lpapi-ble-uni

lpapi-ble-uni 是一款基于 UniApp 本身提供的 BLE 接口和 canvas 所封装的标签编辑及蓝牙打印接口，接口通过 canvas 来按照用户需要进行 2D 绘图，绘制完毕后将图片内容转换为打印机所支持的指令，然后通过 BLE 将数据发送到打印机，然后开始打印图片。

备注：本接口仅适用于德佟印立方系列标签打印机！

接口包的获取

1. 通过 npm/yarn 直接下载安装；

```
npm install lpapi-ble-uni
// 或者
yarn add lpapi-ble-uni
```

2. 通过DCloud 插件市场下载:

打开 DCloud 插件市场后，可以搜索插件 **lpapi-ble**，进入后建议用户点击**下载实例项目 ZIP**，然后通过 HBuilderX 打开 demo，然后进行打印测试；

如果需要跟现有项目进行集成，点击 **下载插件并导入 HBuilderX**，然后选择目标项目导入即可，或者直接点击 **下载插件 ZIP**，然后手动将插件包复制到目标项目的适当位置。

3. 通过德佟印立方官网进行下载：

打开德佟印立方官网后，在顶部点击软件下载，然后选择SDK 开发包下载，找到 蓝牙打印 uni-app SDK（含 API 文档），点击下载即可。

使用方法

1. 在 vue 文件中创建一个隐藏的 canvas 组件，canvas 的宽高需要动态指定。

```
<canvas
    :id="canvasId"
    :canvas-id="canvasId"
    :style="{width: labelWidth + 'px', height: labelHeight + 'px'}"
    style="position: fixed;left: -999999rpx;top: -999999rpx;"
/>
```

2. 获取接口实例

获取到接口实例之后才可以正常调用接口实例进行相关的绘制及打印操作。

引入全局变量 **LPAPIFactory**;

```
// 如果是在cli模式下通过 npm或者yarn安装的插件，则通过如下方式引入接口：
import { LPAPIFactory } from "lpapi-ble-uni";
// 如果是通过插件市场安装的插件，则通本地插件路径导入插件；
import { LPAPIFactory } from "@/uni_modules/DothanTech-LPAPI-BLE/js_sdk/index.js";
```

通过 **LPAPIFactory** 获取创建或者获取接口实例对象

```
// 通过 LPAPIFactory.getInstance 来获取接口的单实例对象。
const lpapi = LPAPIFactory.getInstance({
    // 如果需要查看打印相关日志，需要配置 showLog，值为 0 - 4，
    // 0：表示不显示调试信息，1：表示只显示出错信息，2：表示显示警告或者出错信息，
    // 3：表示显示常规的调试信息以及警告或者错误信息，4：表示显示所有调试信息；
    showLog: 4, // 【可选】
});
// 或者通过 LPAPIFactory.createInstance 来创建一个新的接口实例对象，
// 每一次通过该接口创建出来的接口实例都是一个独立的实例对象。
const lpapi = LPAPIFactory.createInstance({
    showLog: 4,
    canvas: canvas,
    canvasId: canvasId,
});
```

```
export default {
    data() {
        return {
            canvasId: "lpapi-ble-uni",
        };
    },
    onLoad() {
        // 1. 获取接口实例，可以在任何地方调用该方法来获取接口实例，接口实例为单例模式，全局唯一。
        this.lpapi = LPAPIFactory.getInstance();
        // 2. 在需要打印的页面中根据指定的 canvas 来创建绘制上下文对象
        //  (微信小程序可以创建离屏Canvas，可以跳过这一步)
        this.context = this.lpapi.createDrawContext({
            // 设置画布ID，第一步中创建的画布ID，全局唯一
            canvasId: this.canvasId, //  【可选】
            // 通过 createSelectorQuery 来获取 canvas 控件。
            // 目前仅支持微信小程序和支付宝小程序;
            canvas: canvas, //  【可选】
        });
        // 3. 判断绘制上下文是否创建成功，如果常见成功，则将绘制上下文设置到lpapi接口中
        if (this.context) {
            this.lpapi.setDrawContext(this.context);
        }
    },
};
```

3. 根据实际需要创建指定大小的打印任务，开始打印或者预览；

```
function printQrcode(data, callback) {
    // 标签宽度，单位毫米
    const labelWidth = 30;
    // 标签高度，单位毫米;
    const labelHeight = 30;
    // 标签转转方向，值为：0，90，180，270，默认为0，表示不做任何的旋转处理;
    const orientation = 0;
    // 二维码外边可以适当的留一点边距;
    const margin = 2;
    // 二维码大小

    const api = this.lpapi;
    // 创建指定大小的打印任务，单位毫米
    const jobInfo = api.startJob({
        width: labelWidth, // 【必填】标签宽度
        height: labelHeight, // 【必填】标签高度
        orientation: orientation, // 【可选】旋转方向
        // "#!#preview#!#" 和 "#!#transparent#!#" 是两个特殊的打印任务名称，
        // 主要供生成预览图片或者调试使用，不会打印，其他任何字符串都会直接打印;
        jobName: "#!#preview#!#", // 【可选】
    });
    // 更新画布大小，单位像素
    this.labelWidth = jobInfo.canvas.width;
    this.labelHeight = jobInfo.canvas.height;
    // 注意，在uni模式下更新canvas画布大小后，画布大小不会立即生效，
    //       需要等待画布尺寸更新之后在进行后续操作才可以，一般情况下100毫秒足够了。
    setTimeout(() => {
        // 绘制二维码
        api.draw2DQRCode({
            text: "测试测试测试测试测试测试",
            x: margin,
            y: margin,
            width: labelWidth - margin * 2,
            height: labelHeight - margin * 2,
        });
        //
        api.commitJob({
            complete: (resp) => {
                // 执行打印完毕回调函数
                if (typeof callback === "function") {
                    callback(true);
                }
            },
        });
    }, 100);
}
```

接口介绍

1. LPAPIFactory

```
interface InitOptions {
    /** 用于标签绘制的 canvas ID */
    canvasId: string;
    /** Canvas 控件实例对象，可通过 createSelectorQuery 来获取 */
    canvas: any;
    /** 是否显示相关日志信息 */
    logLevel?: number;
}

export interface LPAPIFactory {
    /**
     * 通过配置信息获取接口实例对象。
     */
    static getInstance(context: InitOptions): LPAPI;
    /**
     * 创建一个新的 LPAPI 实例对象。
     */
    static createInstance(context: InitOptions): LPAPI;
}
```

2. LPA_Result

相关错误代码

```
/**
 * 数据打印执行结果。
 */
export enum LPA_Result {
    /** 异步等待中 */
    ASYNC_WAIT = -1,
    /** 打印成功 */
    OK = 0,
    /** 参数错误 */
    ERROR_PARAM = 1,
    /** 未检测到打印机或者未指定打印机 */
    ERROR_NO_PRINTER = 2,
    /** 打印机未连接 */
    ERROR_DISCONNECTED = 3,
    /** 打印机链接失败 */
    ERROR_CONNECT_FAILED = 4,
    /** 数据Notify特征值启动失败 */
    ERROR_START_NOTIFICATION = 5,
    /** 数据发送失败 */
    ERROR_DATA_SEND_ERROR = 6,
    /** 数据接收异常，打印机无响应 */
    ERROR_DATA_RECEIVE_ERROR = 7,
    /** 打印机正在打印过程中不能打印其他标签 */
    ERROR_IS_PRINTING = 8,
    /** 指令发送响应超时 */
    ERROR_RESPONSE_TIMEOUT = 9,
    /** 打印任务创建失败 */
    ERROR_JOB_CREATE = 16,
    /** 打印任务被取消 */
    ERROR_JOB_CANCELED = 17,
    /** 打印数据获取失败 */
    ERROR_GET_IMAGE_DATA = 18,
    /** 打印机状态异常*/
    ERROR_PRINTER_NOT_AVAILABLE = 19,
    /** 其他未知异常 */
    ERROR_OTHER = 32
}
```

3. 相关打印参数

```
/**
 * 纸张间隔类型。
 */
export enum LPA_GapType {
    /** 随打印机设置 */
    Unset = 255,
    /** 连续纸（小票纸） */
    None = 0,
    /**
     * 定位孔
     * @deprecated
     */
    Hole = 1,
    /** 间隙纸（不干胶） */
    Gap = 2,
    /** 黑标卡纸 */
    Black = 3,
    /** 黑标透明贴 */
    Trans = 4,
}
/**
 * 打印速度常用值。
 *
 * 实际有效值为1到5之间，其他表示随打印机设置。
 */
export enum LPA_PrintSpeed {
    /** 随打印机设置 */
    Unset = 255,
    /** 最慢 */
    Min = 1,
    /** 较慢 */
    Low = 2,
    /** 正常速度 */
    Normal = 3,
    /** 较块 */
    High = 4,
    /** 最快 */
    Max = 5,
}
/**
 * 打印浓度常用枚举值。
 *
 * 打印浓度可以1到15之间的任意值，其他表示随打印机设置。
 */
export enum LPA_PrintDarkness {
    /** 随打印机设置 */
    Unset = 255,
```

```
    /** 最淡 */
    Min = 1,
    /** 较淡 */
    Low = 4,
    /** 正常浓度 */
    Normal = 6,
    /** 较浓 */
    High = 10,
    /** 最浓 */
    Max = 15,
}
```

4. LPAPI 接口函数

```
interface LPAPI {
    /**
     * 获取 LPAPI 接口单实例。
     *
     * @param printer 外部提供的打印机设备接口。
     * @returns {LPAPI} LPAPI 接口实例对象。
     */
    static getInstance(options?: LPA_InitOptions): LPAPI;
    /**
     * 创建一个新的 {@link LPAPI} 实例对象。
     */
    static create(options: LPA_InitOptions): LPAPI;
    /**
     * 判断给定的打印任务名称是不是预览任务。
     * @param jobName 目标打印任务名称。
     */
    static isPreviewJob(jobName?: string): boolean;
    /**
     * 判断给定的打印任务名称是不是透明色的预览任务。
     */
    static isTransPrevJob(jobName?: string): boolean;
    /**
     * 判断给定的打印任务名称是否是白底预览任务。
     */
    static isWhitePrevJob(jobName?: string): boolean;
    /**
     * 从目标打印机名称中获取对应的打印机型号。
     */
    static getModelName(name: string): string[];
    /**
     * 判断给定的设备名称是不是接口所支持的打印机设备。
     */
    static isSupportedDevice(name: string): boolean;
    /**
     * 异步加载图片 URL。
     * 图片的绘制需要加载完毕之后才可以进行canvas绘制，否则可能会不显示。
     */
    static loadImageSrc(src: string): Promise<HTMLImageElement | null>;
    static loadHtmlImage(html: string): Promise<HTMLImageElement | null>;
    protected constructor(options: LPA_InitOptions);
    /**
     * 获取当前接口的蓝牙适配器实例。
     */
    get BleAdapter(): IBleAdapter;
    /**
     * 修改接口的蓝牙适配器。
     */
```

```
set BleAdapter(adapter: IBleAdapter);
/**
 * 获取接口的绘制上下文环境。
 */
get Context(): DrawContext;
/**
 * 设置标签绘制的绘制上下文对象。
 */
setDrawContext(value: DrawContext): void;
/**
 * 创建绘制上下文环境。
 * 本接口主要是针对 UniApp 开发环境的，因为在 UniApp 中不支持离屏canvas，在不同的页面中接口所关联的绘制上下文不一
 */
createDrawContext(options: LPA_CreateContextOptions): DrawContext | undefined;
/**
 * 根据给定的错误代码，返回对应的错误描述字符串。
 * @param status 操作失败时的错误代码。
 * @returns 返回给定错误代码对应的描述字符串。
 */
getResultMessage(status: LPA_Result): string;
/**
 * 根据给定的打印机状态码返回对应的描述字符串。
 * @param printable 打印机的状态码，不指定标签当前已连接打印机的状态码。
 * @returns 根据给定的打印机状态码，返回对应的描述字符串。
 */
getPrinterStatus(printable?: number): string;
/**
 * 异步加载图片 URL。
 * 图片的绘制需要加载完毕之后才可以进行canvas绘制，否则可能会不显示。
 */
loadImage(src: string, callback?: (image: HTMLImageElement | null) => void): Promise<HTMLImageElement | n
loadHtml(html: string | HTMLElement): Promise<HTMLImageElement | null>;
/**
 * 设置支持的打印机型号列表。
 * @param models 接口所支持的打印机型号列表，多个型号可以通过";"进行分割。
 */
setSupportPrefixes(models: string | string[]): void;
setSupportTrades(trades: string | string[]): void;
private updateDeviceList;
/**
 * 开始搜索打印机设备。
 *
 * @param {LPA_BleDiscoveryOptions|undefined} options 蓝牙打印机搜索相关现象;
 * @param {string|undefined} options.models 只搜索指定型号的打印机设备;
 * @param {number|undefined} options.timeout 搜索超时时间,
 *      undefined : 表示一直搜索，当用户调用 {@link stopBleDiscovery} 接口的时候或者链接打印机的时候，自动停止;
 *      0         : 表示搜索到目标打印机后立即停止搜索;
 *      number    : 用户指定的超时时间到了之后，自动停止搜索;
 * @param {number|undefined} interval 设备搜索上报的时间间隔;
 * @param {(devices: LPA_BleDevice[]) => void} deviceFound 检测到设备时的回调函数;
```

```
 * @param {(result: LPA_BleAdapterStateChangeResult) => void} adapterStateChange 搜索状态变化时的回调函数;
 * @param {(resp: LPA_Response<any>) => void} success 搜索蓝牙设备启动成功时的回调函数;
 * @param {(resp: LPA_Response<any>) => void} fail 搜索蓝牙设备启动失败时的回调函数;
 * @param {(resp: LPA_Response<any>) => void} complete 搜索蓝牙设备启动成功或失败时的回调函数;
 *
 * @returns {Promise<LPA_Response<any>>} 等同于 complete回调函数。
 */
startBleDiscovery(options?: LPA_BleDiscoveryOptions): Promise<LPA_Response<any>>;
/**
 * 停止蓝牙设备搜索操作。
 */
stopBleDiscovery(): Promise<any>;
/**
 * 获取缓存的打印机列表。
 *
 * @return {LPA_Device[]} 返回打印机设备列表。
 */
getPrinters(): LPA_BleDevice[];
/**
 * 获取已连接的打印机名称。
 */
getPrinterInfo(): IPrinterInfo | undefined;
/**
 * 判断打印机是否已打开。
 */
isPrinterOpened(): boolean;
/**
 * 打开指定的打印机。
 *
 * @param {LPA_OpenDeviceOptions|string|function|undefined} options 打印机链接相关参数;
 *
 * @param {string|undefined} options.name 目标设备名称;
 * @param {string|undefined} options.deviceId 目标设备ID;
 * @param {(resp: LPA_BleConnectionCloseOptions) => void} options.connectionStateChange 打印机断开时的回调函
 *
 * @return {Promise<LPA_Response<any>>} 设备链接结果信息。
 */
openPrinter(options?: LPA_OpenDeviceOptions | string | ((resp: LPA_Response<any>) => void)): Promise<LPA_
/**
 * 关闭已经打开的打印机。
 *
 * @info 关闭打印机时，当前还有未打印的任务/数据将会被自动提交打印，同时所有参数设置将会被保留。
 */
closePrinter(delay?: number, callback?: (res: boolean) => void): Promise<boolean>;
/**
 * 创建打印任务。
 *
 * 创建打印任务时，如果没有链接打印机，则本函数会自动打开当前系统安装的第一个 LPAPI 支持的打印机，用于打印。
 * 当前还有未打印的任务，已有打印数据将会被全部丢弃。
 *
```

```
 * @param {LPA_JobStartOptions} options 标签任务选项。
 *
 * @param {number} options.width 标签宽度，单位毫米，值默认为{@link CONSTANTS.LABEL_WIDTH}。
 * @param {number} options.height 标签高度，单位毫米，值默认为{@link CONSTANTS.LABEL_HEIGHT}。
 * @param {0|90|180|270} options.orientation 标签打印方向，`0`表示不旋转，`90`表示右转90度，`180`表示180度旋转
 * @param {string|undefined} options.jobName 打印任务名称，特殊情况下的任务不进行打印，可用于生成对应的预览图片
 *          预览时的值可参考:{@link LPA_JobNames}，默认为{@link LPA_JobNames.Print}，表示打印任务。
 * @param {number} options.dpi 打印头分辨率（值默认为203）。
 * @param {HTMLImageElement | undefined} options.backgroundImage 背景图，预览模式下有效。
 * @param {string|undefined} options.backgroundColor 背景色，预览模式下有效。
 * @param {number|undefined} options.printPages 打印页面的个数。在分页打印的时候用于指定总的打印页数，方便处理打
 *
 * @returns {JobStartResult|undefined} 返回打印任务及画布相关信息。
 */
startJob(options: LPA_JobStartOptions): JobStartResult | undefined;
/**
 * 创建打印任务，在打印任务中：
 *      如果制定了打印机相关信息，那么会自动链接打印机；
 *      如果未指定打印机，则返回错误信息；
 *
 * @param {number} options.width 标签宽度，单位毫米，值默认为{@link CONSTANTS.LABEL_WIDTH}。
 * @param {number} options.height 标签高度，单位毫米，值默认为{@link CONSTANTS.LABEL_HEIGHT}。
 * @param {0|90|180|270} options.orientation 标签打印方向，`0`表示不旋转，`90`表示右转90度，`180`表示180度旋转
 * @param {string|undefined} options.jobName 打印任务名称，特殊情况下的任务不进行打印，可用于生成对应的预览图片
 *          预览时的值可参考:{@link LPA_JobNames}，默认为{@link LPA_JobNames.Print}，表示打印任务。
 * @param {string|undefined} options.printerName 目标打印机名称；
 * @param {string|undefined} options.deviceId 目标设备ID；
 * @param {number|undefined} options.printPages 打印页面的个数。在分页打印的时候用于指定总的打印页数，方便处理打
 * @returns
 */
startPrintJob(options: LPA_StartPrintJobOptions, callback?: (res: LPA_JobStartResult) => void): Promise<LI
/**
 * 终止当前打印任务，允许重新创建新的打印任务。
 */
abortJob(): void;
/**
 * 获取当前打印任务的预览图片。
 */
toDataURL(): string;
/**
 * 取消后续打印任务。
 */
cancelJob(): void;
/**
 * 提交打印任务，进行真正的打印。
 *
 * @param {LPA_JobCommitOptions|undefined} options 相关打印参数。
 *
 * @param {number|undefined} options.threshold 图片进行黑白转换时的阈值，默认为{@link CONSTANTS.THRESHOLD}，
 * @param {LPA_PrintSpeed|undefined} options.speed 打印速度，默认随打印机设置。
```

```
 * @param {LPA_PrintDarkness|undefined} options.darkness 打印浓度，默认随打印机设置。
 * @param {LPA_GapType|undefined} options.gapType 纸张类型，默认随打印机设置。
 * @param {(resp: IResponse<string>) => void} options.success 打印任务处理成功回调函数;
 * @param {(resp: IResponse<string>) => void} options.fail 打印任务处理失败回调函数;
 * @param {(resp: IResponse<string>) => void} options.complete 打印任务处理完毕回调函数;
 */
commitJob(options?: LPA_JobCommitOptions): Promise<LPA_JobPrintResult>;
/**********************************************************************
 * 绘制相关内容。
 **********************************************************************/
/**
 * 设置后续绘制内容的默认对齐方式。
 *      0: 水平靠左对齐;
 *      1: 水平居中对齐;
 *      2: 水平靠右对齐;
 *      3: 水平拉伸对齐;
 */
setItemHorizontalAlignment(alignment: Alignment): void;
/**
 * 设置后续绘制内容的默认垂直对齐方式。
 *      0: 垂直靠上对齐;
 *      1: 垂直居中对齐;
 *      2: 垂直靠下对齐;
 *      3: 垂直拉伸对齐;
 */
setItemVerticalAlignment(alignment: Alignment): void;
/**
 * 设置后续绘制内容的默认旋转角度。
 *      0  : 不旋转;
 *      90 : 右转90度;
 *      180: 旋转180度;
 *      270: 左转90度;
 */
setItemOrientation(orientation: number): void;
/**
 * 绘制文本。
 *
 * regionCorners regionLeftUpCorner regionRightUpCorner regionRightBottomCorner
 * regionLeftBottomCorner regionLeftBorders regionRightBorders，这些参数都是长度
 * 数组，建议都是通过数组来传递参数，这样接口会对长度都自动转发为接口使用的 0.01mm 的
 * 单位。为了调试方便，这些参数也支持逗号分隔的字符串方式来参数。但是此时参数必须调用者
 * 自己转发为 0.01mm 为单位的长度数据。
 *
 * @param {DrawTextOptions} options 文本绘制相关选项。
 *
 * @param {string|string[]} options.text 待绘制的文本数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。值默认为0。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。值默认为0。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米。
 *          值默认为0，表示绘制宽度不做限制。
```

```
 * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米。
 *          值默认为0，表示高度不做显示，以实际高度显示。
 * @param {string|undefined} options.fontName 绘制对象的字体名称，值默认为{@link CONSTANTS.FONT_NAME}。
 * @param {number} options.fontHeight 绘制对象的字体高度，单位毫米，
 *          值默认为{@link CONSTANTS.FONT_HEIGHT}。
 * @param {LPA_FontStyle|undefined} options.fontStyle 字体样式。
 * @param {LPA_AutoReturnMode|undefined} options.autoReturn 自动换行模式，默认为{@link LPA_AutoReturnMode.C
 *          {@link LPA_AutoReturnMode.None}：没有自动换行；
 *          {@link LPA_AutoReturnMode.Char}：按字换行；
 *          {@link LPA_AutoReturnMode.Word}：按词换行。
 * @param {number|undefined} options.charSpace 字符间距，默认为0，单位毫米。
 * @param {number|string|undefined} options.lineSpace 行间距，单位毫米，
 *          或为枚举字符串（1_0，1_2，1_5，2_0）。默认为 1_0，也即单倍行距。
 *          格式为：`[Width，Y，Height，Width，Y，Height]`，单位毫米。
 * @param {number[]|string|undefined} regionRightBorders 显示区域右边的删除矩形，最多支持删除两个矩形，
 *          格式为：`[Width，Y，Height，Width，Y，Height]`，单位毫米。
 * @param {boolean|undefined} onlyMeasureText 表示仅仅度量、而不真正的绘制文本。
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
drawText(options: DrawTextOptions): void;
/**
 * 打印一维条码。
 *
 * @param {DrawBarcodeOptions} options 一维码绘制相关选项。
 *
 * @param {string} options.text 待绘制的一维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。值默认为0。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。值默认为0。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米。
 *          值默认为0，表示根据 {@link barPixels} 设定的点的大小自动计算对象宽度。
 * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米。
 *          值默认为0，表示根据 {@link barPixels} 设定的点的大小自动计算对象宽度。
 * @param {number|undefined} options.textHeight 一维码中供人识读文本的高度，单位毫米，
 *          值默认为0，表示不显示一维码下面的字符串。
 * @param {LPA_BarcodeType|undefined} options.type 一维码类型，默认为{@link LPA_BarcodeType.LPA_1DBT_AUTO},
 * @param {string|undefined} options.fontName 一维码中供人识读文本的字体名称，默认为{@link CONSTANTS.FONT_NAM
 * @param {LPA_FontStyle|undefined} options.fontStyle 一维码供人识读文本的字体风格，默认为{@link LPA_FontStyl
 * @param {LPA_ItemAlignment|undefined} options.textAlignment 一维码供人识读文本的水平对齐方式，值参考{@link
 *          >= 5 表示表示跟随一维码本身的水平对齐方式，默认为{@link LPA_ItemAlignment.Center}，也即居中对齐。
 * @param {LPA_BarcodeFlags|undefined} options.barcodeFlags 一维码编码参数标志，值参考{@link LPA_BarcodeFlag
 * @param {number} options.barPixels 在不指定一维码宽度的情况下，一维码中每个逻辑点的像素大小，单位像素，值为 1
 * @param {number|undefined} options.textBarSpace 一维码供人识读文本和条码的垂直间距，单位毫米，默认为约2个像素
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
```

```
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为: {@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
drawBarcode(options: Draw1DBarcodeOptions): void;
/**
 * 打印一维条码。
 *
 * @param {DrawBarcodeOptions} options 一维码绘制相关选项。
 *
 * @param {string} options.text 待绘制的一维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。值默认为0。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。值默认为0。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米。
 *          值默认为0，表示根据 {@link barPixels} 设定的点的大小自动计算对象宽度。
 * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米。
 *          值默认为0，表示根据 {@link barPixels} 设定的点的大小自动计算对象宽度。
 * @param {number|undefined} options.textHeight 一维码中供人识读文本的高度，单位毫米，
 *          值默认为0，表示不显示一维码下面的字符串。
 * @param {LPA_BarcodeType|undefined} options.type 一维码类型，默认为{@link LPA_BarcodeType.LPA_1DBT_AUTO},
 * @param {string|undefined} options.fontName 一维码中供人识读文本的字体名称，默认为{@link CONSTANTS.FONT_NAM
 * @param {LPA_FontStyle|undefined} options.fontStyle 一维码供人识读文本的字体风格，默认为{@link LPA_FontStyl
 * @param {LPA_ItemAlignment|undefined} options.textAlignment 一维码供人识读文本的水平对齐方式，值参考{@link
 *          >= 5 表示表示跟随一维码本身的水平对齐方式，默认为{@link LPA_ItemAlignment.Center}，也即居中对齐。
 * @param {LPA_BarcodeFlags|undefined} options.barcodeFlags 一维码编码参数标志，值参考{@link LPA_BarcodeFlag
 * @param {number} options.barPixels 在不指定一维码宽度的情况下，一维码中每个逻辑点的像素大小，单位像素，值为 1
 * @param {number|undefined} options.textBarSpace 一维码供人识读文本和条码的垂直间距，单位毫米，默认为约2个像素
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为: {@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
draw1DBarcode(options: Draw1DBarcodeOptions): void;
/**
 * 打印 QrCode 二维码。
 *
 * @param {DrawQrcodeOptions} options QRCode二维码绘制相关参数。
 *
 * @param {string} options.text 待绘制的二维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米
 *          值默认为0，表示根据 {@link qrcPixels} 设定的点的大小自动计算二维码大小。
 * @param {number|undefined} options.height 绘制对象的显示高度，不指定表示按照: {@link width} 来显示，单位毫米
 * @param {number|number} options.version 二维码编码最小版本号，1~40，默认为根据内容自动计算。
 * @param {LPA_QREccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_QREccLevel}，默认为{@
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
```

```
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
drawQRCode(options: DrawQRCodeOptions): void;
/**
 * 打印 QrCode 二维码。
 *
 * @param {DrawQrcodeOptions} options QRCode二维码绘制相关参数。
 *
 * @param {string} options.text 待绘制的二维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米
 *          值默认为0，表示根据 {@link qrcPixels} 设定的点的大小自动计算二维码大小。
 * @param {number|undefined} options.height 绘制对象的显示高度，不指定表示按照：{@link width} 来显示，单位毫米
 * @param {number|number} options.version 二维码编码最小版本号，1~40，默认为根据内容自动计算。
 * @param {LPA_QREccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_QREccLevel}，默认为{@
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
draw2DQRCode(options: DrawQRCodeOptions): void;
/**
 * 打印 Pdf417 二维码。
 *
 * @param {DrawPdf417Options} options PDF417二维码绘制选项。
 *
 * @param {string} options.text 待绘制的PDF417二维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米
 *          值默认为0，表示根据 {@link p417Pixels} 设置的大小自动计算二维码宽度。
 * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米
 *          值默认为0，表示根据 {@link p417Pixels} 设置的大小自动计算二维码高度。
 * @param {LPA_P417EccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_P417EccLevel}，默认
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
drawPDF417(options: DrawPDF417Options): void;
/**
 * 打印 Pdf417 二维码。
```

```
 *
 * @param {DrawPdf417Options} options PDF417二维码绘制选项。
 *
 * @param {string} options.text 待绘制的PDF417二维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米
 *          值默认为0，表示根据 {@link p417Pixels} 设置的大小自动计算二维码宽度。
 * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米
 *          值默认为0，表示根据 {@link p417Pixels} 设置的大小自动计算二维码高度。
 * @param {LPA_P417EccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_P417EccLevel}，默认
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
 */
draw2DPdf417(options: DrawPDF417Options): void;
/**
 * 打印 DataMatrix 二维码。
 *
 * @param {DrawDataMatrixOptions} options DataMatrix 二维码绘制选项。
 *
 * @param {string} options.text 待绘制的二维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米
 *          值默认为0，表示根据 {@link dmtxPixels} 设定的点的大小自动计算二维码大小。
 * @param {number|undefined} options.height 绘制对象的显示高度，不指定表示按照：{@link width} 来显示，单位毫米
 * @param {number|number} options.symbolShape
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270，默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
 * @returns 成功与否。
 */
drawDataMatrix(options: DrawDataMatrixOptions): void;
/**
 * 打印 DataMatrix 二维码。
 *
 * @param {DrawDataMatrixOptions} options DataMatrix 二维码绘制选项。
 *
 * @param {string} options.text 待绘制的二维码数据。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米
 *          值默认为0，表示根据 {@link dmtxPixels} 设定的点的大小自动计算二维码大小。
 * @param {number|undefined} options.height 绘制对象的显示高度，不指定表示按照：{@link width} 来显示，单位毫米
```

```
 * @param {number|number} options.symbolShape
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270，默认为0，表示不旋转。
 * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 水平对齐方式。不指定表示使用 {@link setI
 *          默认为{@link LPA_ItemAlignment.Start}，表示居左对齐。
 * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 垂直对齐方式。不指定表示使用 {@link setIte
 *          默认为: {@link LPA_ItemAlignment.Start}，表示居上对齐。
 * @returns 成功与否。
 */
draw2DDataMatrix(options: DrawDataMatrixOptions): void;
/**
 * 绘制矩形框。
 *
 * @param {DrawRectOptions} options 矩形框绘制相关选项。
 *
 * @param {number|undefined} options.x 矩形的水平位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y 矩形的垂直位置，单位毫米，值默认为0。
 * @param {number|undefined} options.width 矩形的水平宽度，单位毫米，默认为{@link CONSTANTS.RECT_WIDTH}。
 * @param {number|undefined} options.height 矩形的垂直高度，单位毫米，值默认与宽度相同。
 * @param {number|undefined} options.cornerWidth 矩形的圆角宽度，单位毫米，值默认为0。
 * @param {number|undefined} options.cornerHeight 矩形的圆角高度，单位毫米，值默认为0。
 * @param {number|undefined} options.lineWidth 圆角矩形的线宽，单位毫米，值默认为{@link CONSTANTS.LINE_WIDTH
 * @param {boolean|undefined} options.fill 是否绘制填充圆角矩形，值默认为false，表示显示矩形边框。
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 */
drawRectangle(options: DrawRectOptions): boolean | void;
drawRect(options: DrawRectOptions): boolean | void;
/**
 * 绘制椭圆边框。
 *
 * @param {DrawRectOptions} 椭圆绘制相关选项。
 *
 * @param {number|undefined} options.x 椭圆的水平位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y 椭圆的垂直位置，单位毫米，值默认为0。
 * @param {number|undefined} options.width 椭圆的水平宽度，单位毫米，值默认为{@link CONSTANTS.RECT_WIDTH}。
 * @param {number|undefined} options.height 椭圆的垂直高度，单位毫米，值默认与宽度相同。
 * @param {number|undefined} options.lineWidth 椭圆的线宽，单位毫米，值默认为{@link CONSTANTS.LINE_WIDTH}。
 * @param {boolean|undefined} options.fill 是否绘制填充椭圆，默认为false，表示绘制椭圆边框。
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 */
drawEllipse(options: DrawBorderOptions): boolean;
/**
 * 绘制圆形。
 *
 * @param {DrawCircleOptions} options 圆形绘制相关参数。
 *
 * @param {number|undefined} options.x 水平方向上的圆心坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y 垂直方向上的圆心坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.radius 圆形半径，单位毫米，值默认为{@link CONSTANTS.RADIUS}。
```

```
 * @param {number|undefined} options.lineWidth 圆形边框宽度，单位毫米，值默认为{@link CONSTANTS.LINE_WIDTH}。
 * @param {boolean|undefined} options.fill 是否绘制填充圆形，默认为false，表示只绘制圆形边框。
 */
drawCircle(options: DrawCircleOptions): boolean;
/**
 * 绘制直线。
 *
 * @param {DrawLineOptions} options 直线绘制相关选项。
 *
 * @param {number|undefined} options.x1 点划线起点位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y1 点划线起点位置，单位毫米，值默认为0。
 * @param {number|undefined} options.x2 点划线终点位置，单位毫米，值默认等于x1。
 * @param {number|undefined} options.y2 点划线终点位置，单位毫米，值默认等于y1。
 * @param {number|undefined} options.lineWidth lineWidth: 直线线宽，单位毫米，值默认为{@link CONSTANTS.lineW
 * @param {number[]|undefined} options.dashLens 点化线线段长度的数组。
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 */
drawLine(options: DrawLineOptions): void;
/**
 *  打印指定的URL图片。
 *
 * @param {DrawImageUrlOptions} options URL图片绘制相关选项。
 *
 * @param {CanvasImageSource} options.img 图片对象。
 * @param {number|undefined} options.x 绘制对象的水平坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.y 绘制对象的垂直坐标位置，单位毫米，值默认为0。
 * @param {number|undefined} options.width 绘制对象的显示宽度，单位毫米，值默认为0，表示图片的实际大小。
 * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米，值默认为0，表示图片的实际大小。
 * @param {number|undefined} options.threshold 图片黑白打印的灰度阈值。
 *          0 表示使用参数设置中的值;
 *          256 表示取消黑白打印，用灰度打印;
 *          257 表示直接打印图片原来的颜色。
 * @param {0|90|180|270|undefined} options.orientation 旋转角度，0、90、180、270。
 *          不指定表示使用 {@link setItemOrientation()} 设置的参数。默认为0，表示不旋转。
 *
 * @info        如果之前没有调用 StartPage 而直接进行打印，则打印函数会自动调用 StartPage开始一打印页面，然后进行
 * @info        打印位置和宽度高度是基于当前页面的位置和方向，不考虑页面和打印动作的旋转角度。
 * @info        图片打印时会被缩放到指定的宽度和高度。
 * @info        标签打印都是黑白打印，因此位图会被转变成灰度图片（RGB三分量相同，0～255取值的颜色）之后，然后根据
 *              默认灰度阀值为 192，也就是说 >= 192 的会被认为是白色，而 < 192 的会被认为是黑色。
 */
drawImage(options: DrawImageOptions): Promise<boolean>;
/**
 * @deprecated 请使用 drawImage 替代函数。
 */
drawImagePath(options: DrawImageOptions): Promise<boolean>;
drawHtml(options: DrawHtmlOptions): Promise<boolean>;
/**
 *
```

```
 * @param {DrawTableOptions} options 表格绘制相关参数。
 *
 * @param {number} options.x 表格在画布X轴方向上的位置（单位毫米，值默认为0）。
 * @param {number} options.y 表格在画布Y轴方向上的位置（单位毫米，值默认为0）。
 * @param {number} options.width 表格的宽度（单位毫米，值默认为标签宽度）。
 * @param {number} options.height 表格的高度（单位毫米，值默认为标签高度）。
 * @param {number} options.lineWidth 表格线条的宽度，单位毫米，默认位0.35。当线条宽度小于等于0的时候不显示表格的
 * @param {TableRow[]} options.rows 表格的单元格内容的行列表，列表中的每一项表示表格中每一行的单元格列表。在用ro
 *      可以不用指定表格的 rowCount 和表格的 columnCount，单元格的行列数可以通过该属性的长度以及该属性中每一条数据
 * @param {TableCell[]} options.cells 表格单元格列表，列表长度为表格的行数乘以表格的列数[rowCount * columnCoun
 * @param {number} options.rowCount 表格的行数，值默认为 tableRows的行数。
 * @param {number} options.columnCount 表格的列数，值默认为 tableRows 中最大的值。
 * @param {number[]} options.rowHeights 单元格行高列表，
 *          当数组中的值大于等于1的时候，数组中的值为单元格的固定高度，单位毫米；
 *          当数组中的值未指定，或者值大于0并且小于1的时候，表示当前行单元格所占表格剩余高度的百分比。
 * @param {number[]} options.columnWidths 单元格列宽列表，原理等同于 rowHeights，
 *          当数组中的值大于等于1的时候，数组中的值为单元格的固定宽度，单位毫米；
 *          当数组中的值未指定，或者值大于0并且小于1的时候，表示当前行单元格所占表格剩余宽度的百分比。
 * @param {Rect[]} options.groups 单元格合并信息。Rect中的属性描述如下：
 *          Rect.x：待合并的目标单元格的列索引，从0开始。
 *          Rect.y：待合并的目标大院个的行索引，从0开始。
 *          Rect.width ：从目标单元格开始，要合并的单元格的列数，值大于等于1，1表示列方向上没有要合并的单元格。
 *          Rect.height：从目标单元格开始，要合并的单元格的行数，值大于等于1，1表示行方向上没有要合并的单元格。
 *      备注：单元格的合并信息也可以通过单元格内容中的 rowSpan 和 ColumnSpan 来指定。
 *
 * @interface TableCell extends DrawItemOptions {
 *      type: DrawType;         // 单元格类型
 *      rowSpan?: number;       // 占用单元格的行数
 *      columnSpan?: number;    // 占用单元格的列数
 * }
 */
drawTable(options: DrawTableOptions): boolean;
/**
 * 直接打印指定位图对象。
 *
 * @param {LPA_ImagePrintOptions} options 图片打印相关选项。
 *
 * @param {HTMLImageElement|string|undefined} options.image HTMLImageElement格式的图片对象，或者BASE64格式的
 * @param {number|undefined} options.width 图片打印区域宽度，单位毫米，值默认为0，表示按照实际大小来打印。
 * @param {number|undefined} options.height 图片打印区域高度，单位毫米，值默认为0，表示按照实际大小来打印。
 * @param {number|undefined} options.threshold 图片进行黑白转换时的阈值，默认为{@link CONSTANTS.THRESHOLD}。
 * @param {number|undefined} options.orientation 图片打印方向，默认为0，表示打印前不进行图片的旋转操作。
 * @param {number|undefined} options.copies 打印份数，默认只打印1份。
 * @param {string|undefined} options.jobName 打印任务名称。
 * @param {(res: LPA_Result) => void} options.success 打印成功回调函数。
 * @param {(res: LPA_Result) => void} options.fail 打印失败回调函数。
 * @param {(res: LPA_Result) => void} options.complete 打印完毕回调函数。
 */
printImage(options: LPA_ImagePrintOptions): Promise<LPA_JobPrintResult>;
/**
```

```
 * 将打印输入添加到打印队列中。
 *
 * @param {LPA_ImageDataPrintOptions} options ImageData 打印相关参数
 *
 * @param {ImageData} options.imageData ImageData格式的二进制数据。
 * @param {ArrayBuffer|string|undefined} data 图片二进制数据，或者图片数据的十六进制字符串。
 * @param {number|undefined} width 图片的像素宽度，配合 data 属性一块使用，可以替代 imageData属性。
 * @param {number|undefined} height 图片的像素高度，配合 data 属性一块使用，可以替代 imageData属性。
 * @param {LPA_PrintDarkness} options.printDarkness 打印浓度。
 * @param {LPA_PrintSpeed} options.printSpeed 打印速度。
 * @param {LPA_GapType} options.gapType 纸张类型。
 * @param {number} options.threshold 图片处理灰度阈值。
 * @param {(res: LPA_Result) => void} options.success 打印成功回调函数。
 * @param {(res: LPA_Result) => void} options.fail 打印失败回调函数。
 * @param {(res: LPA_Result) => void} options.complete 打印完毕回调函数。
 * @returns 打印结果成功或着错误编码。
 */
printImageData(options: LPA_ImageDataPrintOptions): Promise<LPA_Result>;
/**
 * @param options JSON 打印相关配置信息。
 *
 * @param {number} options.jobInfo.jobWidth 打印任务宽度，单位毫米。
 * @param {number} options.jobInfo.jobHeight 打印任务高度，单位毫米。
 * @param {number|undefined} options.jobInfo.orientation 打印任务旋转角度。
 * @param {string|undefined} options.jobInfo.jobName 打印任务名称。
 * @param {number|undefined} options.jobInfo.gapType 设置打印纸张类型。
 * @param {number|undefined} options.jobInfo.printDarkness 设置打印浓度
 * @param {number|undefined} options.jobInfo.printSpeed 设置打印速度。
 * @param {number|undefined} options.jobInfo.threshold 设置打印时的灰度阈值。
 * @param {boolean|undefined} options.jobInfo.antiColor 是否进行反色打印。
 * @param {boolean|undefined} options.jobInfo.horizontalFlip 是否需要进行水平镜像翻转打印。
 *
 * @param {string|undefined} options.printerInfo.printerName 打印机名称。
 * @param {string|undefined} options.printerInfo.deviceId 打印机设备ID。
 * @param {number|undefined} options.printerInfo.printerDPI 打印机分辨率。
 *
 * @param {DrawPageItems[]|undefined} options.jobPages 打印页面列表，在一个打印任务中可以同时打印多张标签。
 * @param {DrawPageItems|undefined} options.jobPage 打印页面信息，该参数通常只打印一张标签，如果需要批量打印，可
 *      备注：jobPage 与 jobPages 其中一个必须要设置，否则本次打印为无效的打印。
 * @param {Record<string, any>[]|undefined} options.jobArguments 打印参数，用于进行批量打印。可以通过 jobPage
 * @returns
 */
print(options: LPA_JsonPrintOption): Promise<LPA_JobPrintResult>;
/**
 * 解析并打印 wdfx 格式的文件内容。
 * @param {LPA_JsonPrintOption} options wdfx文件解析与打印相关配置参数。
 *
 * @param {string} options.content wdfx格式的字符串内容。
 * @param {JobExtraInfo} options.jobInfo 打印任务相关参数。
 * @param {string|undefined} options.jobInfo.jobName 打印任务名称。
```

```
     * @param {number|undefined} options.jobInfo.gapType 标签纸类型。
     * @param {number|undefined} options.jobInfo.printSpeed 打印速度。
     * @param {number|undefined} options.jobInfo.printDarkness 打印浓度。
     * @param {number|undefined} options.jobInfo.orientation 打印任务出纸方向。
     * @param {PrinterInfo} options.printerInfo 打印机相关参数。
     * @param {string|undefined} options.printerInfo.printerName 打印机名称，在打印模式下会自动链接该打印机。
     * @param {string|undefined} options.printerInfo.deviceId 打印机设备ID，在打印模式下会自动链接该打印机。
     * @param {Function} options.onJobCreated 打印任务创建完毕的回调函数，在Uni环境下需要在打印任务创建完毕后实时的
     * @param {Function} options.onPageComplete 打印页面处理完毕的回调函数，用户可以在页面处理完毕后实时的预览标签信
     * @param {Function} options.onJobComplete 所有打印页面全部处理完毕时的回调函数。
     * @param {Document|undefined} options.doc 在不知DOMParser的环境下，用户可以使用三方工具来创建一个Document来完
     * @param {DOMParser|undefined} options.domParser 特殊环境下底层不支持默认的DOMParser，此时需要用户自己创建一个
     *
     * @returns {Promise<LPA_JobPrintResult>}
     */
    printWdfx(options: LPA_JsonPrintOption): Promise<LPA_JobPrintResult>;
    /**
     * 特殊情况下，用于销毁相关资源。
     * 譬如：在微信小程序/UniAPP中用于关闭蓝牙适配器，释放相关资源。
     */
    quit(): void;
}
```