Draw the rectangle

Example Introduction

In this section, we introduce the draw_rectangle() method for drawing lines

API Documentation

`draw_rectangle`

```
image.draw_rectangle(x, y, w, h[, color[, thickness=1[, fill=False]]])
```

Draw a rectangle on the image. You can pass in parameters `x, y, w, h` separately or as a tuple `(x, y, w, h)`.

- **color**: An RGB888 tuple representing the color, suitable for grayscale or RGB565 images, and the default is white. For grayscale images, you can also pass pixel values (range 0-255); for RGB565 images, you can pass byte-flipped RGB565 values.
- **thickness**: Controls the pixel width of the rectangle border, the default is 1.
- **fill**: When set to `True`, the interior of the rectangle will be filled, the default is `False`.

This method returns an image object, allowing other methods to be called through chaining.

Compressed images and Bayer format images are not supported.

Sample code

```
# Import required modules
# 导入所需的模块
import time, os, urandom, sys, math

# Import display and media related modules
# 导入显示和媒体相关模块
from media.display import *
from media.media import *

# Define display resolution constants
# 定义显示分辨率常量
DISPLAY_WIDTH = 640
DISPLAY_HEIGHT = 480

def display_test():
    """
    Function to test display functionality
```

```
    测试显示功能的函数
    """

    # Create main background image with white color
    # 创建白色背景的主图像
    img = image.Image(DISPLAY_WIDTH, DISPLAY_HEIGHT, image.ARGB8888)
    img.clear()
    img.draw_rectangle(0, 0, DISPLAY_WIDTH, DISPLAY_HEIGHT,color=
(255,255,255),fill=True)

    # Initialize display with ST7701 driver
    # 使用ST7701驱动初始化显示器
    Display.init(Display.ST7701, width = DISPLAY_WIDTH, height = DISPLAY_HEIGHT,
to_ide = True)
    # Initialize media manager
    # 初始化媒体管理器
    MediaManager.init()

    try:
        # Main decorative rectangle - centered
        # Use different shades of sky blue to create a sense of hierarchy
        # Dark sky blue - central horizontal line
        # 主要的装饰性矩形图案 - 居中调整
        # 使用不同深浅的天蓝色营造层次感
        # 深色天蓝色 - 中央水平线
        img.draw_rectangle(120, 160, 400, 160, color=(0, 191, 255), thickness=2)

        # Small decorative rectangle on the left
        # 左侧装饰性小矩形
        img.draw_rectangle(120, 160, 50, 50, color=(135, 206, 235), fill=True)
 # 浅天蓝色 Light sky blue
        img.draw_rectangle(120, 270, 50, 50, color=(0, 191, 255), fill=True)
 # 天蓝色  sky blue

        # Small decorative rectangle on the right
        # 右侧装饰性小矩形
        img.draw_rectangle(470, 160, 50, 50, color=(0, 191, 255), fill=True)
 # 天蓝色 sky blue
        img.draw_rectangle(470, 270, 50, 50, color=(135, 206, 235), fill=True)
 # 浅天蓝色 Light sky blue

        #Central decorative rectangle
        # 中央装饰性矩形
        img.draw_rectangle(220, 200, 200, 80, color=(0, 191, 255), thickness=2)
        img.draw_rectangle(240, 220, 160, 40, color=(135, 206, 235), fill=True)

        # Thin rectangle with connecting line effect
        # 连接线效果的细长矩形
        img.draw_rectangle(170, 235, 50, 10, color=(0, 191, 255), fill=True)
        img.draw_rectangle(420, 235, 50, 10, color=(0, 191, 255), fill=True)

        # Small decorative blocks
        # 点缀性的小方块
        img.draw_rectangle(200, 180, 15, 15, color=(173, 216, 230), fill=True)
 # 最浅天蓝色 Lightest sky blue
        img.draw_rectangle(425, 180, 15, 15, color=(173, 216, 230), fill=True)
        img.draw_rectangle(200, 285, 15, 15, color=(173, 216, 230), fill=True)
        img.draw_rectangle(425, 285, 15, 15, color=(173, 216, 230), fill=True)
```

```
        # Update display with background image
        # 更新显示背景图像
        Display.show_image(img)
        while True:
            time.sleep(2)

    except KeyboardInterrupt as e:
        print("user stop: ", e)
    except BaseException as e:
        print(f"Exception {e}")

    # Cleanup and deinitialize display
    # 清理并反初始化显示器
    Display.deinit()
    os.exitpoint(os.EXITPOINT_ENABLE_SLEEP)
    time.sleep_ms(100)
    # Release media resources
    # 释放媒体资源
    MediaManager.deinit()

if __name__ == "__main__":
    # Enable exit points and run display test
    # 启用退出点并运行显示测试
    os.exitpoint(os.EXITPOINT_ENABLE)
    display_test()
```

Routine running effect