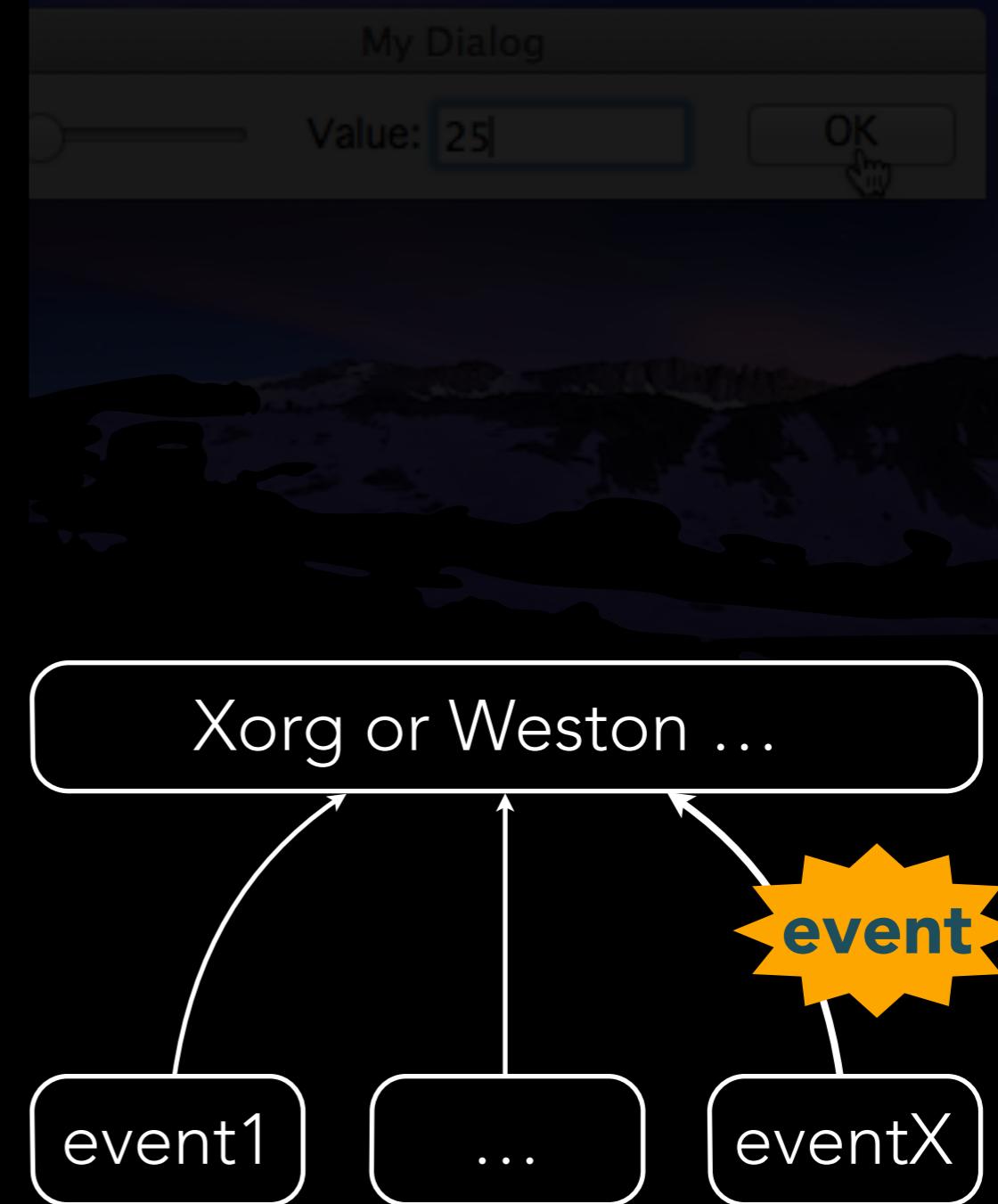


图形按钮一次点击
的背后发生了
什么？



第一步： 分派点击事件

- 图形服务器**监听**输入事件设备
- 图形服务器：Xorg Server、Weston ...
- 输入事件设备：
 - 虚拟的，或者叫做接口
 - 鼠标、键盘、触摸板 ...
 - /dev/input/eventX



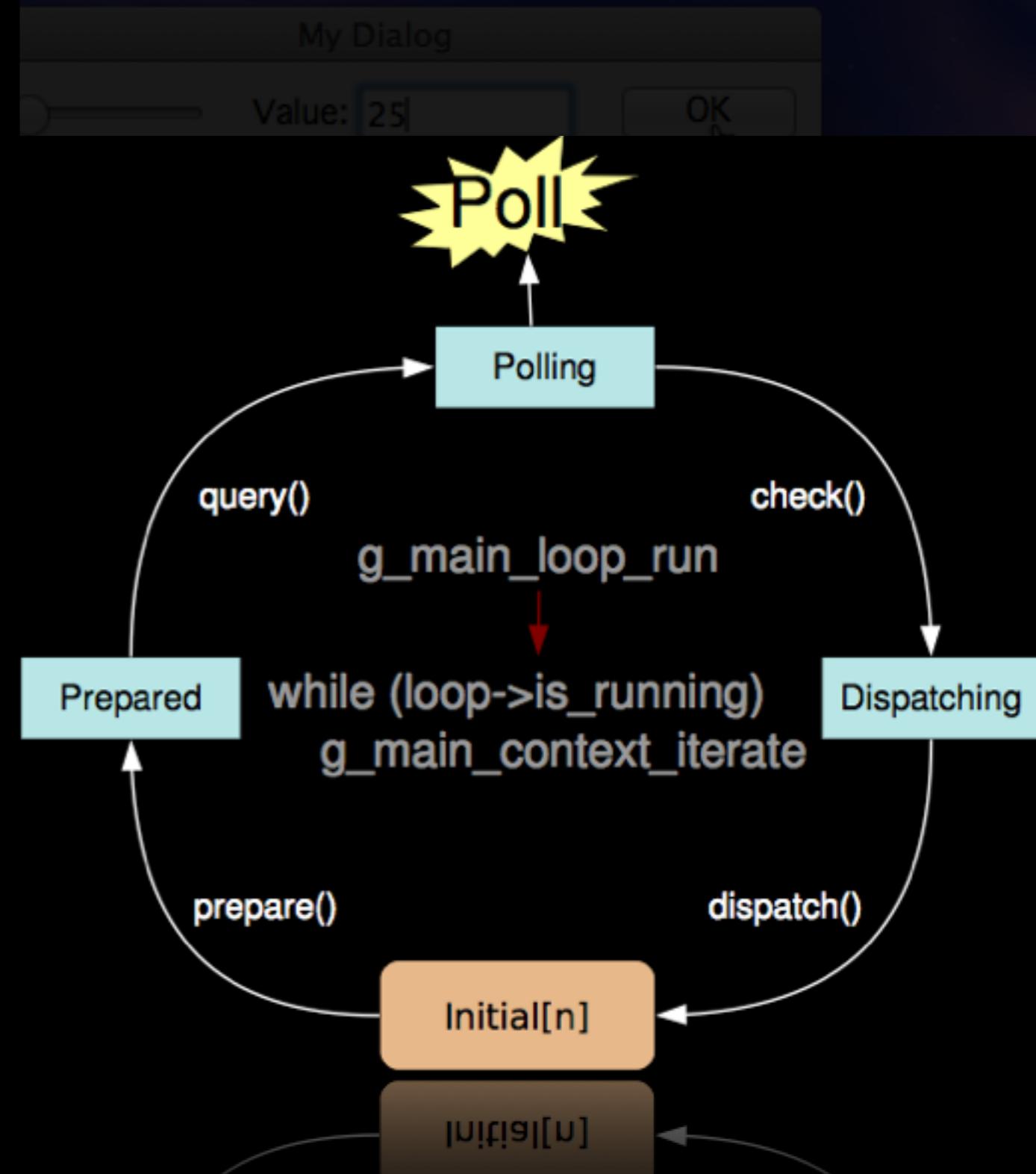
第一步： 分派点击事件

- event: (x, y), press
- Xorg or Weston ...:
 - event → window
- 事件被丢给窗口
 - 确切地说：丢给窗口所在程序的**事件循环**



第二步： 进一步分派事件

- 窗口所在程序的事件循环
 - poll前准备工作
 - polling: 等事件到来
 - dispatching: 分派
 - 开始下一轮循环



第二步： 进一步分派事件

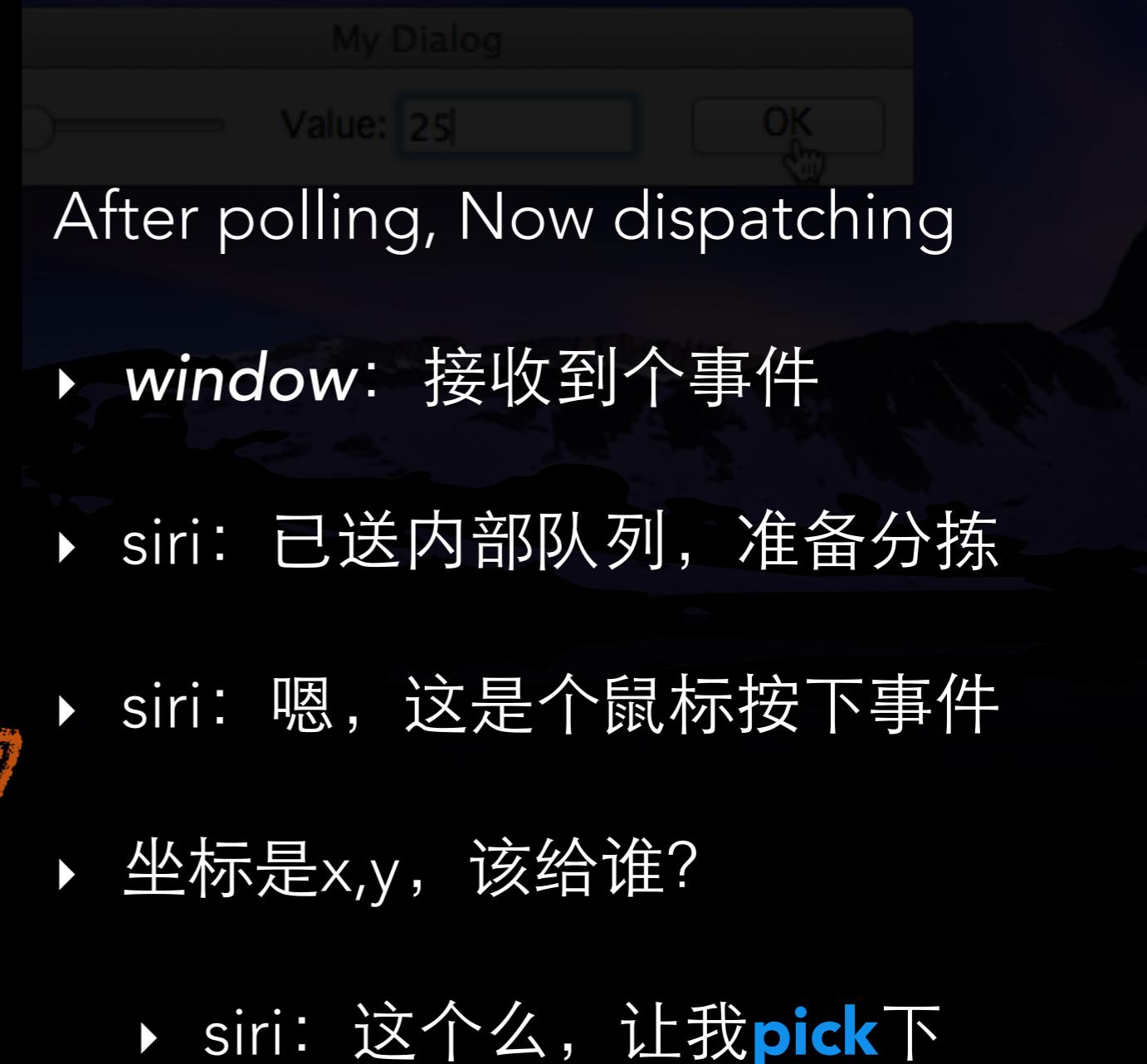
- 窗口所在程序的事件循环
 - dispatch1：将事件封装、转储到内部队列
 -



- **window**: 接收到个事件
- **siri**: 已送内部队列，准备分拣

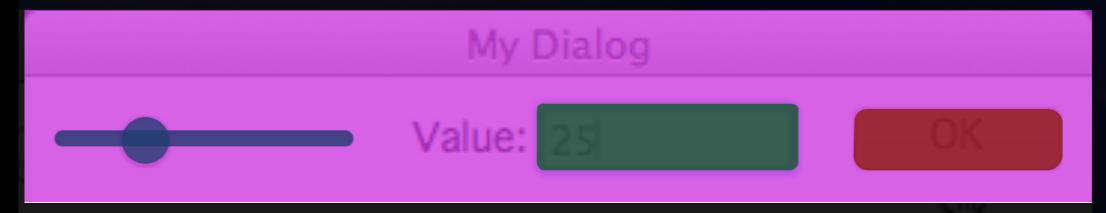
第二步： 进一步分派事件

- 窗口所在程序的事件循环
 - dispatch1: 将事件封装、转储到内部队列
 - dispatch2: 分派事件到.....控件上



第二步： 进一步分派事件

- 窗口所在程序的事件循环
 - dispatch1: 将事件封装、转储到内部队列
 - dispatch2: 分派事件到.....控件上



■ 窗体

■ 滑动条

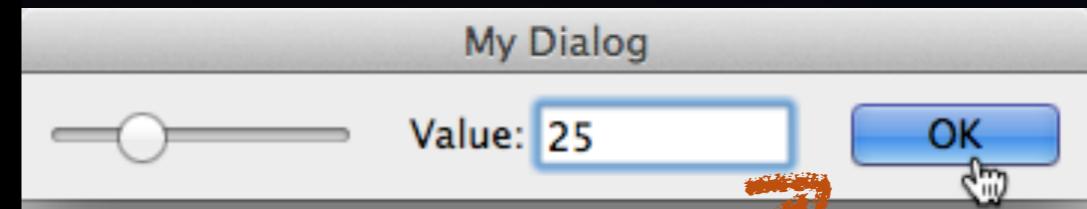
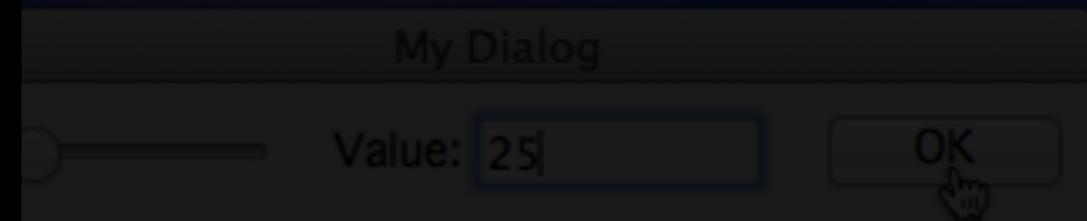
Value: 25

■ 文本输入框

■ 按钮

- 坐标是x,y，该给谁？
 - siri: 这个么，让我**pick**下
 - siri: 好了，x,y点是红色，是按钮控件
 - siri: 发给按钮控件

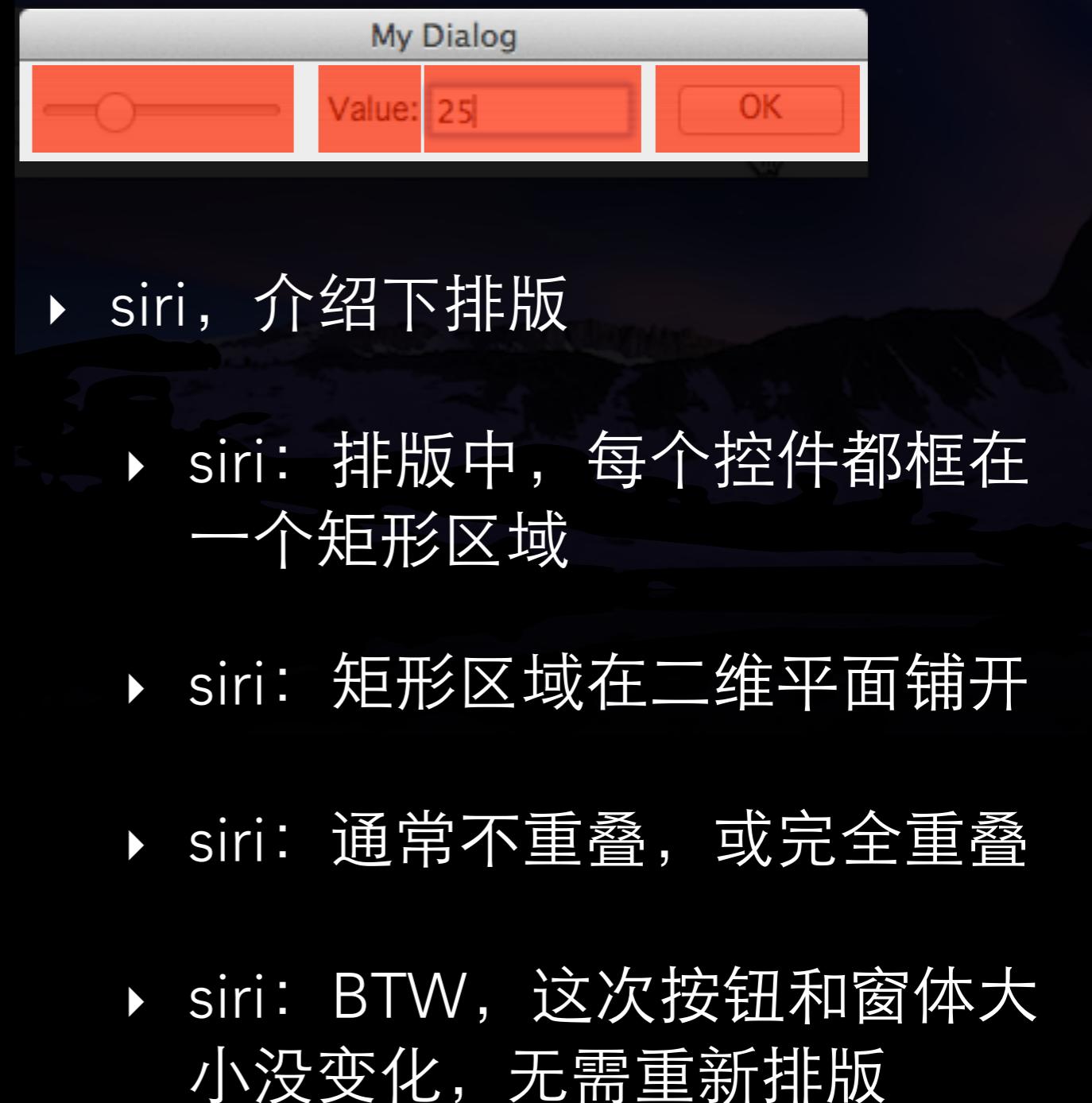
第三步： 按钮给点反应



- 反应：高亮显示
 - ▶ 按钮：收到鼠标按下事件，召唤回调函数(CallBack)
 - ▶ 回调：标记高亮状态，标记重绘

第四步： 重绘界面

- 重新排版
-



第四步： 重绘界面

- 重新排版
- 重新画



- ▶ 先画0，再画1.1，1.2 ...
- ▶ 画画前准备：
 - ▶ 0：设置投影矩阵
 - ▶ 设置模型和视图矩阵
 - ▶ 设置剪辑区

第四步： 重绘界面

- 重新排版
- 重新画



矩阵？？为毛这么复杂！？



► 先画0，再画1.1，1.2 ...

► 画画前准备：

► 0：设置投影矩阵

► 设置模型和视图矩阵

► 设置剪辑区

第四步： 重绘界面

- 重新排版
- 重新画



It's 高大上的 3D 图形界面

▶ 画画前准备：

- ▶ 0：设置投影矩阵
- ▶ 设置模型和视图矩阵
- ▶ 设置剪辑区

矩阵？？为毛这么复杂！？

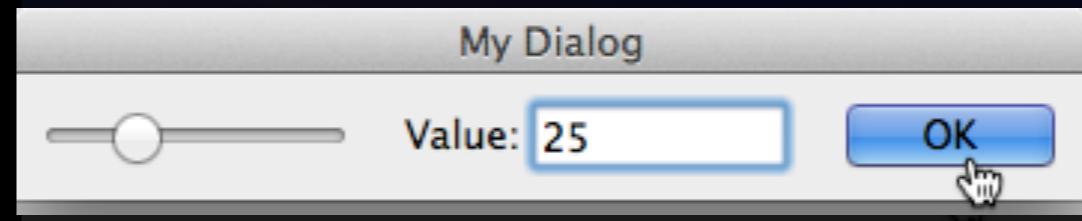
第四步： 重绘界面

- 重新排版
- 重新画



第四步： 显示出来

- 对话框新画面还在后台缓冲 (offscreen buffer)中
- 在帧显示时刻刷到帧缓冲 (frame buffer)中
- 最终GPU把新画面显示在屏幕



- glXSwapInterval(1)
 - 60fps: 每16ms，待显示的后台缓冲 → 帧缓冲
- glXSwapBuffers
 - 提交后台缓冲待显示

总结

- 事件分派：到*window*
- 事件循环：
 - *window*内分派到控件
 - 控件响应，重绘界面：
 - 重新排版
 - 重新画
- 显示出来

