dtpweb

便携式打印机web接口。

注意：js 接口需要与 dtpweb 插件配合使用才可以，否则 js 接口将无法正常打印

- 兼容硬件环境：Windows、Kylin、UOS、Deepin、Ubuntu 等
- 支持的软件环境：Browser、NodeJS、VUE、React 等环境。

下载

```
# npm下载
npm install dtpweb
# yarn下载
yarn add dtpweb
```

使用方法

安装 dtpweb 打印助手

1. window 版本安装：
   在 windows 中，打印机驱动已经集成了 dtpweb 打印助手，直接安装驱动即可。
2. linux 桌面版（deb 版本）：

```
# linux 下 dtpweb服务安装方法
sudo apt update
sudo apt install libcurl4
sudo dpkg -i dtpweb-2.1.xxxx-xxx.deb
# linux 下 dtpweb服务卸载方法
sudo dpkg -r com.dothantech.dtpweb
```

3. linux 服务器版本（rpm 版本）：
   暂不支持

Browser 中使用

1. 检查本地打印助手是否可用（dtpweb 打印接口需要借助于本地的打印助手才能进行打印）；
2. 引入 dtpweb.js 文件；

3. 通过 dtpweb.checkServer()来检查打印助手是否可用；
4. 开始进行标签绘制操作；

eg:

```
<!-- 在合适的地方导入dtpweb.js文件 -->
<script type="text/javascript" src="../../lib/dtpweb.js"></script>
<script>
    /**
     * @type {import("../../lib/dtpweb").DTPWeb}
     */
    var api = undefined;

    window.onload = function () {
        // 检测打印接口是否可用
        dtpweb.checkServer((value) => {
            api = value;
            if (!value) {
                alert("未检测到打印机插件！");
            }
        });
    };
</script>
```

NodeJS/Vue 中使用接口

1. 导入 dtpweb 模块

```
import { DTPWeb } from 'dtpweb'
```

2. 获取接口实例 api

```
let api = undefined;
 mounted() {
    DTPWeb.checkServer((value) => {
        api = value;
        if (!value) {
            alert("打印助手不可用！");
        }
    }
}
```

快速入门

Browser 接口快速入门

```
// 接口初始化
<script type="text/javascript" src="../../lib/dtpweb.js"></script>
<script>
var api = undefined;
window.onload = function() {
    dtpweb.checkServer((value) => {
        api = value;
        if (!value) {
            console.log("No Detected DothanTech Printer Helper!");
        }
    });
}

function printText() {
    const printerName = "DT60 Label Printer";
    const labelWidth = 40;
    const labelHeight = 30;
    const text = "https://www.detonger.com"
    const fontHeight = 3;
    // 检查接口是否可用
    if (!api) return;
    // 1. 打印之前需要先链接目标打印机
    api.openPrinter(printerName, (success) => {
        if(success) {
            // 2. 创建一个指定大小的标签任务
            api.startJob({width: labelWidth, height: labelHeight});
            // 3. 在标签纸上打印目标字符串
            api.drawText({text, width: labelWidth, height: labelHeight, fontHeight});
            // 4. 结束绘制操作，开始打印
            api.commitJob(() => {
                // 5. 关闭已经打开的打印机
                api.closePrinter();
            });
        }
    });
}
</script>
```

vue 环境快速入门

- 安装 dtpweb 接口包

```
npm install dtpweb
或者
yarn add dtpweb
```

```
// 导入接口
import {DTPWeb} from "dtpweb"
// 初始化接口
mounted() {
    this.api = DTPWeb.getInstance();
    DTPWeb.checkServer((value) => {
        this.api = value;
        if (!value) {
             console.log("No Detected DothanTech Printer Helper!");
        }
    });
}
// 打印测试
methods: {
    printQRCode() {
        const labelWidth = 30;
        const labelHeight = 30;
        const margin = 5;
        const qrcodeWidth = labelWidth - margin * 2;
        const printerName = "DT60 Label Printer";

        // 判断接口是否可用
        if (!this.api) return;
        // 1. 链接目标打印机
        api.openPrinter(printerName, (success) => {
            if(success) {
                // 2. 创建一个指定大小的标签任务
                this.api.startJob({});
                // 3. 在打印任务上绘制二维码
                this.api.draw2DQRCode({x: margin, y: margin, width: qrcodeWidth});
                // 4. 结束打印任务的绘制，开始打印
                this.api.commitJob(() => {
                    // 打印完毕后关闭打印机
                    this.api.closePrinter();
                });
            }
        });
    }
}
```

DTPWeb 接口介绍

DTPWeb接口介绍

```
/**
 * PC JavaScript 版本 LPAPI 接口的异步封装，底层基于 dtpweb 打印接口。
 */
declare class DTPWeb {
    /**
     * 检查打印服务是否运行正常。
     * @param options 打印服务相关初始配置信息。
     * @param options.callback: (api?: DTPWeb, info?: LPA_ServerInfo) => void 打印服务检查回调信息。
     * @returns 成功：返回 DTPWeb 实例，失败：返回 undefined;
     */
    static checkServer(options?: InitOptions | ((api: DTPWeb | undefined,
        resp: LPA_Response<LPA_ServerInfo>) => void)): DTPWeb | undefined;
    /**
     * 获取一个单实例接口对象。
     *
     * @param options 接口初始话配置信息。
     *
     * @returns 返回一个单实例接口对象。
     */
    static getInstance(options?: LPA_InitOptions): DTPWeb;
    /**
     * 判断给定的打印机类型是不是本地打印机。
     */
    isLocalPrinter(printerType?: number | LPA_DeviceInfo): boolean;
    /**
     * 检测插件是否可用。
     *
     * @param options 插件检测相关参数。
     *
     * @param {(result: LPA_Response<?>) => void} options.callback 插件检查回调函数，
     *        回调参数表示插件检查结果详细信息。
     */
    checkPlugin(options?: LPA_CheckPluginOptions | ((resp: LPA_Response<LPA_ServerInfo>) => void)): void;
    /**
     * 检查指定的端口号是否可用。
     *
     * @param options 端口检测相关参数。
     *
     * @param {(result: LPA_Response<any>)} options.callback 端口检测回调函数，回调参数表示端口检测详细信息。
     */
    checkPort(options: LPA_CheckPortOptions): void;
    /**
     * 获取dtpweb打印助手的版本信息。
     */
    getVersion(clientType?: number): LPA_VersionInfo | undefined;
```

```
/**
 * 获取dtpweb打印助手相关信息。
 */
getServerInfo(clientType?: number): LPA_ServerInfo | undefined;
/**
 * 默认只支持德佟系列的打印机，其他厂家的打印机需要通过该接口来配置后才可以使用。
 */
setSupportedPrinters(supportedPrinters: string): boolean | undefined;
/**
 * 获取系统默认打印机相关信息。
 */
getDefaultPrinter(): LPA_DefaultPrinter | undefined;
/**
 * 修改系统默认打印机。
 */
setDefaultPrinter(printerName: string): void;
/**
 * 搜索局域网内的打印机。
 *
 * 建议在搜索命令下发2秒钟之后再通过{@link getPrinters()}来获取打印机列表。
 *
 * @param mode 打印机搜索模式，值默认为1。
 * @returns 成功与否。
 */
discoveryPrinters(mode?: number): boolean;
/**
 * 获取打印机列表。
 *
 * @param {LPA_PrinterOptions} 打印机设备相关选项。
 *
 * @param {boolean|undefined} options.onlyOnline 是否只获取在线（已连接）的打印机？默认为true。
 * @param {boolean|undefined} options.onlyLocal 是否只获取本地打印机？默认为true。
 * @param {boolean|undefined} options.onlySupported 是否只获取支持的打印机？默认为true。
 *
 * @return {LPA_Device[]} 返回打印机设备列表。
 */
getPrinters(options: LPA_PrinterOptions | ((devices: LPA_DeviceInfo[]) => void),
    timeout?: number): LPA_DeviceInfo[];
/**
 * 获取指定打印机的设备信息。
 * @param printerName 目标打印机名称。
 * @returns {LPA_DeviceInfo}
 */
getDeviceInfo(printerName: string): LPA_DeviceInfo;
/**
```

```
     * 打开指定的打印机。
     *
     * @param {string|LPA_DeviceOpenOptions|Function|undefined} options 打印机名称或对象。
     *
     * @param {string|undefined} options.name 目标打印机名称，可通过 可通过
     *        {@link getPrinters} 来获取。
     * @param {string|undefined} options.ip 目标打印机所在电脑的IP地址，默认为本机IP，可通过
     *        {@link getPrinters} 来获取。
     * @param {number|undefined} options.port 目标打印机所在电脑的打印助手端口号，可通过
     *        {@link getPrinters} 来获取。
     * @param {number|undefined} options.type 目标打印机类型，可通过 {@link getPrinters} 来获取；
     * @param {Function|undefined} options.callback 打印机打开结果回调函数。
     *
     * @return {boolean} 目标打印机链接成功与否。
     */
    openPrinter(options?: string | LPA_DeviceOpenOptions | ((success: boolean) => void),
        callback?: (success: boolean) => void): boolean;
    /**
     * 获取已连接的打印机名称。
     */
    getPrinterName(): string;
    /**
     * 判断打印机是否已打开。
     */
    isPrinterOpened(): boolean;
    /**
     * 判断当前打印机是否在线。
     */
    isPrinterOnline(): boolean;
    /**
     * 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(): boolean;
    /**
     * 显示打印机属性设置界面或者首选项设置界面。
     *
     * @param {LPA_PrinterPropertyOptions} options 打印机相关相关选项。
     *
     * @param {string|undefined} options.printerName 打印机名称，如果为空则会打开当前已打开打印机的
     *        打印机属性或者打印首选项。
     * @param {boolean|undefined} options.showDocument 是否显示打印机首选项？默认为true。
     *          true: 表示显示首选项设置界面；
     *          false:显示打印机属性设置界面。
```

```
     *
     * @warning 如果在调用该接口前已通过 openPrinter 函数打开打印机，则可以不指定 printerName。
     */
    showProperty(data: LPA_PrinterPropertyOptions): boolean;
    /**
     * 获取打印相关参数。
     *
     * @param {LPA_ParamID} id 打印参数ID，ID值可参考 {@link LPA_ParamID}。
     *
     * @return {number} 值参考 {@link LPA_ParamID} 中不同ID所对应的值类型。
     */
    getParam(id: LPA_ParamID): number;
    /**
     * 设置打印参数;
     *
     * @param {number|LPA_PrintParamOptions} options 打印参数相关选项。
     *
     * @param {LPA_ParamID} options.id 打印机参数ID，ID值可参考 {@link LPA_ParamID}。
     * @param {number} options.value id值所对应打印机参数的value，具体可参考 {@link LPA_ParamID}。
     *
     * @return {boolean} 成功与否?
     */
    setParam(options: LPA_PrintParamOptions): boolean;
    /**
     * 获取已连接打印机的纸张类型。
     */
    getGapType(): LPA_GapType;
    /**
     * 修改已连接打印机的纸张类型。
     *
     * @param {LPA_GapType} value 纸张类型。
     */
    setGapType(value: LPA_GapType): boolean;
    /**
     * 返回已连接打印机的打印浓度。
     *
     * @return {number} 打印机浓度值说明可参考 {@link LPA_PrintDarkness};
     */
    getPrintDarkness(): number;
    /**
     * 修改已连接打印机的打印浓度。
     *
     * @param {number} value 打印浓度。
     */
    setPrintDarkness(value: number): boolean;
```

```
/**
 * 返回已连接打印机的打印速度。
 */
getPrintSpeed(): number;
/**
 * 修改已连接打印机的打印速度。
 *
 * @param {number} value 打印速度，值参考{@link LPA_PrintSpeed}。
 */
setPrintSpeed(value: number): boolean;
/**
 * 获取打印机的分辨率（打印机链接成功后有效）。
 */
getPrinterDPI(): LPA_PrinterDPI | undefined;
/**
 * 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_StartJobOptions} 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，值默认位0，可选值有：
 *        0  ：表示不旋转；
 *        90 ：表示右转90度；
 *        180：表示180度旋转；
 *        270：表示左转90度；
 * @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。
 */
startJob(options: LPA_StartJobOptions): boolean;
/**
 * 创建打印任务。
 *      1：  在打印模式下，如果用户制定了目标打印机，则自动链接目标打印机，打印机链接成功，则调用
 *           startJob 创建目标打印任务，否则返回失败。
 *      2：  在预览模式下，直接调用 startJob 创建预览任务。
 */
startPrintJob(options: LPA_StartPrintJobOption, callback?: (result: boolean) => void): boolean;
/**
 *  取消当前打印任务。
 *
 *  使用说明：当前还有未打印的任务/数据将会被全部丢弃，但是所有参数设置将会被保留。
 */
```

```
abortJob(): void;
/**
 * 提交打印任务，进行真正的打印。
 *
 * @param {PrintOptions|undefined} options 相关打印参数。
 *
 * @param {number|undefined} options.copies 打印份数。
 * @param {number|undefined} options.orientation 打印方向，默认为`0`。
 *          `0`表示不旋转，`90`表示右转90度，`180`表示进行180度旋转，`270`表示左转90度。
 * @param {number|undefined} options.threshold 图片进行黑白转换时的阈值，默认为
 *        {@link CONSTANTS.THRESHOLD}，也即：192。
 * @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 {(success: boolean) => void} option.callback 打印结果回调函数。
 *
 * @param {PrintJobResults | undefined} 返回的打印结果信息。
 */
commitJob(options?: PrintOptions | ((success: PrintJobResults | undefined) => void)):
    PrintJobResults | undefined;
/**
 * 设置后续绘制内容的水平对齐方式。
 * @param {LPA_ItemAlignment} alignment 对齐方式：
 *
 * @deprecated 可以在绘制目标对象的时候直接指定对齐方式：horizontalAlignment。
 */
setItemHorizontalAlignment(alignment: number): LPA_Response<any>;
/**
 * 设置后续绘制内容的水平对齐方式。
 * @param {LPA_ItemAlignment} alignment 对齐方式：
 *
 * @deprecated 可以在绘制目标对象的时候直接指定对齐方式：verticalAlignment。
 */
setItemVerticalAlignment(alignment: number): LPA_Response<any>;
/**
 * 设置后续绘制内容的旋转角度。
 * @param {0|90|180|270} orientation 旋转角度；
 *
 * @deprecated 可以在绘制目标对象的时候直接指定旋转角度：orientation。
 */
setItemOrientation(orientation: number): LPA_Response<any>;
/**
 * 得到最近一次打印任务的标识。
 *
 * @return 打印任务标识。
```

```
 */
getJobID(): number;
/**
 * 得到打印任务的状态信息。
 * @param {LPA_JobInfoOptions} options 任务选项。
 *
 * @param {string|undefined} options.printerName 打印机名称，为空表示当前打开的打印机。
 * @param {number|undefined} options.jobID 打印任务标识，为0表示最近一次的打印任务。
 *
 * @return 返回任务信息,格式为 JOB_INFO_1，为 NULL 用于测量需要的空间字节数。
 */
getJobInfo(options: LPA_JobInfoOptions): LPA_JobInfo | undefined;
/**
 * 得到刚完成的打印任务的打印任务信息。
 *
 * @return {LPA_PageInfo} 返回刚完成的打印任务信息
 */
getPageInfo(): LPA_PageInfo;
/**
 * 得到刚完成的打印任务的页面图片数据。
 * @param {LPA_PageImageOptions} options 参数选项。
 *
 * @param {number|undefined} options.page 通过getPageInfo获取到的页面总数中的索引，默认为0，表示第一页。
 * @param {LPA_SourceImageFormat|undefined} options.format 获取到的图片的数据格式，具体可参考
 *        {@link LPA_SourceImageFormat}，默认为{@link LPA_SourceImageFormat.LPASIF_IMAGEDATA},
 *        表示返回BASE64格式的图片数据。
 *
 * @return {LPA_PageImage} 返回页面图片数据。
 */
getPageImage(options: LPA_PageImageOptions): LPA_PageImage;
/**
 * 开始一打印页面。
 *
 * @info 如果之前没有调用 StartJob，则本函数会自动调用 StartJob，然后再开始一打印页面。此后调用 EndPage
 *       结束打印时，打印任务会被自动提交打印。页面旋转角度非 0 打印时，必须在打印动作之前设置打印页面尺寸信息。
 */
startPage(): boolean;
/**
 * 结束一打印页面。
 *
 * @info 如果之前没有调用 StartJob 而直接调用 StartPage，则本函数会自动提交打印。
 */
endPage(): boolean;
/**
 * 设置绘制函数是否返回绘制的详细信息?
```

```
     *
     * @param options 字符串打印相关参数。
     *
     * @param {boolean|undefined} options.returnDrawResult 绘制函数是否返回绘制的详细信息。
     */
    returnDrawResult(options: LPA_DrawResultOptions | boolean): boolean;
    /**********************************************************************
     * 绘制相关内容。
     **********************************************************************/
    /**
     * 将图片的 URL 转换为 base64 字符串。
     */
    imageSrc2DataUrl(src: string, callback: (dataUrl: string) => void): void;
    /**
     * 将给定的毫米值转换为磅值。
     *
     * 该函数常用于绘制字符串的时候字体大小的单位换算。
     *
     * @param value 待转换的值，单位毫米。
     * @returns 转换后的值，单位磅。
     */
    mm2Pound(value: number): number;
    /**
     * 将给定的磅值转换为毫米值。
     *
     * 该函数常用于绘制字符串的时候字体大小的单位换算。
     *
     * @param value 待转换的值，单位磅。
     * @returns 转换后的值，单位毫米。
     */
    pound2Mm(value: number): number;
    /**
     * 按照给定的分辨率，将像素值转换为毫米值。
     * @param value 待转换的像素值。
     * @param dpi 转换分辨率，默认为203。
     */
    pix2Mm(value: number, dpi?: number): number;
    /**
     * 按照给定的分辨率，将毫米值转换为像素值。
     * @param value 待转换的毫米值。
     * @param dpi 转换分辨率，默认为203。
     */
    mm2Pix(value: number, dpi?: number): number;
    /**
     * 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，默认为{@link LPA_FontStyle.Regular}。
* @param {LPA_AutoReturnMode|undefined} options.autoReturn 自动换行模式，默认为
*        {@link LPA_AutoReturnMode.Char}，其他可选值有：
*            {@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。
* @param {number|undefined} options.leadingIndent 首行缩进的四个参数，默认为0。
*        四选一，leadingIndent 具有最高优先级。
*          0          ：表示没有首行缩进；
*          1 ~ 999    ：表示首行向左缩进 N/10 个中文字符个数（字符高度）
*          1000       ：表示首行向左缩进到中文冒号、英文冒号、英文冒号+英文空格
*          > 1000     ：表示首行向左缩进（N - 1000）的 ScaleUnit
*          -999 ~ -1  ：表示首行向右缩进 -N/10 个中文字符个数（字符高度）
*          < -1000    ：表示首行向右缩进（-N - 1000）的 ScaleUnit
* @param {number|undefined} options.leadingIndentChars，根据指定的中文字符个数进行首行缩进。其值可以为小数，
*        比方说 1.5表示 1.5 个中文字符 / 3 个英文字符。> 0 表示首行向左缩进，< 0表示首行向右缩进。
* @param {number|undefined} options.leadingIndentMM 根据指定的毫米数进行首行缩进。
*          > 0 表示首行向左缩进，< 0 表示首行向右缩进。
* @param {boolean|undefined} options.leadingIndentColon 表示首行向左缩进到
*        中文冒号、英文冒号、英文冒号+英文空格。
* @param {number[]|string|undefined} regionCorners 显示区域四个角的删除矩形，
*        分别为左上、右上、右下、左下，格式为：
*        "[Width, Height, Width, Height, Width, Height, Width, Height]"，单位毫米。
```

```
     * @param {number[]|string|undefined} regionLeftUpCorner 显示区域左上角的删除矩形，格式为：
     *        "[Width, Height]"，单位毫米。
     * @param {number[]|string|undefined} regionRightUpCorner 显示区域右上角的删除矩形，格式为：
     *        "[Width, Height]"，单位毫米。
     * @param {number[]|string|undefined} regionRightBottomCorner 显示区域右下角的删除矩形，格式为：
     *        "[Width, Height]"，单位毫米。
     * @param {number[]|string|undefined} regionLeftBottomCorner 显示区域左下角的删除矩形，格式为：
     *        "[Width, Height]"，单位毫米。
     * @param {number[]|string|undefined} regionLeftBorders 显示区域左边的删除矩形，最多支持删除两个矩形，
     *          格式为：`[Width, Y, Height, Width, Y, Height]`，单位毫米。
     * @param {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 setItemHorizontalAlignment()} 设置的参数，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 setItemVerticalAlignment()} 设置的参数，默认为：{@link LPA_ItemAlignment.Start}，
     *        表示居上对齐。
     */
    drawText(options: DrawTextOptions): boolean;
    /**
     * 计算字符串长度。
     * @param options 字符串计算相关参数。
     * @param {string|string[]} options.text 待计算的文本数据。
     * @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，默认为{@link LPA_FontStyle.Regular}。
     * @param {LPA_AutoReturnMode|undefined} options.autoReturn 自动换行模式，默认为
     *        {@link LPA_AutoReturnMode.Char}，其他可选值如下：
     *          {@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。
     * @returns 字符串计算相关结果。
     */
    measureText(options: DrawTextOptions): MeasureTextResult | undefined;
    /**
     * 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},
     *        表示根据字符串自动采用最佳方式。
     * @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_NAME}。
     * @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_FontStyle.Regular}，表示显示常规字体样式。
     * @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 LPA_ItemAlignment}，>= 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_BarcodeFlags}，默认为 ShowReadDown | ShowStartStop | EanCheckCode。
     * @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 - 7 之间的任意值，默认为2。
     * @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 setItemHorizontalAlignment()} 设置的参数，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 setItemVerticalAlignment()} 设置的参数，默认为: {@link LPA_ItemAlignment.Start},
     *        表示居上对齐。
     */
    draw1DBarcode(options: DrawBarcodeOptions): boolean;
    /**
     * 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 {LPA_QRTextEncoding|undefined} options.textEncoding 字符串编码方式，值参考
     *        {@link LPA_QRTextEncoding}，默认为{@link LPA_QRTextEncoding.UTF8}。
     * @param {number|undefined} options.qrcPixels 表示在不指定二维码显示宽度的情况下，
     *        二维码每个逻辑点的像素个数，默认为2个像素。
     * @param {number|number} options.qrcVersion 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_QREncodeMode|undefined} options.encodeMode 二维码编码模式，值参考{@link LPA_QREncodeMode},
     *         默认为{@link LPA_QREncodeMode.ModeNum}。如果编码内容需要更高级别的编码模式，程序会自动升级模式。
     * @param {LPA_QREccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_QREccLevel},
     *        默认为{@link LPA_QREccLevel.EccLevel_L}。
     * @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 setItemHorizontalAlignment()} 设置的参数，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 setItemVerticalAlignment()} 设置的参数，默认为: {@link LPA_ItemAlignment.Start},
     *        表示居上对齐。
     */
    draw2DQRCode(options: DrawQrcodeOptions): boolean;
    /**
     * 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 {number|undefined} options.textEncoding 字符串编码方式，{@link LPA_P417TextEncoding},
     *        默认为{@link LPA_P417TextEncoding.UTF8}。
     * @param {number|undefined} options.p417Pixels 在不指定二维码宽度的情况下每个逻辑点的像素个数，默认为2。
     * @param {LPA_P417EncodeMode|undefined} options.encodeMode 二维码编码模式，值参考
     *        {@link LPA_P417EncodeMode}，默认为{@link LPA_P417EncodeMode.Auto}。
     *        如果编码内容需要更高级别的编码模式，程序会自动升级模式。
     * @param {LPA_P417EccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_P417EccLevel},
     *        默认为{@link LPA_P417EccLevel.Auto}。
     * @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 setItemHorizontalAlignment()} 设置的参数，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 setItemVerticalAlignment()} 设置的参数，默认为：{@link LPA_ItemAlignment.Start},
     *        表示居上对齐。
     */
    draw2DPdf417(options: DrawPdf417Options): boolean;
    /**
     * 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 {LPA_DMTextEncoding|undefined} options.textEncoding 字符串编码方式，
     *        值参考{@link LPA_DMTextEncoding}，
     *        默认为{@link LPA_QRTextEncoding.UTF8}。
     * @param {number|undefined} options.dmtxPixels 表示在不指定二维码显示宽度的情况下，
     *        二维码每个逻辑点的像素个数，默认为2个像素。
     * @param {number|number} options.symbolShape
     * @param {LPA_DMEncodeMode|undefined} options.encodeMode 二维码编码模式，
     *        值参考{@link LPA_DMEncodeMode}，默认为{@link LPA_QREncodeMode.ModeNum}。
     *        如果编码内容需要更高级别的编码模式，程序会自动升级模式。
     * @param {number|undefined} options.encodeFlags
     * @param {number|undefined} options.minHeight 二维码最小高度，单位毫米。默认自适应。
     * @param {number|undefined} options.maxHeight 二维码最大高度，单位毫米。默认自适应。
     * @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 setItemHorizontalAlignment()} 设置的参数，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 setItemVerticalAlignment()} 设置的参数，默认为：{@link LPA_ItemAlignment.Start},
     *        表示居上对齐。
     * @returns 成功与否。
     */
    draw2DDataMatrix(options: DrawDataMatrixOptions): boolean;
    /**
     * 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;
    /**
     * 绘制圆角矩形。
     *
     * @param {DrawRectOptions} 绘制圆角矩形的相关选项。
     *
     * @param {number|undefined} options.x 圆角矩形的水平位置，单位毫米，值默认为0
     * @param {number|undefined} options.y 圆角矩形的垂直位置，单位毫米，值默认为0
     * @param {number|undefined} options.width 圆角矩形的水平宽度，单位毫米，值默认为
     *        {@link CONSTANTS.RECT_WIDTH}。
     * @param {number|undefined} options.height 圆角矩形的垂直高度，单位毫米，值默认与宽度相同。
     * @param {number|undefined} options.cornerWidth 圆角宽度，单位毫米，值默认为0。
     * @param {number|undefined} options.cornerHeight 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，默认false，表示绘制圆角矩形框。
     * @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。
     */
    drawRoundRectangle(options: DrawRectOptions): boolean;
    /**
     * 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: DrawRectOptions): 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.lineWidth}。
     * @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): boolean;
    /**
     * 打印点划线。
     *
     * @param {DrawDashLineOptions} options 点划线绘制相关选项。
     *
     * @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，值默认为x1。
     * @param {number|undefined} options.y2 点划线终点位置，单位毫米，值默认为y1。
     * @param {number|undefined} options.lineWidth lineWidth: 直线线宽，单位毫米，值默认为
```

```
     *        {@link CONSTANTS.LINE_WIDTH}。线宽是向线的下方延伸的。
     * @param {number[]} options.dashLen 点化线线段长度的数组，默认为{@link CONSTANTS.DASH_LEN}。
     * @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 LPAStartPage 而直接进行打印，则打印函数会自动调用 LPAStartPage(0)
     *        开始一打印页面，然后进行打印。
     * @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。
     */
    drawDashLine(options: DrawLineOptions): boolean;
    /**
     *  打印指定的URL图片。
     *
     * @param {DrawImageOptions} options URL图片绘制相关选项。
     *
     * @param {string} options.imageFile 图片文件或者URL路径。
     * @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，
     *        然后进行打印。
     * @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，
     *        然后根据一阀值将位图再次转换黑白位图再进行打印。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): boolean;
    /**
     * 绘制图片对象。
     *
     * @param {DrawImageDataOptions} options 图片对象绘制选项。
     *
     * @param {string|ArrayBuffer|Blob} options.data 图片数据，一般情况下是图片base64字符串或者ArrayBuffer。
     * @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.drawWidth 图片显示宽度，单位毫米，值默认为0，表示图片的实际宽度。
```

```
     * @param {number|undefined} options.drawHeight 图片显示高度，单位毫米，值默认为0，表示图片的实际高度。
     * @param {number|undefined} options.threshold 图片黑白转换的灰度阀值，默认为{@link CONSTANTS.THRESHOLD}。
     *          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 {LPA_SourceImageFormat|undefined} options.format 目标图片数据格式，默认为
     *        {@link LPA_SourceImageFormat.LPASIF_IMAGEDATA}，表示BASE64图片。
     * @param {number|undefined} options.imageWidth data中图片的实际宽度，单位像素。
     * @param {number|undefined} options.lineSize 位图数据每一行数据的字节数，默认为零。
     *          如果指定 lineSize，则必须 >= 默认长度；如果为零，则采用如下的默认长度：
     *
     *          LPASIF_BPP_1   : (width + 7) / 8
     *          LPASIF_BPP_1N  : (width + 7) / 8
     *          LPASIF_32_RGBA : width * 4
     *          LPASIF_32_BGRA : width * 4
     *          LPASIF_32_RGB  : width * 4
     *          LPASIF_32_BGR  : width * 4
     *          LPASIF_PACKAGE : 报文格式未使用 lineSize 参数。
     * @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。
     */
    drawImageD(options: DrawImageDataOptions): boolean;
    /**
     * 绘制表格对象。
     * @param {DrawTableOptions} options 表格绘制相关参数。
     * @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，值默认为打印任务的宽度。
     * @param {number|undefined} options.height 绘制对象的显示高度，单位毫米，值默认为打印任务的高度。
     * @param {number|undefined} options.lineWidth 矢量图的线条宽度，单位毫米，值默认为0.4毫米。
     *        当线宽小于等于0的时候不显示边线
     * @param {TableRow[]} options.tableRows 单元格内容行列表。
     * @param {number|undefined} options.rowCount 表格的行数。
     * @param {number|undefined} options.columnCount 表格的列数。
     * @param {TableCell[]} options.cells 单元格内容列表。
     * @param {number[]|undefined} options.rowHeights 单元格中单元格行的高度。
     *      备注：当指定的单元格高度大于1的时候，分配指定大小的高度，否则会按比例等分剩余高度。
     * @param {number[]|undefined} options.columnWidths 单元格每一列的宽度。
     *      备注：当单元格的大小大于1的时候，分配指定大小的宽度，否则会按比例等分剩余宽度。
     * @param {Rect[]|undefined} options.groups 单元格合并列表。
     * @param {LPA_ItemAlignment|undefined} options.horizontalAlignment 表格中单元格的默认水平对齐方式。
     * @param {LPA_ItemAlignment|undefined} options.verticalAlignment 表格中单元格的默认垂直对齐方式。
     * @param {number|undefined} options.cellPadding 单元格中内容与边线之间的距离。
     * @param {number|undefined} options.fontHeight 表格的默认字体高度，单位毫米。
     * @param {number|undefined} options.fontSize 表格的默认字号大小，单位磅。
```

```
     * @param {number|undefined} options.fontStyle 表格的默认字体样式。
     * @param {string|undefined} options.fontName 表格的默认字体名称。
     *
     * @returns {boolean} 成功与否。
     */
    drawTable(options: DrawTableOptions): boolean;
    /**
     * 直接打印指定位图对象。
     *
     * @param {PrintImageOptions} options 图片打印相关选项。
     *
     * @param {string|ArrayBuffer|Blob} options.data 打印数据，一般情况下为BASE64格式的图片数据。
     * @param {string|undefined} options.printerName 打印机名称，不指定表示上次连接过的打印机。
     * @param {LPA_SourceImageFormat|undefined} options.format 图片格式，默认为
     *        {@link LPA_SourceImageFormat.LPASIF_IMAGEDATA}，现阶段只支持该格式。
     * @param {number|undefined} options.imageWidth 如果以二进制流的方式传递打印数据，
     *        则需要指定对应图片的宽度，单位像素。
     * @param {number|undefined} options.lineSize 二进制流单行数据大小。
     * @param {number|undefined} options.printWidth 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.printHeight 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。
     */
    printImage(options: PrintImageOptions): boolean;
    /**
     * 直接打印打印机所支持的控制命令数据，可以是打印数据，也可以是参数设置命令等。
     *
     * @param options 命令打印相关选项。
     *
     * @param {Blob} options.data 图片文件的二进制数据。
     * @param {string|undefined} options.printerName 打印机名称，不指定表示上次连接过的打印机。
     * @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。
     */
    printRawData(options: RawDataPrintOptions): boolean;
    /**
     * 根据给定的标签配置信息，打印整个打印任务。
     *
     * @param {PrintJsonOptions} options 打印配置信息。
     * @param {number} options.action 结果返回方式。
     *      0x0001: 表示返回打印用二级制数据;
```

```
     *      0x0002: 表示返回包含""前缀的 BASE64 编码的预览用图片数据。
     *      0x0004: 表示返回预览用图片网址，如：https://d6688.cn/f/f?key=xxx;
     *      0x0082: 表示是否生成透明底色的预览图片;
     *      0x1000: 表示直接打印给定的json数据。
     * @param {PrinterInfoOptions|undefined} options.printerInfo 目标打印机相关信息。
     * @param {JobInfoOptions|undefined} options.jobInfo 标签或打印相关信息。
     * @param {DrawItemOptions[]} options.jobPages 标签内容列表。
     * @param {KeyValue[][]} options.jobArguments 关键字列表。
     *      当 jobPages 页数为 1 时生效，jobArguments中的每个子数组表示一页数据。
     *      在 jobPages中没想数据的 text中，用中括号来设置关键字，eg: "xxx [test] xxx"。
     * @param {(result: PrintJobResults|undefined) =>void} options.callback 打印结果回调函数。
     */
    print(options: PrintJsonOptions): PrintJobResults | undefined;
    /**
     * 解析 wdfx 文件并进行打印。
     * @param {PrintWdfxOptions} options 打印相关配置参数。
     * @param {number} options.action 结果返回方式，每个 bit 位表示一个功能，可以是多个功能的组合。
     *      0x0001: 表示返回打印用二级制数据;
     *      0x0002: 表示返回包含""前缀的 BASE64 编码的预览用图片数据。
     *      0x0004: 表示返回预览用图片网址，如：https://d6688.cn/f/f?key=xxx;
     *      0x0082: 表示是否生成透明底色的预览图片;
     *      0x1000: 表示直接打印给定的json数据。
     * @param {string} options.content wdfx文件内容。
     * @param {PrinterInfoOptions|undefined} options.printerInfo 目标打印机相关信息。
     * @param {JobInfoOptions|undefined} options.jobInfo 用户可以通过 jobInfo 来修改wdfx中标签的相关信息。
     * @param {KeyValue[][]} options.jobArguments 关键字列表。
     *      当 jobPages 页数为 1 时生效，jobArguments中的每个子数组表示一页数据。
     *      在 jobPages中没想数据的 text中，用中括号来设置关键字，eg: "xxx [test] xxx"。
     * @param {(result: PrintJobResults) =>void} options.callback 打印结果回调函数。
     */
    printWdfx(options: PrintWdfxOptions): void;
}
```

纸张间隔

```
/**
 * 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 */
    Gap = 2,
    /** 黑标卡纸 */
    Black = 3,
    /** 黑标透明贴 */
    Trans = 4,
}
```

打印速度

```
/**
 * 打印速度常用值。
 *
 * 实际有效值为0到4之间，255表示随打印机设置，其他为无效值。
 */
export enum LPA_PrintSpeed {
    /** 随打印机设置 */
    Unset = 255,
    /** 最慢 */
    Min = 0,
    /** 较慢 */
    Low = 1,
    /** 正常速度 */
    Normal = 2,
    /** 较块 */
    High = 3,
    /** 最快 */
    Max = 4,
}
```

打印浓度

```
/**
 * 打印浓度常用枚举值。
 *
 * 打印浓度可以0到14之间的任意值，255表示随打印机设置，其他为无效值。
 */
export enum LPA_PrintDarkness {
    /** 随打印机设置 */
    Unset = 255,
    /** 最淡 */
    Min = 0,
    /** 较淡 */
    Low = 3,
    /** 正常浓度 */
    Normal = 5,
    /** 较浓 */
    High = 9,
    /** 最浓 */
    Max = 14,
}
```

对齐方式

```
/**
 * 打印动作的对齐方式。
 */
export enum LPA_ItemAlignment {
    /** 水平居左(垂直居上)对齐 */
    Start = 0,
    /** 水平（垂直）居中对齐 */
    Center = 1,
    /** 水平居右（垂直居下）对齐 */
    End = 2,
    /** 拉伸（多余空间使用空白填充） */
    Stretch = 3,
    /** 放大（多余空间通过放大填充） */
    Expand = 4,
}
```

字体样式和换行模式

```
/**
 * 字体样式。
 */
export enum LPA_FontStyle {
    /** 一般字体样式 */
    Regular = 0,
    /** 粗体 */
    Bold = 1,
    /** 斜体 */
    Italic = 2,
    /** 粗斜体 */
    BoldItalic = 3,
    /** 下划线 */
    Underline = 4,
    /** 删除线 */
    Strikeout = 8,
}
/**
 * 绘制字符串的时候的自动换行模式。
 */
export enum LPA_AutoReturnMode {
    /** 不进行自动换行 */
    None = 0,
    /** 按照字自动换行 */
    Char = 1,
    /** 按照词自动换行 */
    Word = 2,
}
```

一维码类型

```
/**
 * 一维条码编码类型。
 *
 * UPC-A，UPC-E，EAN13，EAN8，ISBN 统称为商品码，编码和显示方式类似；
 * 只能包含数字，对于支持两段的方式的编码，使用“+”来作为前后两段的分隔；
 * 都有校验字符，一般为0～9。对于 ISBN 编码，其校验字符可能为“X”。
 */
export enum LPA_BarcodeType {
    /**
     * 支持长度为：12、12+2、12+5，显示为 1+5+5+1
     * 输入长度为 12：表示已经有校验码；
     *            11：没有校验码，程序会自动添加；
     *          < 11：加上前导零，然后自动添加校验码；
     */
    LPA_1DBT_UPC_A = 20,
    /**
     * 支持长度为：8、8+2、8+5，显示为1+6+1。其中第一位是编码数字类型，只
     * 能为0/1，表示采用的数字系统；第八位是校验位，采用 upc_check() 进行校验。
     * 输入长度为 8：表示已经有校验码，如果第一个字符不是0/1，则强制换成0来处理；
     *           7：没有校验码，程序会自动添加。如果第一个字符不是0/1，则强制换成0来处理；
     *           6：没有校验码，程序会自动添加。同时采用数字系统 0 来进行编码。
     *         < 6：加上前导 0 到长度为 6 之后，再进行编码。
     */
    LPA_1DBT_UPC_E = 21,
    /**
     * 支持长度为：13、13+2、13+5、8、8+2、8+5、5、2。
     * 输入长度为 13：表示已经有校验码；
     *           12：没有校验码，程序会自动添加；
     *         6~11：加上前导零之后，当成长度为 12 的处理；
     * 输入长度为 3/4/5：表示编码成长度为 5 的附加条码；
     *             1/2：表示编码成长度为 2 的附加条码。
     */
    LPA_1DBT_EAN13 = 22,
    /**
     * 在内部和 EAN13 编码统一处理
     *
     * 输入长度为 8：表示已经有校验码；
     * 输入长度大于 8 时，切换成 EAN13 码进行编码；
     * 输入长度 <= 5 时，切换成 EAN13 码进行编码；
     * 输入长度为 7：没有校验码，程序会自动添加；
     *           6：加上前导零，然后自动添加校验码；
     */
    LPA_1DBT_EAN8 = 23,
    /**
     * 1、"0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ-. $/+%"
```

```
     * 2、以 * 为显示用引导和结束字符（编码中没有，仅仅显示用）
     * 3、每个字符用10个编码（显示长度为13）
     * 4、引导字符10个（显示长度为13），结束字符9个（显示长度12）
     * ==》字符数为 10+ 9+10×N
     *        显示长度 13+12+13×N
     *        10个字符 13+12+13×10 = 155像素
     * 如果编码内容中包含不支持的字符，则会切换成 CODE 128 编码；
     */
    LPA_1DBT_CODE39 = 24,
    /**
     * 1、0~9
     * 2、加校验码之后长度必须是偶数，否则在头部加上 0
     * 3、每个字符用5个编码（显示长度为 7）
     * 4、引导字符 4 个（显示长度为 4），结尾字符 3 个（显示长度是 4）
     * ==》字符数为 4 + 3 + 10* (N/2)
     *        显示长度 4 + 4 + 14* (N/2)
     *        10个字符 4+4+14×5 = 78像素
     * 如果编码内容中包含不支持的字符，则会切换成 CODE 128 编码；
     */
    LPA_1DBT_ITF25 = 25,
    /**
     * 1、"0123456789-$:/.+ABCD"，多应用于医疗领域
     * 2、引导/结束字符 A～D，都会被转化为大写
     * 3、加上引导字符/校验码之后，数据统一编码；
     * 4、每个字符用8个编码（显示长度为 10～11）
     * ==》字符数为 8×N，显示长度为 10×N～11×N
     *        10个字符10×10 + 11×2 = 122像素
     * 如果编码内容中包含不支持的字符，则会切换成 CODE 128 编码；
     */
    LPA_1DBT_CODABAR = 26,
    /**
     * 0x00～0x7F
     * 如果编码内容中包含不支持的字符，则会切换成 CODE 128 编码；
     */
    LPA_1DBT_CODE93 = 27,
    /**
     * 0x00～0xFF，CODE 128 A/B 支持全字符，对于 CODE 128 C 编码：
     *
     * 1、有固定方式的校验码，都是数字，必须是偶数长度
     * 2、引导字符 105，结束字符 106
     * 3、条码宽度范围为1～4，每个字符用7个编码（显示长度为11）
     * ==》字符数为 7+7+7+7× (N/2)
     *        显示长度 11+11+11+11× (N/2)
     *        10个字符 11+11+11+11× (10/2) = 88像素
     */
```

```
    LPA_1DBT_CODE128 = 28,
    /**
     * 0~9，最后一位可能为 0~9，X（校验字符）
     * 13：必须是 978/979 前导，用 EAN13 编码，isbn13_check
     * 10：加上 978 前导之后，用 EAN13 编码，isbn_check
     * <=9：加上 0 前导之后，Check，然后再加上 978 前导，用 EAN 13 编码
     * 如果编码内容中包含不支持的字符，则会切换成 CODE 128 编码；
     */
    LPA_1DBT_ISBN = 29,
    /**
     * EXTENDED CODE 39，0x00～0x7F
     *
     * 对于 CODE 39 不支持的字符，采用转义之后，用两个字符来表示
     * 如果编码内容中包含不支持的字符，则会切换成 CODE 128 编码；
     */
    LPA_1DBT_ECODE39 = 30,
    /**
     * 根据编码内容，自动选择最适合的编码类型进行编码。
     *
     * 1、ITF25（内容长度为偶数，并且全部为数字时）
     * 2、CODABAR（如果内容以A/B/C/D开头，又以A/B/C/D结尾的话）
     * 3、CODE 39
     * 4、CODE 128
     */
    LPA_1DBT_AUTO = 60,
}
```

QRCode纠错级别

```
/**
 * QRCode 纠错模式。
 */
export enum LPA_QREccLevel {
    /** Low */
    EccLevel_L = 0,
    /** Middle */
    EccLevel_M = 1,
    /** Quality */
    EccLevel_Q = 2,
    /** High */
    EccLevel_H = 3,
}
```

修改记录

2.6.20240925

1. 增加 GridMatrix 二维码打印功能；
2. startJob的时候增加 jsonMode 的属性(用于接口的测试)；
3. drawTable的时候增加 autoReturn 属性，用于控制表格单元格的默认换行模式；

2.6.20240815

1. 实现了 printRawData 的接口；
2. 实现了绘制表格的时候，当表格线宽(lineWidth) 小于等于0的时候不显示表格边框的功能；
3. 增加 imageSrc2DataUrl 的接口，用于将图片url转换为base64字符串，然后进行打印的功能（在linux中不再支持图片url的下载打印功能）；
4. 解决了通过 startJob 方式来生成 rawData 的时候无法指定指定打印机分辨率和打印头宽度的问题；

2.6.20240730（2024-07-30）

1. 解决了json打印模式下当base64字符串过长的时候打印助手异常退出的问题；
2. 解决了json打印的时候关键字显示异常的问题；
3. 解决了调用 printImage 打印的时候，printWidth和printHeight设置毫米单位后，参数未生效的问题；
4. 实现了表格绘制的功能；
5. wdfx解析打印功能完善，同时解决了解析并绘制表格的时候出现的一些显示异常的问题；

v2.3.2023.0815

- 随底层 dtpweb 打印助手同步更新，同时增加打印助手版本检测功能；
- 性能优化，打印数据准备好之后统一发送；
- 为了避免长时间打印的时候出现超时或者假死状态，数据发送的时候通过回调函数的方式来返回打印结果；

v2.1.20221212

- 底层 dtpweb 服务蓝牙链接相关代码升级；
- 实现了 json 数据打印的功能；

v2.1.5

- 根据最新版本接口，同步更新相关接口文档。

v2.1.4

- 接口优化。

v2.1.3

- 接口文档优化与完善。

v2.1.2

- 解决了 nodejs 中调用接口进行打印的时候无法打印 base64 图片的问题；

v2.1.1

- 完善局域网打印功能；
- 接口用法及注释的完善与更新；

v2.1.0

- 2.1 版本正式发布