

Python DearPyGui 常用控件一



何小有 [关注](#)

2020.11.04 20:33:09 字数 1,074 阅读 22

菜单栏

菜单栏是一个 GUI 应用中重要的控件，始终显示在窗口顶部，并具有三个主要部分：

- menu_bar —— 主菜单功能区
- menu —— 下拉菜单或子菜单
- add_menu_item —— 可以运行回调或可以操作的项目

菜单可以根据需要嵌套，而且任何控件都可以添加到菜单中，例如下面栗子中的“控件列表”菜单。

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 add_additional_font('三极中柔宋.ttf', 18, glyph_ranges='chinese_simplified_common')
5
6 def print_me(sender, data):
7     log_debug(f"菜单项: {sender}")
8
9 show_logger()
10
11 with window("Tutorial"):
12     with menu_bar("Main Menu Bar"):
13         with menu("文件"):
14             add_menu_item("保存", callback=print_me)
15             add_menu_item("另存为", callback=print_me)
16
17         with menu("设置"):
18             add_menu_item("设置项 1", callback=print_me)
19             add_menu_item("设置项 2", callback=print_me)
20
21         add_menu_item("帮助", callback=print_me)
22
23         with menu("控件列表"):
24             add_checkbox("选择", callback=print_me)
25             add_button("点击", callback=print_me)
26             add_color_picker4("选择颜色", callback=print_me)
27
28 start_dearpygui()
```



何小有

[关注](#)

总资产49 (约3.39元)

[Python DearPyGui 进阶](#)

阅读 38

[Python DearPyGui 常用控件二](#)

阅读 7

[Python DearPyGui 常用控件一](#)

阅读 22

推荐阅读

[Android TV开发之使用Leanback传输控件](#)

阅读 41

[12 个实用的前端开发技巧总结](#)

阅读 1,036

[Android 11 新特性 \(含Android Studio 4.0 新特性\)](#)

阅读 6,787

[CSS也可以这么美之登录页面---第二弹](#)

阅读 979

[vue 实现手写电子签名](#)

阅读 652

写下你的评论...

评论0

赞

...

Dear PyGui Logger



dearpygui_menubar.png

目录对话框

通过 `select_directory_dialog` 来调用目录对话框，而且必须为其提供回调方法。回调方法返回的 `data` 参数中将包含 **目录路径** 和 **文件夹路径**。通常，目录对话框是由另一个控件（例如下面栗子中的按钮）调用的。

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 add_additional_font('三极中柔宋.ttf', 18, glyph_ranges='chinese_simplified_common')
5
6 def directory_picker(sender, data):
7     select_directory_dialog(callback=apply_selected_directory)
8
9 def apply_selected_directory(sender, data):
10     log_debug(data)
11     directory = data[0]
12     folder = data[1]
13     set_value("目录", directory)
14     set_value("文件夹", folder)
15     set_value("文件夹路径", f"{directory}\\{folder}")
16
17 show_logger()
18
19 with window("Tutorial"):
20     add_button("目录选择器", callback=directory_picker)
21     add_text("目录路径: ")
22     add_same_line()
23     add_label_text("##dir", source="目录", color=[255, 0, 0])
24     add_text("文件夹: ")
25     add_same_line()
26     add_label_text("##folder", source="文件夹", color=[255, 0, 0])
27     add_text("文件夹路径: ")
28     add_same_line()
29     add_label_text("##folderpath", source="文件夹路径", color=[255, 0, 0])
30
31 start_dearpygui()
```



dearpygui_selectdirectorydialog.png

文件对话框

写下你的评论...

评论0

赞



文件扩展名的过滤，控制显示哪些后缀名的文件。

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 add_additional_font('三极中柔宋.ttf', 18, glyph_ranges='chinese_simplified_common')
5
6 def file_picker(sender, data):
7     open_file_dialog(callback=apply_selected_file, extensions="*.py")
8
9 def apply_selected_file(sender, data):
10    log_debug(data)
11    directory = data[0]
12    file = data[1]
13    set_value("目录", directory)
14    set_value("文件", file)
15    set_value("文件路径", f"{directory}\\{file}")
16
17 show_logger()
18
19 with window("Tutorial"):
20    add_button("文件选择器", callback=file_picker)
21    add_text("目录路径: ")
22    add_same_line()
23    add_label_text("##filedir", source="目录", color=[255, 0, 0])
24    add_text("文件: ")
25    add_same_line()
26    add_label_text("##file", source="文件", color=[255, 0, 0])
27    add_text("文件路径: ")
28    add_same_line()
29    add_label_text("##filepath", source="文件路径", color=[255, 0, 0])
30
31 start_dearpygui()
```

dearpygui_openfiledialog.png

绘制图表

Dear PyGui 具有 `simple_plot`（简单绘图）和 `plot`（绘图）两个绘图方式，两者都是动态的。

`simple_plot`（简单绘图）接受列表参数，并基于列表中的数据数据绘制 y轴 数据，可以是折线图或直方图。

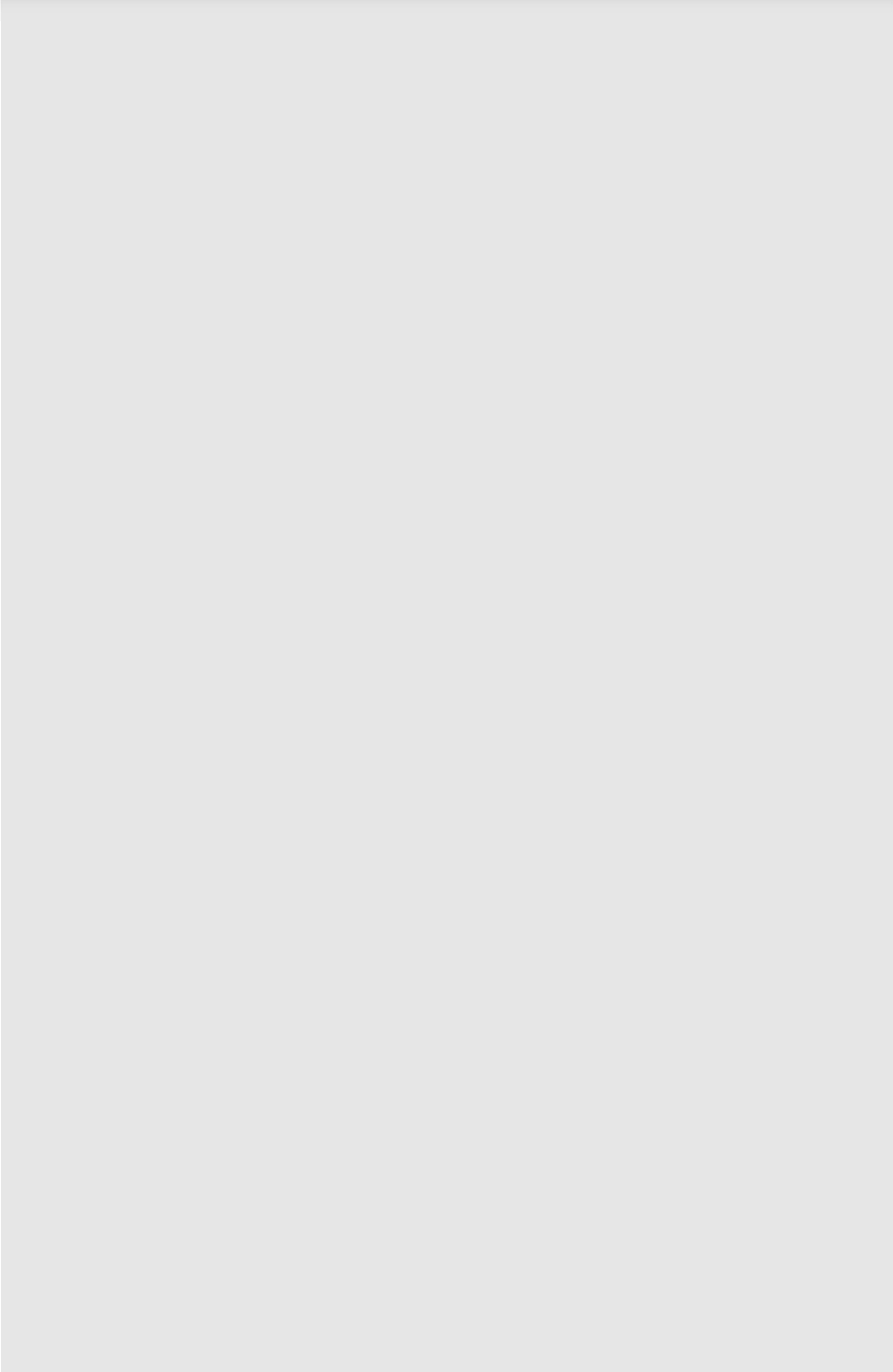
```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 with window("Tutorial"):
5     add_simple_plot("Simpleplot1", value=[0.3, 0.9, 0.5, 0.3], height=300)
```

写下你的评论...

评论0

赞

...



dearpygui_addsimpleplot.png

而 `plot`（绘图）则具有更多的功能，绘图同时使用 `x`轴 和 `y`轴 坐标，使用 `add_plot` 方法创建，然后将数据作为线形图或散布图添加，`plot`（绘图）的特点有：

- 单击 & 拖动 —— 平移绘图
- 单击 & 拖动轴 —— 在一个方向上平移绘图
- 双击 —— 将绘图缩放并移动到数据区域
- 右键单击 & 拖动 —— 缩放区域
- 双右键单击 —— 打开设置
- Shift + 右键单击 & 拖动 —— 缩放并填充当前轴的区域
- 滚动鼠标滚轮 —— 缩放

写下你的评论...

 评论0

 赞

...

```

1 from dearpygui.core import *
2 from dearpygui.simple import *
3 from math import cos, sin
4
5 def plot_callback(sender, data):
6     clear_plot("Plot")
7     data1 = []
8     for i in range(0, 100):
9         data1.append([3.14 * i / 180, cos(3 * 3.14 * i / 180)])
10    data2 = []
11    for i in range(0, 100):
12        data2.append([3.14 * i / 180, sin(2 * 3.14 * i / 180)])
13    add_line_series("Plot", "Cos", data1, weight=2, color=[0, 0, 255, 100])
14    add_shade_series("Plot", "Cos", data1, weight=2, fill=[255, 0, 0, 100])
15    add_scatter_series("Plot", "Sin", data2, color=[0, 255, 0, 100])
16
17 with window("Tutorial"):
18     add_button("Plot data", callback=plot_callback)
19     add_plot("Plot", height=-1)
20
21 start_dearpygui()

```

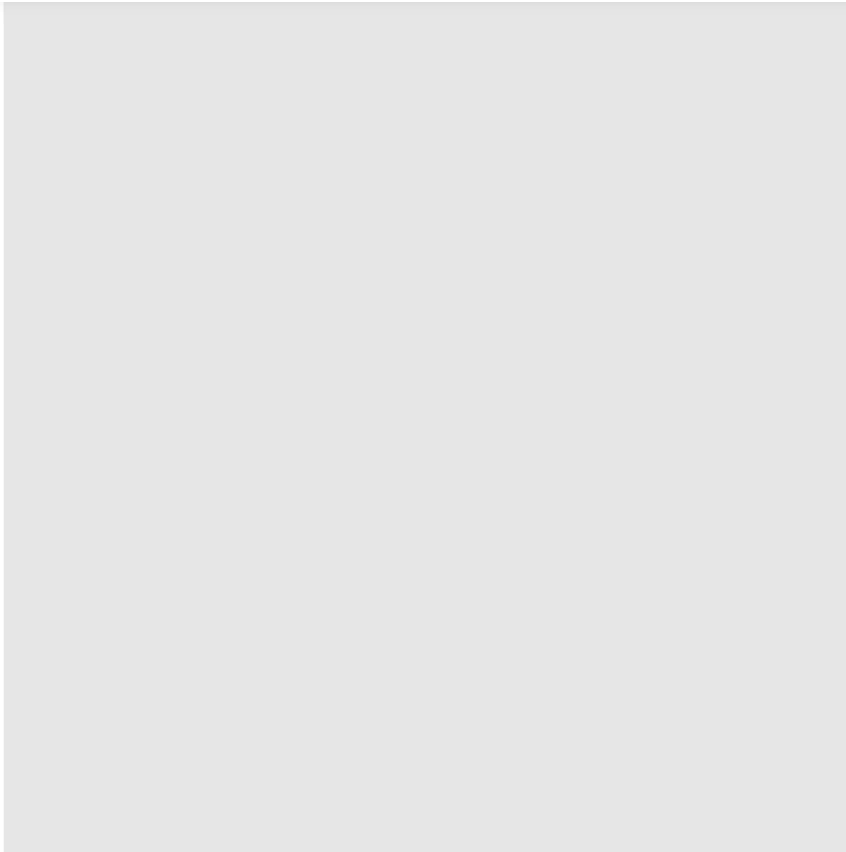
dearpygui_addplot.png

通过 `set_value` 方法可以更改绘图调用的值，使 `simple_plot`（简单绘图）实现动态实时绘制。

```

1 from dearpygui.core import *
2 from dearpygui.simple import *
3 from math import sin
4
5 def on_render(sender, data):
6     frame_count = get_data("frame_count")
7     frame_count += 1
8     add_data("frame_count", frame_count)
9     plot_data = get_value("plot_data")
10    if len(plot_data) > 100:
11        plot_data.pop(0)
12    plot_data.append(sin(frame_count / 30))
13    set_value("plot_data", plot_data)
14
15 with window("Tutorial"):
16     add_simple_plot("Simple Plot", source="plot_data", minscale=-1.0, maxscale=1.0, height=300)
17     add_data("frame_count", 0)
18     set_render_callback(on_render)
19
20 start_dearpygui()

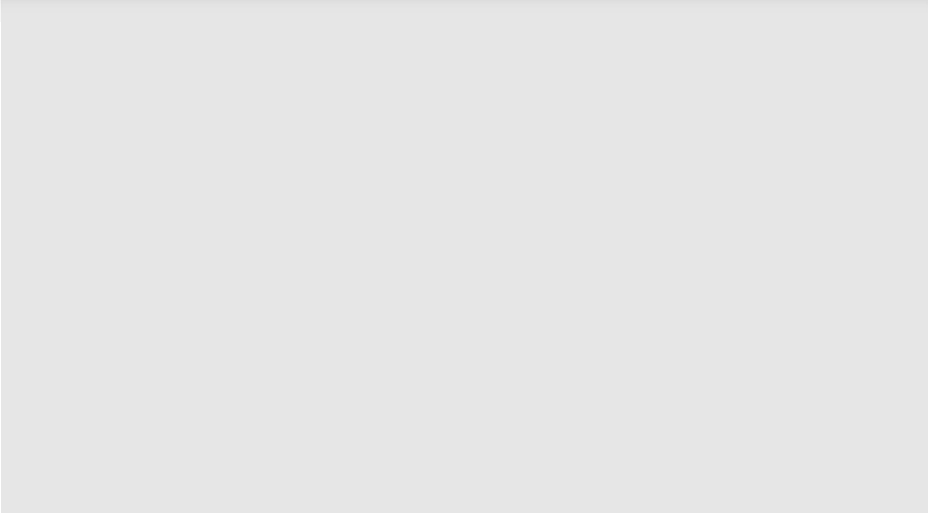
```



dearpygui_simpleplotsetvalue.png

同样的，`plot`（绘图）也可以动态实时绘制，举个栗子，我们使用 `set_render_callback` 设置一个渲染回调实现动态绘制。

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3 from math import cos
4
5 def plot_callback(sender, data):
6     # 跟踪每一帧
7     frame_count = get_data("frame_count")
8     frame_count += 1
9     add_data("frame_count", frame_count)
10    # 更新 plot_data
11    plot_data = get_data("plot_data")
12    if len(plot_data) > 2000:
13        frame_count = 0
14        plot_data.clear()
15    plot_data.append([3.14 * frame_count / 180, cos(3 * 3.14 * frame_count / 180)])
16    add_data("plot_data", plot_data)
17    # 绘制新数据
18    clear_plot("Plot")
19    add_line_series("Plot", "Cos", plot_data, weight=2)
20
21 with window("Tutorial"):
22     add_plot("Plot", height=-1)
23     add_data("plot_data", [])
24     add_data("frame_count", 0)
25     set_render_callback(plot_callback)
26
27 start_dearpygui()
```

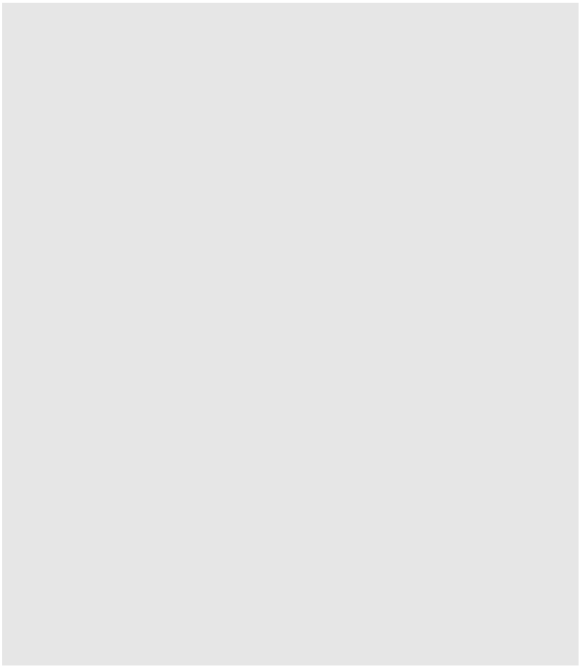


dearpygui_plotsetvalue.png

绘画与画布

Dear PyGui 有一个低级绘图 API，可以用来原始绘画、自定义控件甚至动态绘画。先通过调用 `add_drawing` 方法开始绘画，再通过调用各种绘画方法来添加笔画。需要注意的是，画布的原点位于左下角。

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 with window("Tutorial"):
5     add_drawing("Drawing_1", width=120, height=120)
6
7 draw_line("Drawing_1", [10, 10], [100, 100], [255, 0, 0, 255], 1)
8 draw_text("Drawing_1", [16, 16], "Origin", color=[250, 250, 250, 255], size=15)
9 draw_arrow("Drawing_1", [100, 65], [50, 70], [0, 200, 255], 1, 10)
10
11 start_dearpygui()
```

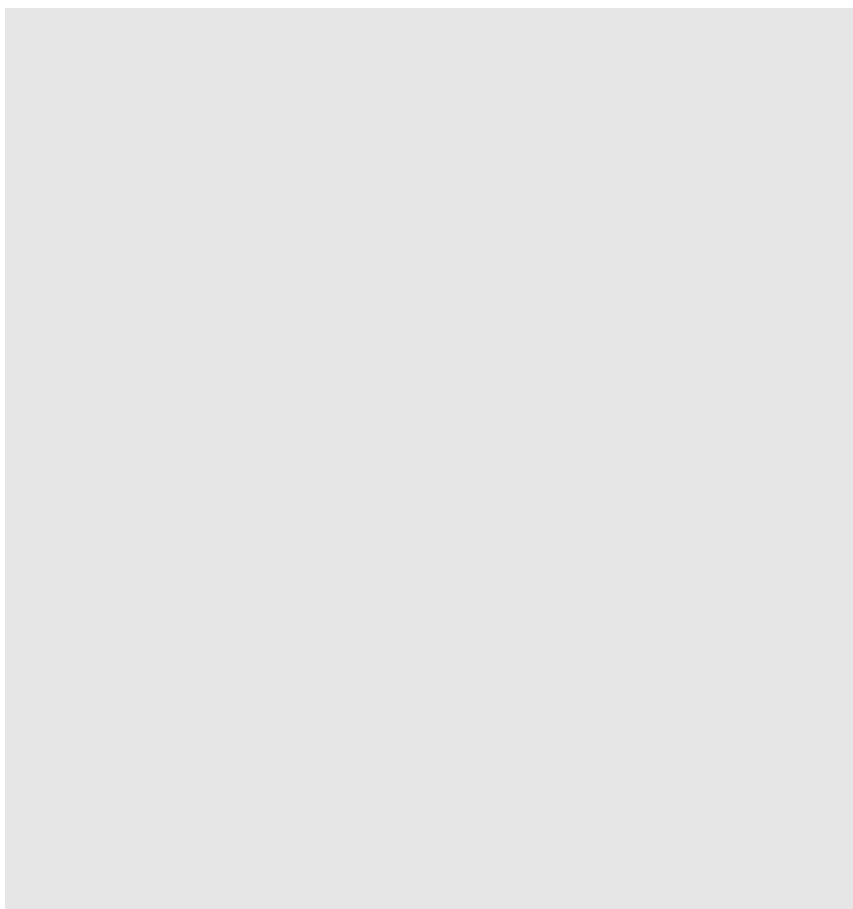


dearpygui_adddrawing.png

写下你的评论...

评论0 赞 ...

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 with window("Tutorial"):
5     add_drawing("Drawing_1", width=240, height=240)
6
7 draw_line("Drawing_1", [10, 10], [100, 100], [255, 0, 0, 255], 1)
8 draw_text("Drawing_1", [16, 16], "Origin", color=[250, 250, 250, 255], size=15)
9 draw_arrow("Drawing_1", [100, 65], [50, 70], [0, 200, 255], 1, 10)
10
11 set_drawing_origin("Drawing_1", 15, 15)
12 set_drawing_scale("Drawing_1", 2, 2)
13 set_drawing_size("Drawing_1", 250, 250)
14
15 start_dearpygui()
```



dearpygui_setdrawingorigin.png

绘画（drawing）可以显示的图像类型有 .png、.jpg、.bmp，使用时需掉用 `draw_image` 以绘制图像。通过 `pmin` 和 `pmax` 参数，我们可以将图像绘制到画布上矩形的左上和右下区域，图像会缩放自动缩放以适应指定区域。

使用 `uv_min` 和 `uv_max` 参数，我们可以控制图像要绘制到哪个区域的 **标量（scalar）**，默认情况下，`uv_min = [0,0]` 和 `uv_max = [1,1]` 将显示整个图像，而 `uv_min = [0,0]` 和 `uv_max = [0.5,0.5]` 则仅显示图形的一部分。

```
1 from dearpygui.core import *
2 from dearpygui.simple import *
3
4 with window("Tutorial"):
5     add_drawing("Drawing_1", width=700, height=700)
6
```

写下你的评论...

评论0

赞

...


```
12  
13 start_dearpygui()
```

dearpygui_pokemonpng.png

尽管我们可以通过清除和重绘整个图来实现图形的动态化，但是 DearPyGui 还提供了一种更有效的方法，要使绘画（`drawing`）动态化，应该使用 `tag` 参数标记要重绘的控件，然后，只要使用相同的标签去调用。这样，我们就能实现仅清除该控件，并将其重新绘制。

```
1 from dearpygui.core import *  
2 from dearpygui.simple import *  
3  
4 def on_render(sender, data):  
5     counter = get_data("counter")  
6     counter += 1  
7     modifier = get_data("modifier")  
8     if counter < 300:  
9         modifier += 1  
10    elif counter < 600:  
11        modifier -= 1  
12    else:  
13        counter = 0  
14        modifier = 2  
15    xpos = 15 + modifier * 1.25  
16    ypos = 15 + modifier * 1.25  
17    color1 = 255 - modifier * .8  
18    color3 = 255 - modifier * .3  
19    color2 = 255 - modifier * .8  
20    radius = 15 + modifier / 2  
21    segments = round(35 - modifier / 10)  
22    draw_circle("Drawing_1", [xpos, ypos], radius, [color1, color3, color2, 255], segments=seg  
23        tag="circle#dynamic")  
24    add_data("counter", counter)  
25    add_data("modifier", modifier)  
26
```

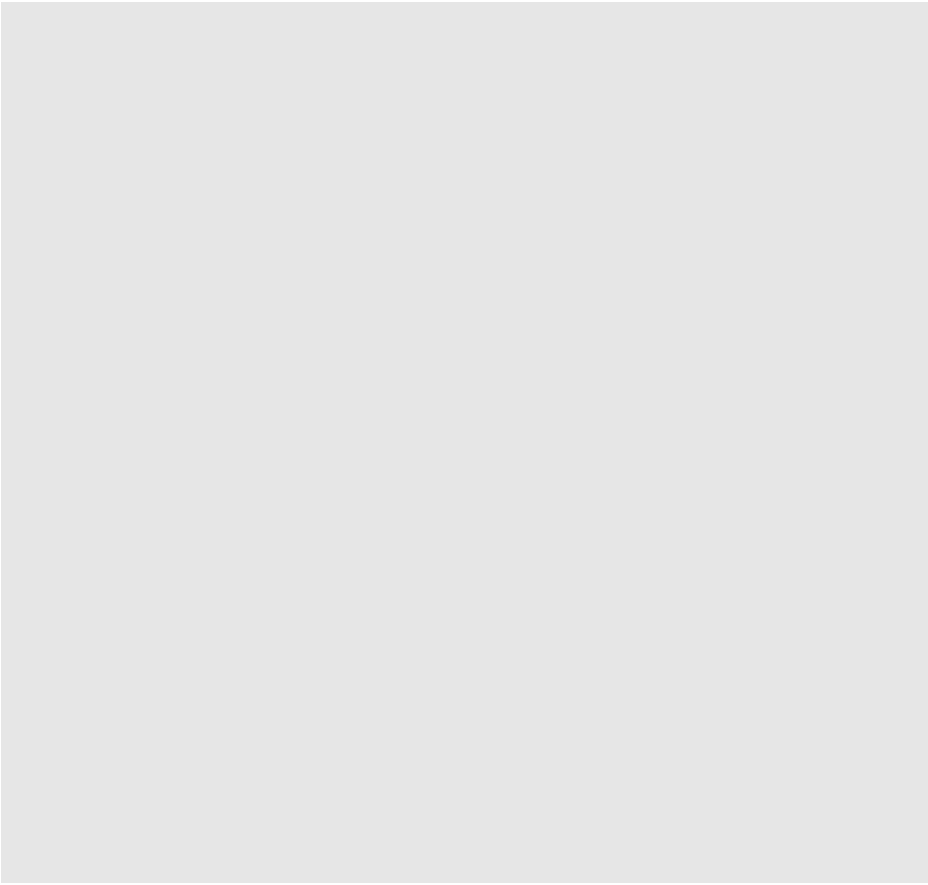
写下你的评论...

评论0

赞

...

```
33 set_render_callback(on_render)
34
35 start_dearpygui()
```



dearpygui_drawcircletag.png



"小礼物走一走，来简书关注我"

赞赏支持

还没有人赞赏，支持一下



何小有
总资产49 (约3.39元) 共写了8.8W字 获得370个赞 共323个粉丝

关注

写下你的评论...

 评论0  赞 ...