uniapp 安卓无障碍操作UTS原生插件

目　录

使用方法

介绍

安卓无障碍操作UTS原生插件，插件集成了各种无障碍操作，包括监听无障碍服务的各种操作，开启无障碍服务，关闭无障碍服务，节点操作，粘贴文本到输入框中等，插件UTS开发，支持uniapp x

联系作者

关注微信公众号可联系作者

插件地址

https://ext.dcloud.net.cn/plugin?id=20019

用法

在需要使用插件的页面加载以下代码

```
import * as module from "@/uni_modules/leven-uts-accessibility"
```

页面内容

```
<template>
  <view>
    <uni-card title="安卓无障碍操作UTS原生插件">
```

使用方法

```
        <button type="primary" @click="setListener">监听无障碍服务</button>
        <button type="primary" @click="removeListener">取消监听无障碍服务</button>
        <button type="primary" @click="openService">开启无障碍服务</button>
        <button type="primary" @click="stopService">关闭无障碍服务</button>
        <button type="primary" @click="isOpenService">无障碍服务是否开启</button>
        <button type="primary" @click="getAllNodes">获取所有的节点</button>
        <button type="primary" @click="getContainTextNodes">获取包含文字的节点</button>
        <button type="primary" @click="clickContainTextNodes">点击包含文字的节点</butto
        <button type="primary" @click="getNodesByIds">根据id集合获取节点</button>
        <button type="primary" @click="clickNodesByIds">点击包含id集合的节点</button>
        <button type="primary" @click="getAllEditableNodes">获取所有的可输入的节点</but
        <button type="primary" @click="pasteTextToEditor">粘贴文本到输入框</button>
        <button type="primary" @click="logStr = ''">清空日志</button>
      </uni-card>
      <uni-card class="uni-card-box" title="日志">
        <view><text style="font-size: 14px; flex-wrap: wrap;">{{logStr}}</text></vie
      </uni-card>
    </view>
  </template>

  <script>
    import * as module from "@/uni_modules/leven-uts-accessibility"
    export default {
      data() {
        return {
          logStr: "",
        }
      },
      methods: {
        //粘贴文本到输入框
        pasteTextToEditor() {
          module.pasteTextToEditor({
            //文本内容
            text: "这是一条测试数据",
            //输入框索引，可通过getAllEditableNodes获取
            index: 0
          }, res => {
            this.writeLog("粘贴文本到输入框：" + JSON.stringify(res))
          });
        },
        //获取所有可输入的节点
        getAllEditableNodes() {
          module.getAllEditableNodes(res => {
            this.writeLog("获取所有可输入的节点：" + JSON.stringify(res))
          });
        },
        //点击包含id集合的节点
        clickNodesByIds() {
          module.clickNodesByIds({
            list: ["com.tencent.mm:id/a3u", "com.tencent.mm:id/a3y"],
```

```
        //点击第几个索引，该索引可以通过getNodesByIds获取
        index: 0
      }, res => {
        this.writeLog("点击包含id集合的节点：" + JSON.stringify(res))
      });
    },
    //获取id集合的节点
    getNodesByIds() {
      module.getNodesByIds({
        list: ["com.tencent.mm:id/a3u", "com.tencent.mm:id/a3y"]
      }, res => {
        this.writeLog("获取id集合的节点:" + JSON.stringify(res))
      });
    },
    //点击包含文字的节点
    clickContainTextNodes() {
      //可通过此方法获取到是否有红包^_^，可以抢红包哦
      module.clickContainTextNodes({
        //包含的文本
        list: ["恭喜发财"],
        //点击第几个索引，该索引可以通过getContainTextNodes获取
        index: 0
      }, res => {
        this.writeLog("点击包含文字的节点：" + JSON.stringify(res))
      });
    },
    //获取包含文字的节点
    getContainTextNodes() {
      //可通过此方法获取到是否有红包^_^
      module.getContainTextNodes({
        //"红包", "恭喜发财", "好"
        list: ["红包", "恭喜发财", "好"]
      }, res => {
        this.writeLog("获取包含文字的节点：" + JSON.stringify(res))
      });
    },
    //获取所有的节点
    getAllNodes() {
      module.getAllNodes(res => {
        this.writeLog("获取所有的节点：" + JSON.stringify(res))
      });
    },
    //无障碍服务是否开启
    isOpenService() {
      module.isOpenService(res => {
        this.writeLog("无障碍服务是否开启：" + JSON.stringify(res))
      });
    },
    //关闭无障碍服务
    stopService() {
```

使用方法

```
        module.stopService(res => {
          this.writeLog("关闭无障碍服务：" + JSON.stringify(res))
        });
      },
      //开启无障碍服务
      openService() {
        //改方法没有回调，开启成功会在监听中回调
        module.openService();
      },
      //取消监听无障碍服务
      removeListener() {
        //该方法没有回调，取消监听后会在监听中回调然后监听失效
        module.removeListener();
      },
      //监听无障碍服务
      setListener() {
        module.setListener({
          // 监听的应用包名
          packageNames: ['com.tencent.mm', 'com.tencent.mobileqq']
        }, res => {
          //因为打印内容比较多所以注释掉当前代码
          // this.writeLog("监听无障碍服务：" + JSON.stringify(res))
          if (res.data && res.data.type) {
            if (res.data.type == "onEvent") {
              if (res.data.data.type == "viewClick") {
                //此处做测试使用，因为没有监听当前应用，所以只能在这里做测试
                this.writeLog(JSON.stringify(res))
                //视图被点击
                // this.getAllNodes();
                // this.getContainTextNodes();
                // this.clickContainTextNodes();
                // this.getNodesByIds();
                // this.clickNodesByIds();
                // this.getAllEditableNodes();
                this.pasteTextToEditor();
              }
            } else {
              this.writeLog("监听无障碍服务：" + JSON.stringify(res))
            }
          }
        })
      },
      // 写日志
      writeLog(str) {
        console.log(str)
        let logStr = uni.$lv.date.format(null, "yyyy-mm-dd hh:MM:ss") + " " + str
        // let logStr = str + "\n";
        this.logStr = logStr + this.logStr;
      }
    }
```

```
    }
</script>

<style>

</style>
```

更新日志

2024-08-28

首次发布

监听无障碍服务

方法名

`setListener`

> 监听后无障碍的回调事件都会在改方法中返回

用法

- 用法如下：

```
module.setListener({
  // 监听的应用包名
  packageNames: ['com.tencent.mm', 'com.tencent.mobileqq']
}, res => {
  this.writeLog("监听无障碍服务：" + JSON.stringify(res))
})
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
packageNames	Array	是	无	要监听的应用程序包名

回调

- 示例

```
{
  "data": {
    "type": "onSuccessListener"
  },
  "message": "监听成功",
  "code": 0
}

{
  "data": {
    "type": "onSuccess"
```

```
  },
  "message": "服务开启成功",
  "code": 0
}

{
  "data": {
    "type": "onEvent",
    "data": {
      "type": "viewScrolled",
      "eventData": {
        "scrollX": 0,
        "scrollY": 0,
        "fromIndex": 0,
        "className": "android.widget.ListView",
        "text": [],
        "nodeInfo": {
          "windowId": 1475,
          "parentNode": {
            "windowId": 1475,
            "childCount": 16,
            "className": "android.widget.FrameLayout",
            "isClickable": false,
            "isScrollable": false,
            "isEditable": false
          },
          "id": "com.tencent.mm:id/j8g",
          "childCount": 36,
          "className": "android.widget.ListView",
          "isClickable": false,
          "isScrollable": true,
          "isEditable": false
        },
        "windowId": 1475,
        "packageName": "com.tencent.mm",
        "currentItemIndex": -1,
        "itemCount": 313,
        "maxScrollY": 0,
        "maxScrollX": 0,
        "eventTime": 1200903714,
        "recordCount": 0
      }
    }
  },
  "message": "",
  "code": 0
}

{
  "data": {
```

```
    "type": "onEvent",
    "data": {
      "type": "viewClick",
      "eventData": {
        "scrollX": 0,
        "packageName": "com.tencent.mm",
        "scrollY": 0,
        "fromIndex": -1,
        "className": "android.widget.TextView",
        "text": [],
        "nodeInfo": {
          "parentNode": {
            "windowId": 1475,
            "id": "com.tencent.mm:id/bn1",
            "childCount": 3,
            "className": "android.widget.RelativeLayout",
            "isClickable": false,
            "isScrollable": false,
            "isEditable": false
          },
          "id": "com.tencent.mm:id/bkl",
          "windowId": 1475,
          "text": {
            "mText": "9XHX-VKLY-3XBT-XVSE",
            "mSpanIndex": 0,
            "mSpans": [],
            "mSpanCount": 0,
            "mSpanData": []
          },
          "childCount": 0,
          "className": "android.widget.TextView",
          "isClickable": true,
          "isScrollable": false,
          "isEditable": false
        },
        "windowId": 1475,
        "currentItemIndex": -1,
        "itemCount": -1,
        "maxScrollY": 0,
        "maxScrollX": 0,
        "contentDescription": {
          "mText": "9XHX-VKLY-3XBT-XVSE",
          "mSpanIndex": 0,
          "mSpans": [],
          "mSpanCount": 0,
          "mSpanData": []
        },
        "eventTime": 1201066037,
        "recordCount": 0
      }
```

```
    }
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.type	String	类型，具体请参考下方说明
data.data	Object	无障碍操作事件数据，**在data.type=onEvent时返回**
data.data.type	String	无障碍操作事件类型，具体类型请参考下方说明
data.data.eventData	Object	当前操作的节点信息
code	Integer	返回类型，0.成功，其他：失败

1. data.type说明

名称	描述
onCancelListener	取消监听
onError	错误事件
onEvent	无障碍事件（监听的应用状态事件）
onStop	服务停止
onSuccess	服务启动成功
onSuccessListener	监听无障碍成功

2. data.data.type说明

名称	描述
viewClick	视图被点击
notificationStateChanged	当通知栏状态发生改变时
announcement	产生一个通知事件
assistReadingContext	辅助用户读取当前屏幕事件
gestureDetectionEnd	结束手势检测

名称	描述
gestureDetectionStart	开始手势检测
touchExplorationGestureEnd	触摸浏览事件结束
touchExplorationGestureStart	触摸浏览事件开始
touchInteractionEnd	触摸屏幕事件结束
touchInteractionStart	触摸屏幕事件开始
viewAccessibilityFocused	视图成为无障碍焦点
viewAccessibilityFocusCleared	视图无障碍焦点被清除
viewContextClicked	视图上下文点击事件
viewFocused	视图获得焦点
viewHoverEnter	鼠标悬停到视图上
viewHoverExit	鼠标离开视图
viewLongClicked	视图被长按
viewScrolled	视图滚动
viewSelected	视图被选中
viewTextChanged	视图文本发送变化
viewTextSelectionChanged	视图选中的文本发生改变
viewTextTraversedAtMovementGranularity	在给定的移动粒度下遍历视图文本的事件
windowsChanged	窗体发生变化
windowContentChanged	窗体内容发生变化
windowsStateChanged	窗体状态发生变化

取消监听无障碍服务

方法名

`removeListener`

> 该方法没有回调，取消监听的回调在 `监听无障碍服务` 方法中返回

用法

- 用法如下：

```
module.removeListener()
```

- 参数说明

  无

回调

无

开启无障碍服务

方法名

`openService`

> 该方法没有回调，取消监听的回调在 `监听无障碍服务` 方法中返回

用法

- 用法如下：

```
module.openService()
```

- 参数说明

  无

回调

无

关闭无障碍服务

方法名

`stopService`

> 关闭成功后会在 `监听无障碍服务` 方法中返回停止事件

用法

- 用法如下：

```
module.stopService(res => {
  this.writeLog(JSON.stringify(res))
});
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败

无障碍服务是否开启

方法名

`isOpenService`

用法

- 用法如下：

```
module.isOpenService(res => {
  this.writeLog("无障碍服务是否开启：" + JSON.stringify(res))
});
```

- 参数说明

  无

回调

- 示例

```
{
  "data": {
    "status": true
  },
  "message": "",
  "code": 0
}
```

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.status	Boolean	是否开启
code	Integer	返回类型，0.成功，其他：失败

获取所有的节点

方法名

`getAllNodes`

用法

- 用法如下：

```
module.getAllNodes(res => {
  this.writeLog("获取所有的节点：" + JSON.stringify(res))
});
```

- 参数说明

  无

回调

- 示例


```
{
  "data": {
    "list": [
      {
        "windowId": 1910,
        "parentNode": {
          "windowId": 1910,
          "id": "com.tencent.mm:id/bkg",
          "childCount": 1,
          "className": "android.widget.FrameLayout",
          "isClickable": true,
          "isScrollable": false,
          "isEditable": false
        },
        "childCount": 0,
        "className": "android.widget.ImageView",
        "isClickable": false,
        "isScrollable": false,
        "id": "com.tencent.mm:id/bkm",
        "contentDescription": "图片",
```

```
        "isEditable": false
      },
      {
        "windowId": 1910,
        "parentNode": {
          "windowId": 1910,
          "id": "com.tencent.mm:id/bl9",
          "childCount": 4,
          "className": "android.widget.LinearLayout",
          "isClickable": false,
          "isScrollable": false,
          "isEditable": false
        },
        "childCount": 0,
        "className": "android.widget.ImageButton",
        "isClickable": true,
        "isScrollable": false,
        "id": "com.tencent.mm:id/bpe",
        "contentDescription": "切换到按住说话",
        "isEditable": false
      },
      {
        "windowId": 1910,
        "parentNode": {
          "windowId": 1910,
          "id": "com.tencent.mm:id/bl9",
          "childCount": 4,
          "className": "android.widget.LinearLayout",
          "isClickable": false,
          "isScrollable": false,
          "isEditable": false
        },
        "childCount": 0,
        "className": "android.widget.ImageButton",
        "isClickable": true,
        "isScrollable": false,
        "id": "com.tencent.mm:id/bqr",
        "contentDescription": "表情",
        "isEditable": false
      },
      {
        "windowId": 1910,
        "parentNode": {
          "windowId": 1910,
          "id": "com.tencent.mm:id/bl9",
          "childCount": 4,
          "className": "android.widget.LinearLayout",
          "isClickable": false,
          "isScrollable": false,
          "isEditable": false
```

```
        },
        "childCount": 0,
        "className": "android.widget.ImageButton",
        "isClickable": true,
        "isScrollable": false,
        "id": "com.tencent.mm:id/bjz",
        "contentDescription": "更多功能按钮，已折叠",
        "isEditable": false
      }
    ]
  },
  "message": "",
  "code": 0
}
```


- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	Array	数据列表
data.list.windowId	Integer	窗体id
data.list.parentNode	Object	父级节点
data.list.childCount	Integer	子集节点数量
data.list.className	String	组件类名
data.list.isClickable	Boolean	是否可以点击
data.list.isScrollable	Boolean	是否可以滚动
data.list.id	String	组件id名称
data.list.contentDescription	String	组件描述
data.list.isEditable	Boolean	是否可以编辑
data.list.text	String	组件文本内容
code	Integer	返回类型，0.成功，其他：失败

获取包含文字的节点

- 用法如下：

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
list	Array	是	无	包含的文本列表

- 示例

- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	Array	数据列表
data.list.windowId	Integer	窗体id
data.list.parentNode	Object	父级节点
data.list.childCount	Integer	子集节点数量
data.list.className	String	组件类名
data.list.isClickable	Boolean	是否可以点击
data.list.isScrollable	Boolean	是否可以滚动
data.list.id	String	组件id名称
data.list.contentDescription	String	组件描述
data.list.isEditable	Boolean	是否可以编辑
data.list.text	String	组件文本内容
code	Integer	返回类型，0.成功，其他：失败

点击包含文字的节点

方法名

`clickContainTextNodes`

> 该方法一般无回调，如果有回调一般是错误信息

用法

- 用法如下：

```
module.clickContainTextNodes({
  //包含的文本
  list: ["恭喜发财"],
  //点击第几个索引，该索引可以通过getContainTextNodes获取
  index: 0
}, res => {
  this.writeLog("点击包含文字的节点：" + JSON.stringify(res))
});
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
list	Array	是	无	包含的文本列表
index	Integer	否	0	点击第几个索引，该索引可以通过getContainTextNodes获取

回调

- 示例

  无

根据id集合获取节点

方法名

`getNodesByIds`

用法

- 用法如下：

```
module.getNodesByIds({
  list: ["com.tencent.mm:id/a3u", "com.tencent.mm:id/a3y"]
}, res => {
  this.writeLog("获取id集合的节点:" + JSON.stringify(res))
});
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
list	Array	是	无	包含的id列表

回调

- 示例

```
{
  "data": {
    "list": [
      {
        "parentNode": {
          "windowId": 1929,
          "id": "com.tencent.mm:id/bkg",
          "childCount": 3,
          "className": "android.widget.FrameLayout",
          "isClickable": true,
          "isScrollable": false,
          "isEditable": false
        },
        "id": "com.tencent.mm:id/a3u",
        "windowId": 1929,
        "text": {
```

```
            "mText": "恭喜发财，大吉大利",
            "mSpanIndex": 0,
            "mSpans": [],
            "mSpanCount": 0,
            "mSpanData": []
          },
          "childCount": 0,
          "className": "android.widget.TextView",
          "isClickable": false,
          "isScrollable": false,
          "isEditable": false
        }
      ]
    },
    "message": "",
    "code": 0
}
```


- 回调说明：

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	Array	数据列表
data.list.windowId	Integer	窗体id
data.list.parentNode	Object	父级节点
data.list.childCount	Integer	子集节点数量
data.list.className	String	组件类名
data.list.isClickable	Boolean	是否可以点击
data.list.isScrollable	Boolean	是否可以滚动
data.list.id	String	组件id名称
data.list.contentDescription	String	组件描述
data.list.isEditable	Boolean	是否可以编辑
data.list.text	String	组件文本内容
code	Integer	返回类型，0.成功，其他：失败

点击包含id集合的节点

方法名

`clickNodesByIds`

> 该方法一般无回调，如果有回调一般是错误信息

用法

- 用法如下：

```
module.clickNodesByIds({
  list: ["com.tencent.mm:id/a3u", "com.tencent.mm:id/a3y"],
  //点击第几个索引，该索引可以通过getNodesByIds获取
  index: 0
}, res => {
  this.writeLog("点击包含id集合的节点：" + JSON.stringify(res))
});
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
list	Array	是	无	包含的id列表
index	Integer	否	0	点击第几个索引，该索引可以通过getNodesByIds获取

回调

- 示例

  无

获取所有的可输入的节点

方法名

`getAllEditableNodes`

用法

- 用法如下：

```
module.getAllEditableNodes(res => {
  this.writeLog("获取所有可输入的节点：" + JSON.stringify(res))
});
```

- 参数说明

  无

回调

- 示例


```
{
  "data": {
    "list": [
      {
        "windowId": 1970,
        "id": "com.tencent.mm:id/bkk",
        "childCount": 0,
        "className": "android.widget.EditText",
        "isClickable": true,
        "isScrollable": false,
        "isEditable": true
      }
    ]
  },
  "message": "",
  "code": 0
}
```


- 回调说明：

获取所有的可输入的节点

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
data.list	Array	数据列表
data.list.windowId	Integer	窗体id
data.list.parentNode	Object	父级节点
data.list.childCount	Integer	子集节点数量
data.list.className	String	组件类名
data.list.isClickable	Boolean	是否可以点击
data.list.isScrollable	Boolean	是否可以滚动
data.list.id	String	组件id名称
data.list.contentDescription	String	组件描述
data.list.isEditable	Boolean	是否可以编辑
data.list.text	String	组件文本内容
code	Integer	返回类型，0.成功，其他：失败

粘贴文本到输入框

方法名

`pasteTextToEditor`

用法

- 用法如下：

```
module.pasteTextToEditor({
  //文本内容
  text: "这是一条测试数据",
  //输入框索引，可通过getAllEditableNodes获取
  index: 0
}, res => {
  this.writeLog("粘贴文本到输入框：" + JSON.stringify(res))
});
```

- 参数说明

参数名	参数类型	是否必填	默认值	参数描述
text	String	是	无	文本内容
index	Integer	否	0	输入框索引，可通过getAllEditableNodes获取

回调

- 示例

```
{
  "data": {},
  "message": "",
  "code": 0
}
```

- 回调说明：

粘贴文本到输入框

参数名	参数类型	参数描述
message	String	消息提示
data	Object	数据对象
code	Integer	返回类型，0.成功，其他：失败