设计模式一：组合模式



前端界面使用组合模式进行设计。前端的用户界面中各个组件之间有着清晰的 “整体—部分”关系，可以按树状结构进行组织。MainView容器主要包含Console、文件编辑器、文件区和绘图区这四个组件，绘图区又由乌龟和画布两个组件构成。容器对象和叶子对象具有生命周期的一致性，叶子对象无法脱离容器对象进行单独渲染；当容器对象的状态更新时，叶子对象的状态也会相应更新。

使用了组合模式后，在容器对象中增删叶子节点都很方便，增加了开发的敏捷性。当需求调整时，我们可以相应地修改模块，符合开闭原则，对以后的维护非常有利。

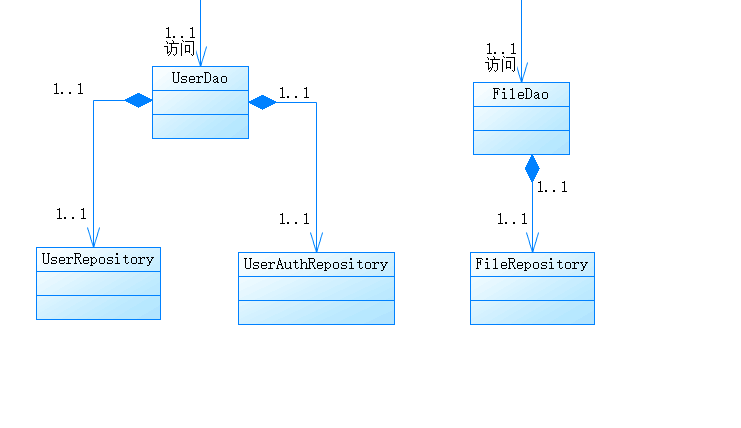
设计模式二：解释器模式



前端需要一个代码解析器将LOGO语言翻译成画图区可执行的命令，我们采用解释器模式进行设计。LOGO语言解释器类是顶层的封装，我们将在Console和文件编辑器使用它。Context是环境角色，包含预定义的一些全局信息。AbstractExpression为抽象表达式，声明一个抽象的解释操作，这个接口为抽象语法树中所有的节点所共享。TerminalExression为终结符表达式，实现与文法中的终结符相关联的解释操作；NonterminalExpression为非终结符表达式，为文法中的非终结符实现解释操作。

采用解释器模式后，增加了LOGO语言的可扩展性。如果需求变更、增加新的语法时，我们可以增加TerminalExression和NonterminalExpression中的子节点，敏捷地增加对新语法的支持。

设计模式三：外观模式



在后端中，UserDao和FileDao作为外观类对Repository进行封装。其中UserDao作为访问UserRepository和UserAuthRepository的入口，FileDao作为访问FIleRepository的入口。从而使上面的Service层访问数据时不必去管数据库的具体构成如何，实现了服务层和数据层的解耦。