | | No | Maximum date and time. | | success | Function | Yes | The callback function for a successful API call. | | fail | Function | Yes | The callback function for a failed API call. | | complete | Function | Yes | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

The returned date formats include:

- yyyy-MM-dd (default)
- HH:mm
- yyyy-MM-dd HH:mm
- yyyy-MM. Pass in `canIUse('datePicker.object.format.yyyy-MM')` to my.canIUse to query if the current version can be used.
- yyyy. Pass in `canIUse('datePicker.object.format.yyyy')` to my.canIUse to query if the current version can be used.

Success callback function

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | date | String | The selected date. |

Error Code

| | | | | --- | --- | --- | | **Error Code** | **Description** | **Solution** | | 11 | The user cancelled the operation. | The user cancelled the operation and no action is required. |

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

---

my.disconnectBLEDevice {#mydisconnectbledevice}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.disconnectBLEDevice

2021-05-09 18:43

Use this API to disconnect from a Bluetooth Low Energy (BLE) device.

Instructions:

- Bluetooth device might be disconnected at any time. It is recommended to listen to my.onBLEConnectionStateChanged callback event. When the BLE device is disconnected, perform the reconnect operation as required.
- After read and write interface are called for a disconnected device, error 10006 is returned and reconnection is recommended.

Note:

Currently simulation in IDE is not supported. Please debug in production environment.

Sample Code

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
```

```
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
```

```
        my.onBluetoothDeviceFound({
          success: res => {
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
        devid = res.devices[0].deviceId;
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain all searched devices
  getBluetoothDevices() {
    my.getBluetoothDevices({
      success: res => {
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  bindKeyInput(e) {
    this.setData({
      devid: e.detail.value,
    });
  },

  //Connect the device
  connectBLEDevice() {
    my.connectBLEDevice({
      deviceId: this.data.devid,
      success: res => {
        my.alert({ content: 'Succeeded to connect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Disconnect the device
  disconnectBLEDevice() {
    my.disconnectBLEDevice({
      deviceId: this.data.devid,
      success: () => {
        my.alert({ content: 'Succeeded to disconnect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Obtain the char ID of the connected device, read and write
characteristics are respectively screened out.
  getBLEDeviceCharacteristics() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.getBLEDeviceCharacteristics({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            //See the related document for more information of the
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
            this.setData({
              charid: res.characteristics[0].characteristicId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Read and write data
  readBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.readBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          //1  Android reading service
          // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
          success: res => {
            my.alert({ content: JSON.stringify(res) });
          },
          fail: error => {
            my.alert({ content: 'Failed to read' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  writeBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.writeBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.charid,
          //Android writing service
          //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
          value: 'ABCD',
          success: res => {
            my.alert({ content: 'Succeeded to write data!' });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },
  notifyBLECharacteristicValueChange() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.notifyBLECharacteristicValueChange({
          state: true,
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          success: () => {
            //Listens to characteristic change events
            my.onBLECharacteristicValueChange({
              success: res => {
                //  my.alert({content: 'Changes of
characteristics  '+JSON.stringify(res)});
                my.alert({ content: 'Obtain the response data = ' +
res.value });
              },
            });
            my.alert({ content: 'Succeeded to listen' });
          },
          fail: error => {
            my.alert({ content: 'Failed to listen' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  offBLECharacteristicValueChange() {
    my.offBLECharacteristicValueChange();
  },

  //Other events
  bluetoothAdapterStateChange() {

my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  onBluetoothAdapterStateChange() {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of the Bluetooth state  ' +
JSON.stringify(res) });
    }
  },
  offBluetoothAdapterStateChange() {

my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStat
  },
  getBind(name) {
    if (!this[`bind${name}`]) {
      this[`bind${name}`] = this[name].bind(this);
    }
    return this[`bind${name}`];
  },
  BLEConnectionStateChanged() {

my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
  },
  onBLEConnectionStateChanged(res) {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of connection state  ' +
JSON.stringify(res) });
    }
  },
  offBLEConnectionStateChanged() {

my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChan
  },
  onUnload() {
    this.offBLEConnectionStateChanged();
    this.offBLECharacteristicValueChange();
    this.offBluetoothAdapterStateChange();
    this.closeBluetoothAdapter();
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | deviceId | String | Yes | The Bluetooth device ID. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

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

---

my.disconnectBLEDevice {#mydisconnectbledevice}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.disconnectBLEDevice

2022-07-03 18:44

Use this API to disconnect from a Bluetooth Low Energy (BLE) device.

Instructions:

- Bluetooth device might be disconnected at any time. It is recommended to listen to my.onBLEConnectionStateChanged callback event. When the BLE device is disconnected, perform the reconnect operation as required.
- After read and write interface are called for a disconnected device, error 10006 is returned and reconnection is recommended.

Note:

Currently simulation in IDE is not supported. Please debug in production environment.

Sample Code

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
```

```
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
        devid = res.devices[0].deviceId;
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain all searched devices
  getBluetoothDevices() {
    my.getBluetoothDevices({
      success: res => {
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  bindKeyInput(e) {
    this.setData({
      devid: e.detail.value,
    });
  },

  //Connect the device
  connectBLEDevice() {
    my.connectBLEDevice({
      deviceId: this.data.devid,
      success: res => {
        my.alert({ content: 'Succeeded to connect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Disconnect the device
  disconnectBLEDevice() {
    my.disconnectBLEDevice({
      deviceId: this.data.devid,
      success: () => {
        my.alert({ content: 'Succeeded to disconnect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Obtain the char ID of the connected device, read and write
characteristics are respectively screened out.
  getBLEDeviceCharacteristics() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.getBLEDeviceCharacteristics({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            //See the related document for more information of the
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
            this.setData({
              charid: res.characteristics[0].characteristicId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Read and write data
  readBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.readBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          //1  Android reading service
          // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
          success: res => {
            my.alert({ content: JSON.stringify(res) });
          },
          fail: error => {
            my.alert({ content: 'Failed to read' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  writeBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.writeBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.charid,
          //Android writing service
          //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
          value: 'ABCD',
          success: res => {
            my.alert({ content: 'Succeeded to write data!' });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },
  notifyBLECharacteristicValueChange() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.notifyBLECharacteristicValueChange({
          state: true,
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          success: () => {
            //Listens to characteristic change events
            my.onBLECharacteristicValueChange({
              success: res => {
                //  my.alert({content: 'Changes of
characteristics  '+JSON.stringify(res)});
                my.alert({ content: 'Obtain the response data = ' +
res.value });
              },
            });
            my.alert({ content: 'Succeeded to listen' });
          },
          fail: error => {
            my.alert({ content: 'Failed to listen' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  offBLECharacteristicValueChange() {
    my.offBLECharacteristicValueChange();
  },

  //Other events
  bluetoothAdapterStateChange() {

my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  onBluetoothAdapterStateChange() {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of the Bluetooth state  ' +
JSON.stringify(res) });
    }
  },
  offBluetoothAdapterStateChange() {

my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  getBind(name) {
    if (!this[`bind${name}`]) {
      this[`bind${name}`] = this[name].bind(this);
    }
    return this[`bind${name}`];
  },
  BLEConnectionStateChanged() {

my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChange
  },
  onBLEConnectionStateChanged(res) {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of connection state  ' +
JSON.stringify(res) });
    }
  },
  offBLEConnectionStateChanged() {

my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
  },
  onUnload() {
    this.offBLEConnectionStateChanged();
    this.offBLECharacteristicValueChange();
    this.offBluetoothAdapterStateChange();
    this.closeBluetoothAdapter();
  },
});
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | deviceId | String | Yes | The Bluetooth device ID. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

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

my.downloadFile {#mydownloadfile}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.downloadFile

2022-07-03 18:44

Download a file resource to a local location.

Sample Code

copy

```
my.downloadFile({
  url: 'http://img.example.com/example.jpg',
  success({ apFilePath }) {
    my.previewImage({
      urls: [apFilePath],
    });
  },
  fail(res) {
    my.alert({
      content: res.errorMessage || res.error,
    });
  },
});
```

Parameters

Property	**Type**	**Required**	**Description**
url	String	Yes	Downloading file address.
header	Object	No	HTTP request Header.
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

Property	**Type**	**Description**
apFilePath	String	Temporary file storage location.

Error Code

Error	**Description**
12	Downloading failed.
13	No right.

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

---

my.downloadFile {#mydownloadfile}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.downloadFile

2021-05-09 18:43

Download a file resource to a local location.

Sample Code

copy

```
my.downloadFile({
  url: 'http://img.example.com/example.jpg',
  success({ apFilePath }) {
    my.previewImage({
      urls: [apFilePath],
    });
  },
  fail(res) {
    my.alert({
      content: res.errorMessage || res.error,
    });
  },
});
```

Parameters

Property	**Type**	**Required**	**Description**
url	String	Yes	Downloading file address.
header	Object	No	HTTP request Header.
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | apFilePath | String | Temporary file storage location. |

Error Code

| | | | --- | --- | | **Error** | **Description** | | 12 | Downloading failed. | | 13 | No right. |

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

---

my.getAppIdSync {#mygetappidsync}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getAppIdSync

2022-07-03 18:44

Use this API to obtain the Mini Program App ID synchronously.

Sample Code

copy

```
const appIdRes = my.getAppIdSync();
console.log(appIdRes.appId);
```

Return Value

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | appId | String | The App ID of the current Mini Program. |

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

---

my.getAppIdSync {#mygetappidsync}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.getAppIdSync

2021-05-09 18:43

Use this API to obtain the Mini Program App ID synchronously.

Sample Code

copy

```
const appIdRes = my.getAppIdSync();
console.log(appIdRes.appId);
```

Return Value

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | appId | String | The App ID of the current Mini Program. |

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

---

my.getAuthCode {#mygetauthcode}

Last updated: 2021-05-10

Path: miniprogram_gcash

my.getAuthCode

2021-05-10 22:04

Get authentication code.

Sample Code

copy

```
my.getAuthCode({
  scopes: 'auth_user',
  success: (res) => {
    my.alert({
      content: res.authCode,
    });
  },
});
```

Parameters

| Name | Type | Mandatory | Description | | scopes | String/Array | N | The scope of auth, there are two types: `auth_base`,
`auth_user`, by default, its value is `auth_base` |

Success Callback Function

The incoming parameter is of the Object type with the following attributes:

| | | | | | --- | --- | --- | --- | | Field | Type | Mandatory | Description | | authCode | String | Y | Auth code | | authErrorScopes | Key-value | Y | The scope that failed to grant auth, key is the scope and value is the error | | authSuccessScopes | Array | Y | The scope that succeed to grant auth |

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

---

my.getAuthCode {#mygetauthcode}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getAuthCode

2022-07-03 18:44

Call the API to obtain the authorization code (authCode). The authorization code can be used to obtain access token, so as to easily obtain the app user userId, nickname, etc.

For more information, refer to the user authorization.

Sample code

copy

```
my.getAuthCode({
  scopes: ['auth_user'],
  success: (res) => {
    my.alert({
      content: res.authCode,
    });
  },
  fail: (res) => {
      console.log(res.authErrorScopes)
  },
});
```

Parameters

Property	**Type**	**Required**	**Description**
scopes	String/Array	Yes	The scope of authorization, including: - auth_base - auth_user (auth_base is silent authorization)
success	Function	No	Callback function upon call success.
fail	Function	No	Callback function upon call failure.
complete	Function	No	Callback function upon call completion (to be executed upon either call success or failure).

Scopes description

Scopes	**Description**
auth_base	Authorized to obtain the unique user ID.
auth_user	Authorized to obtain user information.

Note:

- `auth_base` are used to silently obtain user ID, silent authorization does not pop the frame and directly obtains user information. All the other scopes are used for proactive user authorization.
- The `auth_base` and `auth_user` are legacy scopes and not recommended to be used.

Callback function

The incoming parameter is of the Object type with the following attributes:

Property	**Type**	**Required**	**Description**
authCode	String	Yes	Authorization code.
authErrorScopes	Key-value	Yes	The scope that failed to grant authorization, key is the scope and value is the error.
authSuccessScopes	Array	Yes	The scope that succeed to grant authorization.

Successful response example

copy

```
{
    "authCode":"1591797390204",
    "authSuccessScopes":['auth_user']
}
```

Failure response example

copy

```
{
    "authErrorScopes":{
       "auth_user":"40006"
    }
}
```

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

---

my.getBLEDeviceCharacteristics {#mygetbledevicecharacteristics}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getBLEDeviceCharacteristics

2022-07-03 18:44

Use this API to obtain all characteristics in a Bluetooth device that is connected to the native.

Instruction:

After connection, execute my.getBLEDeviceServices and this interface before data can be exchanged with the Bluetooth device.

Sample Code

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->

<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
```

```
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
        devid = res.devices[0].deviceId;
      },
      fail: error => {
```

```
      my.alert({ content: JSON.stringify(error) });
    },
  });
},

//Obtain all searched devices
getBluetoothDevices() {
  my.getBluetoothDevices({
    success: res => {
      my.alert({ content: JSON.stringify(res) });
    },
    fail: error => {
      my.alert({ content: JSON.stringify(error) });
    },
  });
},
bindKeyInput(e) {
  this.setData({
    devid: e.detail.value,
  });
},

//Connect the device
connectBLEDevice() {
  my.connectBLEDevice({
    deviceId: this.data.devid,
    success: res => {
      my.alert({ content: 'Succeeded to connect!' });
    },
    fail: error => {
      my.alert({ content: JSON.stringify(error) });
    },
  });
},

//Disconnect the device
disconnectBLEDevice() {
  my.disconnectBLEDevice({
    deviceId: this.data.devid,
    success: () => {
      my.alert({ content: 'Succeeded to disconnect!' });
    },
    fail: error => {
      my.alert({ content: JSON.stringify(error) });
    },
  });
},

//Obtain the services of the connected device
getBLEDeviceServices() {
  my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          my.getBLEDeviceServices({
            deviceId: this.data.devid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              this.setData({
                serid: res.services[0].serviceId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Obtain the char ID of the connected device, read and write
  characteristics are respectively screened out.
    getBLEDeviceCharacteristics() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.getBLEDeviceCharacteristics({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              //See the related document for more information of the
  properties of the characteristics. Pair the characteristics according
  to the properties and record the value for later use.
              this.setData({
                charid: res.characteristics[0].characteristicId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Read and write data
    readBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.readBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            //1  Android reading service
            // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
            success: res => {
              my.alert({ content: JSON.stringify(res) });
            },
            fail: error => {
              my.alert({ content: 'Failed to read' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    writeBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.writeBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.charid,
            //Andriod writing service
            //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
            value: 'ABCD',
            success: res => {
              my.alert({ content: 'Succeeded to write data!' });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },
    notifyBLECharacteristicValueChange() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.notifyBLECharacteristicValueChange({
            state: true,
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            success: () => {
              //Listens to characteristic change events
              my.onBLECharacteristicValueChange({
                success: res => {
                  //  my.alert({content: 'Changes of
  characteristics  '+JSON.stringify(res)});
                  my.alert({ content: 'Obtain the response data = ' +
  res.value });
                },
              });
              my.alert({ content: 'Succeeded to listen' });
            },
            fail: error => {
              my.alert({ content: 'Failed to listen' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    offBLECharacteristicValueChange() {
      my.offBLECharacteristicValueChange();
    },

    //Other events
    bluetoothAdapterStateChange() {

  my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
    },
    onBluetoothAdapterStateChange() {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of the Bluetooth state  ' +
  JSON.stringify(res) });
      }
    },
    offBluetoothAdapterStateChange() {

  my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStat
    },
    getBind(name) {
      if (!this[`bind${name}`]) {
        this[`bind${name}`] = this[name].bind(this);
      }
      return this[`bind${name}`];
    },
    BLEConnectionStateChanged() {

  my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
    },
    onBLEConnectionStateChanged(res) {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of connection state  ' +
  JSON.stringify(res) });
      }
    },
    offBLEConnectionStateChanged() {

  my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChan
    },
    onUnload() {
      this.offBLEConnectionStateChanged();
      this.offBLECharacteristicValueChange();
      this.offBluetoothAdapterStateChange();
      this.closeBluetoothAdapter();
    },
  });
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | deviceId | String | Yes | The Bluetooth device ID. | | serviceId | String | Yes | The UUID of the service corresponding to a Bluetooth characteristic. | | success | Function | No | The callback

function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

Success callback function

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | characteristic | Array | The list of device characteristics. |

characteristic

The Bluetooth device characteristic information.

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | characteristicId | String | The Bluetooth device characteristic UUID. | | serviceId | String | The UUID of the service corresponding to a Bluetooth characteristic. | | value | Hex String | The hexadecimal value corresponding to a Bluetooth characteristic. | | properties | Object | The operation types supported by this characteristic. |

properties

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | read | Boolean | This field indicates whether this characteristic supports `read` operation. | | write | Boolean | This field indicates whether this characteristic supports `write` operation. | | notify | Boolean | This field indicates whether this characteristic supports `notify` operation. | | indicate | Boolean | This field indicates whether this characteristic supports `indicate` operation. |

九色鹿

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

---

my.getBLEDeviceCharacteristics {#mygetbledevicecharacteristics}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.getBLEDeviceCharacteristics

2021-05-09 18:43

Use this API to obtain all characteristics in a Bluetooth device that is connected to the native.

Instruction:

After connection, execute my.getBLEDeviceServices and this interface before data can be exchanged with the Bluetooth device.

Sample Code

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->

<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
```

```
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
```

```
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
        devid = res.devices[0].deviceId;
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain all searched devices
  getBluetoothDevices() {
    my.getBluetoothDevices({
      success: res => {
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  bindKeyInput(e) {
    this.setData({
      devid: e.detail.value,
    });
  },

  //Connect the device
  connectBLEDevice() {
    my.connectBLEDevice({
      deviceId: this.data.devid,
      success: res => {
        my.alert({ content: 'Succeeded to connect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Disconnect the device
  disconnectBLEDevice() {
    my.disconnectBLEDevice({
      deviceId: this.data.devid,
      success: () => {
        my.alert({ content: 'Succeeded to disconnect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Obtain the char ID of the connected device, read and write
characteristics are respectively screened out.
  getBLEDeviceCharacteristics() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.getBLEDeviceCharacteristics({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            //See the related document for more information of the
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
```

```
              this.setData({
                charid: res.characteristics[0].characteristicId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Read and write data
    readBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.readBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            //1  Android reading service
            // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
            success: res => {
              my.alert({ content: JSON.stringify(res) });
            },
            fail: error => {
              my.alert({ content: 'Failed to read' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    writeBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.writeBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.charid,
            //Andriod writing service
            //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
            value: 'ABCD',
            success: res => {
              my.alert({ content: 'Succeeded to write data!' });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },
    notifyBLECharacteristicValueChange() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.notifyBLECharacteristicValueChange({
            state: true,
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            success: () => {
              //Listens to characteristic change events
              my.onBLECharacteristicValueChange({
                success: res => {
                  //  my.alert({content: 'Changes of
  characteristics  '+JSON.stringify(res)});
                  my.alert({ content: 'Obtain the response data = ' +
  res.value });
                },
              });
              my.alert({ content: 'Succeeded to listen' });
            },
            fail: error => {
              my.alert({ content: 'Failed to listen' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    offBLECharacteristicValueChange() {
      my.offBLECharacteristicValueChange();
    },

    //Other events
    bluetoothAdapterStateChange() {

  my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
    },
    onBluetoothAdapterStateChange() {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of the Bluetooth state  ' +
  JSON.stringify(res) });
      }
    },
    offBluetoothAdapterStateChange() {

  my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStat
    },
    getBind(name) {
      if (!this[`bind${name}`]) {
        this[`bind${name}`] = this[name].bind(this);
      }
      return this[`bind${name}`];
    },
    BLEConnectionStateChanged() {

  my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
    },
    onBLEConnectionStateChanged(res) {
      if (res.error) {
        my.alert({ content: JSON.stringify(error) });
      } else {
        my.alert({ content: 'Changes of connection state  ' +
  JSON.stringify(res) });
      }
    },
    offBLEConnectionStateChanged() {

  my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChan
    },
    onUnload() {
      this.offBLEConnectionStateChanged();
      this.offBLECharacteristicValueChange();
      this.offBluetoothAdapterStateChange();
      this.closeBluetoothAdapter();
  },
});
```

Parameters

Property	**Type**	**Required**	**Description**
deviceId	String	Yes	The Bluetooth device ID.
serviceId	String	Yes	The UUID of the service corresponding to a Bluetooth characteristic.
success	Function	No	The callback function for a successful API call.
fail	Function	No	The callback function for a failed API call.
complete	Function	No	The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails.

Success callback function

Property	**Type**	**Description**
characteristic	Array	The list of device characteristics.

characteristic

The Bluetooth device characteristic information.

Property	**Type**	**Description**
characteristicId	String	The Bluetooth device characteristic UUID.
serviceId	String	The UUID of the service corresponding to a Bluetooth characteristic.
value	Hex String	The hexadecimal value corresponding to a Bluetooth characteristic.
properties	Object	The operation types supported by this characteristic.

properties

Property	**Type**	**Description**
read	Boolean	This field indicates whether this characteristic supports `read` operation.
write	Boolean	This field indicates whether this characteristic supports `write` operation.
notify	Boolean	This field indicates whether this characteristic supports `notify` operation.
indicate	Boolean	This field indicates whether this characteristic supports `indicate` operation.

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

---

my.getBLEDeviceServices {#mygetbledeviceservices}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.getBLEDeviceServices

2021-05-09 18:43

Use this API to obtain all services of Bluetooth devices that are connected to the native.

Note:

Currently simulation in IDE is not supported. Please debug in production environment.

Sample Code

copy

```
 //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },
```

Return Value Sample

copy

```
{
    "services": [{\
        "isPrimary": true,\
        "serviceId": "00001800-0000-1000-8000-00805f9b34fb"\
    }, {\
        "isPrimary": true,\
        "serviceId": "00001801-0000-1000-8000-00805f9b34fb"\
    }, {\
        "isPrimary": true,\
        "serviceId": "d0611e78-bbb4-4591-a5f8-487910ae4366"\
    }, {\
        "isPrimary": true,\
        "serviceId": "9fa480e0-4967-4542-9390-d343dc5d04ae"\
    }]
}
```

Parameters

Property	**Type**	**Required**	**Description**
deviceId	String	Yes	The Bluetooth device ID.
success	Function	No	The callback function for a successful API call.
fail	Function	No	The callback function for a failed API call.
complete	Function	No	The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails.

Success return value

Property	**Type**	**Description**
services	Array	List of discovered device services

services

Property	**Type**	**Description**
isPrimary	Boolean	This field indicates whether the service is the main service. Valid values are: `true`: The service is the main service. `false`: The service is not the main service.
serviceId	String	The UUID of the service corresponding to a Bluetooth characteristic.

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

---

my.getBLEDeviceServices {#mygetbledeviceservices}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getBLEDeviceServices

2022-07-03 18:44

Use this API to obtain all services of Bluetooth devices that are connected to the native.

Note:

Currently simulation in IDE is not supported. Please debug in production environment.

Sample Code

copy

```
 //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },
```

Return Value Sample

copy

```
{
    "services": [{\
        "isPrimary": true,\
        "serviceId": "00001800-0000-1000-8000-00805f9b34fb"\
    }, {\
        "isPrimary": true,\
        "serviceId": "00001801-0000-1000-8000-00805f9b34fb"\
    }, {\
        "isPrimary": true,\
        "serviceId": "d0611e78-bbb4-4591-a5f8-487910ae4366"\
    }, {\
        "isPrimary": true,\
        "serviceId": "9fa480e0-4967-4542-9390-d343dc5d04ae"\
    }]
}
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | deviceId | String | Yes | The Bluetooth device ID. | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

Success return value

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | services | Array | List of discovered device services |

services

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | isPrimary | Boolean | This field indicates whether the service is the main service. Valid values are:
`true`: The service is the main service.
`false`: The service is not the main service. | | serviceId | String | The UUID of the service corresponding to a Bluetooth characteristic. |

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

---

my.getBatteryInfo {#mygetbatteryinfo}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.getBatteryInfo

2021-05-09 18:43

Use this API to asynchronously obtain the battery level and the charging state of the current device. No parameters are required.

Sample Code

copy

```
my.getBatteryInfo({
  success: (res) => {
    my.alert({ content: 'System information  ' + JSON.stringify(res),
});
```

```
    console.log({ content: 'System information  ' +
JSON.stringify(res), });
  },
  fail: (error) => {
    my.alert({ content: 'Inquiry failed' + JSON.stringify(error), });
  },
  complete: () => {
    my.alert({ title: 'Complete callback', });
  },
})
```

Parameters

| | | | | | --- | --- | --- | --- | | **Property** | **Type** | **Required** | **Description** | | success | Function | No | The callback function for a successful API call. | | fail | Function | No | The callback function for a failed API call. | | complete | Function | No | The callback function used when the API call is completed. This function is always executed no matter the call succeeds or fails. |

Success callback function

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | level | Int | The battery level of the current device. | | isCharging | Boolean | This property indicates whether the device is charging. |

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

---

my.getBatteryInfoSync {#mygetbatteryinfosync}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.getBatteryInfoSync

2021-05-09 18:43

Use this API to synchronously obtain the battery level and the charging state of the current device. No parameters are required.

Return Values

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | level | Int | The battery level of the current device. | | isCharging | Boolean | This property indicates whether the device is charging. |

Sample Code

copy

```
var res = my.getBatteryInfoSync();
my.alert({content: 'System information  '+JSON.stringify(res)});
console.log({content: 'System information  '+JSON.stringify(res),});
```

九色鹿

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

---

my.getBatteryInfoSync {#mygetbatteryinfosync}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getBatteryInfoSync

2022-07-03 18:44

Use this API to synchronously obtain the battery level and the charging state of the current device. No parameters are required.

Return Values

| | | | | --- | --- | --- | | **Property** | **Type** | **Description** | | level | Int | The battery level of the current device. | | isCharging | Boolean | This property indicates whether the device is charging. |

Sample Code

copy

```
var res = my.getBatteryInfoSync();
my.alert({content: 'System information  '+JSON.stringify(res)});
```

```
console.log({content: 'System information '+JSON.stringify(res),});
```

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

---

my.getBluetoothAdapterState {#mygetbluetoothadapterstate}

Last updated: 2021-05-09

Path: miniprogram_gcash

my.getBluetoothAdapterState

2021-05-09 18:43

Use this API to check the Bluetooth adapter status in the Mini Program.

Note:

Currently simulation in IDE is not supported. Please debug in the production environment.

Code Sample

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
```

```
      <view class="page-section-title">Other events</view>
      <view class="page-section-demo">
        <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
        <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
        <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
        <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Scan the Bluetooth device
  startBluetoothDevicesDiscovery() {
    my.startBluetoothDevicesDiscovery({
      allowDuplicatesKey: false,
      success: () => {
        my.onBluetoothDeviceFound({
          success: res => {
            // my.alert({content:'Listens to new
device'+JSON.stringify(res)});
            var deviceArray = res.devices;
            for (var i = deviceArray.length - 1; i >= 0; i--) {
              var deviceObj = deviceArray[i];
              //Pair the target device with the device name or
broadcast data, and then record the device ID for later use.
              if (deviceObj.name == this.data.name) {
                my.alert({ content: 'Target device is found' });
                my.offBluetoothDeviceFound();
                this.setData({
                  deviceId: deviceObj.deviceId,
                });
                break;
              }
            }
          },
          fail: error => {
            my.alert({ content: 'Failed to listen to new device' +
JSON.stringify(error) });
          },
        });
      },
      fail: error => {
        my.alert({ content: 'Failed to start scanning' +
JSON.stringify(error) });
      },
    });
  },

  //Stop scanning
  stopBluetoothDevicesDiscovery() {
    my.stopBluetoothDevicesDiscovery({
      success: res => {
        my.offBluetoothDeviceFound();
        my.alert({ content: 'Succeeded!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the connected device
  getConnectedBluetoothDevices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connecting devices!' });
          return;
        }
        my.alert({ content: JSON.stringify(res) });
        devid = res.devices[0].deviceId;
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain all searched devices
  getBluetoothDevices() {
    my.getBluetoothDevices({
      success: res => {
        my.alert({ content: JSON.stringify(res) });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  bindKeyInput(e) {
    this.setData({
      devid: e.detail.value,
    });
  },

  //Connect the device
  connectBLEDevice() {
    my.connectBLEDevice({
      deviceId: this.data.devid,
      success: res => {
        my.alert({ content: 'Succeeded to connect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Disconnect the device
  disconnectBLEDevice() {
    my.disconnectBLEDevice({
      deviceId: this.data.devid,
      success: () => {
        my.alert({ content: 'Succeeded to disconnect!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },

  //Obtain the services of the connected device
  getBLEDeviceServices() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        my.getBLEDeviceServices({
          deviceId: this.data.devid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            this.setData({
              serid: res.services[0].serviceId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Obtain the char ID of the connected device, read and write
characteristics are respectively screened out.
  getBLEDeviceCharacteristics() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.getBLEDeviceCharacteristics({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          success: res => {
            my.alert({ content: JSON.stringify(res) });
            //See the related document for more information of the
properties of the characteristics. Pair the characteristics according
to the properties and record the value for later use.
            this.setData({
              charid: res.characteristics[0].characteristicId,
            });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },

  //Read and write data
  readBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.readBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          //1  Android reading service
          // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
          success: res => {
            my.alert({ content: JSON.stringify(res) });
          },
          fail: error => {
            my.alert({ content: 'Failed to read' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  writeBLECharacteristicValue() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.writeBLECharacteristicValue({
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.charid,
          //Android writing service
          //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
          //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
          value: 'ABCD',
          success: res => {
            my.alert({ content: 'Succeeded to write data!' });
          },
          fail: error => {
            my.alert({ content: JSON.stringify(error) });
          },
        });
      },
    });
  },
  notifyBLECharacteristicValueChange() {
    my.getConnectedBluetoothDevices({
      success: res => {
        if (res.devices.length === 0) {
          my.alert({ content: 'No connected devices' });
          return;
        }
        this.setData({
          devid: res.devices[0].deviceId,
        });
        my.notifyBLECharacteristicValueChange({
          state: true,
          deviceId: this.data.devid,
          serviceId: this.data.serid,
          characteristicId: this.data.notifyId,
          success: () => {
            //Listens to characteristic change events
            my.onBLECharacteristicValueChange({
              success: res => {
                //  my.alert({content: 'Changes of
characteristics  '+JSON.stringify(res)});
                my.alert({ content: 'Obtain the response data = ' +
res.value });
              },
            });
            my.alert({ content: 'Succeeded to listen' });
          },
          fail: error => {
            my.alert({ content: 'Failed to listen' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  offBLECharacteristicValueChange() {
    my.offBLECharacteristicValueChange();
  },

  //Other events
  bluetoothAdapterStateChange() {

my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState(
  },
  onBluetoothAdapterStateChange() {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of the Bluetooth state  ' +
JSON.stringify(res) });
    }
  },
  offBluetoothAdapterStateChange() {

my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  getBind(name) {
    if (!this[`bind${name}`]) {
      this[`bind${name}`] = this[name].bind(this);
    }
    return this[`bind${name}`];
  },
  BLEConnectionStateChanged() {

my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChange
  },
  onBLEConnectionStateChanged(res) {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of connection state  ' +
JSON.stringify(res) });
    }
  },
  offBLEConnectionStateChanged() {

my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
  },
  onUnload() {
    this.offBLEConnectionStateChanged();
    this.offBLECharacteristicValueChange();
    this.offBluetoothAdapterStateChange();
    this.closeBluetoothAdapter();
  },
});
```

Parameters

The input parameters are displayed in the following table:

Property	**Type**	**Required**	**Description**
success	Function	No	The callback function for a successful API call.
fail	Function	No	The callback function for a failed API call.
complete	Function	No	The callback function for a completed API call (Regardless of whether the call is successful or not).

Success Callback Function

The input parameters are displayed in the following table:

Property	**Type**	**Description**
discovering	Boolean	Indicates whether bluetooth device is being discovered.
available	Boolean	Indicates whether bluetooth is available (BLE should be supported and switched on).

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

---

my.getBluetoothAdapterState {#mygetbluetoothadapterstate}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getBluetoothAdapterState

2022-07-03 18:44

Use this API to check the Bluetooth adapter status in the Mini Program.

Note:

Currently simulation in IDE is not supported. Please debug in the production environment.

Code Sample

copy

```
/* .acss */
.help-info {
  padding:10px;
  color:#000000;
}
.help-title {
  padding:10px;
  color:#FC0D1B;
}
```

copy

```
// .json
{
    "defaultTitle": "Bluetooth"
}
```

copy

```
<!-- .axml-->
<view class="page">
  <view class="page-description">Bluetooth API</view>
  <view class="page-section">
    <view class="page-section-title">The Bluetooth state</view>
    <view class="page-section-demo">
       <button type="primary" onTap="openBluetoothAdapter">Initialize
Bluetooth</button>
       <button type="primary" onTap="closeBluetoothAdapter">Close
Bluetooth</button>
       <button type="primary" onTap="getBluetoothAdapterState">Obtain
Bluetooth state</button>
    </view>
    <view class="page-section-title">Scan the Bluetooth device</view>
    <view class="page-section-demo">
       <button type="primary"
onTap="startBluetoothDevicesDiscovery">Start searching</button>
       <button type="primary" onTap="getBluetoothDevices">All devices
found</button>
       <button type="primary" onTap="getConnectedBluetoothDevices">All
devices connected</button>
       <button type="primary"
onTap="stopBluetoothDevicesDiscovery">Stop searching</button>
    </view>
    <view class="page-section-title">Connect the device</view>
    <view class="page-section-demo">
       <input class="input" onInput="bindKeyInput" type="{{text}}"
placeholder="Enter the device ID of the device to connect"></input>
       <button type="primary" onTap="connectBLEDevice">Connect the
device</button>
       <button type="primary" onTap="getBLEDeviceServices">Obtain
device services</button>
       <button type="primary"
onTap="getBLEDeviceCharacteristics">Obtain read and write
characteristics</button>
       <button type="primary" onTap="disconnectBLEDevice">Disconnect
the device</button>
    </view>
     <view class="page-section-title">Read and write data</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="notifyBLECharacteristicValueChange">Listens to the
characteristic data change</button>
       <button type="primary" onTap="readBLECharacteristicValue">Read
data</button>
       <button type="primary"
onTap="writeBLECharacteristicValue">Write data</button>
       <button type="primary"
onTap="offBLECharacteristicValueChange">Un-listens to characteristic
value</button>
    </view>
     <view class="page-section-title">Other events</view>
     <view class="page-section-demo">
       <button type="primary"
onTap="bluetoothAdapterStateChange">Changes of the Bluetooth
state</button>
       <button type="primary"
onTap="offBluetoothAdapterStateChange">Un-listens to Bluetooth
state</button>
       <button type="primary"
onTap="BLEConnectionStateChanged">Changes of Bluetooth connection
state</button>
       <button type="primary" onTap="offBLEConnectionStateChanged">Un-
listens to Bluetooth connection state</button>
```

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

copy

```
// .js
Page({
  data: {
    devid: '0D9C82AD-1CC0-414D-9526-119E08D28124',
    serid: 'FEE7',
    notifyId: '36F6',
    writeId: '36F5',
    charid: '',
    alldev: [{ deviceId: '' }],
  },

  //Obtain the Bluetooth state
  openBluetoothAdapter() {
    my.openBluetoothAdapter({
      success: res => {
        if (!res.isSupportBLE) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
        my.alert({ content: 'Succeeded to initialize!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  closeBluetoothAdapter() {
    my.closeBluetoothAdapter({
      success: () => {
        my.alert({ content: 'Bluetooth closed!' });
      },
      fail: error => {
        my.alert({ content: JSON.stringify(error) });
      },
    });
  },
  getBluetoothAdapterState() {
    my.getBluetoothAdapterState({
      success: res => {
        if (!res.available) {
          my.alert({ content: 'Sorry, your mobile Bluetooth is
unavailable temporarily' });
          return;
        }
```

```
          my.alert({ content: JSON.stringify(res) });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Scan the Bluetooth device
    startBluetoothDevicesDiscovery() {
      my.startBluetoothDevicesDiscovery({
        allowDuplicatesKey: false,
        success: () => {
          my.onBluetoothDeviceFound({
            success: res => {
              // my.alert({content:'Listens to new
  device'+JSON.stringify(res)});
              var deviceArray = res.devices;
              for (var i = deviceArray.length - 1; i >= 0; i--) {
                var deviceObj = deviceArray[i];
                //Pair the target device with the device name or
  broadcast data, and then record the device ID for later use.
                if (deviceObj.name == this.data.name) {
                  my.alert({ content: 'Target device is found' });
                  my.offBluetoothDeviceFound();
                  this.setData({
                    deviceId: deviceObj.deviceId,
                  });
                  break;
                }
              }
            },
            fail: error => {
              my.alert({ content: 'Failed to listen to new device' +
  JSON.stringify(error) });
            },
          });
        },
        fail: error => {
          my.alert({ content: 'Failed to start scanning' +
  JSON.stringify(error) });
        },
      });
    },

    //Stop scanning
    stopBluetoothDevicesDiscovery() {
      my.stopBluetoothDevicesDiscovery({
        success: res => {
          my.offBluetoothDeviceFound();
          my.alert({ content: 'Succeeded!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain the connected device
    getConnectedBluetoothDevices() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connecting devices!' });
            return;
          }
          my.alert({ content: JSON.stringify(res) });
          devid = res.devices[0].deviceId;
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain all searched devices
    getBluetoothDevices() {
      my.getBluetoothDevices({
        success: res => {
          my.alert({ content: JSON.stringify(res) });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },
    bindKeyInput(e) {
      this.setData({
        devid: e.detail.value,
      });
    },

    //Connect the device
    connectBLEDevice() {
      my.connectBLEDevice({
        deviceId: this.data.devid,
        success: res => {
          my.alert({ content: 'Succeeded to connect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Disconnect the device
    disconnectBLEDevice() {
      my.disconnectBLEDevice({
        deviceId: this.data.devid,
        success: () => {
          my.alert({ content: 'Succeeded to disconnect!' });
        },
        fail: error => {
          my.alert({ content: JSON.stringify(error) });
        },
      });
    },

    //Obtain the services of the connected device
    getBLEDeviceServices() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          my.getBLEDeviceServices({
            deviceId: this.data.devid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              this.setData({
                serid: res.services[0].serviceId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Obtain the char ID of the connected device, read and write
  characteristics are respectively screened out.
    getBLEDeviceCharacteristics() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.getBLEDeviceCharacteristics({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            success: res => {
              my.alert({ content: JSON.stringify(res) });
              //See the related document for more information of the
  properties of the characteristics. Pair the characteristics according
  to the properties and record the value for later use.
              this.setData({
                charid: res.characteristics[0].characteristicId,
              });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },

    //Read and write data
    readBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.readBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            //1  Android reading service
            // serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            // characteristicId:'00002a38-0000-1000-8000-00805f9b34fb',
            success: res => {
              my.alert({ content: JSON.stringify(res) });
            },
            fail: error => {
              my.alert({ content: 'Failed to read' +
  JSON.stringify(error) });
            },
          });
        },
      });
    },
    writeBLECharacteristicValue() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.writeBLECharacteristicValue({
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.charid,
            //Android writing service
            //serviceId:'0000180d-0000-1000-8000-00805f9b34fb',
            //characteristicId:'00002a39-0000-1000-8000-00805f9b34fb',
            value: 'ABCD',
            success: res => {
              my.alert({ content: 'Succeeded to write data!' });
            },
            fail: error => {
              my.alert({ content: JSON.stringify(error) });
            },
          });
        },
      });
    },
    notifyBLECharacteristicValueChange() {
      my.getConnectedBluetoothDevices({
        success: res => {
          if (res.devices.length === 0) {
            my.alert({ content: 'No connected devices' });
            return;
          }
          this.setData({
            devid: res.devices[0].deviceId,
          });
          my.notifyBLECharacteristicValueChange({
            state: true,
            deviceId: this.data.devid,
            serviceId: this.data.serid,
            characteristicId: this.data.notifyId,
            success: () => {
              //Listens to characteristic change events
              my.onBLECharacteristicValueChange({
                success: res => {
                  //  my.alert({content: 'Changes of
  characteristics  '+JSON.stringify(res)});
                  my.alert({ content: 'Obtain the response data = ' +
  res.value });
                },
            });
            my.alert({ content: 'Succeeded to listen' });
          },
          fail: error => {
            my.alert({ content: 'Failed to listen' +
JSON.stringify(error) });
          },
        });
      },
    });
  },
  offBLECharacteristicValueChange() {
    my.offBLECharacteristicValueChange();
  },

  //Other events
  bluetoothAdapterStateChange() {

my.onBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterState
  },
  onBluetoothAdapterStateChange() {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of the Bluetooth state  ' +
JSON.stringify(res) });
    }
  },
  offBluetoothAdapterStateChange() {

my.offBluetoothAdapterStateChange(this.getBind('onBluetoothAdapterStat
  },
  getBind(name) {
    if (!this[`bind${name}`]) {
      this[`bind${name}`] = this[name].bind(this);
    }
    return this[`bind${name}`];
  },
  BLEConnectionStateChanged() {

my.onBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
  },
  onBLEConnectionStateChanged(res) {
    if (res.error) {
      my.alert({ content: JSON.stringify(error) });
    } else {
      my.alert({ content: 'Changes of connection state  ' +
JSON.stringify(res) });
    }
  },
  offBLEConnectionStateChanged() {
```

```
my.offBLEConnectionStateChanged(this.getBind('onBLEConnectionStateChang
  },
  onUnload() {
    this.offBLEConnectionStateChanged();
    this.offBLECharacteristicValueChange();
    this.offBluetoothAdapterStateChange();
    this.closeBluetoothAdapter();
  },
});
```

Parameters

The input parameters are displayed in the following table:

Property	**Type**	**Required**	**Description**
success	Function	No	The callback function for a successful API call.
fail	Function	No	The callback function for a failed API call.
complete	Function	No	The callback function for a completed API call (Regardless of whether the call is successful or not).

Success Callback Function

The input parameters are displayed in the following table:

Property	**Type**	**Description**
discovering	Boolean	Indicates whether bluetooth device is being discovered.
available	Boolean	Indicates whether bluetooth is available (BLE should be supported and switched on).

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

---

my.getBluetoothDevices {#mygetbluetoothdevices}

Last updated: 2021-05-10

Path: miniprogram_gcash

my.getBluetoothDevices

2021-05-10 03:43

Use this API to get all the bluetooth devices that are discovered, including those that are connected to the current device.

Instructions:

- You may not get the advertisData and RSSI in the simulator. Please debug in the production environment.
- For Integrated Development Environment (IDE) and Android devices, the device ID is the MAC address of the device; for iOS devie, the device ID is the UUID of the device. Therefore, do not hard code the device ID. You need to process the device ID on different platforms; iOS devices can be dynamically matched based on properties such as localName, advertisData, and manufacturerData.

Note:

Currently simulation in IDE is not supported. Please debug in the production environment.

Code Sample

copy

```
my.getBluetoothDevices({
  success: (res) => {
    console.log(res)
  },
  fail:(res) => {
  },
  complete: (res)=>{
  }
});
```

Parameters

The input parameters are displayed in the following table:

Property	**Type**	**Required**	**Description**
success	Function	No	The callback function for a successful API call.
fail	Function	No	The callback function for a failed API call.
complete	Function	No	The callback function for a completed API call (Regardless of whether the call is successful or not).

Success Callback Function

The input parameters are displayed in the following table:

Property	**Type**	**Description**
devices	Array	A list of all the devices that are discovered.

Device Object

Property	**Type**	**Description**
name	String	Name of the bluetooth device.(For some devices, there's no name.)
deviceName(Compatibal with initial version)	String	Name of the bluetooth device.
localName	String	Name of

the local device. | | deviceId | String | Device ID | | RSSI | Number | Received Signal Strength Indicator | | advertisData | Hex String | Advertisement data of the device | | manufacturerData | Hex String | Manufacturer data of the device |

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

---

my.getBluetoothDevices {#mygetbluetoothdevices}

Last updated: 2022-07-03

Path: miniprogram_gcash

my.getBluetoothDevices

2022-07-03 18:44

Use this API to get all the bluetooth devices that are discovered, including those that are connected to the current device.

Instructions:

- You may not get the advertisData and RSSI in the simulator. Please debug in the production environment.
- For Integrated Development Environment (IDE) and Android devices, the device ID is the MAC address of the device; for iOS devie, the device ID is the UUID of the device. Therefore, do not hard code the device ID. You need to process the device ID on different platforms; iOS devices can be dynamically matched based on properties such as localName, advertisData, and manufacturerData.

Note:

Currently simulation in IDE is not supported. Please debug in the production environment.

Code Sample

copy

```
my.getBluetoothDevices({
  success: (res) => {
    console.log(res)
  },
  fail:(res) => {
  },
  complete: (res)=>{
```