Android SDK - Content Layout

1. Set label size and view direction

The rotation angle direction when switching from the label view to the edit view is counterclockwise. The width and height in the interface parameters are taken from the edit view.

1.1 Taking a 0° rotation as an example

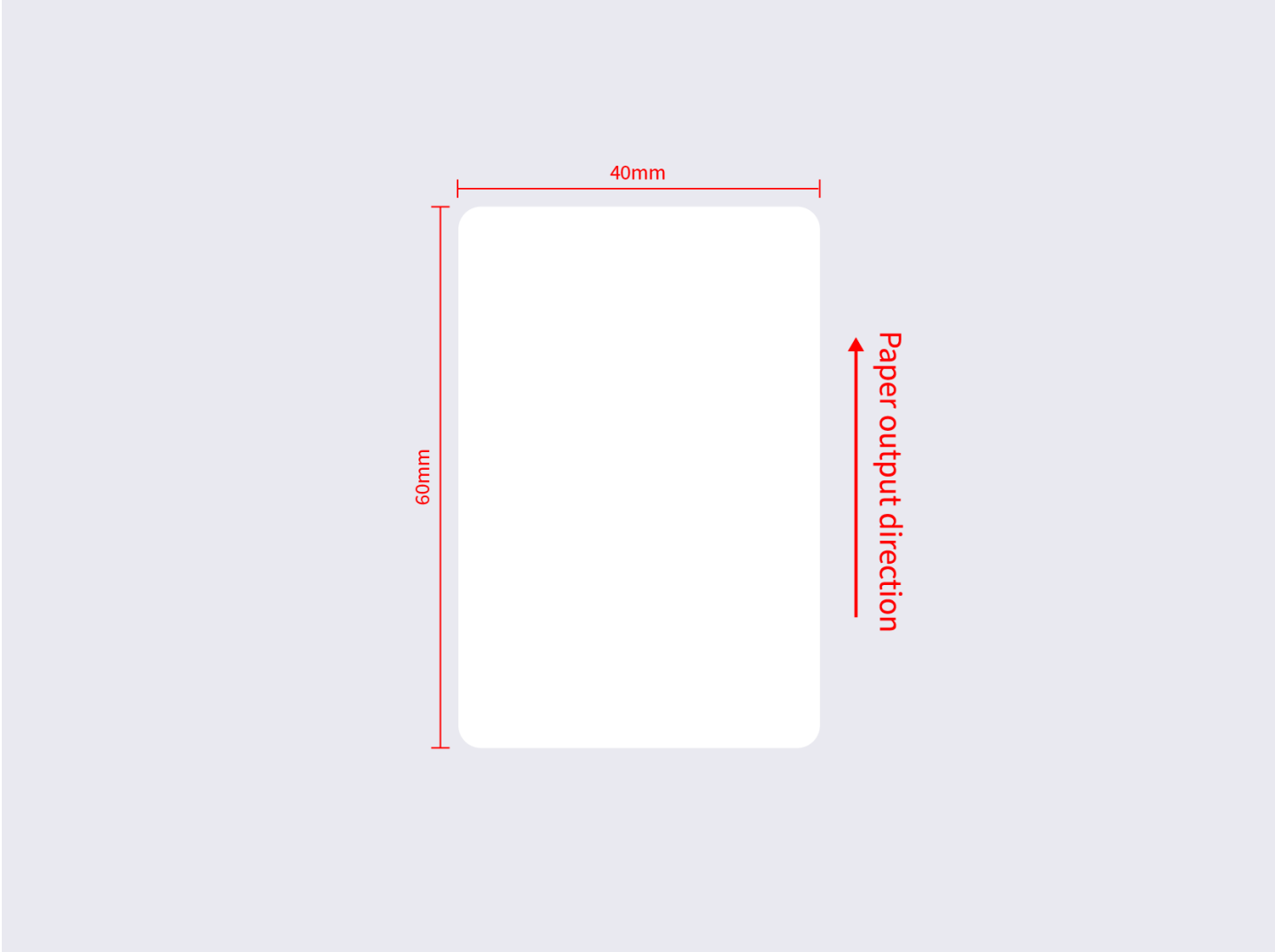
Take a label with dimensions of 40-60 (unit: mm) as an example

Sample code is as follows:

代码块

```
/*
* Set the canvas size
*
* @param width The width of the canvas
* @param height The height of the canvas
* @param orientation The rotation angle of the canvas
* @param fontDir The font path is currently unavailable, just use the default ""
*
*/
api.drawEmptyLabel(40, 60, 0, "");
```

Example Diagram of Edit View:



1.2 Taking a 90° rotation as an example

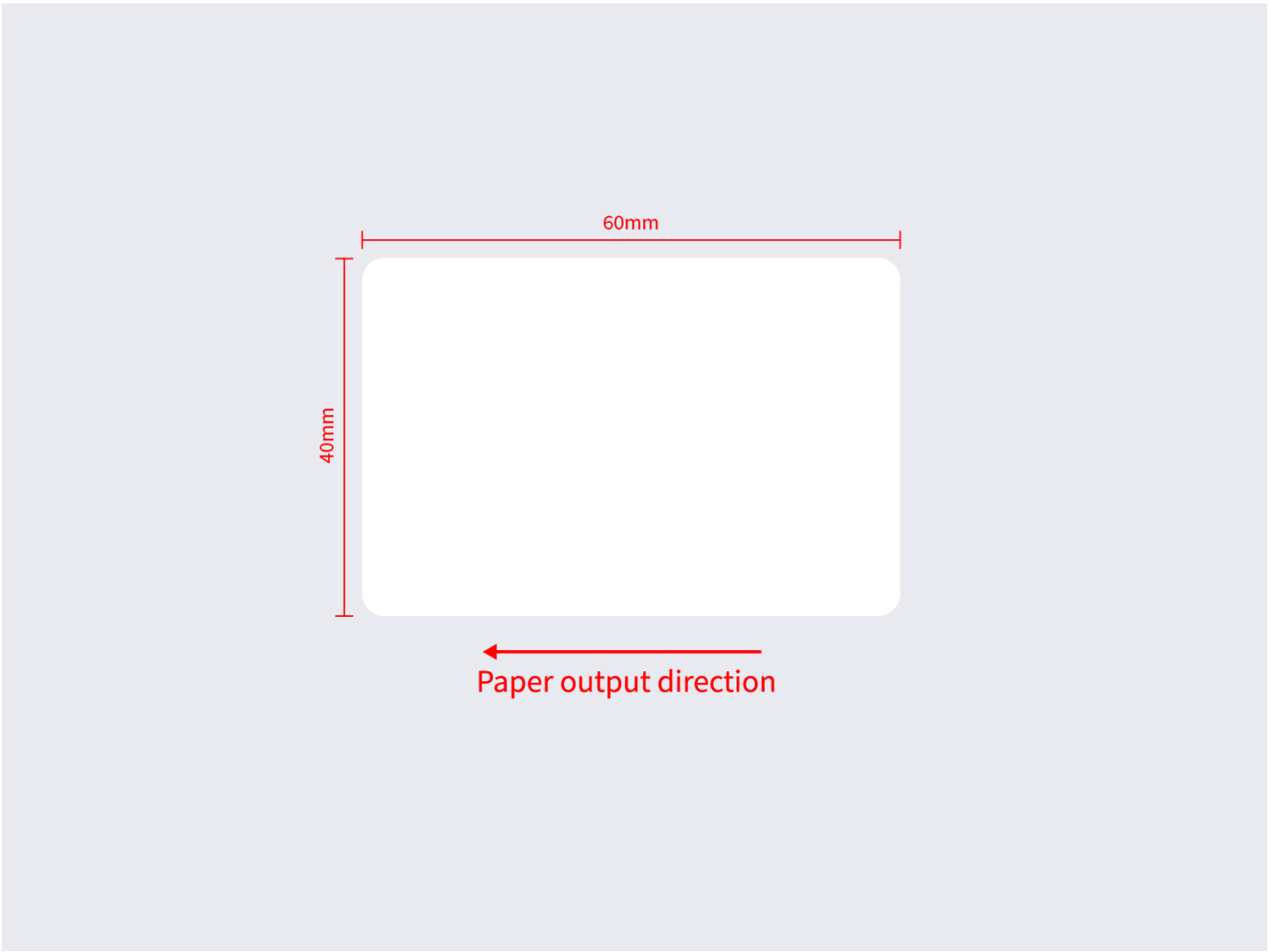
Take a label with dimensions of 40-60 (unit: mm) as an example

Sample code is as follows:

代码块

```
/*
* Set the canvas size
*
* @param width The width of the canvas
* @param height The height of the canvas
* @param orientation The rotation angle of the canvas
* @param fontDir The font path is currently unavailable, just use the default
""
*
*/
api.drawEmptyLabel(60, 40, 90, "");
```

Example Diagram of Edit View:



2. Element Drawing

The drawing of the element is based on the current editing view. The coordinates of the upper-left corner of the editing view are 0,0. Alignment parameters or interfaces are only valid for text boxes, representing the alignment of text relative to the text box.

2.1 Text Box Drawing

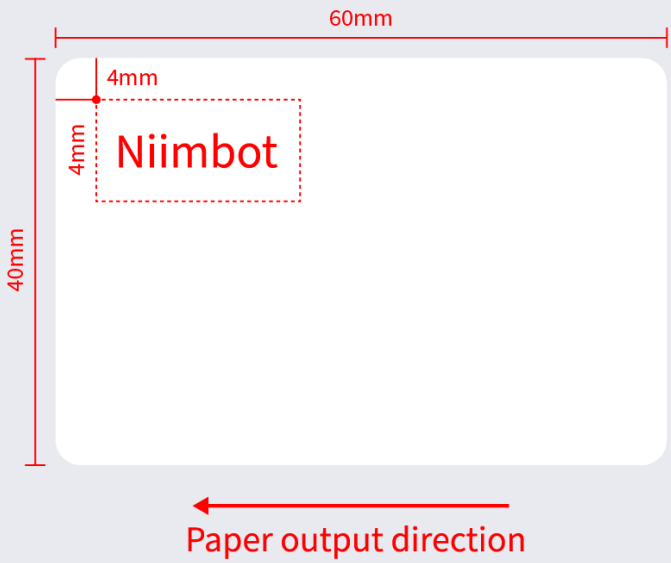
Sample code is as follows:

代码块

```
api.drawEmptyLabel(60, 40, 90, "");
/**
 * Draw text
 *
 * @param x Position x
 * @param y Position y
 * @param width Width
 * @param height Height
 * @param value Content
```

```
 * @param fontFamily Font family name, use the default font when the font is not passed or is an empty string, temporarily use the default font
 * @param fontSize Font size
 * @param rotate Rotation
 * @param textAlignHorizontal Horizontal alignment: 0: Left alignment 1: Center alignment 2: Right alignment
 * @param textAlignVertical Vertical alignment: 0: Top alignment 1: Vertical center 2: Bottom alignment
 * @param lineModel 1: Fixed width and height, content size self-adapts (font size/character spacing/line spacing scaled proportionally) 2: Fixed width, height self-adapts 3: Fixed width and height, add... after exceeding content 4: Fixed width and height, directly crop exceeding content 6: Fixed width and height, automatically shrink when content exceeds preset width and height (font size/character spacing/line spacing scaled proportionally)
 * @param letterSpace Standard spacing between letters, unit mm
 * @param lineSpace Line spacing (multiple distance), unit mm
 * @param mFontStyles Font styles [Bold, Italic, Underline, Delete underline (reserved)]
 */
api.drawLabelText(4.0F, 4.0F, 20.0F, 10.0F,  "文本内容", "", 3.0F, 0, 1, 1, 6, 0.0F, 1.0F, new boolean[]{false, false, false, false});
```

The renderings are as follows (the actual drawing effect of the text box does not include border lines):



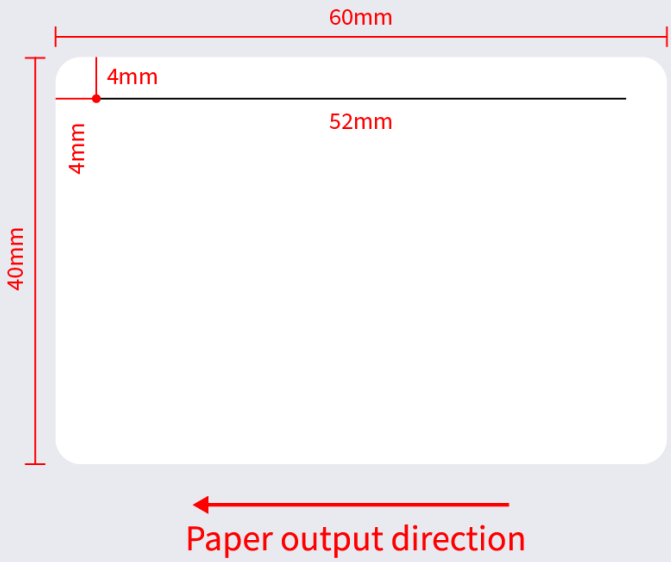
2.2 Line Drawing

Sample code is as follows:

代码块

```
api.drawEmptyLabel(60, 40, 90, "");
/**
 * Draw a line
 *
 * @param x Horizontal coordinate
 * @param y Vertical coordinate
 * @param width Width, unit: mm
 * @param height Height, unit: mm
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param lineType Line type, 1: solid line, 2: dashed line type, solid-to-empty ratio 1:1
 * @param dashWidth Width of the dashed line, [length of solid segment, length of empty segment]
 */
api.drawLabelLine(4.0F, 4.0F, 52.0F, 0.5F, 0, 1, new float[]{});
```

The renderings are as follows:



2.3 One-dimensional Code (Barcode) Drawing

代码块

```
api.drawEmptyLabel(60, 40, 90, "");
/**
 * Draw a one-dimensional barcode
 *
 * @param x Horizontal coordinate
 * @param y Vertical coordinate
 * @param width Width, unit: mm
 * @param height Height, unit: mm
 * @param codeType One-dimensional barcode type 20: CODE128, 21: UPC-A, 22: UPC-E, 23: EAN8, 24: EAN13,
 * 25: CODE93, 26: CODE39, 27: CODEBAR, 28: ITF25
 * @param value Text content
 * @param fontSize Text font size
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param textHeight Text height
```

```
15    * @param textPosition Text position, int, display position of one-dimensional
      barcode text recognition code, 0: display below, 1: display above, 2: do not
      display
16    */
17    api.drawLabelBarCode(4.0F, 4.0F, 38.0F, 15.0F,20 "12123141341", 2.0F,0,2.5F,
      0);
```

The renderings are as follows (the actual drawing effect of the text box does not include border lines):

2.4 QR Code Drawing

代码块

```
1    api.drawEmptyLabel(60, 40, 90, "");
2    /**
3     * Draw a QR code
4     *
5     * @param x Horizontal coordinate
6     * @param y Vertical coordinate
7     * @param width Width, unit: mm
```

```
8    * @param height Height, unit: mm
9    * @param value Text content
10   * @param codeType One-dimensional code type, 31: QR_CODE, 32: PDF417, 33:
     DATA_MATRIX, 34: AZTEC
11   * @param rotate Rotation angle, only supports 0, 90, 180, 270
12   */
13  api.drawLabelQrCode(4.0F, 4.0F, 15.0F, 15.0F, "Niimbot SDK", 31, 0);
```

The renderings are as follows :

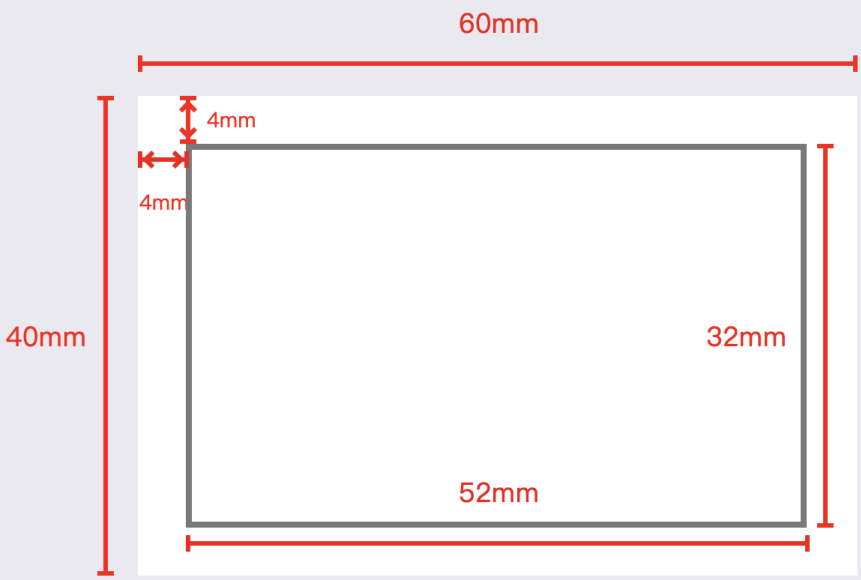
2.5 Graphics Drawing

代码块

```
1   api.drawEmptyLabel(60, 40, 90, "");
2   /**
3    * Draw a shape
4    *
5    * @param x Horizontal coordinate
```

```
 * @param y Vertical coordinate
 * @param width Width, unit: mm
 * @param height Height, unit: mm
 * @param graphType Graph type, 1. Circle 2. Ellipse 3. Rectangle 4. Rounded rectangle
 * @param rotate Rotation angle, only supports 0, 90, 180, 270
 * @param cornerRadius Corner radius
 * @param lineWidth Line width
 * @param lineType Line type, 1: Solid line, 2: Dashed line type, solid-to-empty ratio 1:1
 * @param dashwidth Width of dashed line, [length of solid segment, length of empty segment]
 */
api.drawLabelGraph(4.0F, 4.0F, 52.0F, 32.0F, 3, 0, 0.0F,0.5F, 1, new float[] {})
```

The renderings are as follows:



2.6 Image Drawing

Graphic drawing needs to be converted to base64, similar to other elements. For details, please refer to the documentation in conjunction with the DEMO example. If you have any questions, you can directly consult the system access technical support staff.

3. Conversion between Physical Dimensions of Labels and Image Dimensions

Reference: Printer Parameter Query - Case Number 0001

IV. Precautions for Printing Images (Non-LOGO)

Some users generate images themselves for label printing. Please note the following points

4.1 Image Dimensions

The image size should be passed in as the actual size after converting the physical size of the label into pixels. If the aspect ratio is not 1:1, issues such as scaling, blurriness, and incomplete printing may occur during printing.

4.2 Image Background

The SDK on some terminals may not handle transparent images properly, resulting in black blocks during printing. Some dark backgrounds may also cause black blocks to appear during printing. **When printing images, the background is required to be set to white.**