1. **什么是MVVM设计模式，它和MVC的区别是什么**

**Q1：MVVM（Model-View-ViewModel）是**一种软件设计模式，常用于用户界面的构建，它将应用程序分为三个核心组件：

Model：包含业务逻辑和数据结构，负责处理应用程序的核心数据操作。

View：只关注视觉呈现，负责展示用户界面。

ViewModel：直接持有Model实例，作为Model与View之间的gatekeeper，为View提供封装的业务逻辑接口，并监听视图的变化来更新Model的数据。

**Q2：MVVM和MVC的区别有**：

**组件定义与分工不同**：MVVM的Model 仅负责业务逻辑和数据结构，并不直接持有模型实例，实例在ViewModel中，MVC则是在Model中直接声明一个实例；MVVM用ViewModel取代MVC中的Controller和View与Model间的数据流。

**数据流与交互方式不同**：MVVM以ViewModel作为gatekeeper，没有Model与View间的直接数据流动；MVC虽然View通过Controller调用Model的服务，但Model与View间还是有直接的双向数据流动。

**组件耦合度不同**：MVVM实现了真正的数据与界面解耦；MVC即使通过观察者机制等降低耦合度，View与Model之间还是有依赖性。

**代码复杂度不同**：MVVM通过数据绑定机制解除了View与ViewModel之间的显式交互，ViewModel自动适应UI的更新，相较于MVC的Controller显式处理用户输入，减少了代码的复杂度，提高了泛用性。

**使用场景不同**：MVVM适用于复杂的应用，频繁的交互和更新页面；MVC适合静态页面与简单的页面更新。

1. **在Stanford的Memories项目中，M/V/VM分别对应的代码是什么，他们之间的逻辑关系是怎么样的？**

M对应：MemorizeGame.swift文件

V对应：EmojiMemoryGameView.swift文件

VM对应：EmojiMemoryGame.swift文件

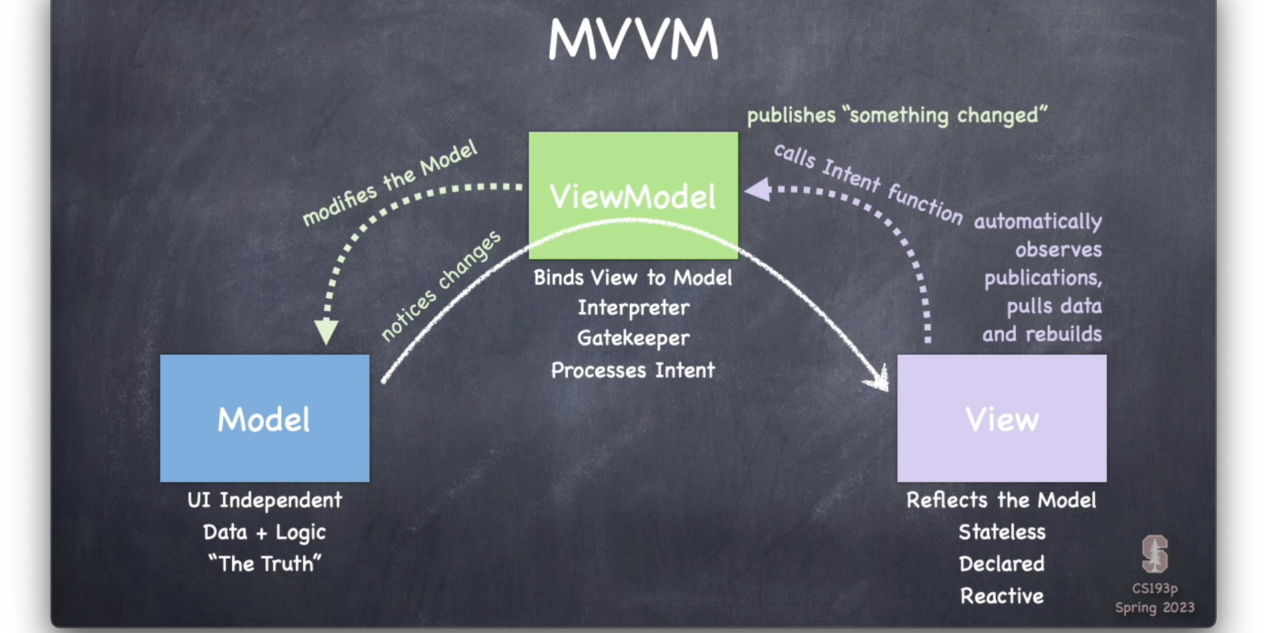
逻辑关系：

M定义游戏的核心逻辑和数据结构，将处理业务的方法封装提供ViewModel，并通过可观察属性将数据变化自动提交给ViewModel。

VM充当中间层，持有Model的实例，通过实例将Model的数据提供给View，并通过调用实例方法，将View的用户交互事件转发给Model处理。

V负责展示UI，并通过数据绑定机制与ViewModel交互，完成自动更新界面和用户交互事件传递。

**3. 请解释图片中的所有名词：**



1. **Model**：由data、structure和basic logic组成。

**UI Independent**：完全不需要在Model中导入任何UI内容，是generic Model。

**Data + Logic**：Model是data和logic两者的combination。

1. **ViewModel**：View与Model的connection，创建只读的data flow。

**Binds View to Model**：ViewModel是将View绑定到Model上的connection。

**Interpreter**：VM执行解释工作，关注Model中的变化，并将其publish，subscribe此发布的View会自动修改UI。

**Gatekeeper**：通过控制数据单向流动，保护Model不直接受View影响。

**Processes Intent**：有用户意图函数的定义，供View调用，来获取用户意图、影响Model。注意，用户意图是一个generic定义，并不直接依赖于某个操作或某个按钮。

1. **View**：UI界面上可视化的展示。

**Reflects the Model**：View必须与Model一致，Model什么样View就得长什么样。

**Stateless**：因为只需要展示Model，不推荐额外使用@state包装的属性。

**Declared**：View仅仅是Model的映射，不需要实现data状态的改变。

**Reactive**：用户在UI的交互事件也必须传递给Model，并在UI显示出响应。

1. **Notices changes**：当Model改变时，通过ObservableObject的订阅发布机制，View重新渲染UI界面。Update在View中完成，省去了MVC中controller里update函数的代码，使ViewModel与View解耦合。

**Calls Intent function**：当用户做出交互时，调用对应的意图函数，来告诉ViewModel用户的意图。

**Modifies the Model**：将用户意图传递给Model，使Model做出相应改变。

**4.提交完成的Memories项目**

项目地址：https://github.com/evo2233/Tongji-MAD-MemorizeGame