PC 端 Web SDK Vue 接口说明文档 V4.0.6

文档修改记录

序号	版本号	修改内容	修改者	修改日期
1	v3.1.2	• 文档建立	周功成	2021/4/7
2	v3.1.4	• 更新 3.1.4 内容	石坤	2021/11/15
3	v3.1.8	• 兼容新机型 • 修复旋转后裁切异常 bug	石坤	2022/8/8
4	v3.1.8	更新接口封装	张彬	2023/6/7
5	v3.2.1	• 支持 K3 及 K3W 机型 • 增加 WIFI 相关接口	张彬	2023/10/12
6	v3.2.2	• 支持 M2 机型 • 增加 M2 相关说明	张彬	2023/10/30
7	v3.2.5	• 支持 B3S_P 机型 • 支持 B21S 机型 • 支持 B31 机型 • 更新图像库	张彬	2024/9/12
8	v4.0.3	• 支持 M3、K2、B21Pro 系列机型 • 完善错误码 • 新增绘制带 logo 二维码接口	张彬	2025/4/29

		• 提高 Websocket 通讯速度 5.Demo 支持黑标间隙纸		
9	v4.0.6	• 新增 closePrinter 接口 • 修复 Wifi 搜索接口 BUG • 修复历史遗留 WIFI 连接 BUG	张彬	2025/9/13

DEMO 目录结构

代码块

```
PC-SDK-VUE/
├── node-module/                              # NPM 加载的项目依赖模块
├── public/                                   # 存放公共资源和项目主入口文件
├── src/                                      # 项目核心文件夹：包括项目源码、静态资源等
│   ├── asset/                                # 存放静态资源的文件夹
│   ├── router/                               # 保存各类路由相关配置
│   │   └── index.js                          # 路由配置
│   ├── units/                                # 存放接口文件的文件夹
│   │   ├── printData/                        # 存放打印数据的文件夹
│   │   │   ├── Barcode.js                    # 一维码打印及预览示例数据
│   │   │   ├── Batch.js                      # 批量打印及预览示例数据
│   │   │   ├── Combination.js                # 组合打印及预览示例数据
│   │   │   ├── Graph.js                      # 图形打印及预览示例数据
│   │   │   ├── Img.js                        # 图片打印及预览示例数据
│   │   │   ├── Line.js                       # 线条打印及预览示例数据
│   │   │   ├── QrCode.js                     # 二维码打印及预览示例数据
│   │   │   └── Text.js                       # 文本打印及预览示例数据
│   │   ├── Print.js                          # 打印接口文件
│   │   ├── Socket.js                         # 打印服务连接文件
│   │   └── PrintElementFactory.js            # 绘制工厂策略类
│   ├── views/                                # 存储页面文件
│   │   └── HomeView.vue                      # 打印示例文件
│   ├── App.vue                               # 页面入口文件
│   ├── balel.config.js                       # 全局配置文件
│   ├── jsconfig.json                         # 指定了根文件以及JavaScript语言服务提供的功能选项
│   ├── package-lock.json                     # 项目版本管理使用的文件
│   └── package.json                          # 项目的基本配置文件
```

```
└── vue.config.js                          # 配置文件
```

产品目的

JCAPI 接口为调用者提供易用的方法完成标签绘图、打印操作。本接口中提供了标贴的绘制方法，包括：文字、一维码、二维码，图形、线条、图像绘制，同时还能进行绘制对象的旋转，调用者还可以调用方法获得绘制完成的标签图片用于标签预览，打印。方便用户在二次开发中调用接口，缩短开发周期，加快开发

打印机支持

支持打印机型号
B1
B203
B21 /B21_Pro/B21S
B3S / B3S_P
B31
B4
B11
K2
K3/K3W
B50/B50W
B32/Z401/B32R
M2
M3

准备工作

- 安装精臣打印服务（jcPrinterSdk.exe）
  - 前置：关闭杀毒软件（如 360，易误报）

  - 关键：**必须默认路径安装（C 盘）**
  - 注意：勿禁用服务开机启动

- 安装对应机型驱动

机型系列	系统要求
B50/B11	Win7/10/11 均需装驱动
其他机型	Win10/11 无需装，仅 Win7 需装

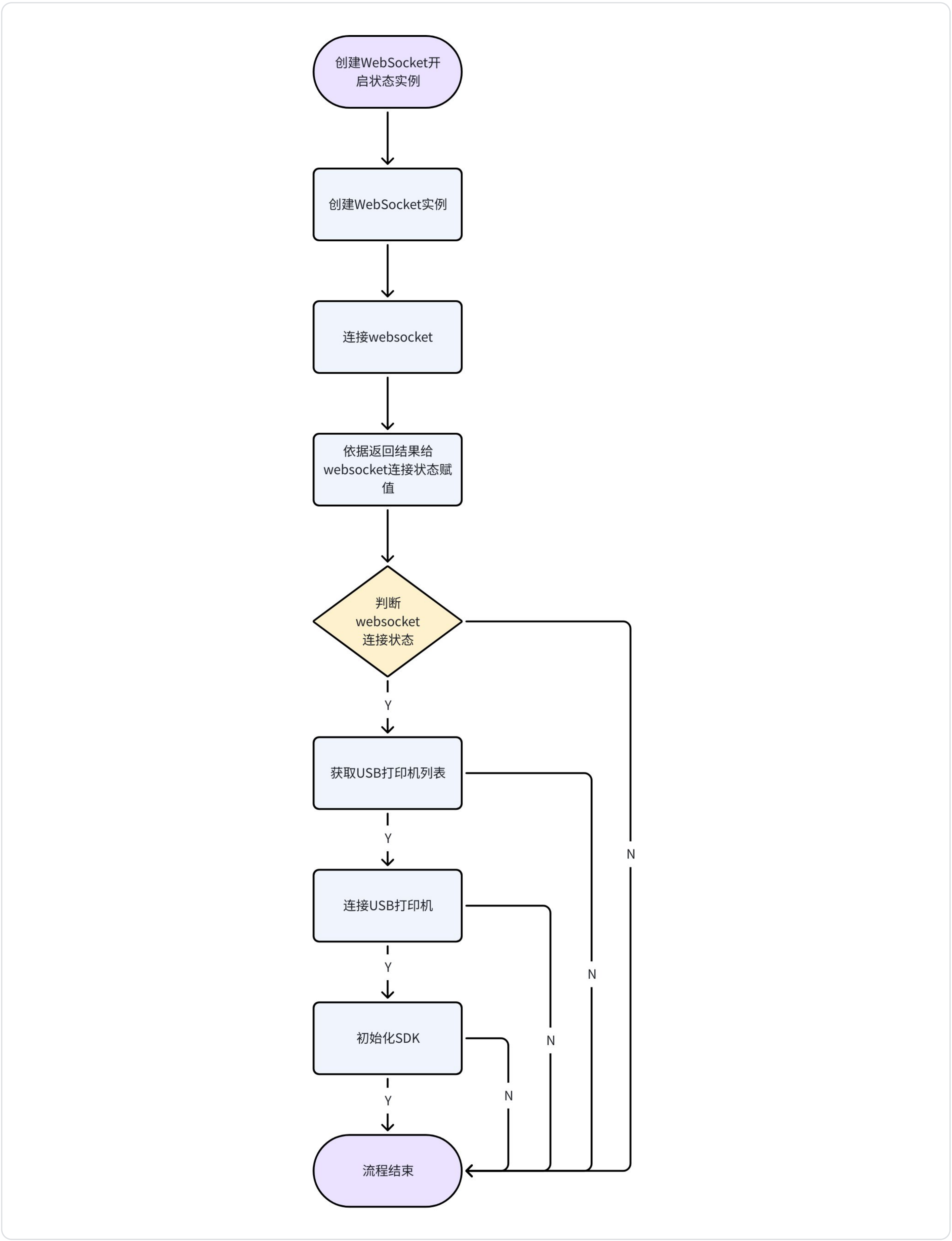
- 设备连接（2 种方式，不支持蓝牙）
  - USB 连接
    - 系统：仅支持 Windows
    - 驱动：可能需装（参考第 2 步）
    - 特别：**不支持驱动打印**（已用驱动打印需下载专用驱动
  - WIFI 连接
    - 机型：**仅支持 K3W 机型**
    - 系统：仅支持 Windows
    - 驱动：无需安装

一、初始化及打印调用流程、打印流程

1.1 初始化流程

1.1.1 USB 打印初始化流程

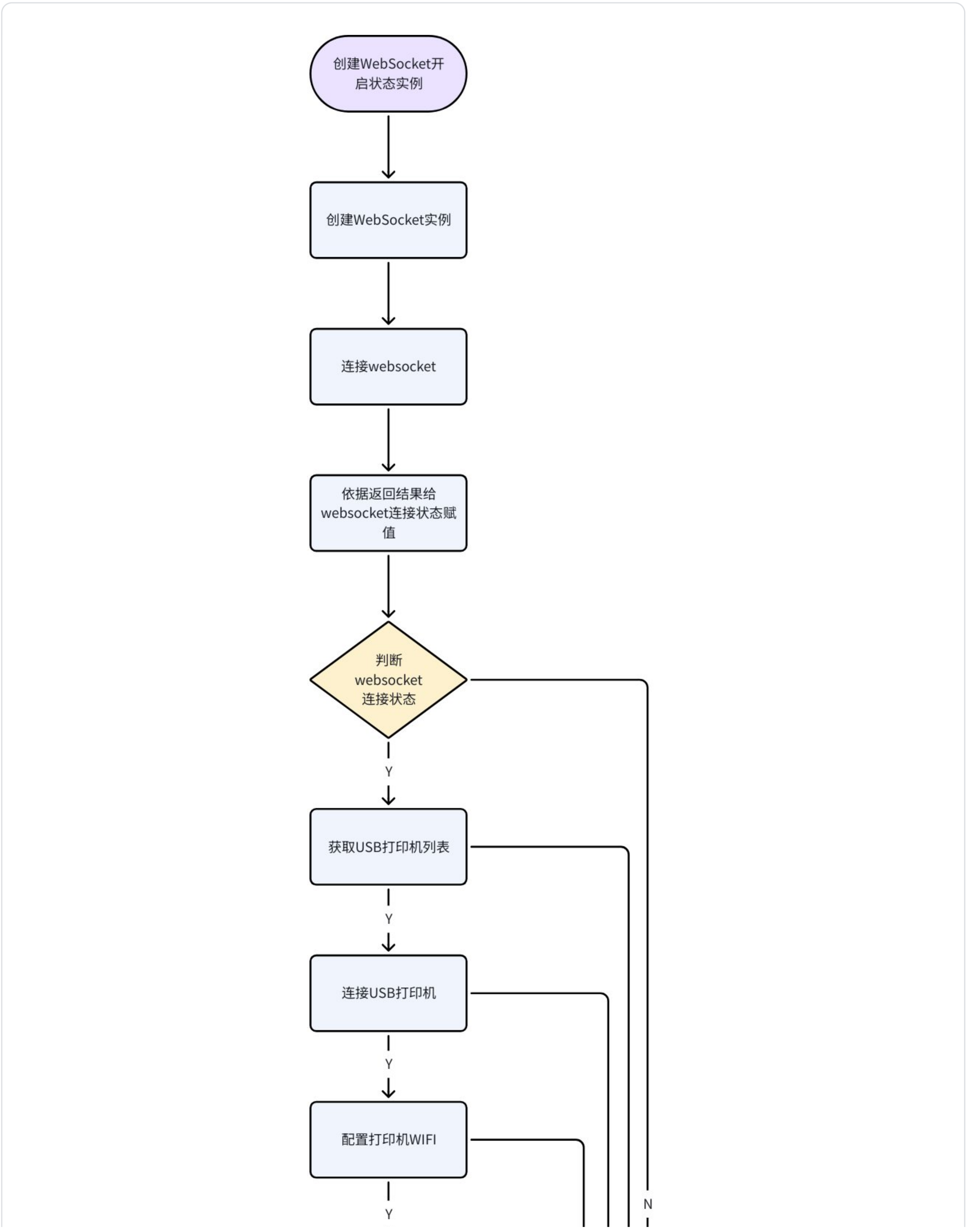
- WebSocket 建议页面加载时进行初始化，在 WebSocket 初始成功后回调中进行获取打印机、选择打印机、初始化 SDK 等操作
- 因为所有接口均为异步操作，调用下一接口需要验证当前接口结果后再执行下一接口
- 记录打印机列表获取状态、连接状态、初始化状态，打印机需要检查对应的状态

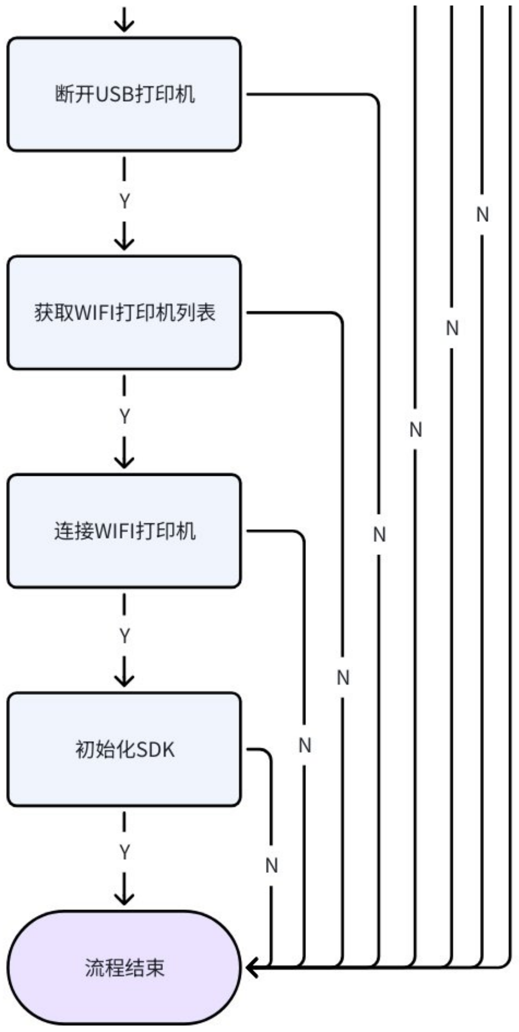


1.1.2 WIFI 打印初始化流程

- WebSocket 建议页面加载时进行初始化，在 WebSocket 初始成功后回调中进行获取打印机、选择打印机、初始化 SDK 等操作

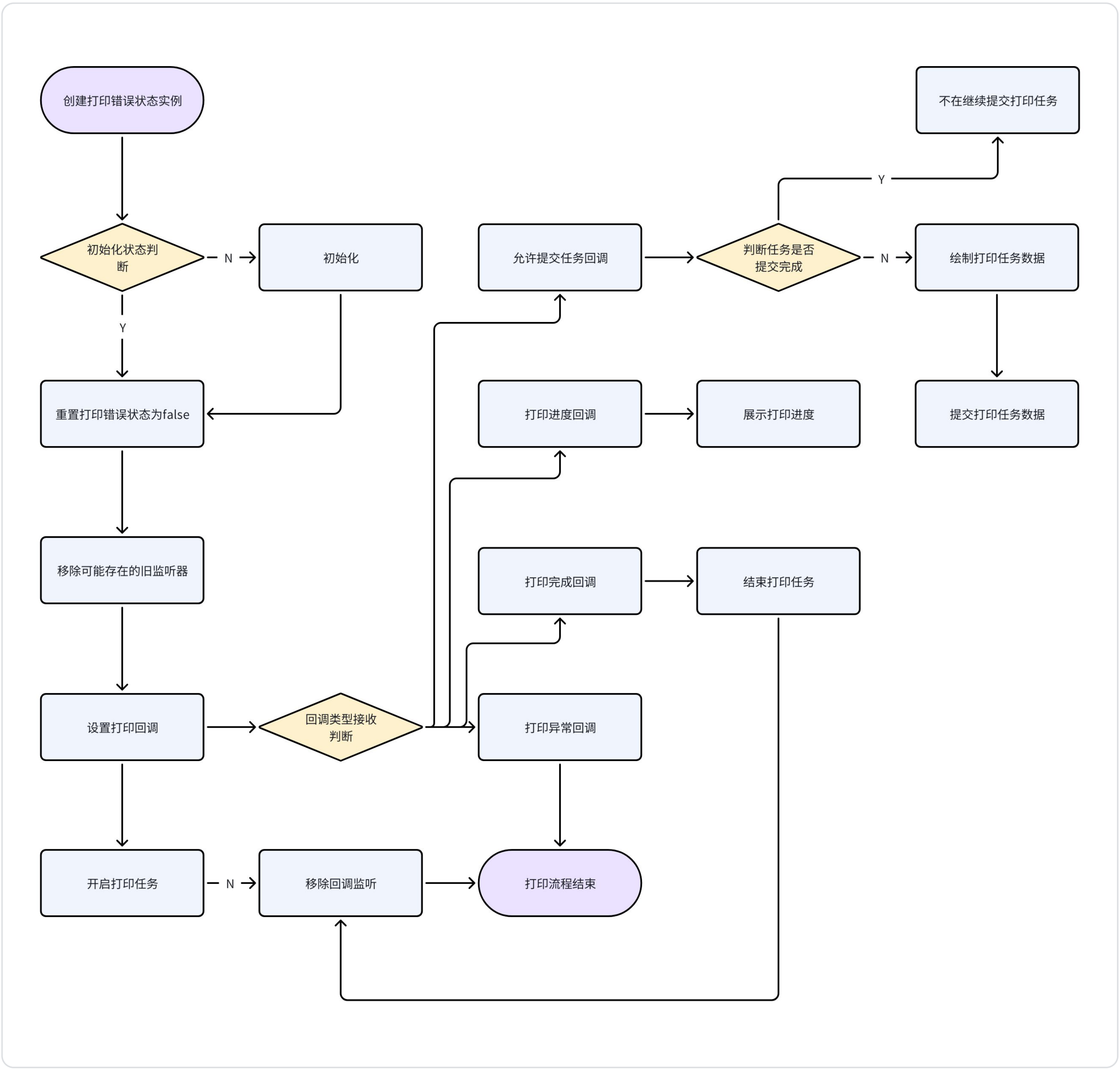
- 因为所有接口均为异步操作，调用下一接口需要验证当前接口结果后再执行下一接口
- 记录打印机列表获取状态、连接状态、初始化状态，打印机需要检查对应的状态
- **打印机 WIFI 配置成功后，后续直接搜索连接，无需多次进行配置（省略 USB 打印机获取、打印机连接、打印机网络配置）**





1.2 打印流程

- 打印前建议判断 WebSocket 是否初始化成功、SDK 是否初始化成功（包含初始化 SDK，获取打印机、选择打印机三个流程）
- 因为所有接口均为异步操作，除 WebSocket 初始化调用是在单独的回调接口中判断是否初始化成功外，其他接口应通过添加 await 关键字调用方法后，等待方法返回结果，解析返回的结果数据后进行判断再进行下一接口调用
- 打印回调监听会有多种回调，包含异常取消、页码回调，可参考流程图及 DEMO 进行处理



二、页面初始化相关接口

2.1 初始化打印服务及接口实例（包含打印机状态回调）

代码块

```
export default class Socket {
 /**
  * 打开 WebSocket 连接并返回一个解析为 WebSocket 实例的 Promise。
  *
  * @param {function} openChange - WebSocket 连接打开时要调用的回调函数。
  * @param {function} onMessageCallback - 接收到消息时要调用的回调函数。
  * @return {Promise} 一个解析为 WebSocket 实例的 Promise。
```

```
   */
  open(openChange, onMessageCallback)
}
```

代码块

```
// 创建socket实例
const socketData = new Socket();
socketData.open(
  (openBool) => {
    console.log(openBool, "openBool");
    this.printSocketOpen = openBool;
  },
  (msg) => {
    if (msg.resultAck.callback != undefined) {
      const callbackName = msg.resultAck.callback.name;
      const msgInfo = msg.resultAck.info;
      if (callbackName == "onCoverStatusChange") {
        //盒盖状态: 0-闭合、1-打开
        console.log("盒盖状态", msgInfo.capStatus);
      } else if (callbackName == "onElectricityChange") {
        //"power" : 0-4, // 电池电量等级（共5档）
        console.log("电池电量等级", msgInfo.power);
      }
    }
  }
);
```

2.2 初始化 SDK initSdk

代码块

```
export default class NMPrintSocket {
  /**
   * 初始化SDK，在打印服务连接成功后调用此接口。
   * 在调用SDK的绘制接口之前，必须先调用此接口。
   *
   * @param {object} json - 包含必要参数的JSON对象,格式如下:
   *  {
   *    "fontDir": string, //字体文件目录，默认为""，暂不生效
   * }
   *
   * @return {Promise} 返回一个 Promise，解析为初始化SDK的结果
   */
  initSdk(json)
}
```

代码块

```
//初始化SDK参数JSON
{
  "fontDir": ""
}
//初始化成功返回JSON
{
  "apiName": "initSdk",
    "resultAck": {
      "errorCode": 0,
      "info": "initSdkApi ok!",
      "result": 0
    }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//进行初始化
async init() {
  if (!this.printSocketOpen) return alert("打印服务未开启");
  //初始化数据
  try {
    const res = await this.nMPrintSocket.initSdk({ fontDir: "" });
    if (res.resultAck.result == 0) {
      console.log("初始化成功");
      this.initBool = true;
    } else {
      console.log("初始化失败");
      this.initBool = false;
    }
  } catch (err) {
    console.error(err);
  }
}
```

2.3 获取 USB 打印机列表 getAllPrinters

代码块

```
export default class NMPrintSocket {
  /**
   * 获取所有当前PC上连接的精臣打印机
   *
   * @return {Promise} 返回一个Promise，解析为打印机列表。
   *
   * @description
   * 需要在打印服务连接成功后调用此函数。
   */
  getAllPrinters()
}
```

代码块

```
//返回结果
{
        "apiName": "getAllPrinters",
        "resultAck": {
                "errorCode": 0,
                "info": "{\"e623012991\":\"31\"}",//打印机名称及类型
                "result": "true"
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//调用流程
async getPrinters() {
  if (!this.printSocketOpen) {
    return alert("打印服务未开启");
  }
  console.log("开始获取打印机");
  try {
    const allPrintersRes = await this.nMPrintSocket.getAllPrinters();
    console.log(allPrintersRes, "allPrintersRes");
    if (allPrintersRes.resultAck.errorCode === 0) {
      const allPrinters = JSON.parse(allPrintersRes.resultAck.info);
      this.printers = { ...allPrinters };
      this.selectPrinter = Object.keys(this.printers)[0];
      console.log("printers", this.printers);
    } else {
      alert("没有在线的打印机");
    }
  } catch (err) {
    console.error(err);
  }
}
```

2.4 获取 WIFI 连接的打印机列表 scanWifiPrinter

代码块

```
export default class NMPrintSocket {
  /**
   * 搜索Wifi打印机
   *
   *
   * @return {Promise} 返回一个 Promise，解析为打印机Wifi配置信息
   *
   * @description
   * 需要在打印服务连接成功后调用此函数。
   */
  scanWifiPrinter()
}
```

代码块

```
//返回结果
{
        "apiName": "scanWifiPrinter",
        "resultAck": {
                "errorCode": 0,
                "info": "[{
      "deviceName":"K3W-E828013369",
                        "IP":"192.168.1.10",
                        "tcpPort":"9200",
                        "avaliableClient":"0"
          }]",
    "result": "true"
        }
}


// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//调用流程
async scanWifiPrinters() {
  const allPrintersRes = await this.nMPrintSocket.scanWifiPrinter();
  console.log("allPrintersRes", allPrintersRes);
  const errorCode = allPrintersRes.resultAck.errorCode;
  //处理搜索结果
  if (errorCode === 0) {
    const allPrinters = allPrintersRes.resultAck.info;
    this.wifiPrinters = {};
```

```
      allPrinters.forEach((item) => {
        this.wifiPrinters[item.deviceName] = item.tcpPort;
      });
      console.log("wifiPrinters", this.wifiPrinters);

      this.wifiSelectPrinter = Object.keys(this.wifiPrinters)[0];
      console.log("wifiSelectPrinter", this.wifiSelectPrinter);
    } else {
      alert("没有在线的打印机");
    }
  },
```

2.5 连接 USB 打印机 selectPrinter

代码块

```
export default class NMPrintSocket {
  /**
   * 选择并打开需要使用的打印机名称，及端口号
   *
   * @param {string} printerName - 打印机名称。
   * @param {number} port - 连接端口。
   * @return {Promise} 返回一个Promise，解析为连接结果
   *
   * @description
   * 需要在打印服务连接成功后调用此函数，建议在getAllPrinters调用成功后调用该接口，以保证传入的打印机名称和端口的打印机状态正常。。
   */
  selectPrinter(printerName, port)
}
```

代码块

```
//返回数据示例
{
        "apiName": "selectPrinter",
        "resultAck": {
                "callback": {
                        "name": "onConnectSuccess",
                        "printerName": "e623012991"
                },
                "errorCode": 0,
                "info": "select printer ok!",
                "result": true
        }
}
```

```

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//调用流程
async selectOnLinePrinter() {
  if (!this.printSocketOpen) {
    return alert("打印服务未开启");
  }
  console.log("开始连接打印机");
  try {
    const res = await this.nMPrintSocket.selectPrinter("e623012991",31));
    console.log("选择打印机", res);
    if (res.resultAck.result) {
      console.log("连接成功");
      this.onlineBool = true;
    } else {
      console.log("连接失败");
      this.onlineBool = false;
      alert("连接失败");
    }
  } catch (err) {
    console.error(err);
  }
}
```

2.6 连接 WIFI 打印机列表中的打印机 connectWifiPrinter

代码块

```
export default class NMPrintSocket {
  /**
   * 发送消息以选择打印机。
   *
   * @param {string} printerName - 打印机名称。
   * @param {number} tcpPort - 端口号。
   * @return {Promise} 返回连接结果
   *
   * @description
   * 需要在打印服务连接成功后调用此函数，建议在scanWifiPrinter调用成功的回调接口中调用该接口，保证传入的打印机名称和端口的打印机状态正常。
   * 注意：此函数仅能连接 WIFI 打印机列表中的打印机。
   */
  connectWifiPrinter(printerName, tcpPort)
}
```

```
//示例返回成功数据
{
        "apiName": "selectPrinter",
        "resultAck": {
                "callback": {
                        "name": "onConnectSuccess",
                        "printerName": "e623012991"
                },
                "errorCode": 0,
                "info": "select printer ok!",
                "result": true
        }
}
//示例返回失败数据
{
  "apiName":"connectWifiPrinter",
  "resultAck":{
    "callback":{
      "name":"onDisConnect",
      "printerName":"K3_W-F612010061"
    },
    "errorCode":0,
    "info":"success",
    "result":false
  }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//调用流程
async selectOnLineWifiPrinter() {
  if (!this.printSocketOpen) {
    return alert("打印服务未开启");
  }
  try {
    const wifiConnectRes = await this.nMPrintSocket.connectWifiPrinter(
      this.wifiSelectPrinter,
      parseInt(this.wifiPrinters[this.wifiSelectPrinter])
    );
    //此版文报存在问题，errorCode连接成功与连接失败一致，暂时先用result判断
    const result = JSON.parse(wifiConnectRes.resultAck.result);
    if (result) {
      console.log("连接成功");
      this.onlineWifiBool = true;
      this.onlineUsbBool = false;
    } else {
      console.log("连接失败");
```

```
        this.onlineWifiBool = false;
        alert("连接失败");
      }
      console.log("wifiConnectRes", wifiConnectRes);
    } catch (err) {
      console.error(err);
    }
  },
```

2.7 断开打印机连接 closePrinter

代码块

```
export default class NMPrintSocket {
  /**
   * 断开打印机连接。
   *
   * @return {Promise} 返回一个Promise，解析为关闭结果
   */
  closePrinter()
}
```

2.8 配置打印机的 WIFI 信息 configurationWifi

代码块

```
export default class NMPrintSocket {
  /**
   * 配置打印机的Wifi网络
   *
   * @param {string} wifiName - wifi网络的名称。
   * @param {string} wifiPassword - wifi网络的密码。
   * @return {Promise} 返回一个 Promise，解析为打印机Wifi配置结果
   *
   * @description
   * 注意:仅支持2.4G频段网络，且需要在连接成功后配置。首次配置建议在USB连接成功后配置
   */
  configurationWifi(wifiName, wifiPassword)
}
```

代码块

```
//示例返回数据
{
  "apiName":"configurationWifi",
  "resultAck":{
    "errorCode":0,
    "info":"configuration wifi printer ok!",
    "result":true
  }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//调用流程
async setWifiConfiguration() {
  if (!this.printSocketOpen) {
    return alert("打印服务未开启");
  }

  try {
    if (this.wifiName.trim() !== "") {
      const wifiConfigurationResult =
        await this.nMPrintSocket.configurationWifi(
          this.wifiName.trim(),
          this.wifiPassword.trim()
        );

      console.log("wifiConfigurationResult", wifiConfigurationResult);

      const errorCode = JSON.parse(
        wifiConfigurationResult.resultAck.errorCode
      );

      console.log("errorCode", errorCode);

      if (errorCode === 0) {
        return alert(
          "网络配置成功，请断开USB线缆后使用WIFI搜索连接打印机（PC需要和打印机在同一网络）"
        );
      } else {
        return alert("网络配置失败");
      }
    } else {
      return alert("wifi名称不得为空");
    }
  } catch (err) {
    console.error(err);
  }
},
```

2.9 获取打印机的 WIFI 相关配置 getWifiConfiguration

代码块

```
export default class NMPrintSocket {
  /**
   * 获取打印机的wifi配置。
   *
   * @return {Promise} 返回一个 Promise，解析为打印机Wifi配置信息
   */
  getWifiConfiguration()
}
```

代码块

```
//示例返回成功数据
 {
   "apiName":"getWifiConfiguration",
     "resultAck":{
     "errorCode":0,
     "info":"{
       \n\t\"wifiName\" : \"Test\"\n
       }\n",
      "result":"{
       \n\t\"wifiName\" : \"Test\"\n
       }\n"
   }
 }
//示例返回失败数据
 {
   "apiName":"getWifiConfiguration",
     "resultAck":{
     "errorCode":23,
     "info":"select printer connect first!",
     "result":false
   }
 }

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//调用流程
async getWifiConfigurationInfo() {
  if (!this.printSocketOpen) {
    return alert("打印服务未开启");
  }
  try {
    const wifiInfo = await this.nMPrintSocket.getWifiConfiguration();
    const errorCode = JSON.parse(wifiInfo.resultAck.errorCode);

    if (errorCode === 0) {
      const info = JSON.parse(wifiInfo.resultAck.info);
      console.log("wifiInfo", info);
      alert("wifiInfo:" + info);
    } else {
      alert("wifiInfo:获取失败");
    }
  } catch (err) {
    console.error(err);
  }
},
```

三、绘制打印数据相关接口

3.1 创建画板 InitDrawingBoard

代码块

```
export default class NMPrintSocket {
  /**
   * 初始化绘制画板
   *
   * @param {Object} json - 包含初始化绘制画板所需数据的JSON对象。格式如下:
   * {
   *   "width": number, // 画板的宽度，单位为mm
   *   "height": number, // 画板的高度，单位为mm
   *   "rotate": number, // 画板的旋转角度，仅支持0、90、180、270
   *   "path": string, // 字体文件的路径，默认为"", 暂不生效
   *   "verticalShift": number, // 垂直偏移量，暂不生效
   *   "HorizontalShift": number // 水平偏移量，暂不生效
   * }
   * @return {Promise} 返回一个 Promise，解析为初始化绘制画板的结果
   *
   * @description
   * 增加接口说明:
   * 1.在调用绘制接口之前，必须先初始化SDK。
   * 2.绘制元素前，必须先初始化画板，否则会引起崩溃!
   * 3.初始化画板时会清空画板上次绘制的内容!
   */
  InitDrawingBoard(json)
}
```

代码块

```
{
        "apiName": "InitDrawingBoard",
        "resultAck": {
                "errorCode": 0,
                "info": "init draw board success!",
                "result": 0
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
asnyc InitDrawingBoard(){
  const InitDrawingBoardParam={
  "width":48,
  "height":30,
  "rotate":0,
  "path":"ZT001.ttf",
  "verticalShift":0,
  "HorizontalShift":0};
  //设置画布尺寸
  try {
    const res = await this.nMPrintSocket.InitDrawingBoard(
      InitDrawingBoardParam
    );
    if (res.resultAck.result != 0) {
      return;
    }
    // //进行下一步操作,绘制元素
  } catch (err) {
    console.error(err);
  }
}
```

3.2 绘制文本 DrawLableText

代码块

```
export default class NMPrintSocket {
  /**
   * 绘制标签文本。
   * @param {object} json - 包含标签文本信息的JSON对象。
   *   JSON格式要求如下:
   *   - x: x轴坐标，单位mm
   *   - y: y轴坐标，单位mm
   *   - height: 文本高度，单位mm
   *   - width: 文本宽度，单位mm
   *   - value: 文本内容
   *   - fontFamily: 字体名称，暂不生效，使用默认字体思源黑体
   *   - rotate: 旋转角度，0:0，1:90，2:180，3:270
   *   - fontSize: 字号，单位mm
   *   - textAlignHorizonral: 水平对齐方式：0:左对齐 1:居中对齐 2:右对齐
   *   - textAlignVertical: 垂直对齐方式：0:顶对齐 1:垂直居中 2:底对齐
   *   - letterSpacing: 字母之间的标准间隔，单位mm
   *   - lineSpacing: 行间距（倍距），默认1
   *   - lineMode: 1:宽高固定，内容大小自适应，预设宽高过大时字号放大，预设宽高过小时字号缩小，
   *     保证内容占据满预设宽高（字号/字符间距/行间距 按比例缩放）
   *     2:宽度固定，高度自适应
   *     4:宽高固定,超出内容直裁切
   *     6:宽高固定，内容超过预设的文本宽高自动缩放
   *     建议设置为6
   *   - fontStyle: 字体样式[加粗，斜体，下划线，删除下划线（预留）]
   *
   * @return {Promise} 返回一个 Promise，解析为绘制标签文本的结果
   * @description 绘制标签文本前必须先初始化画板
   */
  DrawLableText(json)
}
```

代码块

```
//返回数据示例
{
        "apiName": "DrawLableText",
        "resultAck": {
                "errorCode": 0,
                "info": "draw bar code success!",//此处返回信息有误，下个版本修复
                "result": 0
        }
}


// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async DrawLableText(){
  const DrawLableTextParam = {
    "x": 20.0,
    "y": 10.0,
    "height": 10,
    "width": 50,
    "value": "精臣SDK",
    "fontFamily": "宋体",
    "rotate": 0,
    "fontSize": 4.0,
    "textAlignHorizonral": 0,
    "textAlignVertical": 0,
    "letterSpacing": 1.0,
    "lineSpacing": 1.0,
    "lineMode": 0,
    "fontStyle": [false, false, false, false]
  }

        const res = await this.nMPrintSocket.DrawLableText(DrawLableTextParam);
  if (res.resultAck.result != 0) {
    return;
  }
  //进行下一步操作,继续绘制或提交
}
```

3.3 一维码绘制 DrawLableBarCode

代码块

```
export default class NMPrintSocket {
  /**
   * 绘制一维码条形码。
   *
   * @param {Object} json - 包含一维码条形码信息的JSON对象。格式如下:
   * {
   *   "x": number, // x轴坐标, 单位mm
   *   "y": number, // y轴坐标, 单位mm
   *   "height": number, // 一维码宽度, 单位mm
   *   "width": number, // 一维码高度, 单位mm (包含文本高度)
   *   "value": string, // 一维码内容
   *   "codeType": number, // 条码类型:
   *                      // 20: CODE128
   *                      // 21: UPC-A
   *                      // 22: UPC-E
   *                      // 23: EAN8
   *                      // 24: EAN13
   *                      // 25: CODE93
   *                      // 26: CODE39
   *                      // 27: CODEBAR
```

```
     *                     // 28: ITF25
     *   "rotate": number, // 旋转角度，0: 0, 1: 90, 2: 180, 3: 270
     *   "fontSize": number, // 文本字号，单位mm，字号为0则文本不显示
     *   "textHeight": number, // 文本高度，单位mm，高度为0则文本不显示
     *   "textPosition": number // 一维码文字识别码显示位置:
     *                          // 0: 下方显示
     *                          // 1: 上方显示
     *                          // 2: 不显示
     * }
     *
     * @return {Promise} 返回一个 Promise，解析为绘制一维码条形码的结果
     *
     * @description
     * 1. 绘制元素前，必须先初始化画板
     */
    DrawLableBarCode(json)
  }
```

代码块

```
//返回数据示例
{
        "apiName": "DrawLableBarCode",
        "resultAck": {
                "errorCode": 0,
                "info": "draw bar code success!",
                "result": 0
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async DrawLableBarCode(){
  const DrawLableBarCodeParam = {
          "x": 20.0,
          "y": 10.0,
    "height": 10,
    "width": 50,
    "value": '12345678',
    "codeType": 20,
    "rotate": 0,
    "fontSize": 4.0,
    "textHeight": 0,
    "textPosition": 0,
  }
```

```

        const res = await
this.nMPrintSocket.DrawLableBarCode(DrawLableBarCodeParam);
  if (res.resultAck.result != 0) {
    return;
  }
  //进行下一步操作,继续绘制或提交
}
```

3.4.1 二维码绘制 DrawLableQrCode

代码块

```
export default class NMPrintSocket {
  /**
   * 绘制二维码。
   *
   * @param {Object} json - 包含二维码信息的JSON对象。格式如下:
   * {
   *   "x": number, // x轴坐标, 单位mm
   *   "y": number, // y轴坐标, 单位mm
   *   "height": number, // 二维码高度, 默认宽高一致
   *   "width": number, // 二维码宽度, 单位mm
   *   "value": string, // 二维码内容
   *   "codeType": number, // 条码类型:
   *                      // 31: QR_CODE
   *                      // 32: PDF417
   *                      // 33: DATA_MATRIX
   *                      // 34: AZTEC
   *   "rotate": number, // 旋转角度, 仅支持0、90、180、270
   * }
   *
   * @return {Promise} 返回一个 Promise, 解析为绘制二维码的结果
   *
   * @description
   * 1. 绘制元素前, 必须先初始化画板
   */
  DrawLableQrCode(json)
}
```

代码块

```
//返回数据示例
{
        "apiName": "DrawLableQrCode",
        "resultAck": {
                "errorCode": 0,
                "info": "draw qr code success!",
                "result": 0
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async DrawLableQrCode(){
  const DrawLableQrCodeParam = {
          "x": 20.0,
          "y": 10.0,
    "height": 10,
    "width": 10,
    "value": "精臣SDK",
    "rotate": 0,
    "codeType": 31,
  }

        const res = await
this.nMPrintSocket.DrawLableQrCode(DrawLableQrCodeParam);
  if (res.resultAck.result != 0) {
    return;
  }
  //进行下一步操作,继续绘制或提交
}
```

3.4.2 二维码绘制 DrawLableQrCode

代码块

```
export default class NMPrintSocket {
   /**
 * 绘制带logo的二维码。
 * @param {*} json - 包含二维码信息的JSON对象。格式如下:
 * {
 *   "x": number, // x轴坐标, 单位mm
 *   "y": number, // y轴坐标, 单位mm
 *   "height": number, // 二维码高度, 默认宽高一致
 *   "width": number, // 二维码宽度, 单位mm
 *   "value": string, // 二维码内容
 *   "codeType": number, // 条码类型:
 *                     // 31: QR_CODE
 *                     // 32: PDF417
 *                     // 33: DATA_MATRIX
 *                     // 34: AZTEC
 *   "rotate": number, // 旋转角度，仅支持0、90、180、270
 *  "correctLevel": 2,//纠错级别，取值范围1-4，默认2
 *  ""logoBase64": ": string,//logo的base64编码(不含数据头，如
data:image/png;base64,)
 *  ""logoPosition": ": 0,//logo的位置，取值范围0-4，默认0:居中，3右下·
 *    "logoScale": 0.25,//logo占据二维码的比例
 * }
 */
DrawLableQrCodeWithLogo(json)
}
```

3.5 线条绘制 DrawLableLine

代码块

```
export default class NMPrintSocket {
  /**
   * 绘制线条。
   *
   * @param {Object} json - 包含线条信息的JSON对象。格式如下：
   * {
   *   "x": number, // x轴坐标，单位mm
   *   "y": number, // y轴坐标，单位mm
   *   "height": number, // 线高，单位mm
   *   "width": number, // 线宽，单位mm
   *   "lineType": number, // 线条类型： 1:实线 2:虚线类型,虚实比例1:1
   *   "rotate": number, // 旋转角度，仅支持0、90、180、270
   *   "dashwidth": number // 线条为虚线宽度，【实线段长度，空线段长度】
   * }
   *
   * @return {Promise} 返回一个 Promise，解析为绘制线条的结果
   *
   * @description
   * 1. 绘制元素前，必须先初始化画板
   */
  DrawLableLine(json)
}
```

代码块

```
//返回数据示例
{
        "apiName": "DrawLableLine",
        "resultAck": {
                "errorCode": 0,
                "info": "draw line success!",
                "result": 0
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async DrawLableLine(){
  const DrawLableLineParam = {
          "x": 2.0,
          "y": 2.0,
    "height": 2,
    "width": 50,
    "rotate": 0,
    "lineType": 2,
    "dashwidth": [1,1],
  }

        const res = await this.nMPrintSocket.DrawLableLine(DrawLableLineParam);
  if (res.resultAck.result != 0) {
    return;
  }
  //进行下一步操作,继续绘制或提交
}
```

3.6 绘制图形 DrawLableGraph

代码块

```
export default class NMPrintSocket {
  /**
   * 绘制图形。
   *
   * @param {Object} json - 包含绘制图形信息的JSON对象。格式如下:
   * {
   *   "x": number, // x轴坐标，单位mm
   *   "y": number, // y轴坐标，单位mm
   *   "height": number, // 图形高度，单位mm
   *   "width": number, // 图形宽度，单位mm
   *   "rotate": number, // 旋转角度，仅支持0、90、180、270
   *   "cornerRadius": number, // 圆角半径，单位mm，暂不生效
   *   "lineWidth": number, // 线宽，单位mm
   *   "lineType": number, // 线条类型: 1:实线 2:虚线类型,虚实比例1:1
   *   "graphType": number, // 图形类型: 1:圆，2:椭圆，3:矩形 4:圆角矩形
   *   "dashwidth": number // 线条为虚线宽度，【实线段长度，空线段长度】
```

```
     * }
     *
     * @return {Promise} 返回一个 Promise，解析为绘制图形的结果
     *
     * @description
     * 1. 绘制元素前，必须先初始化画板
     */
    DrawLableGraph(json)
  }
```

代码块

```
//返回数据示例
{
        "apiName": "DrawLableGraph",
        "resultAck": {
                "errorCode": 0,
                "info": "draw graph success!",
                "result": 0
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async DrawLableGraph(){
  const DrawLableGraphParam = {
          "x": 2.0,
          "y": 5.0,
    "height": 30,
    "width": 40,
    "rotate": 0,
    "graphType": 3,
    "cornerRadius": 0,
    "lineWidth": 4,
    "lineType":2,
    "dashwidth": [1,1],
  }

        const res = await
this.nMPrintSocket.DrawLableGraph(DrawLableGraphParam);
  if (res.resultAck.result != 0) {
    return;
  }
  //进行下一步操作,继续绘制或提交
}
```

3.7 绘制图像 DrawLableImage

代码块

```
export default class NMPrintSocket {
  /**
   * 绘制图片。
   *
   * @param {Object} json - 包含绘制图片信息的JSON对象。格式如下:
   * {
   *   "x": number, // x轴坐标，单位mm
   *   "y": number, // y轴坐标，单位mm
   *   "height": number, // 图片高度，单位mm
   *   "width": number, // 图片宽度，单位mm
   *   "rotate": number, // 旋转角度，仅支持0、90、180、270
   *   "imageProcessingType": number, // 图像处理算法，默认0
   *   "imageProcessingValue": number, // 算法参数，默认127
   *   "imageData": number, // 图片base64数据，不含数据头
   *                      // 如原始数据
为“data:image/png;base64,iVBORw0KGgoAAAANSU”
   *                      // 传入的数据需要去除头部，数据为，“iVBORw0KGgoAAAANSU”
   * }
   * @return {Promise} 返回一个 Promise，解析为绘制图片的结果
   *
   * @description
   * 增加接口说明:
   * 1. 绘制元素前，必须先初始化画板
   */
  DrawLableImage(json)
}
```

代码块

```
//返回数据示例
{
        "apiName": "DrawLableImage",
        "resultAck": {
                "errorCode": 0,
                "info": "draw image success!",
                "result": 0
        }
}

// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
```

```
// 调用流程
async DrawLableImage(){
  const DrawLableImageParam = {
    "x": 2.0,
    "y": 2.0,
    "height": 10,
    "width": 50,
    "rotate": 0,
    "imageProcessingType": 0,
    "imageProcessingValue":127,

"imageData":"iVBORw0KGgoAAAANSUhEUgAAAZAAAACgCAYAAAAisjrVAAAAAXNSR0IArs4c6QAAAA
RnQU1BAACxjwv8YQUAAAAgY0hSTQAAeiYAAICEAAD6AAAAgOgAAHUwAADqYAAAOpgAABdwnLpRPAAAA
AlwSFlzAAAOxAAADsQBlSsOGwAAROVJREFUeF7tnQmYFcW1gFtEDe64Am5BIyhxe/jcRTQquBDc17gb
FfdEfUYlKu6JccWnKIoYiIIGNVEB81TcggsEFQFN3BWjKIILaIG+9V/pmuoqam+3ffevjPPzPm/r76
Z27e6e6urrvvvXXqLHVqia+++ipeuPD7qF27JaM45m+7aIkllpCiKIqiKHEcS/n++++NbGhn/i40ssLIid
mzZ8dLLdU+at9+qegHP/hBvQBRFEVRFIsVIP/+97+j//znu2u2Fnfx0v4Wfxxe9fPXKIWH/7OdbxMgsmsUVuxkZQVPSv8xZSKfdZTUUxRMg8WGjCHMp8Fpo2ZEkjtdCIxSmI7QYXRu3wqHqKEVHs34pP
ZR0vvvvuu2jBgm9RONqp8FAURVFyY2XGEgsXfh9jy1pcWbhwahW8+fPj/71r38eo/Ih27dOkSLbPM
MmqOUxRFqQHffPOvAkz+MbJ6yDffvttNGrUqGj8+PYyxxxnYGdYTfb2xvIBtaNdtkLxTH/nmfRqtq+XCUyRFt1BPzn86eOPuxjAv61HRcXjLvx+AzTK9WEDt8RzjX9rxnEbbQu8YJr/MJvPKRyVHOPnfJeTTs/YWGZ1Pyo6LTNqfYBlsM0H4ywh1bCsJ0ZlazG9zVY
2muuvRUOHDo1++MMfSjuKojQ/LyIjSIEIGTCXE1hsBF9Cp8rUFFppasZRl68ygqqSX0qHqy+6ENEDPEkQKCvu6fHSuMsBXDOBi+QsXQUbeyrZDBucZbCfxCrbxYqJ16QOl/XLFA6gDWb9LNb4m+zzFmK7wcKiMB9HBFYCMoTB14J1eApNRHZHDPCq4xSg+B9y
fkQk8YDAptfaQH/KlOxzwoBJUDVtcHhWv24ORLcrBhZQl7vLWxJwR/5iBgAlCnGf7xdYj1rAG10TMDv3++L2SnMqaSMPKdEX
akNMbN3FSjvPBAi4KqoJs8WRh5nAVDnTL+cJgN2zqA9aM5+6hLNJ3gzLoLlP2tSHgvyB5EaCfTS2Kt2TlS+hNsyfetcjv6K1Pj2/aFZfZsrdw3NlapZnDIVuFamVSdR9mcKnKqBkr0d2mWm8ZPP2kjptj
dLR4+rE7IEghoBQpEPsWcCTbagaiMX3fr9+Gs0Pnx5FlrgagaPnp3DRCwXRyaNP3Ai3LX4SBp6jNoRRO7KFwqIGhuo8JUNVYP30YR2TrBo8qM+I/BjuJOJRSeyARCAW/hbEvi4TcK+5KqaTUAnEy5TE4JXoVNhVS5XVjzKUF2A2c8YDh9PQR4wdWcmyKbmn
gvkCqnsqYBnQGjnfxNYuTDSPUh8BtJAy5JpOpC4ehcgQCOxbVzQHVMqcADJGUgIqr5TeG6bHqeMKtVxRHIyx20Nn1lCN3Vb1VaxXrNDaw4LGwa+9bZvIOpp2yHE0X0Fd
mn5sT0tm4DO+HX3WWP8EPjjuMNiYABzGOcNzt4KBwBGRN0ibVnbL64D8RiJUcyhZ/3HRoADZt+2mfYgnjuYsKhCRRYzmAwfnWOCEQuU5ehtFx7oV2Fa+B
gTvnJcUC3AGGaG1S/1HK+sqYANJ5cqXIcpvRrCE2i2rGdwUF7vcgN1Y92TzYKNEj22Ae59uf38RnHt7B/Yw9d/ZjXsq5iHyxl3iMRIhXrn80c6o0O5H6rgUXzaX
4mL87hl1gfZijSFkLCLb/CMPrjjNbwywgL6TGe8Qb/ItYOmY+/Z6MmuoFUaK2X1IMzgxr/fbdvkebs94wPA0QdpfvjfSvN6+z/T2bu/nCdz0Oe14cjnG+dFgc6QV
xkzXlRGhOKY2mM92uV4KIcE66uU8H6M6Bu1KN6NzNRn/cUM9tbBxk17fAHAB0bQ77pV3+QcuhVP8INxa5r7wCPCBVhy/BHm6bsNNW1eBBtHVCBl1vyUb9ozZdD/ZZ
XEkdaBzsxsvevIkJtjMKG+ZZ1tb4bZ3Ozk/TWyzSaUsTd2W6CoTpHmzD38+ljNgNeT5Vy5Ec5sBpUD/GUL8N9xkLvScSZwc4+o0JAUXfnwcHxFqzbzrUNfoPp
5KE+dHJq8bWCTtwdFjRd28xTGLgbbgOF/6MDwi4M4s6+/i1cOMdf4v6TwSXIEB0MFPYyNbMFb1PyDBM7GRbH+MmdZh7c1pS6bRM5iN7QIKuRsM2b5h/HLAiHBm
lWGjW/C33wjBR76vJmVkUZyCxz/4gW+uPGQjR3Bns0XXnQr0Mh0h+cbeHYkfMOXebk9wVqF9zSSTv8hWd0YC0WQHGjUF2DiAgpP+Bqfr+2F/uLezMqJb9+PMeBMdMuY
X33wkVQJTYxB63XtXwt1rvk3MK0j8zu+WLUYLqMwK9khBIO3Ffqq0OqMQj9lEb1izF+ymX4I0KfjWX7KJL+j5D8KEnLTAwKs2wRCb7YT9UvaaG8eHWvLJ85r+Wc0
hV0VBtDdrfVF1zvDLuS0UY43fX0a5DDW0JYu0PsrfTIGtm0tEuwtsK3iAdxEzv4xLvPLHPPYbpgoU1n+0njj9ciSgtoSAoGSmkQuMp9Dsm66TjiQ5TRcewcnTGnz
bVX7y9ZBzk7YTKN/mNNWVUBqq9oyWfqRkJ2yU3Mq7PzyL1XUwI9vjHRJXVyBu4KY4Qy8jrIRJJZCRm4xGe0g+YtQzVQgQsQjbh7mmMBAvA/tvcRLzs77W5
W0zqzHcS/MHbGmLfzCrDRqNIWHp/QnCCkkGafGhZ2HHtqxChfl/9eZI8hGM6pbtVD3vOBOGMF3qVxNFbzkmRFD/WuECXbT1/lqR8vvO9hhxjmz0Mr4c1mjZCW/kC8/1B/
cG8u/3b3GHb83/zLZrOTDmqt6SmdPhDJSDbOL9Gq+yQz4GgHu13sIDgr4LhxnWe3Zh7r3IZMVezmEWSs8GeSvTe7K4ExrVFzMgNndwOeHUl0Hvo0Z5TFyf6Q
HgUMIQyXwBxDnJhBEMZd0nmWkiJ3v9/5s5tN8cL3i4eW5pPGyQ/l4+UxK41wNP/xm/1H8BgyIJKOxvC1JEGl89FXxk1ITdaW2ovDUSdeY9+wEnmH2W7RAJ
5/UHRcrLiYW9nkZjBeUFLF3eN3S3uNWJYMqV2vf3k5Yg1uDpUB+qqj4RrZpD4AXxfEeH4J6JZNdg1ZeEAEF57NbKubCtmvwDCv2m+aD8uMRoUWqGZQUeVRbtHMFc/d
FKjadSZK/7qQXNLxqSMQnOI5RBGGnTbSKV4L3A+Q/ELYaHgtTM0CoIVb9hcCqlvYf2yjyyaxaYNGfgLoKqmXK9UuFvR+yJKs7QJpmVu5qG2BdNVI58hNeevkvHJm
V3sCzS0Tx3gWDs+AKXVMFZ3cwdh/+0fz8lbSgaLHNlVkb8IVYGJVb7IswwIcX0i+KHvd0BpVN6iycQe7vGqvUUWjMdIpySAnqEnFJqbV8kXcSs4tJoxf71Bq7qdFs
W2PWDgTUESSVS0TwGUbS0KdgUNxKdm+M3q6vUPOVn0C3mS8ULCTfq0Z6N2xF1zFjsbv7yfq0w2GGNnWU0vPAbP/RLlYL/J3sM9vbwXDw5jNmHTbqMy3VE4+yk/Lfp
cBgAhhEUXv5uwFIS2G+IGw9y4JT0drmrM9l/hehWIyhE7q8CLCh00MYpjlnlWqIEAwVvb4e0h27ANN06ruuqebyBNMvkPHWNN1y6KWQAC/pINGjPE6pKWAu0wGn
4LkFjWaq0zqSQP+3M0Hko6DDhX+2KY2mOKWKfTcs+e4uAfaj5EdFF0OJ/NnNfq+nTdUO+qWp+Skk5K+u+9L3AEJ8JHfu6Mpw7Sy72V4wLRM/Uj0l0g95udRXKehA
KyKjXWNN10YvU7Ik1KeW9Jb6ck4tNq0YTLBz15jQL9JbXYtQhRPGNAWTZNNSK2gGDuEQU7bFqFz0wZzQgbnZ2/NqwD10aJOfXb6xFcUGuRyIGG6PYhh5J0vtOPSk
DtI/pIQ8LPvXBsHvXo0DHhWOHxI9BD2YFE0mIXCaOJAdr9bkJfvP8/Fim0AqGzjKZGQjb+i9lMaTnfhwq99VNe3A5T7JlYtMh7yfajVVRJC0PAGeN6aELQ4Yay3Aq
yd2T0I6/HQuTUxp6O6ubABCOL3bdrhiWo76VnQWd7GYjnT9ozH6yDpmHBB2SwZGhS2e2s4n2H6pNY2A9w9UMHWGmoG38hjyvIEMCkZJ9twgNhBx6K6WXCfXtQBAL
i9BbbyrJKxjXsF2uuE1tT4KFV5tSw5rTuH/pN3CM5qfY7gb9Pd0XfYo1eNm8kYYtGtZ5ybD/GTCYFCaGkLXJvW6H6WNUkmiL9yM1QLhCO4I3ZeyYsdFZPHt9g
M5Lz3TFSy8nOm5DW8AFbWWsqywVOKr3sMdnEP0NkOE2LUjHrrDbPF+y0PrAHiF+P9x1PvV6lBKyRm4kbqw1rFLqYSX0y0FL+NdwH67+i1n+L0JGmpMvH4Gk66
YWC0CAeSVIUt6fcwjnLAUY8PPjsqmn6LlH3MG9I6hz0c+cFo/Mmdg2hFgbXhyO3oX6nwbFbzBT+mexZoD0nM4FAUWLNBxNnqGPFlMIp9xW8TjX/dN5uBN+nPr5Q7
q0YjbyNkCMtMQxVJ5nD2oTIuBs6gRVZvkGVKyNrP4RdHSJ1YqLO3w/tPPDbGIzJ7yYLLh9Z2H2UAEdfRk4ZnrNt9n9fb4i6wPxWIBWjq8vkuX2yHRDpmWYOTV
0opDCwY9ZC3UMNRbBgliJnDH4M/4tBVAMzADQfPthf/OIXYg5DCGH2ImLif//3f+UY57MiHhX30EMPF
cGSFpKnKMriCdFOmK6LILTq/MUXX0zebQhjoFuP8cyFQKA25YM1EjUX77zzTmwG8dgIkuRI5cyYMSM2
Qqh+RbkZ+KWUgq13jRaSvGrI66+/LqvajUBKjsTxzJkz4y+++CJ5pShVhEvmmzatxMJn015a9
```

```
E90lbdb7eeuvFw4cPry+DBg0K1nOLmfzGHTp0qKhflT4LSt62iqSsleg4rY866qig87tcWLOx/fbb15
uqWN2ZtcITkxXrPUIQn00ctOt4x8zFGhRFUVonRfka0iKYyOztOtHJoJEFGkhzRUQ1B7UzIiqKomRgF
7wRRWmLXe1d6hgTRLJfYFKqljSfBdGcrgBhPRum9lL9YpKLmb2SfvnPwrYN/jH72h4r4jlUQuZ+IErb
AV8Ruc2IeCsVnq0oeQkNzltuuaX4UoE0SEcccUQj5zTnlTrGX76vrEH7wx/+IMfwSfz3f/+3/O/iXi9
E2nkhsJpg8XDx+4oWQmqlcvsVehb2+fnH3NeAVcdeD9Ku6Z9XDYXtB6K0Dj755JO4d+/eYlv++9//nh
wtDvPDis2sLt5vv/3Eb6W0fsw406hsvfXWybvFwp5Boev5/gGftPNCZdttt03Oyk+aP6JWzwHS7qlI8
IE0iQbCGo0333wzU81ac801xcfyzjvvJEdKs+6664oqmxdmLIMHD47Oussido69dRTk3fyY2fooa1y
89wns4cf/ehHjXw+zFpYC4NviHbcVa9uVEfe42gQLOIMJXvLggWgRLSRuqFITYR7JEszedPIIMDOkkr
rxs6gXZjB29ky71cTjsvwxfcKmMGzHMDHn+kTdUVdC9k08O/mwW+L35qZbCWvFvWH1E7du3eXY2nagP
scoJpn4T4HSHsWRQ73TaaBvPfeew2kYFphj5Azzzwz+F6onHzyyckV8kPEhfmgYvOhJkfyQyTYYYcdJ
tf+7W9/mxxdxMcffxwb1bVRP/1y2223JWcswnz4sgfK5ptvHt97773J0UUYQdBorxMi4jju88orr8Rb
bLFFvRYxZ84cea5GONWXn/zkJ/Hee+8d77TTTvXH2AmSnR3NF1t2f+zZs2f9e2awj/v37y/nufXPOee
csqLd5s6dK7u2FRHNp7R8Qt//pi6+BhKKumL3VDfqKi3iKk9bFHffkDQNpDlKkZQVhVUNLPYz1ytZyJ
1lBidZVX7hhRdGhxxySLAexc4A3H2Gkb5//OMfRYqXKmbgkzaYSYTed8tDDz2UtF4HkWD33HNPNHbsW
FmwyJ7sU6dOTd6NZLbOPicXXHBBoz5TZs6cKe2GosN4H80D7Yt7RxOzheRszEy4nts/IuM4zvu2Ln0w
wkPyitGebZuYeVbX2sI+LdwHKRvsMVbVch6OQjQ7kmHa9/7xj3/I8yAHmT1GIf0+Mfl8fm7f0gp9pT4
ztND7frnqqqvkHhSlKPwoKV7zmySS0zrM86QtAb8tfo9kyciKKm0tNIkJi0uwVwcL/A488MDk6CIYhP
/rv/5LUgUYLSS65pprovvuu08EAqF0PldffXVkNIBo3LhxkoUXAYLDhkY06DO008/LfmzzjjjjKhXr
17JO41h8CKlCpEeaZA52A0l/NOf/iTmH/r3P//2pBPPvlETHlpGO1LhxkoUXuAYLDhkY06DO008/LfmzzjjjjKhXr
G/QxDUxQRGikwUKpfv36iXqNKTCLxx9/PNp9991l4RRmH1QmV2zgAv9RGAghMhkYBkxYoRErIgqqHID
zUP0RysriSanvaFOByRjTssU3KWH+RXgwubGTU34fpC3x8dsK1WM5AZM7Wy+treagyOGe322TCRCEBx
lzGbQRDDa3Fe8NGTIkMmqjZO9l5ssgj3bBjPenP/2p1APqMuPFXsmHzup0tIJy+POf/yyzZdrea6+9k
qOl8X0b/CgQbHzpWKHPandmyo8++mh0zjnnRL/73e+kngt9Z3ZP330hh1AgZPDkk08wYRdFTttCGbv
FO4/zV7Ke0SL4K9BUyJ5pQ8fvn2ODOSlfug8cz47PpuRI0cGU81kgTZDG9ttt1107bXXXRr/85S+jDz7
4QCYKSuuFyU5zQ2r2rAglfktMYq0AYTJERJSP31aoHr5WNworra3mgHsviiaPwnr33XfFto4d8dpuJg
L4Zz/7Wfzzn/9cbPz4AqhnPuD4oosuSmotwkhzuumlUrdcnn/+++djjMEMrygRjtSa7pFqNJi0fm
74ZiOPu3bvHZnaenFEafBTjx+PjcCMjWBtsII+C+pzfd8fUglGE4pXXXHFF+VsKbMJc02hYYZHyMJph
bASu7Hdvnz9/zz77bPG5KEpT4vskIllmGfF/+P4NpTT4QJpUgICZXYuDl1BRnOukNGHpPwOqhQH1kks
uEUfUW2+9lRyNZdBE2PBc66F/0lt4H4piirLBechLe6IDxw4PMALQhCMTZok9xxefvvlqW
e0iOTMhpjZSzxgwAC5ttHS4mWXXbZB4ZkQSrj66qs3eo8fAufhBPf7nIbrQMqQ2kE2MTwr
wTu2WhVsdEy5fUzzzwjnxWCEHiOZuYnjn5FaQp8AWK08Lhr164qQMqE8a9ZFhLiD8F3gQ8BZxPhbqNH
j673J9ClV199VcwdZN/FqQ7/93//F/Xt21dSChgBI8eAjJiontj9cSRjlsHchKnHhXazfcCA4kDGLce7
XX38tfoStttqq3uSGn4XwOFJJ46vBnIMPphteornkh5Tw7M/oQDHDZZZeJMx0TXblwrhm0xdSEfyEL7p
Okkw8//LD4KLDdWqzPBr/OvffeW3aKBKyV7NFy/fXJM4Pnj/+EvAQ1cHzMlzvkZM2Y0uL6y+INDO
guCbNzfM4EdgwYNSl6F4XfC755QWHtu2nl++3z3QiHk/PbtmJGnDxa/fZ+8/SqStGuGxpxKaXITlg/m
C9OP2AiG5EhDzCAo7xPWiumKWTahvsz205g3b54kfkwDMwwmlLFjxyZHGvPxxx/HRngkrxCzrEJczH6PP
ITznllAYaCJx33nnxAQcckLwKc/rpp8dAQNpA8UWUNm8hBBjNbq+99hhKzUTkMHDhQnv
XgwYPlHjEp8hoNpVyM8JawX86fPHlycrQuVPmpp54SDQSNzOWCCy6Q+jxxpfXAZ5pV1llnnaR2HWmhs
X5Bs3XPTTvPbz8PeftAyWq/yH7lJe2aRcL412y5sJjt2lTIF198cX0eAVDV
Bnxw1HIO2sCECROSo4tg1mId4aHFfrzPYj5SwqMhoIGUwndiE3mFs33TTTcNRhbts88+ov0QCmwXGZU
Cx3YlpRIuv/xy0VzQYIgC47NhFkNQDkQSEBIJCn7CR12nZX0zYYw+6HMV1xxRXTLLLbdEv/nNb+TzQx
tR2ga+1uk7uENgZWAG756bdl4lWm2ePliy2k8bryrpV17K6X9VJMKkSRk3bpxIQ2zz9RpRrvWVtw1
vzQQw/Jwj/qhxbvubDQj5mzGdiDDm1m96SkRwMxg3hydBHMKo1iPmzzIS3EgIH+oo43MmjVXLXgP3
tcIKK8S33nqrpJk3g6L033JkOnFC7NNnDo4aVCDDWT27NnxDTDTs7PAGTREwIVbtnmjk0/FPXXnvlOt
06FR/jQWNEEc8mhXn0a8//77MNJ//77M/oxPyPOA+EzMBkM+HZ2OPPU/AdEZTA8zIUCuL4faBBkoQs+x5ZYOn
XqlPSwPG0jb3Hb53sfqhMq7nkhqu1rkTSLEx2TCDdy4okn888M776rz88suL2cMuNDhA+bjHzBX1
0vUFf/jWB41apTU8cF8dcIJIgKbJ4iJBEEQkKuAa5nIgk69Onj/TpNNO
```

```
S95dxN133y3vUejPk08+mbxTGgSI0chktTbObQb1vIX6OATzCBAG5Ztuuinu0aNHfT+JeuP+ged/zDH
H1L/HivXQCnkw2pZ8Pttss40IVGA/Fntu3oIznfsHhBCfP+ZNFSCLL6HPudKy7rrrxnfccYeUCy+8ML
NOqXpuBgcmL6E6edvyizU/W/gt3nnnnbnaCmWWcEnra6j4/acUSZMJEC7ETHKNNdaQG0MDCcFgwcDLQ
Eg9BiQ7oFl++ctf1j+gfffdN3700UclsgusAAnNcCnPPvts/Otf/1reZ6buv89M4Q9/+EO84YYbBgUI
g9uDDz5YH/HF9S1EiyHYrFChoHEcfvjhMqvgmvhvmNETplyKm2++uWINBOFz0EEHJa/qoN9EPxFJha/
E9o/oN+7Tf8Y+PBe0LnsepW/fvpKShXDcckKK2RQM4Zi1gZjSOnC/M9UWN5HheynpkUgG6oJVIVRv++
23T2rUWRVCddzrQdo13cJ3Gz9kVkRXWr+yzkvra6j4z6JomkyAYFo6/vjjZebOA/DBbHHEEUfU3zg5l
myYZwic3EhwZtvMVO1AT04mHpr7ECstrO3wTVgMxI8//ni84447xk8//bQ4hN0ZPIUQWvpEqK4LYcr7
779/g7qdO3eOzzzrrLAnt5Z7Jx4Nj0L6PGSykaaSVlVZaqf5cwo8RxDyPqVOnxtdcc40cRzthXcebb76
Z9Kw8XnrppfjKK68UkyPtYQq0AjwPPD/OQ6AprR/7fSyiMHGz8NsL1eE7f/TRR9eXfv36ZdYjJ1yojn
s98MN/QwVzO5NFVxAwYXT7REnrV6XnhYr/LChF0mQCJA9sczts2LDkVfNy0kkniakN00wW1157bXzLL
beUvX0uiwhJROj6Tlo7zLoQQAhgzJaKwuQxNPj5s/885G0rVI+FrlnXzCNAEDoM3K4gKMdvkee8PM8m
7VkUCQKk2aKwfNgQ5bjjjkteNS+33nqrRB8R6ZEFqeFJQeJHFWXB2gpSnpTKe9XaIAqLdBGkciFSTVF
YuxXCjE/Jf/lJa8uP8PTrsTUD1wtFgpYL33GKS1oUVhZp5+V5NqyRaQpajABRFKX1Qe4lv7hJN9MGbc
LsQ+fawsJh9rtx2yIvXQhy0Ln411y4cKGcmzUwk6MuC5YC+PXynFcOWc+G8tdkiUSt0S1tFUWpGf5sH
Ng3/MMPP5T/GeyqWbPgtkXSTzIq+Ky66qoNkrL611xyySUlg8WGG25Yn0wxRKh9hJO74rtdu3ai0WBR
sW3lOc+ypbNhVbXPhnUy/jXyZAbIC2vdVIAorQK+xqSh+Oabb2QwKDftilIbQgKE/WpYsAuVDpJ2ISG
phGxbefGvyYDfqVMnSf1TSoCEII0R/bDwvVtxxRUlW3eptvzzLEUKkG233Va0tFqBAFETltIqwCzBvg
tsSkbhB1o0ZCZgJnnKKafIKn2lMlxTU6V+B7SG9u3by99qQcjRFhOPcmFLB5+Q0PRhM7YQbh82q2A7a
pdKfS/loBpIBjwe9tUguSF/0+DDsruQkSwyC2Y9zEDKcaLTLqldSLLIPhoHHXRQ8k4+sPUyQw/toZL3
PpnxbbDBBrLrmguDAn06/vjj5TU/SMAmTLHY4/TF/erZ43ZAYSY2atQoMSuUwyeffBKdcMIJ8lkMGzY
sOVoMJP+84447ohtvvvFH2lenZs2fyjpIGn6OPu6dGaK8MvqPuPuNpIDzQZuyMndk2CTp90C7cAbvamX
0WCBC+G6U0kLz3TVtoWzyzNPI+LyhyuJd0TaZBpQSscRgzZgxPPbOwsHD33XcPvhcqV199dXKV/LB4j
3T4rIUpBzMwy3oTQmhZI0KiShfSkbBGgxDEUF9tIRMAiyl9zJdYQpOJqWfdiQvx7KGFkSTIJPWMD+0c
csghsnCLzAT0m/MJCaSYH3/cq1cvSca51VZbNTjOayOYYJOayOYJSWKPc5977bbbvF2221XX5/zCdcuZyH
kX/7ylwaL0JTiyZPFgEWwrLnic7SkrdL2V3ebgT1Yr4jCQkK2Wth6662Tq+WH6WDC/EVGa1f3J+eySQzr
qE37yasDJAU2AHQ/O8Usvs2bNlD2WSPpIo8Ve/+pXsUBiqSyGlMljurOgzz77TLZ7ZcZRqpx00kz
yGqc+h9t5Ai3WJna6Sypm9m4JZdH+22m7xPKDLmH2aHoX6z5TCJDkMaDFoGmgsp4DfffHPRyCjMnkjd
z6wP9dztnxG40cSJE2W7W1sfJ+T+fOrT9OdAp9QDv
i+RjBIMfRHEjOyLa806dPr69PXcId2VGS5JZu39IK2wlw7dB7oVJk+uy2Ar8nnh2/PVv4brjHzMDaaG
dMvjuh8/L6w9xz/euVOubC95XvFN8vvvO2uNFigNbA79gtvqOda/Hdx7dnIaIrdF6evlKKRk1YOeALQ
XZZvtj+PgIMbOyVwZoOTCbrrLOOmDiIDGG/kC5duiQ160CVvvLKK+WDNqsGRsGA+vHHfUjAkjKKdq59++t
xcs6CtaesKYiDRx5DOShwd7CAOv+AC699NLohhtukIzHx9+eHk0Dq59++23R4MHDxYhkZZ2NmGdV6os
6YsQIaafU3vXAj6bUD592KOxDjwDKgvsaOnSo/NgwwbnQ5zfeeCMy9/kyCL4ESJc+Wu0D9mOFzCtsS
8/z5nzQ9AvPt88+7Ioi0jbryMEpk87wObZ5wNqbcIK4UaLQZq5LYTRsjLNdHkpcrhXE1ZOzGxCEpGtv
fbakosLs4qFFZ+kVSEVC2lazEArSQrJp+Wn68BUQs4tcmKde+65cdqeI2mY2YikESHJIOp6Hlj9TZ8w
A9EfCueSgRcwhZmBuD4lCznEQmDKw8RlM+7a0rVrV1kxSw4uTGOo76RgQZX3i/khS5oHzgu9TzHCR1K
ykJqFZ0j2ZB/uiZXsJNnkuqXAdMc9kyqGPUe4j0rgHskHhimNLMHs60J7Ju6ICru5ba
UVzM4u7JMTqucWfiuYsFwzXdp5lfS9WtSElRNm8TiHWaHOLntGiCTvRGJCeffdd2UT/VVWUVm3926d
ZMwPvY8cWEWhIZBYWaKGl4OqOjMZtEa0HTywMwZUw/ayk477SRl6623Fu4Mitbfflt0DzaGoG8sxUBg+AC
ZCG4r1133VXuadq0adETTTzwhJqHx48fLzpHEm9Me5iXUd7Qbv3D/7PnBKvTQ+xRUdExOmPTQCo466qi
kB4vgnni+vXv3jsaOOSyjTQDjC9cZ9GcFekxnOffOaHHHKIPD/WFdAWn/2pp56a1FIWF2yghoXvCF
pKJd8N38KQhht5BnyH80A9NzILq0KISvpeCIkwUXJCLqeNN95YnLgkTBkTBw6dKjMLnAIA7Nj9sFgPxGct
jgELa+99po4mSk2BxYzYzMaqmXAvMoBcfdthhsreKoxtHy2JfdDQkq6n4hfe4/1I7PpKk
kfvm2ubL30C7wPGJxkLadrQM9z20E9vnffbZR/qZhRFW8S9+8QtxkIcyHHHHPfaGxkQyQYYQolz4LPl8zeS
gPoM0ec/YL4bPkizGaCSHHnqoZIJWGmM/0+Ys/h4bfHZ2F09K9+7dg+flLXyv3fZC5Wc/+Zc/+1ly9Tqqce
T71yOYIFTPv++iQQNRAVIBpJ0nygf1EjPLGWec0WAAwTRFlBMbtMtt11yDLMXgzamGTdpJGapOXPmy
```

```
OBMhA+p0zGX8D8p8N1C5lsGq7XWWiu+/PLLG73PIIeZibbYf4RNnkhSaU1u/KV9BmgiwBAglQow9v8I
QeQU29LeeOONyZHywLTH88TslofXX39doqlQ8xEoLnYjL0xNPOdyIeU/eyoQFQb0iVT7CBCbDBKhS2Q
XkV7lbOLVVvC/N81RsvbYqGTvGrdUYj6qRoD410vrf9Z9V4sKkCpgps2g9aMf/Uhm5T5k58U3woybsF
d8KAgddjLELp8GvgnsqiHy+kDwaxi1PHm1iG+//baRALEgAEktT/ulYMbdv3//BpqVC9oXe7KwJwIzd
77EeQszQUJwzzzzzNwCBKiLDwqtBsHNs9l1111FwNvBvxzYSoC+4xdCIFpCAgTs9ZkZ8r6yiNDA1tTF
9SGE4PcbOi9vyWo/RDn7evgFf54LY0aoXq3DzdUHUgWTJ08W2zvRWexzbp5n8k4dhMQOGDBAokIIS2W
/dGz2ZoYdXD1LuB7t7bHHHrInu426cMEvQPgfdfET+LBS2gygksLgoosuqouSKMFSTpQTEUP0kzDkgw
8+OCIs14UwRNo1A6f4T9Zee+3kndJgw81TqoFIKO77pptuklXihA2vvPLKEjVH1uNyYE927pNILXxVb
roJrsNz4vOjfQvHCdclMg6fSJ8+fcTnoiwe+N/1cpk6dWryXxiyFpCDyi12wW0l8Ft022IJQQgWF7r1
KIVTJ0uUvDCLxzTC5k/M1tmznVkzs3p/1owWgNmKx0xJ24oX0C6Y1f70pz+V2bi/GA+YtbCjIjZPzGg
+zAiYceMT8H0cYDUQNvbClg9PPPGEmMOszwHfDDsacg36S/TWcccdJ/03X8BGbfqggWDCYlOtSsCEVa
4Ggk2bc9AA8Keg6dFfZmqYAkvBc8f0heZgBIFoLY888oi8x7a+rg+I3S+JwrOmSXucnS75vPAjsdGRv
T4zQN4rpXG2dux33y2VmHzyQERk6HpZGkI55qRK+o5Juqi20u4xbykSNWGVCb4NI+0lvI4VzBa2qWW7
XjaisjCQI2T40DDpsOKa//GZsJuhDwMZ9nTMQzjGPvjgg+SdReDDsCYs67T3QSCwC6FvogIGMgQCQor
95zHRcC+hwZ7wWPulw2TDlr15QICw+6SZpctgXE5hh0lMX3kECM8HYWNXzuNbYpX+/Pnz5X0GdjcrAF
kCeDYh3n77bREM7Nhm/Sh8fvbcvIVdLAGTAuZLPkvCLlWANCz4i2rBK6+8ErxelgAhGCZ0XqhU0vc0A
VVJW2n3mLcUCQJEFxJmgHpLaCqmHTOoiumGhYNGS0hq1IH5iBz8JPSjHrBBFovyzMxYXmOCwsRCKDCg
xlKHRU2EEZqZtOS6YtW0GYBk1bfL559/Ht1///2RmeVGZpCNttpqq+SdOghHnDRpkoTlYoY644wzGoT
3kVaaxXQXXHCBmI2MNiOLB4GFjYT7ssjPrprF/IZ5hvuibTMgRoceeqj0t1T4IgvsWIFOH8vl/PPPl+
eFCc6CuY4+GKEgecD4PIBQXkxG3Ke/ItiF58FCTzcz6YEHHhj169dPwpv9z7IUY8eOja644gpZja5kE
zJPmgG93kSLSdZfpZ0Hhi3CwjEl2jxRaQsE3euBf03+P/LII5NXpSHHmv3+5SVvv/IwceLEaMcdd0xe
lU+Rwz0mchUgGSBAGNCvu+46SXzIwGZmy8m7dau5+YIwGE+YMEGOsXaBev6KZwuCgtXqrJxlACelB3Z
1vkysACehYbWwLoOUKgzkFgZikgDST9Z7YCNFYM2aNSupEUkqE5IRmtm7rHcAviicR59feOEFOQasd+
GHt9FGG8n16Lfrd0F4hfw9afDDtuCfwY+ET+jEE0+UxIoIN65Hahn8FLZ/eWHAwXfF6nWjNYrQJhsAg
tL1B6XB+hY3mSKpYZTSZAmQaldW59lbxB+o814T/5e/nwbfOXdvkTwUKUDS9jzJS5F+EPmtI0CU6sDX
Yb4kEkLb3ODDwBxz9913ByOxXKhrNKTYDKjJkXxgGiOkmHOx+7cVMI9h2sIXlNek19Yx40yjYgbO5N1
8K7JDBZMnkXLuCm8zGAfruteDvNfEFFwEaRFX+MgWZ9SEpShKTcnSQNJm51mQlQFTJ9pvVlv+TD/vNc
ks4WsbaCWXXHJJ8qouwsrXUnzQGsiG4JOnfZ+062WdVwtUA1GUFgAr3lmHQxAFDv3WhBlnGhVXI0jTG
vIUAjXytOVrINVckzxoLmkRVpUWv32ftOtlnVcL0EB0HYiieDCzssWuESB/kj3273//W44BPrDQ8XJg
ls4OigQD4PNx/UxFgX+JTM5777137s2HWjLkhyKfXFPj70BYdKbl0A6HLmnXyzqvVqgJqxXD4EfkF4k
PQ7skMnBRWByHKQBHMlFcwIDpfzWoi1Oc5JIkZCQpYl7Ye4PAAtKq48AmMWE5EAX2/vvvi+Pbj07jPj
ETEJSAQ5WFlm7f6TcmD/YJwWFO9JYLgz8mBvpEgIRNWEc7tMezI3273XESZzwBA1yTRZ84+4E+sr8Kw
RYM1jzTcsBJz2JErkviTnexYrXQ57vvvlsWtPJ9KCfyrBp49j5FmLCIusOhTTRgVlu+CauaSCY+U7ub
IpQTwZUHv32ftOtlnWfhWRQFvxsVIK0YBg0EAuGqoZXrLoT9IkjuvPPO5EhpCGUlHDgvfM0QHhRCcRn
I84KAwPZLVBkhuQyCZMdlNoZgYMBlUyi2+EXIlIIQbO7VBc0B4cM9MYCzMZiFcN3zzjtPVrpjZ7bYqD
T+unuuEKVFHwnfXnfddSWCj/5awUx/mT0THcdnwmcEVjgTDs4PkzZ5TX2EGq+pz3vAZ0U02mWXXdbg+
qV44IEH5F4effTRRkK4VtRKgDCRQMDymZQrQCqNZOK7leXvSIN+uudW01Y1FDncqwBRZMAibHefffaR
QioPBlT+hhgzZowM3ggBQn7hyy+/lDQirF0pCtakhIQMg99ZZ50lAyCbW/GXrzDpJAhJZn1HKLUD60g
IkWbTL0KUXRjEWVtDuDCaFwM/MHgzOGMeQLMg/JP32UyLEGY0ElL5o3kA7ZA+hfDt4cOHS8p9BB9pTu
zOj/zoSIdP6DTPD0FgYbC163Zok/vieqTZZ3Ds2bNnvVZDPdLo9+rVS3aXZFZdNAjDalJuQK0ECEKYz
wnhWq4AqRTMjO4kohwIO3fXIVXTVjUULUBoUGnFEHLLqnVWafuwsp50JWymY77ckk32oosukhXcofTo
rNJmlfg222wj6UMsrKInFQrpPNIKKVTIskuaDzbcCtWxxajp4qDLixm4JUmlGexjo2EkRxeBg5r7J1V
JKGSZcGdCLbknnlcahC0bwZGaSBII9aUdVtLTrxBm9it7ZoeyDYQgUwBJO0ng6WMEjCTfDD1HyvTp08
U5b2a7jBxSSMVC5mZWYJM+J3SevYdqsdd0ixnQk3frEleG6uQpbPDGc7EYIRGs516vGkgFFGo/T3H7C
Wl9rXUpEg3jbQOgTTC7xvzDzHjIkCFiowf8I8zimR3zl1kdi+wwTVGPTZMszIQxE7FSnJnw5Zdf3mCV
ex5IOsmKcFb15108xqpfbPYkJ2Q2i0bACnz8FPgn0IgwFzGjZ8bsmp8s9J3Fm2gutGXBF4K5yW4vTFg
o5jDq+3DMCBkpaAHuYlIL/cO0YjUJNBoSVLrwc6PfZCugzvXXX99gsacL18Shzor+vfbaS86h/XJhK1
```

```
5Mh/hr8IOgsfBc2Z7X1YBqQZYGgk3/iCOOkP/LAbMmnxW+HGv7z6uBVLt4MQtMj/ho7Ap54H/XR5HWV
76D7nk+BENUEwRR5HCvGkgbgpTV5LQiDxaaBzADNQOc5NdCi2DGjLaCBuLm+gLzZZF9TMg5ZbeZNT9i
+ZsH6l533XWyvwZ7leQFzYU07eb7Wl/Qgpjps7EVKejZXtbdj6UU1CNUlpk5ienKgeeEllNqQ6088Cz
Z4IqFcDZxYwjukXtlSwBS1FcCGpcZqGTPF3K2MRPm+ZOU84ADDqj/LtQK93OzpSiNwCevBsKzDNUrop
DMk3xuWYkS0/qadV61e5cUiWogbQxm8cz2sL8yW2dWzhaz+DOYpQEORtK2kPOJaClm23xF2MqWMFNmS
DjaiYDB5o9vwfpCioYZGtqT3dKTGTlpYugTGpXVgEhPgrOaSC/CakNfaWaFaA30G3t0KB090Vbk1sIx
zvNwNSxm/mhua6yxhmy1i+/IBQc3z5Ntf3luWeGdzN7QnEaMGCHanh8VxL2QU4zPieeNBlIO+FDsZ4t
/Cl8J1yJ1DlomviOcuDw7Ir7w69SCkAbC7L8WPhu0Kqtdu/gaSKV+lzzYFDw8T34baaT1YYcddih5Hl
YDGw1YCUUO93yHVYC0Mdjj4uijj5Z9PfixkRwRc5Q1ozAAE5F02mmnST3MWZgKGMwomKGsyYGvDiYyB
BMDFj8cVHe+5AgXFwbE++67T6KgyBvG3uIu1uxD8kbaZICmPRzYDP70gT5jxuGHhwDByU3ySqLMcOTn
hYHTJrx0YQDCpMT1+VsOmBbY64Xnd/XVV+cKwcUcRoTWs88+K8+F0GjgXglIIJDhqquuknstB6O5yPm
EJvN5sT8Jz5NEnAgQzH7k8cLsx6SAwZXr+8k5iyAkQEIrsIsgbcV3UwoQvpNMNjANuveIw9xdKc5vLB
QkkvVs0u4xL0ULEBpU2hjkwCIPD+aokDkJpzJO5y5dusSDBw+OzWAamwExHjhwYFIjDA5RHMRGU0iON
ARnMCYs9swIgXmJlOtpzmLMcJjgzA+xkbP7iiuuiI3QS16FOeSQQ2RLYIIBQuBIx9HPfZMHib3d8xae
J85pUvd//vnnSYv5wMHN82WbYJ6fEa6ybfGECROSGvnBJIaZiv6494k5DNMjfSSlvwup543gF1OgGaC
So8VgxplmL74JyzcfmYmLBJJkmY9C5N2fI89K8bxtVdLPWoAJS1eit0GYwbC+gdnMlVdeKX9d0CJYrE
SqeWaohMfiDMaBHoIZM469c845R3YAdGd7FruGwXzvZIbsw6wfBzfZQjHt8DoNq624EL6LQ5bQVj+bM
WngcbrPnDlTtINSJgD6Zx20mMzyFupzXiWgDWHGYk0Hs0/uD8f+LrvsktTIhmeLNoM2hnaHici9T8yA
zG7R5nwzD1oJz5w+sJvj448/nrzTOpg7d27yXx2+doz2jPZJCHa5hBbohsgT/JC3rUr6WSvUhNWGwMz
DAIrNG8HBYIuQAOz2ZuYq/wORQkRmETUFDDKs/whF7TBwsh6CRWpmdltvInFBgHBd7PIMVKTUcOFHzE
DPYEo6D9fHAdaEhRkAHwYRTvhpuB4mMQZH+sH5XJ81HwiNkSNHRuPGjRPfBia4kEnFwiBCtBJmCMx65
YIAZTV5XhMW98xgTXQO98LAztoGTCz4UEi5z+JIf+W8hZ8ugwl+Dp4Lz4AIN/wpRJchSO3PG9Ma608e
e+wx8SPZ/Vx4HpgNEToIFyK0MK/wXeBZ9O3bt6LIL0s5C0ZrBf4Iouws+PN4thYmTHxeZCpw6+XBbys
Nvw8himyrKVATVhuCtQJE3Sy77LLx0KFDk6N1UU5EZ5mBQ16b2bSs0+jQoUNsBhcxq1xzzTXymnUUmL
R8zOAu6c2NFiDrJNKim9hGF1MBpq4QmKh69eoVNFFxjRkzZsR9+vSJN910UzHTsAukERBJjUWYmXi9u
o85Km8EE+tgMOPwjDAhYXbIWzp37iz9MYItM8U9pikj4GIjjKWPPH/SxNt1IzxLoq/sPbDF8bXXXism
Nh92P8TEyD3b6xrto1HkWlah38Bah+HDh8trnjfPXVFCaBRWK4dZO45aoniYfaMVsCqa9SAuaCbjx4+
X1d/sWAiYrTCpuDN2tBAc0HxldtttN9ldEW2BGRyzXWbgbFbF6mU/yonZCs5iIkzIOcWaBBfMQGgxzH
7RHnwNBA2GVeRcA9MMazrOPvtsMb9h+sJkg6ZBigjWitAGs3hm+ERHsV6AFeiYaLg2GxGhabhgNkMzQ
zPgeZULGhrOdDQt+9zQEB5++GGZ+bMeg10lgb5grqOfpbBrVEhDgoYEONuJ9uIz6N27t2gOeeAZobmh
yeU9R1HSUA2klcPs0Qyo4vxmXYed4Vp4zR7LRgjUz0TZZzw007XgmGU9hxnA4rFjx8qMlesY9Tvu27d
vfTvVFNZ2mC9ncsU6mO2YATg2AkRWSbPinCAA9zz2JMcJT10XZuRoWG7dNddcUzQd2txxxx1FgzCCSb
Qu3keb4Fje4vaFc1m7QX+mTJkiGhyaGc5q1tn4/csL60/uuOMOWaeDk5z78j/TNNBO0ADRruiDolSLa
iCKaBP4Owi9ZX/w5gYfCTZ/ZuiltplFW0C7YF0HvpxyYCUyocc4rNGe2gJoVWgz5BILaYiKUi5oICpA
FEVRlLJBgGgYr6IoilIRKkAURVGUilABoiiKolSEChClTYIzmVXzhNyGXluyjtvit5NW8l7PL2nnpfX
DJ+16Pmnt++dltZfVTtHX8Uvedv2Sdf3mOi+tnl/Szktrp2pwotcCmm4Jxd8UJ1SHUsnmOaF2KG5bof
eboygNYWEez+Wwww4LvrZkHbfFbyet5L2eX9LOS+uHT9r1fNLa98/Lai+rnaKv45e87fol6/rNdV5aP
b+knZfWTjUQxtvqNRB/wZRNDW7p0aOHLCirZGGV3xaL01hA5rbl11FaBnZDKDL+gv/aknXc4reTRt7r
+aSdl9YPn7Tr+aS175+X1V5WO0Vfxydvuz5Z12+u89Lq+aSdl9ZOtbQ5ExY5gVxIYEYOJSNQkyORrNZ
mHwa/+EnYfFjR7bajKIrSmmlzAsQXAgz6Pscee2x0wgknCqlMsRCKMusoihKa6XJBQgz9FoU8h/
lIX
Quxc0ym2Z28lOI+1gTVh5CfSiq5H0WiqIo1dDkAsTfCrQoKt2LIURaH7PSXmSZuFxq9RygyGehKIqSR
rOasJitV1PYe6EWkGWWfEFuYU8E//q+ECAzLBpAJfhtl1Nq9RwURVFK0awCpNIIJfZurjRyKg/sI+07
0O2e4aVYY401ZECvhGqitTQ1t6IozcFi6URn74iWiJqOFEVpSyyWAaoNcWo5WG+//faS2twteXjuyrRa+//faS2twteXwWbPh
TqQlLURRlcWOxFCA9e/ZstJtckbBfBHsnuMwCrLFj3bq1KlTxSYsRVGUGUxY3FUoCwCwWstcUN6S4EQcf
FfK4qitGYWSwHCDnpFDdahVef+anWL71z3682bN0+FiKlObN2qy95hENb2V9bURQFmlWAENkapSAENm04oor
RiuvvvHJ9gaKG2JxVKAdzYMfmvetLCZ7fZZptos02bN0+FiKlobYbFUoDkiYjKS2jVOYLhhRdeiKZOnVpf0
nToEJ122mmN6rloNJSiKG2JxVKAdOzYMfmvetLCZ7fZZptos802qy95hENb2V9bURQFmlWAENm04oor
RiuvvHJ9gaxjOK8xFdUyEmvzzTdvFIVFJstS/VpllVXkWCX98p+FxT8WejYaPqwoSnPQrAKEFd/9+/e
```

```
P+vXrV19Y9Z11bM0115RjtZzxEyrsR2HtueeeJfuFoKm0X/6zoB2/ff+1Paaaj6IozUGTCxDXZIRTeu
TIkQ0KzumsY2PGjJFjbnLDpgifHTVqVIN++P0aPXp0o36l4ZvO/GdBO377/mt7zL+ehhIritIUNLkAC
aUJKaKkRU75+FFX//rXv5J3svEXE/pt3XbbbfI3T1LFWj0HSt5noSiKUg1NLkD8SKaiSt5B04+6sov/
Jk6cGL300ktSiLjKg9/WgAED5G/WviFQq+dAUQGiKEpT0Kw+kObANx3xGgGyvZO+JO8qd78tK4xquUp
eURSlpdDqR7qsCCVWteMgd81TvqnK4rfl15s1a1aj6LC0thRFURZ3aiZAbBRRc5esCKWZM2c2Wj0e2g
8k1JZf76ijjmpUL62tpi6KoiiFYwbPNsVyyy2HtKgv3bt3j9u1axfPnz8/qaG0BYAAYMGSKf/4ABA4KvL
VnHbfHbSSt5r+eXtPPS+uGTdj2ftPb987Lay2qn6Ov4JW+7fsm6fnOdl1bPL2nnpbVTDd98803c5oz1
voMZhzcmJ/M8kiOKoihKHmomQKxDOqsYKZacUYfrzE4rm2yyifx1z007z2/fD7F9//33k/33k/8WkacPtvj
t++TtV5GErqkoilI0NRMg7iruUsXPMRXai8Mv7EjIX/fctPP89qR8cjCUkcpbv/IhS1ZhLCTUR
SlaJrdhOWvovZDY0Mw2BMq656bdl6eVeE+efpgyWo/LXNwJf3KSzn9VxRFqZRmFyCsm3BLWtgrAsOWK
VOmiN/CP8+tHQWG3ji1FFFWplMVmZJkwYYYKYZihps/qnn346mTpNYYfwSxAwVmzbnLDpgifHWS96cq19ZZq1yNAv/WSi
KohTNYiNAevfuHW299dVWZS0hYH7rjGGwWW25ZX9IG5DypStzrUfL4LEL9wuHv9qlUv7JAOOXFfxaKoi
hFs9gIkyf/NQ7FtYT28Ajh1kur414vL59++qyqk45F1CEVFp13Sp9DyL/ywURVGKpmYCycI9ySCLp2h
goRqutGSYXep63QHh5Z9TbaaCPxEfj13OvlxebCcsH85faJQvtZpJ2Xp4SehaIoStE0qwaSZoqqhLQU
6r4m4V+TXFiQNbu39UrB4O1T6f7t1URShfqhKIpSNDUTIKF9ihY6aB8+0I6bO
PPsslyPKkaJ83b16j6+U5rxzy7COi6dwVRWkKaiZAQvtUUPIMqFl7ZZCg8MQTT2zQFmYau5+HW7Jm8k
Rk5XFOh9r3nfEh53ie88ohzz4iCBB3fxOKoihK0TS5CauINQk9e/ZstL7BdxrbknnW9lVZaSQSIG9obI
tS+LzCWXXbZRsIodF5an7L6UA6+E15RFKVomlyAuOaicqOzFaVtK/UwKAYCxynbSFmaUCQw/HpfcfLCgJ
1I+hKEpLp2YCJLQnBSVrr4r4dsn76bzxhtvNBBnZbCVfEMyZMyfXwJwJ92327bdq/
2o48+ulFdt6Q9L7ctiqIoStHUTICEnLsUSsH+mDT5J4hY7ETwhc0dSJwGF34YpESSxnpwnLdxmoOAt
zXXXc1quuWavwpiqIo1dDkJqwiIYILILNq/AG2gHuM/9FA3Nk+juPQrieLyByB8k
1tIKHDfLva1q+Wknef2yfYrq6+KoijV0uIECIMCMt6dr5awu4x1588UUfL0qUugOvLPuHrIEI4LxWO4NEa
Un7k2bHHHtvgfYofehsKxU07z+2T7Vc1z0JRFCUPLU6AVBPLbGN+uvR6OuOpCCQ3SHHhQNzGGHhc7NK2kp6n7z
mNhfbtqs9pJ0X6ptfFEVRiqbFCZBa4zu+mZ1XOsDmjbrKW8/FnlNJDaiKEpTUDMB4kcBNVfxo7D8CK
g333xTBEjo3KyCMMqKkqJUIgSSoMu6Hr4Ot44tWdFniqIo1dLqBMh8WbiLxSvcWFzwHoqnOk47kHWFhTa
qLW1HGuKEqtaXMmLJ9q/QOVpH1HQ6g1Gt6rKEqtafMCpNZUu6+Hj58aJa0t3Q9EUZRa0+QCxEYFFV3y
RkRVQ+i6WSawavb18Etonw+Oh9D9QBRFqTVNLkCqNqmmX2XopJIyjHnFSJuSot9LaS56DrORRRFaUk0qwk
rjwM6VEL7geTFv2ZowV4aabP9pqLovUUURVGqoVkFfSGhldZ5y6qmnNtoPJC/aSdMm3zSrueT1r5s5/XlZ7ea9nq
mzADAvWWnmQ0ybNq1Rnyjuvh9pRakMm9TS1z7tcYt93z/u49fLateSdl5aP3zSrueT1r5/XlZ7ea9nq
fY6PrZ+Vrs+WddvrvPS6vn49WpNswqQSgfkrbbaSgbgSgbhvNfcZpttos0226xBqeR6aSaySqKkSCLp
94niOsvTilIZP/zhD6Ntt9022nTTTZMjddjjttj3/eN+8etltWtL2nlp/fBJu55PWvv+eVnt5b2epdr
r+MXWz2rXL1nXb67z0ur5xa9Xc+IaQdOhYiRkUiOOzWAerJNVllxySfmbpy23DuS9phmY4y222KJB+f
rrr5NW6thuu+0a1fGLESAVt++Tdr2s85TGDBkyRD6HAQMGBF8rxaLPu2lpiuf9zTffxM2qgVRKpZpLO
fhRTBTfUR+KsPKLec5J7Ybkad8n7XqVBhAoiqJUw2IpQH784x9x9XZE6ChQsXJv+Vjx+yW+0qdp+s9tLe
L7ofiqIoeVgsBcjbb79dcURUNZFMfhqUciK48hDaaMotadfLOo+iKEpSOGLMqgE0HSpF+EDWXXdd8S1
U4gN55ZVX4X4pdeeqns0qFDh2D7ecrEiRMLa6vvFojjREbfJNiz7vpqUpnbSRbfXXVUqObsvBTfOQt1W
yN66czqdT81pb56KOPxN8zY8aMJg9kVLIcPxr0hLIcPPvhAwkPwq/AwqYJr/YzIrPJ18cffxxFlsjMb9fPnll0mNM
J988kk0adIKuf9PP/203ueFRvr+++9LW375+9//nqYJ/+7+9//nqpt0wd8cG49v29+8a/d0rDP/eWXX5bvShru51Pq
ub/33nvbeLF5U/p7bWvSi36LqEdIL9l9/H/hPnjxjjqoAS1eS0HacXR4L6tytHKaanF0kDJPVTkj33tu4TgvVjnz
UUqm2RHH7CdW0VWlZ3Pjuu+/iZ599Nu7+Xr19Ry4bpi/vSQzKDbazzOsGWDaxpHDHzfDGyYuVs4bWnPgDH
LJJfHSSy/doK9EC/7617+OjQCQerNmzYPnPeNWUBnUoaNPnnHNObAY3qefy1FNPxO07d5Zz23/yI+3/fHzTT
TfFCxYYskPdmzpwZH3HmdzvNPB8nmx4ssrXfvLqlpx52vrP+bfVtvvrHm9yt/XnSr77+uT/c+ccjRvOVXb
8cclOtHtJ2W33XaLJ0YC9cuFdjsGs3G3bt0a1Pk11lmPPfYYTVBwB23fRk0b+aeSMPNMnm9fPPPk9KuKGpb4LzbCuKy4m
1qzd9+vR47733bl+DHHlt69e7fVvuu1CuHWjxvHzHHRLk4aRoCs0czgMr/LAi0e3+Xg+l4tOyyy4
rt36ZKx5EcmgG4
14O0fhUBqdWZzbFnB/t0WNx+Qlpf/fN8fv/738tMsRJq9DHXFDOoyWyxY8eO0v+LL
75YZsBXXnmlrNPMpKdNCn119/PfrnP/8Z9enTR47a7NnFRHrPVuu6/MOI1w
```

```
iA4++ODkaBQ98sgjkRng5PvbpUsX0RZ4ffjhh0u6fjQGIxREmxgzZkxkhENyZkPeeOMN6cO4cePkGRp
hlbyTDf0688wzo+7du0eXXXaZrD9oKXDfZ599tmgVjz76aNS1a1c5zv/coxmUo3PPPTd64oknIiPIo3
XXXTcaOnRotP7668tnRp0777xTPqczzjgj+stf/hJdcMEF0XrrrScZK8xkRdrjmQ0bNiw68sgj5VmMH
z++sHrnn3++fM+NcBZNZPTo0VJncUAWMyJAaoFpP1iQWrUgbVbvayC1nP0zU2R2kUVaH7JgBho6L09Z
3GEm3qtXL5lBvfjii8nRlosRIPGhhx4x4ad+rUKR4+fHhsBL8c//bbb+VezGQh3nHHHeOtttpK6vlaFVr
BT37yk3ivvfaKn3nmmdgMRHGPHj1iM4mQNsAIrPjkk0+Wz9cM8KKtofFMnTpV3gc0uT/96U/xpptuGu
+3337xY489JrNRrjl48GDpZxr4C3nmnPvXv/4ODdpyQLO//fbbRSs1E4r4Oksv
muied1xxx3xOuusE5sBOzmzDiNY5N6MAJfnYoS8rM/if5g2bZo8f/sb2nnnneM//vGPhdabPHlybCYf
8YEHHihjB/3ZZ599RNt755135LyWSrP4QGq1T0UtNYu8mGea/FeaFm3TbIGgdVWqeTUH//jHPyRNzsS
JE6NBgwaJxmB9aISRz5s3T+z0M2fOfOB/GAw88EP3qV7+SGSsa+YQJE0TT6NmzZzR27NjIDOKitaLBmE
lK/WZhaLXs6b/lllvKa+z/zGTx8+2www2www6Sjgc7/1tvSXAyaYoMPPhjtvvvvu0a233ho9/vjjMus2A2t0w
w03NEqNgbYzcOBAscFccUV0l5Lg+zUP//5z6WfPE8jGEVLQuMywjnaYIMNxFLBPR9wwAHRyJEjZZyw
xUxGxCeBNsL9Y63AujFixIoW7dusqob7dcIg8gM8PId/OKLLwqtx+eNhrnFFluIfIfxQLBr4xNBM0pf3
3318+ixZLnSwpHpoOldAK7CIK0jt0vabUQNq3by9//XtkFuTin2dL1rNJu8c8ZXFn0qRJ8S677BLvsc
ceflBJUdbHmPGjBFNoW/fvvGrr76aHC2NERwyPnz7ymW2yySaicW299dbxWmutFZvBUmutFZvBU
Oz4ZuCX2XUazL7xE15//fWiYdx2220ySz/99NPFFLm9B60EDOv744xv4X4yAE01mlVVWER+B78NrKfAM
jjjiiNgMvPHDDz8sx+gr/qi11147PVHEE0VLszBjRrMzgjaeO3dufN9998nv6uyzz5bjRhj4fZXz
xxfX3jSaGhsNnOm7cuELr4f8IMWfOnLh///7xeuutF999TrF9s+VwHGG8ft2rVL3i
0PHJBuW2mFH2QWedtqS/DcU09xpt94442xmVnHZrYtJom//e1vMjSc7QGSwzVFq7W+LUf44Gj6eUR
++Y2/iL85l6RhuIn3vuufj5558X09Sdd94pDt9VV11VHKscQ0AY7UO+m127dhXhhstJKK8VG84hXW221
2Gg0Mgjy3eY6tEWo+EMPPSSDFoIG05M19fGsuMbqq68uAxn05WHDhonw+8Uj87HnJ/p+HKHL3KVLl/i
YY46pd/S2NPjcH330URWhPUjJDEHYWrj0dJ/M3uPjQYlJj++KwzWfBZ8Vscdd5wEPRjtpH4Q57lsv/
328lwYtJm0/O53v4vXWGMNEQKjR4+W6gr0gc+/R/p/p/7733St+ZkGLOevPNN6V/LY02IUD8wdt/n
x9mqF4e/LbSSp62Q+eFSiX9XFxhgCi1Zgab9ueff57Ubj6mTJki2mOojxQb6QAxn0fkUBhg7rnnnnnjL
L79MWmwMWgJ177rrruRI+Pl07NgxHjRokAyYcFgYphMhBBx3UoUC6DL9dlIAD+4lNA08WfhzBrydBfhAF
amntfyyyzjPhCrPaBL4FoJrcOmhyTEldAEq3FoI/vya2L5ku01/fff194PXxUI0aMaPA+Zf311xcf1a
effipttUR4/jWLwtp5552T/5oX7Mqu32XPPPfdsYO8luoUoCGzN5fpn/LbS8PsQosi2FEVRag3jVc0Ei
KIoitJ6QYDokmhFURSlIlSAKIqiKBWhAkRRFEWpCBUgiqIoSkWoAFEURVEqQgWIoiiKUhEswU7+VRRF
UZR8IDra/ec/le8RriiKorRNUD7aLVaGaIAPnuu+8k5TBphRVFURSlFMgKZMbXX8+
P/h/2RoQk7TZ4uwAAAABJRU5ErkJggg==",
  }

        const res = await
this.nMPrintSocket.DrawLableImage(DrawLableImageParam);
  if (res.resultAck.result != 0) {
    return;
  }
  //进行下一步操作,继续绘制或提交
}
```

3.8 标签预览 generateImagePreviewImage

代码块

```
export default class NMPrintSocket {
  /**
```

```
     * 生成图像预览图像。
     *
     * @param {number} displayScale - 图像显示比例，表示 1mm 的点数，可调整预览图大小。
     *                              例如，200dpi 的打印机可设置为 8，300dpi 的打印机
可设置为 11.81。
     *
     * @return {Promise} 返回一个 Promise，解析为生成图像预览图像的结果
     *
     * @description
     * 增加方法说明：
     * 1. 在调用此函数之前，必须确保图像数据已准备好，否则无法生成预览。
     */
  generateImagePreviewImage(displayScale)
}
```

代码块

```
//返回数据示例
{
        "apiName": "generateImagePreviewImage",
        "resultAck": {
                "errorCode": 0,
                "info": "{\n\t\"ImageData\" :
```
\"iVBORw0KGgoAAAANSUhEUgAAAZAAAADwCAIAAAChXqV1AAAgAElEQVR4AezBeaznd33f++fr8/2dbWbOeMYzY+MZb+DxgAk2lN5QGiVRC71RK4SaRKJqdRM1EamKyI3C4pKNViit
qlQJ/aMNNA2JaEoANxhsg5fx2B4vs585y+/7eV7zTY80R/POVUa3FhrpPB5R2bVr167rQVR27dq163o
QlV27du26HkRl165du64HUdm1a9eu60FUdu3atet6EJVdu3btuh5EZdeuXbuuB1HZtWvXrutBVHbt2r
XrehCVXbt27boeRGXXrl27rgdR2bVr167rQVR27dq163oQlV27du26HkRl165du64HUbkWSaioVJJQU
akkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQq
SaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSUJFpZKEikolCRW
VShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZ
KEikolCRWVShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQU
akkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQq
UbkWSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRW
VShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSU
JFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQU
akkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUk
VFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRW
VShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSU
JFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqUbkWSaioV
JJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUk
VFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSUJFpZKEiko
lCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSSU
JFpZKEikolCRWVShIqKpUkVFQqUbkWSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKpUkVFQqSahcv
rwxn2/O5/OtrS1Aba0BR44cofLCC88XLCC88lCzCXEVsyQDftxgMHqZw9e5ZJEjUT9YYbbqBy9uzZJCqQBEgC
7N+/n8rZs2eBJCqQhMkNN9xA5dy5c5c733JGomTg4ePEjl7JnTI7MkdIHIONNx8ciNq1See+45rqAC6k0

```
33URFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUakkoaJSicq1SEJFpZKEikolCRWVShIqKpUkVFQqSa
ioVJJQUakkoaKy0wgb5y/svWE/lbNnz6pMMmFyww03ULl8+TLQJkDvfRgGYDabUVHZpgJJ1NYaFRVQg
SRM1NYaFZVKEioqoAJJABVorVGZz+dJeu/qMAxA711dXFykcvHiRcBtgAocOHCAygsvvKDyoiwwdtPN
XL3p8M1UVCpJqKhUklBRqSSholJJQkWlkoSKSiUJFZVKVK5FEioqlSRUVCpJqKhUklBRqSSholJJQkW
lkoSKSiUJFXXtt3/n3GvvWZ4Nw8rSrA2ttWEYFhcXqaiACiTpvbfWHnvssTvvvJOKyjY1CZMkVObZs0d
B7d9J7d7Jv3z4qzz33nBPAbb33Y8eOUXn44YeB3ruTPlHvvfdeKp/73Od67/P5XJ3P530yn8//5t/8m
1Tuv//+PmGnd7/73VTuv//+JOrevXs3NjbOnj27urp65MiR7//+76cyn8+dJFHn83lrTV1ZWaFy8uRJ
4NixY+yUhIpKJQkVlUoSKiqVJFRUKkmoqFSSUFGpROVaJKGiUklCRaWShIpKJQkVlUoSKiqVJFRUKkm
oqFSSULl48WISFWitDcOQycLCApXe++bmprq0tJTkYx/7RJLz58//w3/4f1E5fPjw8vLyysrKMAyttd
47kwcffJDKK17xN49h7P336NJXl5WV3SgLM53MqS0tLTDJProQGttQsXLlC56aabgDYBWmtJg
CeeeILKPffcA7TWkrTWkrTWkvyP//E8h/2/8+mqmn5p36OHLRAQkgAo8+uijVG655
Rag9+5OL7zwApXV1VU1CZAESAKcO3eOyk033Ozk033dRaA1prucLjjz9O5fjx40CSzYmaBHjqqaeozGazG
g+9973s/85nPfOELX3jta1+7Z8+eo0ePvvvGNb6SiUkl0CkqlBRqUTlWiSholJJQkWl
lkoSKSiUJFZVKEioqlSRUVCpJqKhUklBZW1tLwrZhGJIACwsLVO66e09ZcuHDh4sWLwOLi4s
LCwqlTp6jccst6sbGxnw+T6T6ImAc6fP0/l8OHDSVprSVprSYAkJ0+epHLHHXfcfS/7ppqaeozGazD37w
g+9973s/85nPfOELX3jta1+7Z8+eo0ePvvvGNb6SiUklCRaWShIpKJQkVlUoSKiqVJFRUKkmoqFSici2
SUFGpJKGiUklCRaWShIpKJQkVlUoSKiqVJFRUKkmorK2tqcMwJPnYxz721z721ULnvvvvuA3v
ulS5fW19dXV1cXFxfPnj178uRJKi972cvW19fHcWytAa01JJmfOnKFy6NAhJioTJ2fPnqWyd+/eq9O
iTA2toalUOHDjFprWUCJHn66aep3HbbbbUycAE6eeuopKocPH3bCNhU4+YMlcOHDzNRmahnJnn/+eSor
KytqEiAJkMnFixepNbbRhixTUKTpUk9c74vs/+sjVg4KUz3+c8SLjk9BPe6GASHOkP6BmyRlrsFPs
KytqEiAJkMnFixepqFSSUFGpJKGiUklCRaWShIpKJQkVlUoSKiqVqFyLJFRUKkmoqFSSUFGpJKGiUkl
CRaWShIpKJQkVlUoSKr3397///T/5kz+5urr62c9+i/xb/4oz/6oydPnvxP/+k/Ufme7/me3vvGxs
a5c+c2Njb27Nkzm81673/8x39MZXV1dXFxcRgGtqnAs88+S+S2X//v1uA1Qma2trVFZXV4EkQBK2nTt3j
srq6iqgsi2JevHiRSr79u1u5QhI1yblz56gcOHAASASOIEUIHz589T2bdvH5WLFy9S
+ehHP/pbv/VbP//zP/rrbc6aa0BSaioVJJQUakkoaJSSUJFpZKEikolCRWVSlR2ffvM5/Nnn30M5
eHobhN3/zN7/85S//9E//NPBHf/RHt99++1/6S3/p0KFD8/l3bW3t3LlzeybDML7ezybDMLBTEvXy5
4+cODAMAxMVKD3VkD3vrm5ef78+ef5ef78+eeff35tbU2Ua25tbU2Ua2v/OjUJExIojJJoiZhogJJADUJOyXpv
QOZqExUtrXWVP4UKtuSACqTJGoSIAnghEkSKkm4SpLe+1/4C3/hM5/5zEc/+tHXve513/zmN7/whS98
6EMfYqImOXPmzMGDB9m1U1R2ffucevIJMsxms49//OOPP/74Rz7ykazz985Sv33nvvvF77whbe//e2z2cz
J1tbWpUuXxnFcWVkBxnFUx3Hsvavjr0Kz+Xwcx957EhVQgd67ylWScIUkKjslUYEkbfOBJPyZQmAJC
oTNQmgAkkAlUkSlaskUZMwUZPwZ5CEbWoStqmtNa6gJvEKgNpaG4YhiQQoSZgk4SpJuEISrqCurq4+9
NBDn/rUp17/+v/j137tV37f/+v/j137tV37u536Oyebm5rlz544cOckKuK0Rl17fJxceefGEpfX3z4x//+Hvf+9719fXb
b7/92LFjy9e3LNnzxve8L3/qgovo1+fGjjMzE3ozlWkCsNLwxe7n8/pnny5NGjRw8F/gyovXegT+bzee/9OnzjzZ32mHr
s2LG9e/NL6hDDuDDz/55JPr+6tAktaavyzv8mKpCE/9+SMFGTqEn4M3OSROUKKldQk6hAEp
U/gyTDMLTWAJUrJOEKSdgpCTslUVtrN91006c//VfF/bWvffVv/+2/3Xs/c+bMoUH2HHWFq
Oz6Nnn69DMf+Cf/5BW33/GTP/mTqvEN/zCfzOPUGGN7xxfX19YWHh4ex4uvv/51+5vvuvrtvrvvvvvvffAgQPs
XllZUcdxVMeJurWstbm5eerUqZe//OVQm+8+eJwOtsQq2vnYtLr1E9/1+s1uPYpxYyaLprsyyC5JKeKGkiqsS
iUJFRUKkmoqFSSUFGpJKGiUklCRaWShIpKJQkVlUoSKiqVqFyLJFRUKkmoqFSSUFGpJKGiUklCRaWS
ZhaK0BKldIwiQJOyVhpyRM1EyAX/zFX/qDP/iDf/SP3velL33/n4/l4S33+fr5nd//+NmPAAHP3j//fcn4nQqLi4v33l
ZuP3PQTP/ETJ0+e/OwKNmQ9Hj//oy/e/9/6lZ+i7/5iW+4+/mtj+O72PxzxcWXcfk7H+xtcnx967ur6+
/sgjjzzzxxBO33377LbfgMgxD730cDqtNyeHHgUzVBYyFYU8ONxgN8vRfhjdG+wzzWuLJsAwDG2
ShEkSrqAChCcF4h0Vidp7T7T7K5ufmNb3yj9/7hD3/7Lfwn03tf/+9rf1/1E/91Gw20378+Pr6+g/90A998I
Mf7L2rfckvXeVyU033QR8+MMb/lt/62+bP/mbP/mp/5A8H6pqlPxC/cL1gJyn+N46g++9k3oP/6c/'
6JyFTUJoCZhJ5WrqOyUhG2tNaC1xk5JuEISSKkm4QhImSZh853d+55vf/H++5S1vOX366366XvvvvvffAgQPs
ukJUdn2b/I2/8Tc+9KEP9Kc+9KEP9Kc+9KEP9Kc+9KEP9Kc+9KEP9Kc+9KEP9Kc+9KEP9Kc+9KEP9Kc+9
SAPPJ2traV7/61VOnTt1zzz033nhj732+X0IDOSsXlcUj7Nz6V+Ueuf8Vn7lfgmJBJnAn4TB
kGtNaYJAGSqICapPeuMlGTjOMIOPmxH/uxH//xH++T97///'Z/61KeAcRyrTPP7448Dtt99+4403bs
Xzutt67muTf/Jt/873f+73PPPPPPMa17zmmeffbb3nuTo0aNPPfVU7/3jH//gBz7wgwd//mbP/mvP
Xzutt67muTf/Jt/873f+73PPPPMa17zmmeffbb3nuTo0aNPPfVU7/3jH//gBz7wgwd/mbP/mbP/IjP9J7T
```

```
2JaEnZJwlSRsywR417vedeedr3jPe97zu7/7X9/85jex6wpR2fVt8ta3vvXXfvXXzl84//f//t//t//
23/beh2EAnABJhmFQgTYBVEBlogJOgGEYXvayl33Xd33X3XffvbCw0Ht37Ovr6w9+/Wu/8zu/c/78eZ
XJ0tLSn/tzf251dXV5eXlpaWllZWVxcXFhYaFNlpeXW2uz2ay1NgxDmyRprSUBMgGSqExUJr13t12+f
Ln3nqRPtra25pONjY3FxcWFhQUmSba2tlprW1tbrbV//+//fZK//Jf/8uHDhy9duvTrv/7r4ziqb5os
LS0Nw/Af/+N//MxnPuOktQaoSXgJJGHSWkvCFZJQSQIkoZKESRImSdT19fW/83f+3r/4Fz/TWtuzZw+
7rhCVXd8m7/nRv/eRj/zixUsX/+pf/au///u/r3KNVP50SdimcgU1CX9mSVT+PyWh8h/+w39YXV2955
57Tpw48cADD3z4wx9+3/ve9+lPf/rnf/7nH374YfWFF1648cYbl5aW1tbWkgCbm5uXL19+y1ve8slPf
vJlL3vZuXPnNjc3jxw5Apw+ffqmm24CXvOa1/zxH/8xf4ok/G+ShG25AldJwrYkXCUJkEQFkjBJwhV+
6Zd+6U1vetPBgzcuLMzYdYWoXIskVFQqSaioVJJQUakkoaJSSUJFpZKEikolCRWVq7ztB9/6a5/6dSr
DMKhsU5moVJJQUakkUblGSdhJpZKESZLjx49/9rOfnc/n3/Ed33H+/Hkqv/qrv/rII4/83b/7d48ePX
r58uWf+7mfe8973nPw4EGVyvyLy8ubmpspOKpXWGpXeO5XWGpMkXGEcRypLS0tJgCRsS7K2tkZl7969Q
BIgCZMk58+f56Wkcj2LyrVIQkWlkoSKSiUJFZVKEioqlSRUVCpJqKhUklBRucp3fdcbP//AymU
rjKOI5XWmspVVCpJ1CTspFJJwhXUJIBKJQmQpPd+8ODBT37yk//n1nWAA7YTm8A+JHbR6JJKA
6nFRv2ZP/Ktx4tolYQEqZgI2kt/1Nk3zP1y//bPb+ARWlLwTUD7W2dDfYXhDepIpEK/ZZSS9wyZoFz+
RwsHT3//J0xRz7pBl6ifOhJR4Ddmo/+dPQmjgUU7e/+1c4e1mb8Dd/g2OyTiqt8zLFlg/dNpR/PJc/
uvVF7bqRQYWZCXtFQBhSu5Bu5xfLcKFrS+Xzc/d7JGaSl6mUKBUWvWhRtBCmXrMKt82dD/T8PSXaY
DG7GGoKzbEMrExXLdMVPpd6E/dt6tu3PoMjpq0V1vtQeh8WD0yXsx7lIpNFiYFzE9yCh1Cq9m2D4fsz
Gby5ltPrW3kMaWQpQYfrQ+9Zd9dhfnWtcfxdEePcnn9tt7NqPNdJ8S2j7eu/UdZtmwtFbFOmDQMcaAQe
4jR8jCZGJ8JNENbX5IP6rz1wnY5UsdsH2J58XQDdRr9gqhTLoRSrH4ZN6lIEzQaGoETX1WcoDGGFWNmo
FNZOZJ9Gf7sO4wCaH9ed7yKZRwDuA97vVFEgrGS0bSRuacIGXGSN7Dz1eewdVHwXJH0FyHBV1S/QNlv
Xf5SQ6wdVS1JuOOwHdVZz5JVWBkMiyQ0DF1CV0lDmtl3YCaxQgnKAHNKFgz2vzyOD9jSN2gRAHW/f9e
OneXeAZxZeoUBWnHt4k9FYDUzZAkDbDM2AoZ4DgcRQXmVT0BUdMJS5yHF8VM1X0j3F8cq4S2kdR2jU
```

```
FSSUFGpJKGiUklCRaWShIrKTg899NDKysptt91G5eLapSFNTTIMg8pkaWmJytrausrEbUlWV/dSef75
55kkUZOowKFDh6g899xzSQAnQBLg8OHDVE6fPp1ETQKoQJIjR45QefbZZ1UmKpMkN910E5Wnn35aBZK
wTb3llluoPPPMM2rvPQmgAkluueUWKqdOnVKZJOm9M7n11lupPP7440nUJICTJLfffjuVr3/960nGcQ
RUQE0yDENr5EXw8lccZ9dOUbkWSaioVJJQUakkoaJSSUJFpZKEikolCRWVShIqKjt985uPLC0uHrv1d
irjxlprDbAFUJMAbVii0t2MqICaDE5mC0tU5lsbahIVaKICs+UVKhvra6213ruaBFBba4tLK1TW19eZ
qEASJ3v27KGytramMlGBJMDevXupXLp0SWXSWgNUYO/evVQuXboEDMPQewd676213vu+ffuoXLhwQeV
bGqAmAfbv30flzJkzKt/SAJXJoUMHqTz66KPQ/4QTIJCktZYEuPPld7Frp6hciyRUVCpJqKhUklBRqS
SholJJQkWlkoSKSiUJFZWdHv3mI4uLi8duvZ3KfP3SMAyALS/iRQmQLFDRkUoyUOl9nkRlEv6XtBmVP
m4kAdQkKpM2LFHp4wYTNYmaP9EWqYzzdRVoraltGPo4qrOFFSrzrcsDAdQkgC1AG5ao9HEDUJk06eFF
w2yZyrixxiSJkyTqbHkvlfXLFwx0uULvfe++A1SeOnWy7+SktZZETfLyVxxn105RuRZJqKhUklBRqSS
holJJQkWlkoSKSiUJFZVKEioqOz36zUcWFxeP3Xo7lfn6pWEYSAxJ+BN50QIVFToTNQmTZKCiI47j6D
AMJNDtXR1my1Tsm4CaRGXsmQ1A2iIV++Y4jq21+XzeWhuGofcODLNlKvOty621tGbvahKg9z5bWKEy3
7qcbpIeBtJ7twWYLaxQcWtdZZJEBdRhaQ+VvnlZBZIAKpNhaQ+VjfWLTdR5pAukO+K+1YNUTj35hNon
6jiOvXcgSWsNaK3dcecr2LVTVK5FEioqlSRUVCpJqKhUklBRqSSholJJQkWlkoSKyk6PfvORxcXFY7f
eTmW+fsmWWRoJLbwoAZIFKn7LmERNBuiA2tqMim7htyQhsfckahuWqPRxo/feWlN778MwRDoOs2Uq43
y9tab23ofZrI9jEnWYLVOZb10G1GEYeu9JAHW2sEKljxtM1CRqXtTNwjKVvnkZSNJ7b8Ng70za4gqVc
WNNba2pQO+doTUZlvZQ2Vy7AKjAiC9Kou7dd4DKU6dOqr33cRx77+o4jsDdJ17FS0nlehaVa5GEikol
CRWVShIqKpUkVFQqSaioVJJQUakkoaKy08lHH2uNY3fcSWW+finbbAGSkBctUNFRBZJwhWSgMo5bLfb
eW2tA7+RFrSUDlXG+3lpTe+9sSzLMlqn0cQNQW2skfRyBJllYpjLO15vYorbWeu9qe9GwRMWt9R4a6d
hao4sCWVimMs7Xm6hJfFFLE7UtrlDpm5fVJCMmcTKQYWkPlY31i036i8KLVKD3vm/1IJVTTz4BjOPYe
1d772rv/e4Tr+KlpHI9i8q1SEJFpZKEikolCRWVShIqKpUkVFQqSaioVJJQUdnp0Ue/sbgwHLv1Tirz
9UvZxtD4E3nRAhUdVbYlAXrvw7BAxT4Heu+JfEvTUZ0trFDp44baex+GQWWSpA1LVMb5OmNvwzDvY+W
39EQAACAASURBVJJhNptvbSWZLaxQGefrQJLee17UZdIWV6g43+jjCLTWSIB5H2dtyGyJilvrKttUJs
PSHip98zKgAgGSEQfSFleobF2+mKT3zmRu70h33+pBKqeefELtvY/jqPbe1d77iVfew0tJ5XoWlWuRh
IpKJQkVlUoSKiqVJFRUKkmoqFSSUFGpJKGistOj33xkcXHx2K23Uxk31lprvCgxvCiJ2oYlKjpyJb8F
aMMCFfscOjCOY7apbViiMs7XkwBqEibqMFumMt+6nKG10RHbbHDsQJI2LFEZ5+tAa42uvfeQri2zhRU
q863LTdTWmppEBdriCpWtzTVgEJIRgXSTDEt7qIwba6213nsStfc+m81678PSHirrly8ATXpQASd79x
2gcurJJ9Q+GcfRCXD87lfyUlK5nkXlWiSholJJQkWlkoSKSiUJFZVKEioqlSRUVCpJqKjs9Nij31hcX
Dx67DYq48Ya0IahYyZqXtQWqfQ+B5JARw1hUFubUenjhtpaG8cRSKImGWbLVOZbl4fZzN7Z5mS2sEJl
vnW5SQ9N1MyGdHsYZstU5luXgYEAPaSbRG2LK1Tsm31rztB670mAdJO0xRUqmxuXmgzD0HtPgsq3DEt
7qGxdvpgESKICakuG5b1Uti5fHBFQARVQ9+47QOWpUyf7VZIcv/uVvJRUrmdRuRZJqKhUklBRqSShol
JJQkWlkoSKSiUJFZVKEioqV1Aff+ybi4uzo8fuoNI3LwdIenhRa03Ni9oilS9/6YskaBIgE+A7XnMfl
f/5wJdba2prTU3Se09yz6tfQ+XrX/uqmqT3nonaWrv7xKuoPPT1B9UkY+9J0NYacPeJV1F5+KGvAa21
3nsSJ0mO3/1KKg99/cEkTNRMgLuOn6Dy0NcfHIZhPo6BJL33YTaz9+N3v5LKNx55CGitAWprLZPbbr+
TysULZ1prQBIVUIG9+w5QOfXkE2rvfRzH3rvaewfuPvEqXkoq17OoXIskVFQqSaioVJJQUakkoaJSSU
JFpZKEikolCRWVK1y6dGnv3r2nTp06duwYlXFjrbUG2AIkUV944YXDR26Dh8sY3vrG1NgzDfD5XoUNL8
vnPf57Kd37ndwIq4KT3rn7pS1+icu+997bWgHEcl5eX19fXk6gPPPAAlde//vUbGxtqktaaCiR+5Sv/
k8prXvMalUkSYDabbWxsfPWrX6Xy+te/Xh0nS0tLTpJ88YtfpPLa174WUJmoSYAvf/nLVN74xjf03od
h0Lyo/S989rP/lcr3fu93t9bUcRxba+rW1tab3/zmf/7P/28qJ594LMk4jn2bmuT43a/kpaRyPYvKtU
hCRaWShIpKJQkVlUoSKiqVJFRUKkmoqFSSUFG5wtbW5lOnnlxYWDh27DYq8/VLwzD0kERtw4DpvQ+zR
Sr33/9xJmPf6r0PbQFoA+/44XdjHfmk2DLPZrE+S9N6T/PA73kXllz95P+BkNpvN5/Pee2vh9/x
Lir/+v6PA22ytbWVTWt/dAPv5PKJz7+0WYmu99yQqkOTtP/xzh+/xzh+h8iu
//K9ba1tbW7PZbBxHoLU2n8n8/f8c4fofLLn7wf6L0nGccxCZN3vPNHqPzi//ORWWbG5u/sZv/Mb6kbZrH3/3d36Ny6k967euBOVeL69ReeVaJKGiUklCRaWShIpKJQkVlUoSKipVJFRUKkmoqDh61//2r69i
...
```

```
723sdxVHvvToZhANSl2dKtd97Brp2ici2SUFGpJKGiUklCRaWShIpKJQkVlUoSKiqVJFRUrvD8889eu
nRpZXn5pptvoTJurLXWAFuAJEzSFqn0cSutofwvnQRNW6TSx40kQO+9tTafz4dhUIfZMpX51uUkgNok
yYjAbGGFyrixBqiZzO0DsWW2sEJlvnUZGIgKqK01IAvLVPrWmpqEidpaAzJbodI3LzPpvScBWgJkcYX
KuLEGqElUJrYsLO6hsrl2AVBHfBHgt2R1/wEqp558om8bxxHovQNJWmvqwvLibcfuYNdOUbkWSaioVJ
JQ8f9tD/5ibT0LO7//vs+79t7nj41tbIxtQpikabAh7QUNMGkhqtSZqGqVaJRqRpOOhokmE1VVe1Gph
Y4qFUGmTG/SXuRmmuk0N71opVZRLiJ1rqYalJIATjMNJIMZwPgPAYIdjPH5u9f7fLv8Og897n7Z56zT
/dK9lp/PRzMHyBw1c4DMUTMHyBw1c4DMUTMHyBw1t3jhhW9dvfLywcHRY297e+aMN66WUgKSDSCJpHC
YOXW8ASQlYF0nAZJQDjNnXF8vpWRjrAHJhjqsLmROvXkNqOQ1rsdhGGqtw9GlzLl548pqtVqv16WUJL
VWNqqrC5czZ339yjAM6jiONEnK4cXMqTevAev1GkgCqGXj8GLm1JvXMlGBJECtdTi6lDnjjauV1LiSW
mspRQWGo0uZc+3qy5moSWqtmdxz7wOZ8/xzzySptY7jWGt1koRJksuXL7/1kcfSvR5qtgFkjpo5QOao
mQNkjpo5QOaomQNkjpo5QOaomQNkjppbfP1Pn1/fGFfD8Njb3p45442rpZRsgATIhHKYOdabNQKpJgG
cDKsLmbM+vpYEyOsNqwuZsz6+BqhAMd/DwYXMGW9cTcJGKeN6PQxDJerq4GLmrI+vARnrGEspVDeA4e
hS5tSb14CxJFWqQCbl8GLmrK9fATJRK9kopawOLmbO+voVlUkSJ8DqwuXMuXb15TROaq1J7n3TmzPn+
eeeSVIbJ7VWoJSS5PI99771rY+kez3UbAPIHDVzgMxRMwfIHDVzgMxRMwfIHDVzgMxRMwfIHDW3eO65
Z4ZSgMfe9vbMWV+/UkoBAhJAZaMcZo4eJ0VNohZQ2SirzBnX15OUYK2shlqrmmR1cDFzxhtXVSZjZKN
aycHhpcw5vnkVSFLMRq21lMLGwYXMWV+/ApRSnCQZYzGrC5czZ7x+BTABxjgENclwdClz1tevAEnUTI
Akw9GlzFlfv5KmErUYYHXhcuZcv/ZdNROqY9xIcs+9D2TO155/Vh3Hsdaq1lrVJEyS3Hvfg2956M3pX
g812wAyR80cIHPUzAEyR80cIHPUzAEyR80cIHPUzAEyR80tvvLVL184OKSUxx77ocwZb1wFklCKZANI
QjnMHOvaBIzmVUVHNsph5ozr68VYGMdxRVljCcVwcCFzPL6ujrGUQrXWWkpRh6NLmTPeuJoEqMRJMRu
rC5czZ7xxVR1KqTpGNqobBxfvyZz19StJABVIoiZZXbicOeONqyqgMqm1AsPRpcxZX78CqJmoSYDVhc
uZc+3qy5k4SaLUWt903wOZ8/xzzyQZx7E2ahKglKLe/8D9Dz74cLrXQ802gMxRMwfIHDVzgMxRMwfIH
DVzgMxRMwfIHDVzgMxRc4uvPv3lg4NhWB0++ujbMufmjStA2TCVMElCOcwc680aCyuV1EC1YspwlDl1
vKEmqbUySQKU4Shz6s1rSSwU40aBapJyeDFzxhtXcwsVSDIcXcqc8cbVJGoppdYKZDIcXcqcmzeuDGH
DCeDk4OI9mbO+fiUJoGZSSlGHo0uZc3ztFSCJClTymsOjy5lz7erLSWwyqTVvuu+BzPna88+q6/Varb
U6qbWWSa31zQ899OYHHkz3eqjZBpA5auYAmaNmDpA5auYAmaNmDpA5auYAmaNmDpA5am7x3NNfYZVy4
ehtD/9Q5ow3rpZSKgGSAJlQDjPHutaRUqKZ1FpLKZTDzBnX14EkGGslCajl8GLmeHw9G3A8roEkQ3jV
wYXMWV+/UkqptWYC1FqHYSiHFzNnvHG1ErUYNpJaa1bD6uBi5ty4/gqQZgi1VmB14XLmjDeullKO6wi
k4FiHsFEOL2bOjeuvZDKESkq1FopZXbicOdeuvpxEkuprktRa33Tfg5nzteefrbdIUmvNBEjy8EMP33
v//eleDzXdD9xXn/7ywcGwWh2uDodSykAZNgKrYQhDKUkoJZBEAiSREJKSV9VsaBpJqmmATGqtmVCtZ
EMtG+bPDSVjVTMUCTAer7/zne889dS/eObZ5+PxQw899M4nHn/LW95yMKyGUMexDEMSYG1Vh1BKWa/X
maxWq3EcgSQqkKTWmqFQLVAVSKMCKpBEBVQgiQqoJCaVqAdlGMexlKIC0WPrASXJSIAkQ1BJqg7DEEh
hHMchBNZ1XFFqrUmASlSqpRR1jEmollKSjDEJ1QJVK9mgmskYM5GkulFrTVJrLWW1Xq+d1FqTjOOoju
PopDZqkn/5xx/PktTsMtR0P3BfffrLh4erYTgYDspAWa1WQyilrIbBpJTCa0pJItkAMpEQjEnIoIaah
GCtgAKmGceRiVprTcL3GMkLf/bif/vf/Nrf//wn/HvP/zww2NMQh
XIxEk0MMZSCtUxZkJ1jEncICWotdYktVYYxnG0qRN1HEe1TtQ6SfLj73wiS1Kzy1DT/cB95ctPXbh4e
TUMZcVAGSZAaZKUUpIASYA0EoIxCSETJVRCNImaxAmvqW4A2QBr5WD1j/7Rb3zoQx7UqrUMYx3G1
WtVak1SiHh8fHx4eqoCaRk3yqU/93uHR0Qf+4vpJNwkRNAiS
ptQKVFENSFai1llKSWFCBYqKU8k/+6SfqYtSWqoOCMAwqoOYWgJoEUIEkwCc+8V9/5CP/2U
EZNsZYzNqaZEUZY17PSZJaK5DESRL/HEnGcawTtTZqrTXJOI5qrTXJOI5qrTXJj7/ziSxJzS5DTfcD98UvfuHgYey
5dLKavVahgok4MylCZJKUgAXIryIavSiEJRgVUQM1ELSYQX5WE1VCP10CG8l9+GMf9HfmoX/vXXbu/vK5LTl7
xER9/Cf+lS8/9YX1X1YX1ej3G3/md/YaDuel+90/u/8r3l8/+/+hrj1ok8GM2U/rUsBUk9bGi9yzmF2WMpp5pwJEKXY9M+u55g4pnZ5Z0gRvf79KrfTxJw
18QhYSFLMhpMCJpUMwULGWmCsNUNhXf6Ho4MbDAdr3zJc+fmsbtb1wXpIcSMNoCYBnAAWitkAysHq7/
zyf/Df/YN/UOtaLeZ7Di5e+vjHP/7Nb3zgtTr88If/05s3byYBSinjOJZS1Oef/
5PPff7z/+6/82/OhbZSAKMkSowxiTFqJVsqEmptSbRMYlaazacjOPopE7GcUxSa1Vroz7+xLuzJDW7
DDXdD9wXv/iFey5fBoZhODgDYmRFWQ1DYCgFqIXXZALkBEmqQBoVE82kEoljLeZ7xrharT72/5+K9
```

```
87PjmTbWY16i11p/6wAd/7/c+tV6vV6tVklprklLKhz/8n//qr/4qriO11iTA1atX/8f/6X/+D3/57z
gBkqjAGIvZKKU4GeMQKhmCmuSfvXL1yQcfvllzIRnH+kv1JsVSR5NK2KgClahJinkNUEkxQCUbf/fv/
hef+MTfo8pqqLUCSf7J//5P3/++9/3ar/3aL/7ih5IMw5AE+JW/94mPfPjDP//zP//7n/m0CmRSyQsv
/Nnv/u7v/pW/8rMZHaMEs0E1CVCJEyATNUmt0TGJkqTWtZqUOnEyjmOtNUmtdb1e29Ra1cefeHeWpGa
Xoab7gXvmmacLHBwcAAcHQyllGIYVBSilrFarJEwqYZIT1CSUEs2txlpJCfFVdQBD9VW1OpQhZGMov/
Ebv/FLv/TLx9evWVCTFLMBqEnKweqll15+8KGHjm9eX6+Ph5QkFmosYQif/8JTX/zSv/j3fvZn11Y1k
1KKCtRaATUTNRuFoQZIUzcIkAZIUsz3jDGJmkIJxVRSzEYlG8Dn/ujzv/epT//tv/2LQ0hCYnJjPf72
b//2K6+8cjwZx/H4+Hi9Xt+4cQP44Ac/+IEP/OtYUwag1vriiy/+wR/8Xz/zM3+J6qsKmahJVCATNYm
aBFBrrTDUWnVMoiRZr9fAOI7qOI5qrVWtr5cEeOfj78qS1Owy1HQ/cC9tfPvFYRgODw9LyTAMpZRhGE
opwDAM6kEZMpRiNlgNadQkQBIVSKJm4qSYSooZYzEqkESt5DVUk5RSMpT/5X/9ze+89NLf+lt/0/U4D
MMYqY6RoWBeozI5Pj7+6Ec/+tM//+zM/8pbIRSVGTVKIyUZOwUU2ztpZS1DTFjCUlFLNRCeAGwaiA
CiRRgSQqoCYBktRah2FQj46OfvM3f+vg4OAv/+V/i2oBk0qKqeRWtdYhJAFevvLKr//6f//8Lv/ALjz7
61rJhKlGTAMWolbxGzQRQk6hJ8VWjmhS1ToD1eg3UWtfrNTBOktRax3F0kkR94l0/kSWp2WWo6f7/8K
UvffFgtTo4OCilDAPDBFitVqUUmlJKklJKvj+MRE2iFqMmqbVaqLXm9dQhjDEJkEQFVECttQ7DcHx8f
OXKlVrrhQsXjo6OgFrrwcFBknFcF9lQk1Ty/6ICaQA1CVBrBZK4QUpQSylqCqkmUZMAOUEF1EzYqCYp
pYwxE8lQQ3JsHYbhc5/73Le+9a3r129+97vfPT4+vnjx4oULF4aB97znzPQ888ACTUopaSlFrrUlKKUm
KsWADJFGBWitQa4UkJPFVJKm1JlFrrWqdqHWijuNYa1XrRE0CPPGun8iS1Owy1GwDyJLUvGE8/fSXh1
KAg4Nho0xWq1UpBVhRNsY4DAOTJICaphi1ku+hupFkjEmOLtyTM3XllZeSALXWUkqtdRiGi5ieLbKK
y+luXzP/Znzyne/DSS5fM/9OVNXXnmp1lpKUZMAKpDECZBEzRxAzQRQc4shHNeRiQqoSUlSa83EE2qt
6jiOap2o6/U6iVonSdRSiro6OLh48fLb3/72dCegZhtAlqTmjeRLX/riUMAOLh48fLb3/72dCegZhtAlqTmjeRLX/riUMpqtSqlHBwMQnCnl4OAgSSk
FKBOglKImAtJR83pqJmqtncly/flTL3b/3y3W+reb173/TmNC9/50Ugifqm+x7MnO++/GeAeu+b3pwz9f
J3XkwC1Fp5TTVJJSqgHUNICaCZBEzUQF1LyqgHUNICaCZBEzUQF1LyqJNGRSa01iQpDklprEie1VhWotaq1Vnn
eT/USmrCxcu/OiP/li6OajZBpAlqXmD+aPP/S/GWsWalVagWsVqtSSSpJhpJKkavk4gRI
oiaptQJJ1ETNxj33PpAz9Z2XXlABNQmZ5L77H0rz0re/lQnkvvvfkjnfeemFTO67/6Gcqe+8+Sm0134e
ahIkKqPlzJalqGkBNSiY6JgGS1FphUDNxAqi11iRqrTWTegt1HMdaq5Naa5qqlHpwePxH38i3feBmm
0AWZKaN6TPf/4MRAyr1QostwDKBFCSmkRJKqAmUfKqmkSpdQ088OaHc6Ze+sRe+NY3mSQ1Ve
PChR9K8+MI3kgKoDz70cOa8+MKfZvLgQw/nTL3w3GkXa1Vz4iVVUnB5tnd2V3Rh1q3xeuqaO6f8m6xxp/7w30WnjT6pDAASs0TnkeVSl2cdPa9XCmTb9S4zK9VPAwHEoGHm+fVq/9VFTYOl49Yj9yZXNAQdCqJH8T6F4v3fFAWm9dEUcUo5OVSh3wdNiyzDVDdq/Z7qDJ32Gw4jR8UKQNNwaNvbWMcTKGpFOwAS6nNCLbc8g3zvs4WlnDDyHn2U3Wz2
0cHCQ5ODgoJQCDMOQZBgoJSiMslETaICtVZArbUCSd76yGNpdvdH1ryVRSym11SFDUJkESttQKllC
QqoD7y6NvS/Ok3v57Jw299NHO++Y0/AdQkb33ksTTf+PrTI+Q7hBuHzTeLVcaP/3m1zNRmdQJkyQqoAJJa
q2A+uhjP5TmG1//mgqogFpKkcZJbqEmAmA3KLWCqiAmjlqEhVQkzhJoiZRa61kqrk1qrWmtmVa61Jaq2Z
uD7m0uEFLj7x7nelux3UbAPIktS84Wx8HT+S5rlvPKtm8ojrvff9D7gUfvfSDjkTVCSmkYJkAVUAEnUQF1x4X5J
ETQOoQCZAEhXILX74HT+S5rlvPKtm8oE+KG3vyNnzm2zzzydhE+JM2zzzydhE
mtNZNSSpJxHJmogJNSipN3/IUfTfPsM09oJxkkQFMlGBWiuQRz/wHS3NVWVEK/ctAXN1yQcHR0dPdP3G+N6ffE+6O4abQBZkpkDX2/6BZWFp
lqrTa3V1yQqcHR0dP3G+N6ffE+6O4OabQBZkpruFp/9/c9Q3SilDMPApJSkEkmgBMgiVpKUd/9E/9q
mj/6/B8CaibqMAy11iRqRk1SSnn8iXeXeneeoLf6wC73z8XZnzhX/+R5kA73z8XWme+sIfAyqgZvL
Ox9+V5otP/fM0ak5QgUwANYn6+BPvTvPUF/5YTQKoSYBSygKoKSYBaayml1poT1CSUEFMnFSSqm1JgHUTAMT
hJU2tVk6hAErXWal7F5Morxx/86Z9Ktz3UbAPIktR030et9Q8+85nvXQxnRnbAPIktR030et9Q8+85nvXruyXq+BYRiAJKUoJSijGgSA
Oq/8YGfTvO7n/qdTIDCMAZIIAahrgfe//qTRPfvbTapL3vu0kEkD9yfe+P82Tn/qdTn/10EkAF1FLK
e/6196b5/Sc/A6iZqEAVHPDdW+9ePmiVZjjMmQUUGjGwnGQoz4D/qUmATNRMADWvByRRk5j4moz
jzVK9ds/Fb7/nfX813f83qOn2F5BGTQPkVGoaII2aCZA5auYAadRMgDRqGiCnUtMAadTsMiBLUrPLUN
PtLyCNmgbIqdQ0QBo1EyBz1MwA0qaYB0qjZZUCWpGaXoabXgpgDRqJkDmAdKoaYCcSk0DpFGzy4AsS
c0uQ023v4A0ahog0kDpFGzy4AsS c0uQ023v4A0ahogp1LTAGnUNMAckUdMAadScAR0wA5lZoGSKNm
lwFZkppdhhppufwFp1DRATqWmAdKoOQFIo6YB0qg5AUijpgFyKjUNkEbNLgOSJDW7Xd/gLSqGmAnEp
NA6RRcwKQRk0DpFFzApBGTQPkVGoaIIaXQZkSWp2GWq6/QWkWp2GWq6/QWkUdMAOZWaBkijZhZBkij
bIqdQ0QBo1uwzIkttsMtR0+wtIo6YBciYBciq1o+Z2gDRqGiCNsIo+Z2gDRqJkAaNQ2QU6lpgDRqdhmQJanZZajZBpAlq
pAlqenODpBGTQPkVGoaIII2aCZBGze0A0aRR0wBpFGzy4AsSc0uQ023v4A0ahoADRqGiCnUtMAadR0+ws12wCyJDVvGE8/fSXh1KAg4Nho0xWq1UpBVhRNsY4DAOTJICaphi1ku+hupFkjEmOLtyTM3XllZeSALXWUkqtdRiGi5ieLbKK
pAlqenODpBGTQPkVGoaII2aCZBGze0AadRMgDRqGiCnUtMAadR0+ws12wCyJDXd2QHSqGmAnEpNA6RRcwKQRk0DpFFzApBGTQPkVGoaIII2aCZBGze0AadRMgDRqGiCnUtMAadR0+ws12wCyJDXd2QHSqGmAnEpNA6RR
```

```
MwHSqLkdII2aCZBGTQPkVGoaII2abn+hZhtAlqSmOztAGjUNkFOpaYA0aiZAGjW3A6RRMwHSqGmAnEp
NA6RR0+0v1GwDyJLUdGcHSKPmrgBp1EyANGrmAJmjZgKkUXNXgDRquv2Fmm0AWZKa7uwAadTcFSCNmg
mQRs0cIHPUTIA0au4KkEZNt79Qsw0gS1LTnR0gjZq7AqRRMwHSqJkDZI6aCZBGzV0B0qjp9hdqtgFkS
Wq6swOkUXNXgDRqJkAaNXOAzFEzAdKouStAGjXd/kLNNoAsSU13doCcKTUTII2aBkijpgHSqJkAOVNq
uv2Fmm5/ATlTaiZAGjUNkEZNA6RRMwFyptTsMiBLUrPLUNPtLyBnSs0ESKOmAdKoaYA0aiZAzpSaXQZ
kSWp2GWq6/QXkTKmZAGnUNEAaNQ2QRs0EyJlSs8uALEnNLkNN120PyN1S080BsiQ1uww1Xbc9IHdLTT
cHyJLU7DLUdN32gNwtNd0cIEtSs8tQ0+0vICeouQWQiZo5QE5Q0wCZo+Z2gJxKTQPkBDX7AsiS1Owy1
HT7C8gcNRMgjZo5QE5Q0wCZo+ZUQG5HTQPkBDX7AsiS1Owy1GwDyJLUdGcHyBw1EyCNmjlATlDTAJmj
5lRAbkdNA+QENd0bA2q2AWRJarqzA2SOmgmQRs0cICeoaYDMUXMqILejpgFygprujQE12wCyJDXd2QH
SqLljQBo1DZAT1DRAGjV3DEijpgHSqDkBSKOm21+o2QaQJanpzg6QRs0dA9KoaYCcoKYB0qi5Y0AaNQ
2QRs0JQBo13f5CzTaALElNd3aANGruGJBGTQPkBDUNkEbNHQPSqGmANGpOANKo6fYXarYBZElqurMDp
FFzx4A0ahogJ6hpgDRq7hiQRk0DpFFzApBGTbe/ULMNIEtS050dII2aBsgJahogp1LTAJmjpgFyx9Sc
CkijpntjQM02gCxJTXd2gDRqGiAnqGmAnEpNA2SOmgbIHVNzKiCNmu6NATXd/gLSqGmAnKCmAXIqNQ2
QOWoaIHdMzamANGr2BZAlqdllqOn2F5BGTQPkBDUNkFOpaYDMUdMAuWNqYDMUdMAuWNqTgWkUbMvgCxJzS5DTbe/gM
xRMwHSqLkdIKdS0wBp1JwApFHTAGnUNEAmavYRkCWp2WWo6fYXkDlqJkAaNbcD5FRqGiCNmhOANGoaI
I2aBshEzT4CsiQ1uww13f4CMkfNBEij5naAnEpNA6RRcwKQRk0DpFHTAJmo2UdAlqRml6Gm219A5qiZ
AGnU3A6QU6lpgDRqTgDSqGmANGoaIBM1+wjIktTsMtR0bwxAGjV3DMgdUzMHSKNmAqRR0wCZo2YCZI6
aXQZkSWp2GWq6NwYgjZo7BuSOqZkDpFEzAdKoaYDMUTMBMkfNLgOyJDW7DDXbALIkdN7ygDRq7hiQO6
ZmDpBGzQRIo6YBYBMkfNBMgcNd3+Qs02gCxJTbcMIBM12wByx9TMATJR0wBp1DRA5qiZAJmjpttfqNkGk
CWp6XYBkEbNHCCNmhOANGoaIII2aBshETQOk3t2PMgAABxxJREFUUdPtL9RsA8iS1HS7AEijZg6QRs0J
QBo1DZBGTQNkoqYB0qjp9hdqtgFkSWq6XQCkUTMHSKPmBCCNmgZIo6YBMlHTAGnUdPsLNdsAsiQ13dk
BcqbUTIDMUXM7QO6KmgmQOWq6/YWabQBZkpru7AA5U2omQOaouR0gd0XNBMgcNd3+Qs02gCxJTXd2gJ
wpNRMgc9TcDpC7omYCZI6abn+hpttfQM6UmgmQOWpuB8hdUTMBMkfNLgOyJDW7DDXd/gLSqLkrQBo1J
wC5K2rmAGnUNEAmavYRkCWp2WWo6fYXkDlqJkAaNbcD5FRqGiCNmhOANGoaIII2aBsgJau4YkEZN98aAmm0AWZKa7uwAadQ0QE6lpgHSqJkA2YaaUwGZ
o6YBcio13f5CzTaALElNd3aANGoaIKdS0wBp1DRAGjUNEAmavYRkCWp2WWo6fYXkDOlZgJkjpoGyBw1Ew
CNmjlATlDTAJmj5naAnEpNA6RRcwKQRk0DpFHTAJmo2UdAlqRml6Gm219A5qiZAGnU3A6QU6lpgDRqTg
DSqGmANGoaIBM1+wjIktTsMtR0bwxAGjV3DMgdUzMHSKNmAqRR0wCZo+ZUQG5HTQPkBDX7AsiS1Owy1
HT7C8gcNRMgjZo5QE5Q0wCZo+ZUQG5HTQPkBDX7AsiS1Owy1GwDyJLUdGcHyBw1EyCNmjlATlDTAJmj
5lRAbkdNA+QENd0bA2q2AWRJarqzA2SOmgmQRs0cICeoaYDMUXMqILejpgFygprujQE12wCyJDXd2QH
SqLljQBo1DZAT1DRAGjV3DEijpgHSqDkBSKOm21+o2QaQJanpzg6QRs0dA9KoaYCcoKYB0qi5Y0AaNQ
2QRs0JQBo13f5CzTaALElNd3aANGruGJBGTQPkBDUNkEbNHQPSqGmANGpOANKo6fYXarYBZElqurMDp
FFzx4A0ahogJ6hpgDRq7hiQRk0DpFFzApBGTbe/ULMNIEtS050dII2aBsgJahogp1LTAJmjpgFyx9Sc
CkijpntjQM02gCxJTXd2gDRqGiAnqGmAnEpNA2SOmgbIHVNzKiCNmu6NATXd/gLSqGmAnKCmAXIqNQ2
QOWoaIHdMzamANGr2BZAlqdllqOn2F5BGTQPkBDUNkFOpaYDMUdMAuWNqYDMUdMAuWNqTgWkUbMvgCxJzS5DTbe/gM
```

```
Wp2GWq6rjsfgCxJzS5DTdd15wOQJanZZajZBpAlqem6rpuDmm0AWZKaruu6OajZBpAlqem6rpuDmm0AWZKaruu6OajZBpAlqem6rpuDmm0AWZKaruu6OajZBpAlqem6rpuDmm0AWZKaruu6Oajpuu58ALIkNbsMNV3XnQ9AlqRml6Gm67rzAciS1Owy1HRddz4AWZKaXYaaruvOByBLUrPLUNN13fkAZElqdhlquq47H4AsSc0uQ03XdecDkCWp2WX/N3y+mOLHwCY+AAAAAElFTkSuQmCC\",\n\t\"errorCode\" : 0,\n\t\"errorInfo\" : \"No error!\"\n}\n",
                "result": 0
        }
}



// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async generateImagePreviewImage(){
  const generateImagePreviewImageParam = {
                "displayScale":8
        }

        const res = await this.nMPrintSocket.generateImagePreviewImage(generateImagePreviewImageParam['displayScale']);
  //预览图生成失败，退出流程
  if (res.resultAck.result != 0) {
    return;
  }

  //解析处理数据
  const obj  = JSON.parse(info);
  const data = obj.ImageData;

}
```

四、打印接口说明

4.1 设置打印回调

代码块

```
export default class NMPrintSocket {
  // 添加打印监听方法
  addPrintListener(callback)
 }
```

代码块

```
let printListener = null;

printListener = this.nMPrintSocket.addPrintListener(async (msg) => {
  const resultAck = msg?.resultAck;

  if (resultAck?.errorCode === 0 && resultAck?.info === "commitJob ok!") {
    await strategyFactory.handleCommitSuccess();
  }
  //已接入历史版本客户仍可以使用printQuantity和onPrintPageCompleted字段获取打印进度

  if (resultAck?.printCopies != null && resultAck?.printPages != null) {
    strategyFactory.handleProgressUpdate(resultAck);
  }

  if (resultAck?.printCopies === printQuantity &&
    resultAck?.printPages === list.length) {
    await strategyFactory.handleCompletion();
  }

  if (resultAck?.errorCode !== 0) {
    strategyFactory.handleError(msg);
  }
});
```

4.2 移除打印回调

代码块

```
 export default class NMPrintSocket {
   /**
   * 移除指定的打印监听回调函数
   * @param {Function} callback - 需要移除的回调函数
   * @returns {void}
   */
  removePrintListener(callback)
 }
```

代码块

```
const cleanupListener = () => {
  if (printListener) {
    this.nMPrintSocket.removePrintListener(printListener);
    printListener = null;
  }
};
```

4.3 开始打印

代码块

```
export default class NMPrintSocket {
    /**
     * 开始一个打印任务。
     *
     * @param {number} printDensity - 打印浓度，根据不同打印机型号取值范围不同，具体如下:
     *                                    - B3S、B203、B1、K3、K3W、M2: 取值范围 1~5，默认为 3。
     *                                      - B50、B11、B50W、B32、Z401: 取值范围 1~15，默认为 8。
     * @param {number} printLabelType - 纸张类型，可选值:
     *                                 1: 间隙纸
     *                                 2: 黑标纸
     *                                 3: 连续纸
     *                                 4: 定孔纸
     *                                 5: 透明纸
     *                                 6: 标牌
     *                                10:黑标间隙纸
     * @param {number} printMode - 打印模式，可选值:
     *                              1: 热敏
     *                              2: 热转印
     *                              注意，不同打印机型号支持的打印模式有限制，具体如下:
     *                              - D11、D101、D110、H10、B16、B18、B3S、B203、B1、K3、K3W、B11 仅支持热敏。
     *                              - B50、B50W、B32、Z401、M2 仅支持热转印。
     * @param {number} count - 总打印份数，表示所有页面的打印份数之和。
     *                              例如，如果你有3页需要打印，第一页打印3份，第二页打印2份，第三页打印5份，那么count的值应为10（3+2+5）。
     * @return {Promise} - 返回一个 Promise，解析为开始打印任务的结果
     * @example
     * //返回数据示例
     * {
     *     "apiName": "startJob",
     *     "resultAck": {
     *         "errorCode": 0,
     *         "info": "startJob ok!",
     *         "result": 0
     *     }
     * }
     * @description 返回结果中的 errorCode 含义如下:
     *              - 0: 成功
     *              - -1: 失败，info 表示原因
     *              - -2: 打印机忙碌, info 表示原因
     *              - -3: 打印机接收到不支持的参数, 主要是浓度、纸张类型、打印模式,
info 表示具体原因
     */
  startJob(printDensity, printLabelType, printMode, count) {
    // 方法实现将在这里
  }
}
```

代码块

```
//返回数据示例
{
        "apiName": "startJob",
        "resultAck": {
                "errorCode": 0,
                "info": "startJob ok!",
                "result": 0
        }
}
// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//总打印页数
const printDataLength = 5;
// 调用流程
async startJob() {
  const jsonObj = {
    printerImageProcessingInfo: {
      printQuantity: 1,
    },
  };
  const density = 3;
  const label_type = 1;
  const print_mode = 1;
  const printQuantity =jsonObj.printerImageProcessingInfo.printQuantity;
  try {
    const startRes = await this.nMPrintSocket.startJob(
      density,
      label_type,
      print_mode,
      printDataLength*printQuantity
    );
    if (startRes.resultAck.result == 0) {
      // 提交打印任务
      await this.printTag(list, 0);
    }
  } catch (err) {
    console.error(err);
  }
}
```

4.4 提交打印任务

代码块

```
export default class NMPrintSocket {
  /**
   * 提交打印任务，并执行回调函数。
   *
   * @param {string} [printData=null] - 打印数据的 JSON 字符串。
   * @param {string} printerImageProcessingInfo - 打印机图像处理信息的 JSON 字符串，包含打印份数信息，格式如下：
   * {
   *   "printerImageProcessingInfo": {
   *     "printQuantity": 1 // 用于指定当前页的打印份数。例如，如果需要打印3页，第一页打印3份，第二页打印2份，第三页打印5份，则在3次提交数据时，printerImageProcessingInfo 中的 "printQuantity" 值分别应为 3，2，5。
   *   }
   * }
   *
   * @return {Promise} 返回一个 Promise，解析为提交打印任务返回信息
   *
   * @description
   * 需要先开启打印任务，完成绘制后再提交打印任务
   */
  commitJob(printData, printerImageProcessingInfo)
}
```

代码块

```
//数据提交成功返回数据示例
{
        "apiName": "commitJob",
        "resultAck": {
                "errorCode": 0,
                "info": "commitJob ok!",
                "result": 0
        }
}

//打印进度返回示例1: 此回调的含义为第一页第一份打印完成
{
        "apiName": "commitJob",
        "resultAck": {
                "errorCode": 0,
                "info": "",
                "onPrintEPCCodeCompleted": "",
                "onPrintPageCompleted": 1,//打印完成份数回调
                "onPrintPageLengthCompleted": "38.00",
                "printQuantity": 1//打印完成页数回调
        }
}

//打印进度返回示例1: 此回调的含义为第一页第二份打印完成
{
        "apiName": "commitJob",
        "resultAck": {
                "errorCode": 0,
                "info": "",
                "onPrintEPCCodeCompleted": "",
                "onPrintPageCompleted": 2,//打印完成份数回调
                "onPrintPageLengthCompleted": "38.00",
                "printQuantity": 1//打印完成页数回调
        }
}

//打印进度返回示例1: 此回调的含义为第二页第一份打印完成
{
        "apiName": "commitJob",
        "resultAck": {
                "errorCode": 0,
                "info": "",
                "onPrintEPCCodeCompleted": "",
                "onPrintPageCompleted": 1,//打印完成份数回调
                "onPrintPageLengthCompleted": "38.00",
                "printQuantity": 2//打印完成页数回调
        }
}


// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
//打印数据长度
const printDataLength = 5;
// 调用流程
async commitJob() {
  const jsonObj = {
    printerImageProcessingInfo: {
      printQuantity: 1,
    },
  };

  try {
    this.nMPrintSocket.commitJob(null, JSON.stringify(this.jsonObj));
  } catch (err) {
    console.error(err);
  }

}
```

4.5 结束打印任务

代码块

```
export default class NMPrintSocket {
  /**
   * 结束打印任务
   *
   * @return {Promise} 返回一个 Promise，解析为结束打印任务的结果
   *
   * @description
   * 收到最后一页最后一份打印页面后调用该函数结束打印任务
   */
  endJob()
}
```

代码块

```
//返回数据示例
{
        "apiName": "endJob",
        "resultAck": {
                "errorCode": 0,
                "info": "endJob ok!",
                "result": 0
        }
}
// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async endJob() {
  try {
    const endRes = await this.nMPrintSocket.endJob();
    if (endRes.resultAck.errorCode == 0) {
      console.log("结束打印完成");
    }
  } catch (err) {
    console.error(err);
  }
}
```

4.6 取消打印任务

代码块

```
export default class NMPrintSocket {
  /**
   * 取消当前的打印任务，并执行回调函数。
   *
   * @return {Promise} 返回一个 Promise，解析为取消打印任务的结果
   */
  cancelJob()
}
```

代码块

```
// 创建打印实例,此实例只需创建一次
this.nMPrintSocket = new NMPrintSocket(socketData);
// 调用流程
async cancelJob() {
  try {
    const cancelJobRes = await this.nMPrintSocket.cancelJob();
    if (cancelJobRes.resultAck.errorCode == 0) {
      console.log("取消打印成功");
    }
  } catch (err) {
    console.error(err);
  }
}
```

五、回调说明

代码块

```

/**
 * {
 *   "apiName": string, // 调用的 API 名称
 *   "resultAck": {
 *     "errorCode": number, // 错误代码，0 表示成功，其他值表示错误
 *     "info": string, // 信息字符串，描述操作结果
 *     "result": number // 结果代码，通常与 errorCode 一致
 *   }
 * }
 */
{
  "apiName": "commitJob",
  "resultAck": {
    "errorCode": 0,
    "info": "commitJob ok!",
    "result": 0
  }
}
```

六、错误码相关说明

6.1 错误码说明描述

代码块

```
* 0-无错误
//打印机返回部分
* 1-盒子打开
* 2-缺纸
* 3-电量不足
* 4-电池异常
* 5-手动停止
* 6-数据错误
* 7-温度过高
* 8-走纸异常
* 9-正在打印
* 10-未检测到打印头
* 11-环境温度过低
* 12-打印头松动
* 13-未检测到碳带
* 14-不匹配的耗材
* 15-用完的碳带
* 16-不支持的纸张类型
* 17-设置纸张类型失败
* 18-设置打印模式失败
* 19-设置浓度失败
* 20-写入rfid失败
* 21-边距参数错误
* 22-超时错误
* 23-断开连接
* 24-画板参数设置错误
```

```
 * 25-旋转角度参数错误
 * 26-json参数错误
 * 27-出纸异常（关闭上盖检测）
 * 28-检查纸张类型
 * 29-碳带与打印模式不匹配
 * 30-设置浓度不支持
 * 31-不支持的打印模式
 * 32-标签材质设置异常，请重新设置
 * 33-不支持该标签材质，请更换或重新设置
 * 34-不支持RFID写入
 * 50-非法标签
 * 51-非法碳带和标签

//内部使用
//E_UNKNOW_ERROR = 255,
```