

原创

Vic.GoodLuck

已于 2023-05-18 20:11:27 修改


阅读量3.2k

收藏 15

点赞数 3

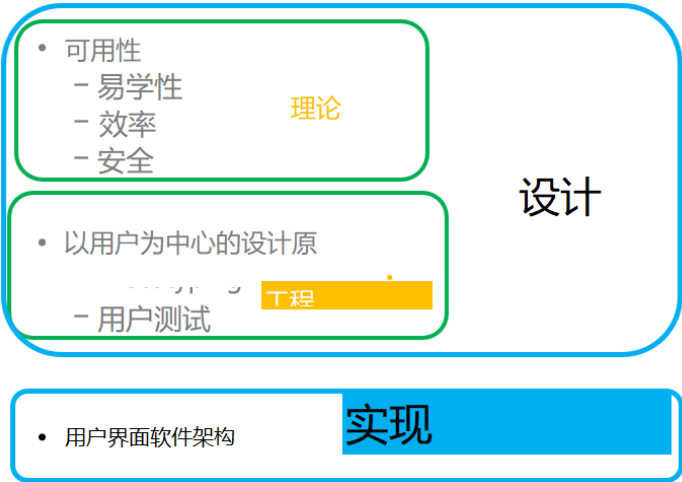
分类专栏: 软件用户界面设计

文章标签: 界面设计

 软件用户界面设计 专栏收录该内容

26 订阅 10 篇文章

界面设计中的“设计”与“实现”，本节的UI架构属于“实现”部分。



1.GUI 设计模式 (Design patterns for GUIs)

(1) 视图树 (View tree)

①定义：GUI结构是一个视图树。视图是一个对象，显示在屏幕的某个区域，可以是一个控件或者其他元素。

②视图树的使用：

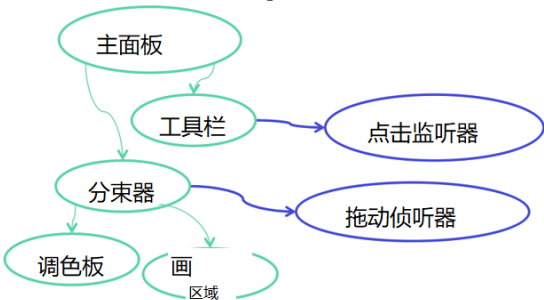
输出：GUI通过改变视图树来改变输出；重绘算法自动重绘受影响的视图

输入：GUI将监听器绑定到视图，来接收键盘和鼠标的输入

布局：自动布局算法通过遍历树来计算视图的位置和大小

③输入处理：

输入处理程序与视图相关联，被称为监听器 (listeners)、事件处理程序、订阅器、观察器等



(2) 监听模式 (Listener Pattern)

①（上面提到的）GUI输入处理是监听模式的一种

②事件源产生一系列离散事件（例如鼠标事

 Vic.GoodLuck

已关注

3

15

0

专栏

④当一个事件发生时，事件源将事件分发给所有绑定的监听器

(3) 模型视图 (Model-view)

①目的：分离前后端

输出：由视图树表示

输入：由绑定在视图上的监听器处理

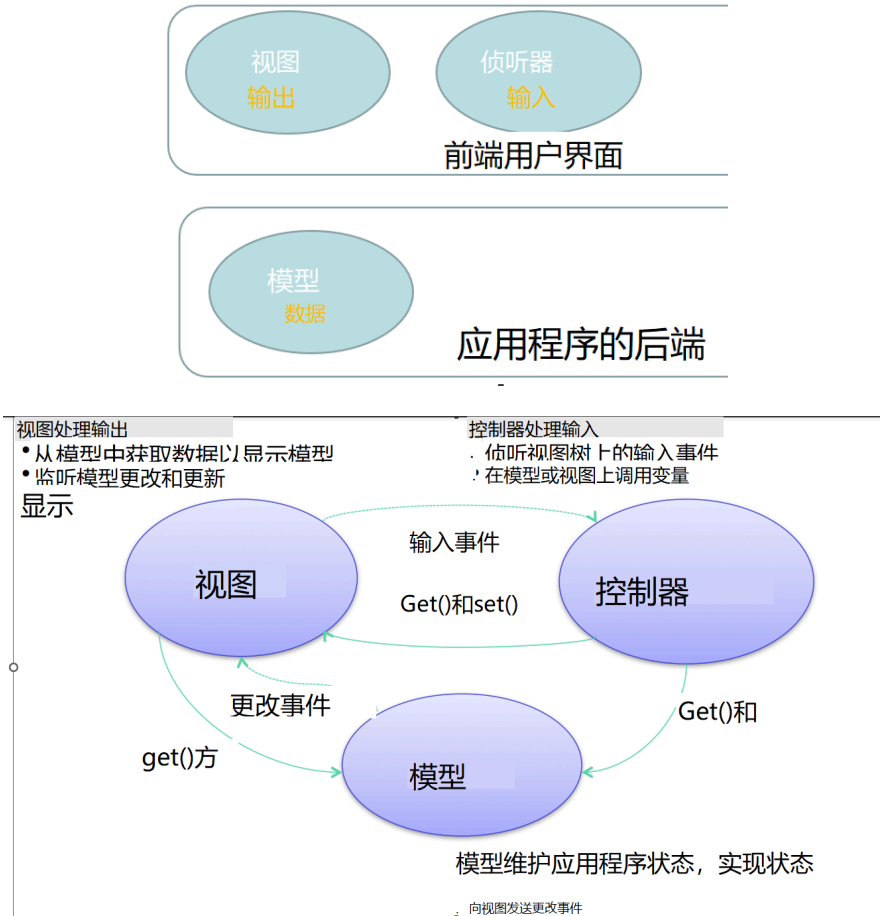
后台（又称模型）：保存用户界面正在展示和编辑的数据

②模型-视图-控制器模式 (Model-View-Controller Pattern, 简称MVC)

模型 (Model) 维护应用程序状态

视图 (View) 显示数据

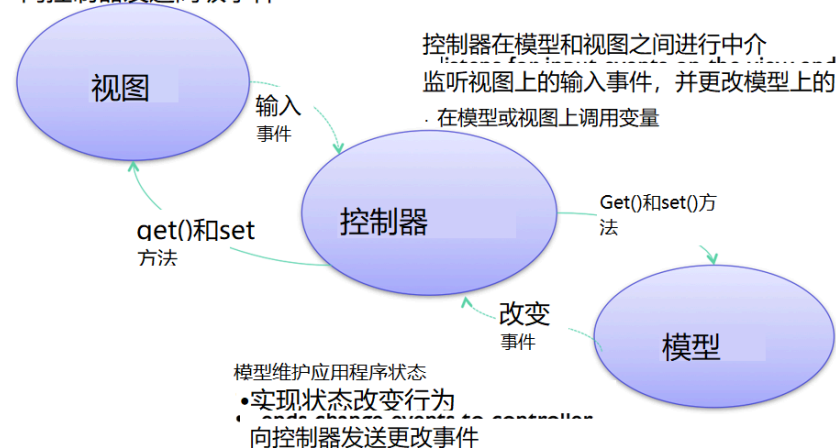
控制器 (Controller) 处理输入





视图处理输出和低级输入

- 向控制器发送高级事件



③模型视图的优点

[1]责任分离：一个模块只负责一个功能 模型Model-数据 视图View-输入输出

[2]解耦(Decoupling)：视图和模型彼此分离，可以独立修改；模型可以被多个视图复用；多个视图可同时使用同一个模型；不被其他模型复用

④控制器 (Controller) 和视图 (View) 很难分离

[1]控制器需要输出：视图必须给控制器提供功能可视性（eg滚动条的拇指）还要对控制器状态的反馈（eg按下的按钮）

[2]控制器和视图共享的情况下谁来管理选择：必须由视图显示；必须由控制器来更新和使用；通常不应当在模型中选择，有独立的选择(例如，同一文档上的两个窗口)，其他视图需要同步选择(例如，表视图和图表视图)

⑤小部件:紧密耦合的视图和控制器

小部件是一个可重用的视图对象，它同时管理输出和输入，有时被称为组件(Java、Flex)或控件(Windows)，例如 滚动条、按钮

2.GUI编程方式 (Approaches to GUI programming)

面向过程的 (procedural)：代码表示，如何得到你想要的

声明式的 (declarative)：代码表示，你想得到什么

直接操作 (direct manipulation)：直接在操作界面创建你想要的（画图）

①标记语言HTML 声明性地指定视图树

②视图树操作JavaScript 循序渐进地改变视图树

③直接操作HTML编辑 AdobeDreamweaver

优点和缺点：

①说明性编程更简洁，程序员只需知道怎么说，不需要知道如何实现

②说明性编程可能更难调试，不能设置断点，不能单步调试，需要更多的试错

③说明性编程的规范使直接操作的创作工具成为可能，因为说明性编程的规范可以通过工具加载并保存，而过程性（程序性）编程不



Vic.GoodLuck 已关注

3



15



0



专栏