

# dotFlow

# 设计文档

刘家昌  
Jason Lau  
计算机科学与技术系 42 班  
Tsinghua CST 42  
2014011307

M +86 185-1162-1217  
E [i@dotkrnl.com](mailto:i@dotkrnl.com)

## 简要描述

对 Android 及 iOS 上的 Flow Free 游戏进行了复刻，使用 Qt 开发框架进行跨平台开发。  
由于原作版权原因，对于 dotFlow 的任何应用，应当仅限制于科研学习用途。

## 基本功能

实现了 Regular Pack 游戏内容

以鼠标左键长按连接颜色匹配的管线，建立水流通道。将所有颜色进行配对，使管线覆盖整个区域，即可顺利过关。如果发生交叉或重叠，管线将会破裂。

鼠标左键长按状态下，鼠标所在位置以圆形标示

并且，鼠标所在位置的颜色，与原作一致，为当前正在进行连接的管线颜色。

水流通道连通时或管道破裂时，播放声音提示

在通关时，以及操作按钮时，也有对应的声音提示。声音来源于原作游戏，原作保留一切权利，因此对于 dotFlow 的任何应用，应当仅限制于科研学习用途。

包括 5×5、6×6、7×7 的棋盘布局，包含原作所有关卡

关卡数据来源于原作游戏，原作保留一切权利，因此 dotFlow 应当仅限制于科研学习用途。

在关卡选择器中，可以进行重新开始，上一关、下一关的操作

这一功能的使用，请点击左上角的第一个图标。见后文操作文档。

## 特色功能

能够自动计算 Flow Free 的解

对于游戏中包含的所有关卡数据，或者使用下个扩展功能描述的加载关卡功能载入用户输入，可以长按或点击游戏上方的问号按钮，显示其答案。

除程序自带的关卡外，可以使用关卡选择器从文件中加载游戏

可以正常使用，功能与自带关卡一致。

提供了跨平台的用户游戏存档功能

无论何时关闭游戏，下次打开仍将显示您的游戏进度以及最佳成绩，dotFlow 依旧。

设计了用户友好的用户界面

在当前关卡结束后，会显示历史最佳成绩，以及本次操作是否完美。若不完美则提供重新玩本关、玩下一关的选择。若完美，则可以随机关卡，或者下一关，并显示五角星奖励符号。长按提示按钮松开自动隐藏，短按提示按钮会保留显示。包含两个循序渐进的教学关，保证用户能够轻松上手。还有更多的人性化设计等待你的发现。

使用 StyleSheet 完成了控件自定义，支持了 Windows :-(

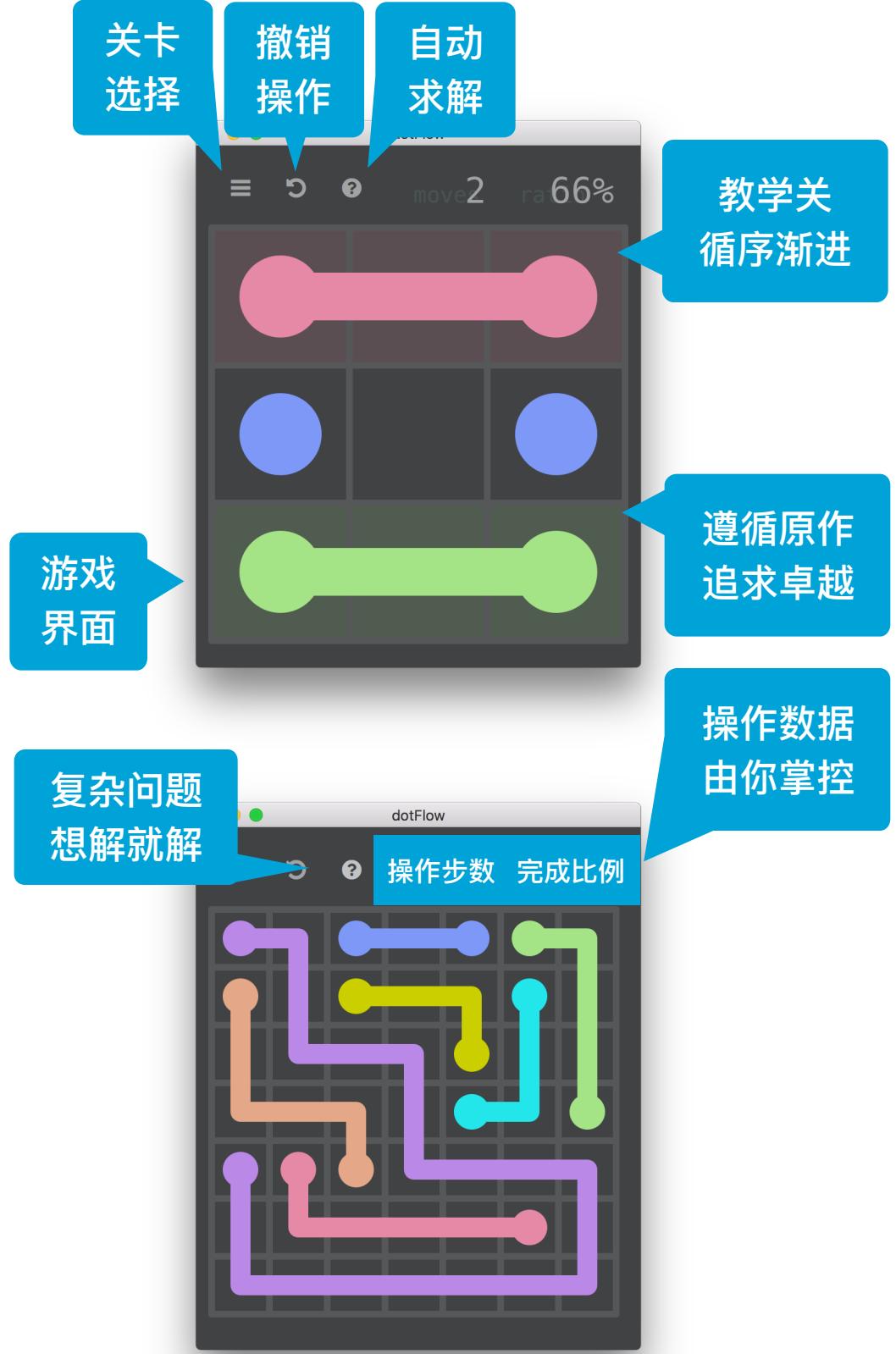
Windows 的字体大小处理不一致，导致需要额外的工作使之在各个平台都能有较好的表现。

提供了撤销操作的功能

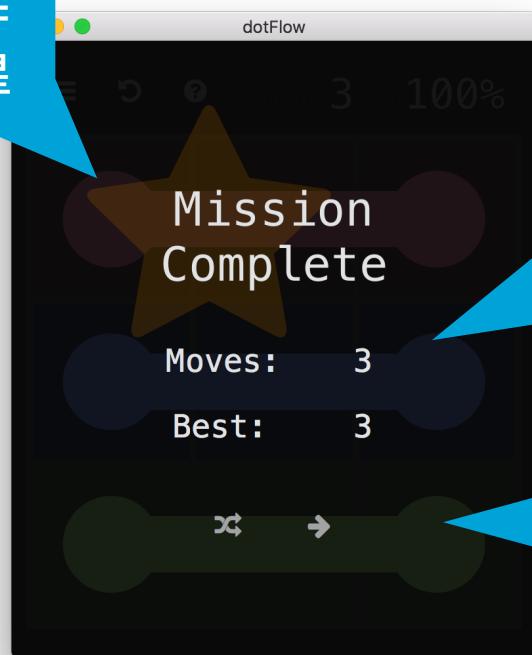
点击游戏上方的撤销按钮，可以撤销用户的误操作。撤销是无限次的。

## 界面展示





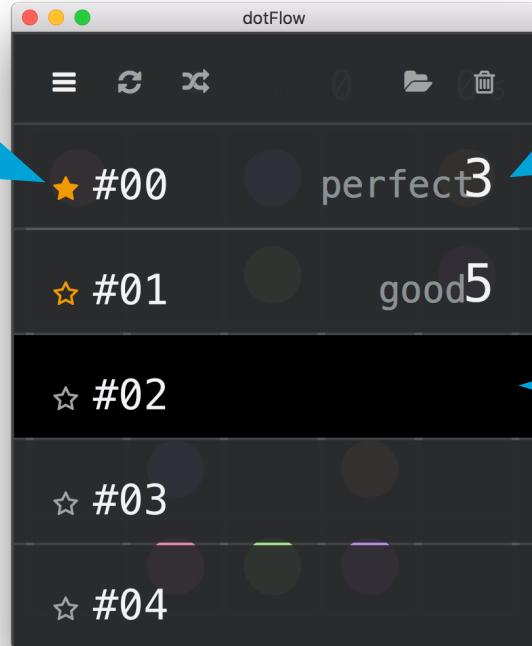
最佳操作  
的鼓励星



当前得分  
最优得分

过关  
界面

过关最佳  
的鼓励星



机智的  
下一操作  
我知道如果不是最优  
会想要重新来过哦

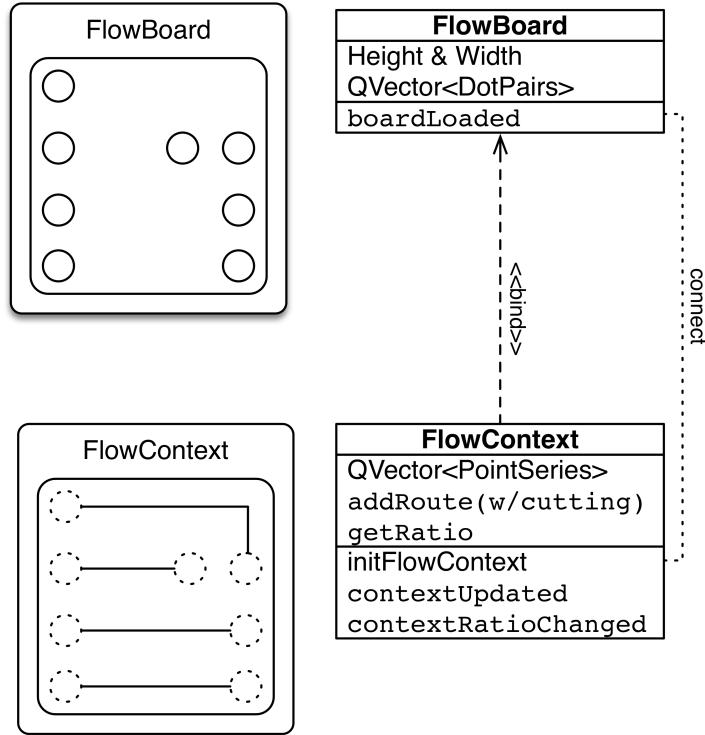
进度存档  
过关纪录  
与你随行

关卡  
选择

当前关卡



## 设计架构

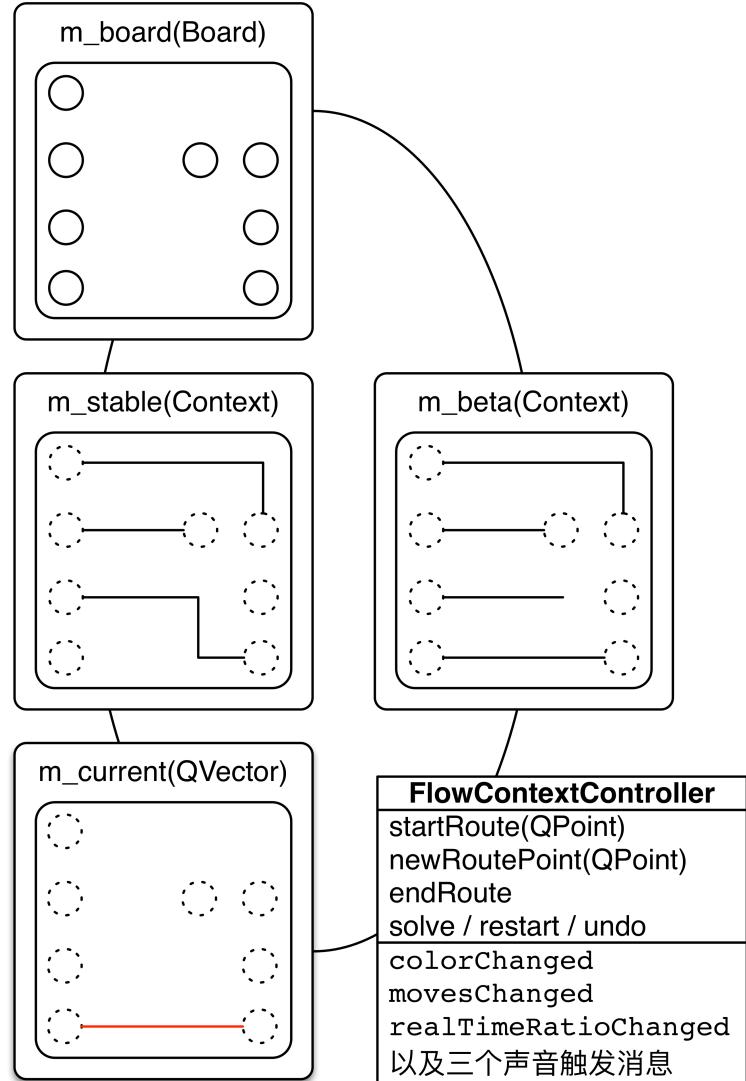


Model 类

我实现了两个 Model 类，一个用于储存局面数据的点对，另一个用于储存链接了的 Flow 数据。在 FlowBoard 会发送 boardLoaded 消息，而 FlowContext 对其进行了监听，当其重新加载时，可以立刻初始化局面。同时 Context 也发送更新消息，这两个可以用于 View 的更新。

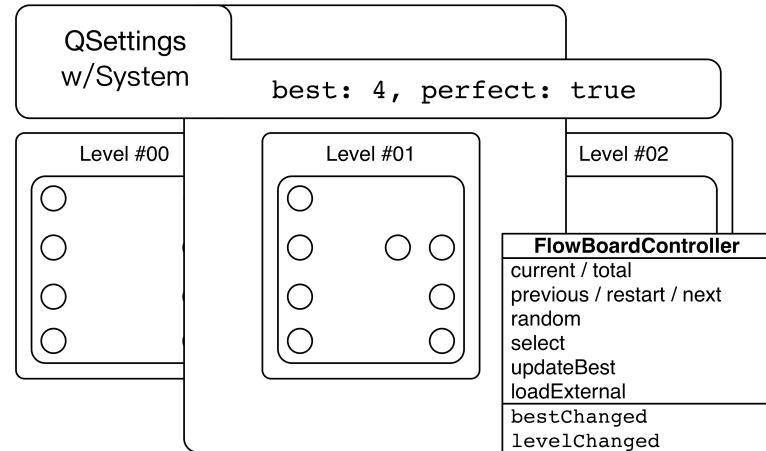
为了管理 FlowContext 这一 Model，我创建了 FlowContextController 这一 Controller 类。它接收 startRoute、addRoutePoint 以及 endRoute 三个消息，使用 m\_board 维护当前拉出的颜色 Flow 数据 m\_current，以及拉出前，除当前颜色以外的所有 Flow 信息 m\_stable。并利用这些信息，在每次 addRoutePoint 时，产生当前面板上应当显示的 Flow 情况 m\_beta。在 endRoute 时，结束这一操作过程，以及保存至 m\_stable。

这一 Controller 类还维护了移动次数的信息，创建后文描述的 FlowSolver 对象自动求解，以栈的形式保存历史 Context 并提供 undo 支持，发送操作事件消息等等，细节不展开讨论。



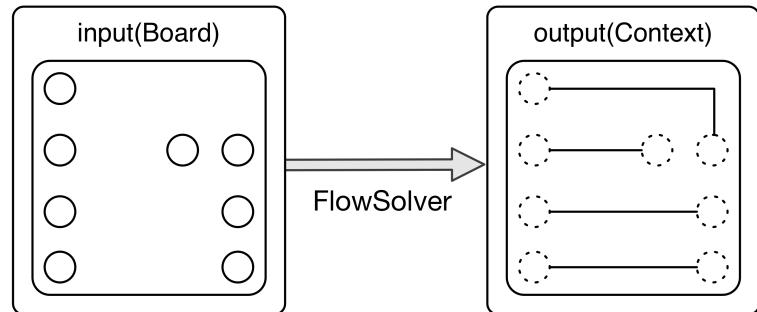
关于 `m_stable` 的维护，主要为判断端点清空 Flow，将其中对应的颜色路径转移至 `m_current` 中。而对于 `m_current` 的维护，主要为判断回环截断、判断端点不可达状态、保证完整 Flow 不被延长以及自动寻路加点。`m_beta` 这一显示数据的处理则极为简单，直接将 `m_stable` 使用 `m_current` 进行截断操作，之后将对应颜色替换为 `m_current` 即可，畅通无阻。

这一实现策略，使得逻辑极为清晰，并且由于分出了稳定状态和易于撤销的状态，并用其导出原本难以维护的状态，使得可以轻松回拉，不出 bug 且对用户更加友好。



FlowBoardController 类

为了管理 FlowBoard 这一 Model，我创建了 FlowBoardController 这一 Controller 类。其读取资源文件，或者外部文件，提供一个稳定的 FlowBoard 对象。这一对象可以进行上一关、下一关等图中可见操作。这一关卡管理器还同时维护了 QSettings 这一 Model 中的各个关卡的 best 以及 perfect 信息，提供了一个平台无关的游戏存档。在关卡的最好成绩 best 改变、或是当前关卡改变时，会发送对应的消息，提醒 View 对于其订阅的消息进行更新。

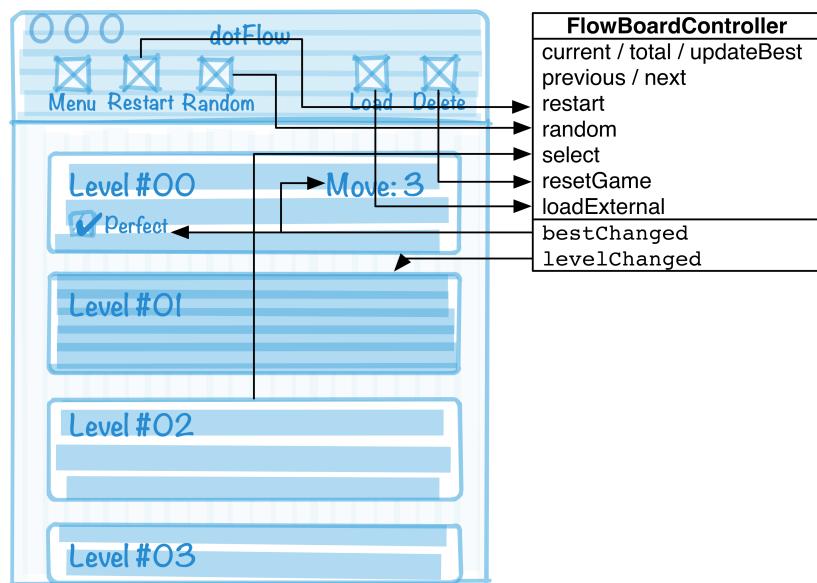
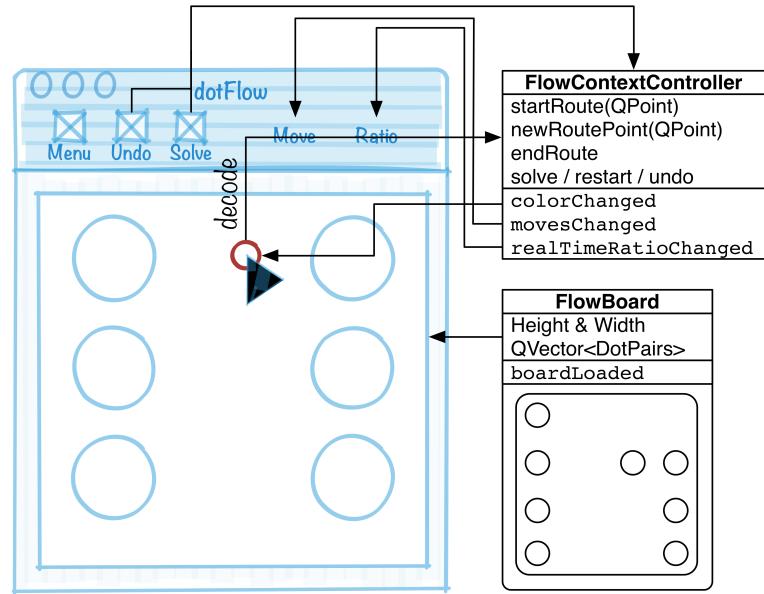


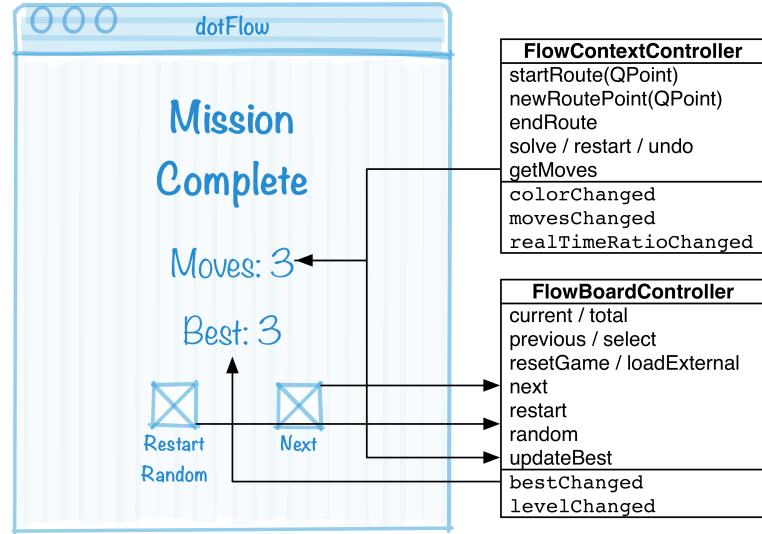
FlowSolver 类

在 FlowContextController 中提到了 FlowSolver 类，这是一个极为简单的解答器，其接受一个 FlowBoard，通过调用 solve 方法，即可返回一个解答，为 FlowContext 的实例。这一实例会被放置于 m\_beta 中予以显示。

上面描述了 Model 的结构以及各个 Controller 与其之间的关系。

下面将以图片的形式展现程序中各个 View 与 Controller 及 Model 的关系。





## 收获感想

这一周对 Qt 的学习，使得我对于利用信号实现事件驱动编程有了很大的启发，给了我很好的练习 MVC 架构的机会，使得我对于图形界面的实现不再感到恐惧。

总的来说，我在这一周的学习中收获很大，也希望再接下去的学习中能够有更大的收获。

感谢老师，以及助教。