Draw the crosshair

Example Introduction

In this section, we introduce the draw_cross() method for drawing a crosshair

API Documentation

```
image.draw_cross(x, y[, color[, size=5[, thickness=1]]])
```

Draw a crosshair on the image. Parameters can be passed in `x, y` separately or as a tuple `(x, y)`.

- **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.
- **size**: Controls the size of the crosshair, default is 5.
- **thickness**: Controls the pixel width of the crosshair, default is 1.

This method returns an image object, allowing other methods to be called in a chain.

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:
        # Center cross - coordinates adjusted to the center
        img . draw_cross ( 320 , 240 , color =( 0 , 191 , 255 ), size = 40 ,
thickness = 3 )

        # Inner circle small cross surround - coordinates adjusted to center
        for  i  in  range ( 8 ):
            angle = i  * ( 360  /  8 )   # Evenly distributed on the
circumference
            x = int ( 320  +  50  *  math . cos ( math . radians ( angle )))
            y = int ( 240  +  50  *  math . sin ( math . radians ( angle )))
            img . draw_cross ( x , y , color =( 135 , 206 , 235 ), size = 15 ,
thickness = 2 )

        # Smaller cross on the outer circle - coordinates adjusted to center
        for  i  in  range ( 12 ):
            angle = i  * ( 360  /  12 )
            x = int ( 320  +  80  *  math . cos ( math . radians ( angle )))
            y = int ( 240  +  80  *  math . sin ( math . radians ( angle )))
            img . draw_cross ( x , y , color =( 173 , 216 , 230 ), size = 10 ,
thickness = 1 )

        # Decorative crosses at the four corners - coordinates adjusted to
center
        img . draw_cross ( 240 , 140 , color =( 0 , 191 , 255 ), size = 25 ,
thickness = 2 )
        img . draw_cross ( 400 , 140 , color =( 0 , 191 , 255 ), size = 25 ,
thickness = 2 )
        img . draw_cross ( 240 , 340 , color =( 0 , 191 , 255 ), size = 25 ,
thickness = 2 )
        img . draw_cross ( 400 , 340 , color =( 0 , 191 , 255 ), size = 25 ,
thickness = 2 )

        # Center dotted with a small cross - coordinates adjusted to the center
        img . draw_cross ( 320 , 240 , color =( 173 , 216 , 230 ), size = 8 ,
thickness = 1 )

        # 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()
```

Example running effect

As you can see, we use crosshairs of different sizes and tones to construct a radial drawing in the center of the screen.