dtpweb

便携式打印机接口。

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 = dtpweb.getInstance();

    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

```
mounted() {
    const api = DTPWeb.getInstance();
    DTPWeb.checkServer((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 = dtpweb.getInstance();
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

```
/**
 * 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>) =
    ): DTPWeb;
    /**
     * 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_Device): boolean;
    /**
     * 检测插件是否可用。
     *
     * @param {(api?: DTPWeb) => void | undefined} callback 插件检测结果回调函数。
     */
    checkPlugin(callback?: (resp: LPA_Response<LPA_ServerInfo>) => void): void;
    /**
     * 获取dtpweb打印助手的版本信息。
     */
    getVersion(): LPA_VersionInfo | undefined;
    /**
     * 获取dtpweb打印助手相关信息。
     */
    getServerInfo(): LPA_ServerInfo | 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, timeout?: number): LPA_Device[];
/**
 * 打开指定的打印机。
 *
 * @param {string|LPA_Printer|undefined} printer 打印机名称或对象。
 *
 * @param {string|undefined} printer.printerName 目标打印机名称。
 * @param {string|undefined} printer.ip 目标打印机IP地址，不指定表示链接本地打印机。
 *
 * @return {boolean} 目标打印机链接成功与否。
 */
openPrinter(options?: string | LPA_Device, callback?: (success: boolean) => void): void;
/**
 * 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的打印机属
 * @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
 * 当前还有未打印的任务，已有打印数据将会被全部丢弃。
 *
 * @param {JobOptions} 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，`18
 * @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
 *          预览时的值可参考:{@link LPA_JobNames}，默认为{@link LPA_JobNames.Print}，表示打印任
 */
startJob(options: LPA_JobStartOptions): boolean;
/**
 * 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
 * @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。
 */
commitJob(options?: PrintOptions | ((success: boolean) => void)): void;
/**
 * 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_JobInfo} 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_JobInfo): LPA_JobInfoOptions | 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
 * @param {LPA_SourceImageFormat|undefined} options.format 获取到的图片的数据格式，具体可参考 {
 *          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 En
 *       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;
/**
 * 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;
/**
 * 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
 * 当前还有未打印的任务，已有打印数据将会被全部丢弃。
 */
startJob(options: LPA_JobOptions): boolean;
/**
 * 提交打印任务，进行真正的打印。
 */
commitJob(options?: PrintOptions | ((success: boolean) => void)): void;
/**
 * 得到刚完成的打印任务的打印任务信息。
 */
getPageInfo(): LPA_PageInfo;
/**
 * 得到刚完成的打印任务的页面图片数据。
 */
getPageImage(options: LPA_PageImageOptions): LPA_PageImage;
/**
 * 开始一打印页面。
 */
startPage(): Promise<boolean>;
/**
 * 结束一打印页面。
 */
endPage(): Promise<boolean>;
/**
 * 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} 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.FON
* @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.Re
* @param {LPA_AutoReturnMode|undefined} options.autoReturn 自动换行模式，默认为{@link LPA_Aut
*          {@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，leading
*          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，<
* @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、
* @param {number[]|string|undefined} regionCorners 显示区域四个角的删除矩形，分别为左上、右上、
*          `[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：`[Wi
* @param {number[]|string|undefined} regionRightUpCorner 显示区域右上角的删除矩形，格式为：`[W
* @param {number[]|string|undefined} regionRightBottomCorner 显示区域右下角的删除矩形，格式为:
* @param {number[]|string|undefined} regionLeftBottomCorner 显示区域左下角的删除矩形，格式为:
* @param {number[]|string|undefined} regionLeftBorders 显示区域左边的删除矩形，最多支持删除两个
*          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
*          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
*          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
*          默认为：{@link LPA_ItemAlignment.Start}，表示居上对齐。
*/
```

```
drawText(options: DrawTextOptions): boolean;
/**
 * 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.LI
 * @param {string|undefined} options.fontName 一维码中供人识读文本的字体名称，默认为{@link CONS
 * @param {LPA_FontStyle|undefined} options.fontStyle 一维码供人识读文本的字体风格，默认为{@lin
 * @param {LPA_ItemAlignment|undefined} options.textAlignment 一维码供人识读文本的水平对齐方式，
 *          >= 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}，t
 * @param {LPA_BarcodeFlags|undefined} options.barcodeFlags 一维码编码参数标志，值参考{@link L
 * @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
 * @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，
 * @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
 *          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
 *          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
 * @param {LPA_QRTextEncoding|undefined} options.textEncoding 字符串编码方式，值参考{@link LPA
 * @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
 * @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_QRE
 *          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_QREccLe
 * @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
```

```
 *          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
 *          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
 * @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，
 * @param {LPA_P417EncodeMode|undefined} options.encodeMode 二维码编码模式，值参考{@link LPA_P
 *          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
 * @param {LPA_P417EccLevel|undefined} options.eccLevel 二维码纠错模式，值参考{@link LPA_P417E
 * @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
 *          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
 *          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
 * @param {LPA_DMTextEncoding|undefined} options.textEncoding 字符串编码方式，值参考{@link LPA
 * @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
 * @param {number|number} options.symbolShape
 * @param {LPA_DMEncodeMode|undefined} options.encodeMode 二维码编码模式，值参考{@link LPA_DME
 *          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
 *          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
 *          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.REC
 * @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 CONSTAN
 * @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 CONSTAN
 * @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 CONSTAN
 * @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.R
 * @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.L
```

```
 * @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.RADIU
 * @param {number|undefined} options.lineWidth 圆形边框宽度，单位毫米，值默认为{@link CONSTANTS
 * @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 C
 * @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;
/**
 *  打印指定的URL图片。
 *
 * @param {DrawImageUrlOptions} 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
 * @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
 * @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
 * @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）
 *                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: DrawImageUrlOptions): boolean;
/**
 * 绘制图片对象。
 *
 * @param {DrawImageDataOptions} options 图片对象绘制选项。
 *
 * @param {any} options.data 图片数据，一般情况下是图片base64字符串。
 * @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
 * @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
 * @param {number|undefined} options.threshold 图片黑白转换的灰度阀值，默认为{@link CONSTANTS.
 *          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_
 * @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;
/**
 * 直接打印指定位图对象。
 *
 * @param {PrintImageOptions} options 图片打印相关选项。
 *
 * @param {string} 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_SourceI
 *          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
 * @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
 * @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
 * @param {number|undefined} options.threshold 图片进行黑白转换时的阈值，默认为{@link CONSTANTS
 * @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
 * @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): void;
    /**
     * 根据给定的标签配置信息，打印整个打印任务。
     *
     * @param {PrintJobOptions} options 打印配置信息。
     *
     * @returns 打印成功与否
     */
    print(options: PrintJobOptions): void;
}
```

修改记录

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