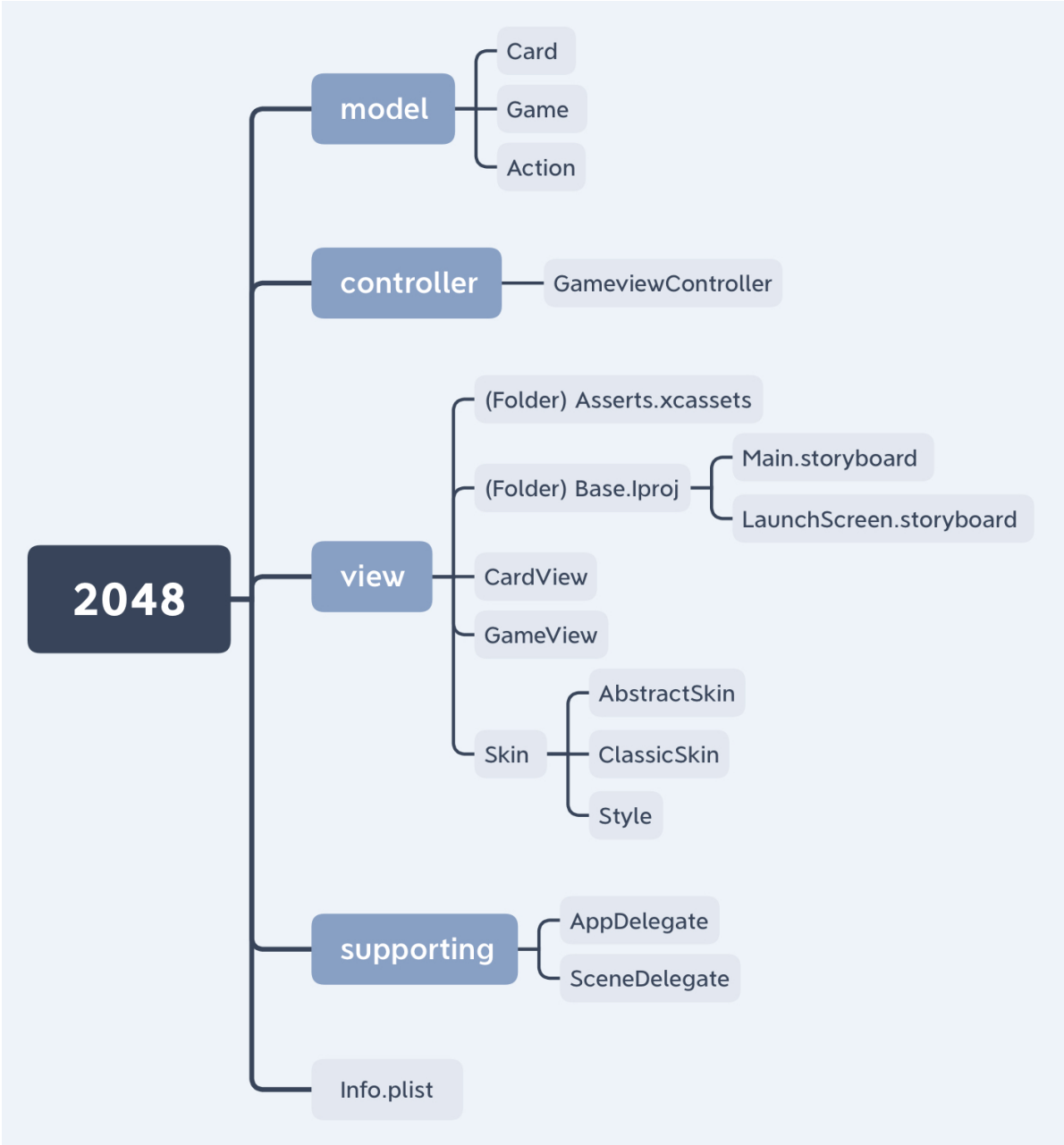
2048游戏app说明文档

秦铭悦 20373843 计算机学院

1. 按照 MVC 架构进行分类的文件

在Xcode中生成新的project后，工程文件内部会自动生成的若干文件。

将自动生成的文件按照MVC模式进行归类，同时新增一些文件，效果如下：



2. 各文件、各函数的功能说明

Model

- **Card**：定义每个小方格的类card
  - 用card类的属性记录**单个小方格的值**
  - 用card类的方法实现每次移动后中**单个小方块内取值的变化**

        - **func getValue ()** : 保持不变
        - **func upgrade ()** : 值×2
- **Game**：主要 定义游戏过程的类
    - **enum Direction**：移动方向常量定义
    - **struct Position**：行列坐标定义
    - 用game类的**属性**定义**表格尺寸**和游戏使用的二维数组**World**
    - game类内定义 **enum status**标识游戏的进行状态
    - 用game类的**方法**实现游戏功能（**start、move、end**）（循环实现"**滑动** - 随机位置出现2或者4 + 现存块全部向滑动位方向移动 + 相邻的值相同块合并"）
    -
        - **move**算法说明：
            1. 每次移动时先建立一个空的表格 **func getCleanWorld()**
            2. ① 根据滑动方向确定的遍历方式（滑动向左：必须从左往右遍历；滑动往右：必须从右往左遍历；滑动往上：必须从上往下遍历；滑动往下：必须从下往上便利）；

               ②当在旧表格中读入不是空白后，记录数字，该数字在新表格中的位置是就表格中的位置**向移动方向移到底**的结果；

               ③如果将数字填入表格、在移动方向上碰到已有的数字，比对连数字，相同则upgrade合并，不同时则数字的最终位置确定。

               ④在任意空位置，生成一个新数2或4 **func generateNewCard()** (具体实现方法：将所有空白的Card放进一个pool里面，然后在pool里面随机拿出一个Card并随机放入2或4)
            3. 当所有数字遍历处理完毕，将新表格作为当前表格，一次操作完毕。

        （Model层的操作存储在antions数组内）
    - 游戏失败法方法 **func checkFailure**
    - 溢出检查方法**func inBound**
    - 对Model层算法结果打印到输出、以实现Debug功能的方法**func debugPrint**
- **Action** ： 通过**enum Action**枚举常量来标识游戏计数的不同状态和动作（*move*，*upgrade*，*new*，*success*，*failure*）

view

- **GameView** : 实现游戏的总体动画效果 --->class GameView
    - **方法 func drawBound** 画游戏的边界，margin为边界离手机边框的距离
    - **方法 func boundSize** 确定游戏边框的尺寸

      func cardSize 确定游戏界面中内部小方框的尺寸
    - **方法 func getRectOf** 转化size*size个小方框的行列坐标
    - **func draw** 根据getRectOf得到的位置绘制内部的size*size个小方框
    - **方法 func performActions** 为**Model**层中实现的不同操作实现不同动画（不同的操作已经存储在 [actions] 类型的数组里）
        - func newCard
        - func moveCard
        - func upgrade
    - **方法 func touchesBegan** 标记玩家触碰开始、记录触屏初始位置

func touchesMoved 标记玩家触碰结束、记录触屏结束位置，计算触 碰距离distance是否达到有效触碰的标准

func touchesEnded 判断并根据判断结果改变 touchingDetectable 的值，标记系统可以开始检测新的触屏操作

func distance 计算touch的起始点和终止点的几何距离

- 重载运算符 extension CGPoint 当计算对象类型都为CGPoint的时候，重载"-"运算符为距离计算

- **CardView**： 具体实现游戏中每个小方框内的动画效果
  - **属性 value**： （label实现）如果值为0，隐藏数字；如果值不为零，显示数字。同时控制label的颜色。
  - **方法 func set**： 初次给value赋值，并设置label的方块的显示方式（根据长短调整防治方法，居中显示），并让label上下左右都顶着view的边
  - **方法 func createAnimation**： 实现card第一次生成时形成的动画效果（时间设置，边框大小变化）
  - **方法 func flash**：实现card值翻倍时的动画效果和颜色变化
- **Skin** ：控制游戏不同部分的颜色
  - class AbstractSkin：颜色选择模板
  - class ClassicSkin： 根据小方块内数字的不同，给小方块确定不同的颜色
  - struct Style： 规定绘制每个小方块时的结构体属性（*content*标签内容，*labelColor*标签颜色，*backgroundColor*背景颜色）

Controller

GameViewController

- gamesize：定义 size * size 的游戏界面大小
- lazy var game：调用游戏**game**的类，**与gameView建立outlet连接**

==后面**Controller层**通过调用Model中的类来实现功能与界面动画的联系

- **方法 func viewDidLoad**： 启动游戏画面，设定游戏启动时的延迟
- **方法 func startGame**： 调用游戏
- **方法 func slideEnded**： 将offset触屏移动方向转换成**enum direction** 中的方向，即Model层game中的move方法中进行计算
- **方法 func showMaskView**

  func maskViewDisappear ：实现游戏结束后出现的像幕布一样的暗淡动画
- **方法 func showSuccessView**

  方法 func showFailureView ：在游戏成功后（达成2048）或者游戏失败后（整个游戏界面没有空白格，且任意相邻格子value不相等），启动有些结束后的暗淡动画