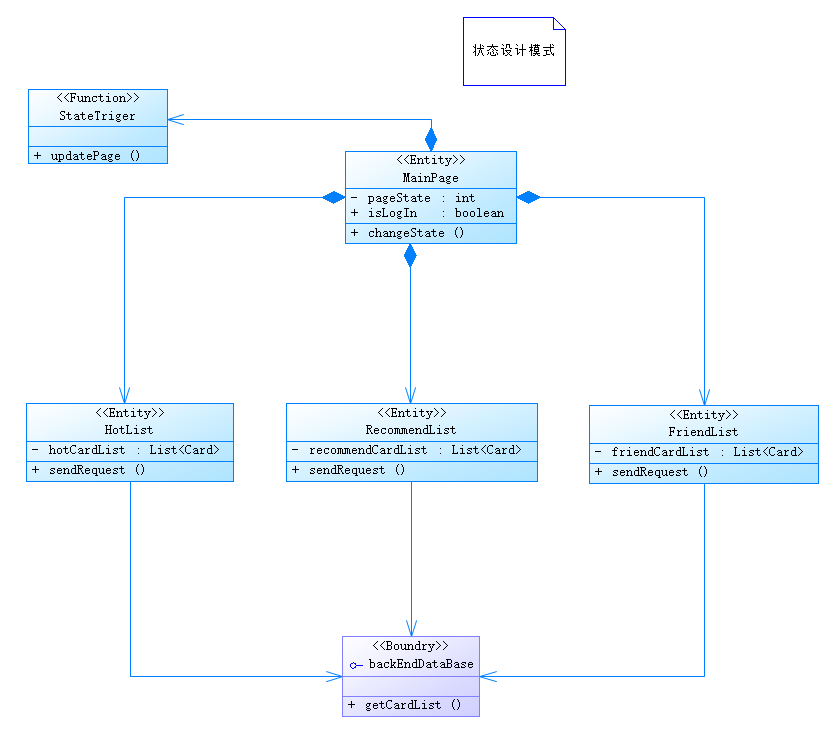
面向对象设计——GOF设计模式

这部分文档用于描述系统中主要的GOF设计模式，其中包括了针对主页的状态设计模式、针对搜索页面的职责链设计模式，后端工程的工厂方法设计模式。附以图片和文字说明，介绍各个实体类和数据传输的具体流程。

1. **针对主页的状态设计模式**

设计模式的UML状态图如下所示：



图一 UML状态图

我们大学生问答网站的主页会显示“热门”、“推荐”、“关注”这三个部分，网站的内容可通过点击进行切换，这里存在一个状态变量，所以我们认为使用UML的状态设计模型更加合适。

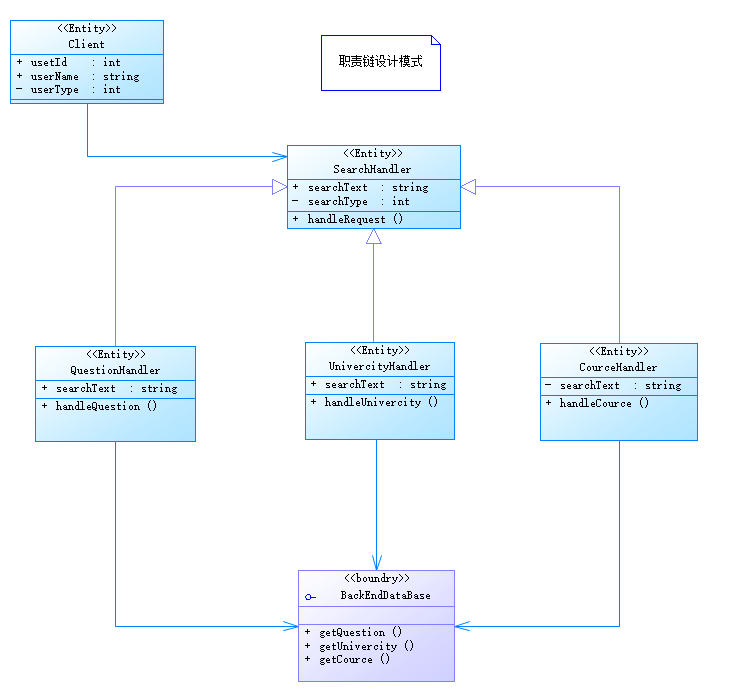
图中， pageState 描述页面所处的状态，当 pageState = 1时，主页显示热门内容；当pageState = 2时，主页显示推荐内容；当 pageState = 3时，主页显示关注内容。

每当用户点击按钮，则会触发 changeState() 函数，更改pageState，而pageState 一旦被更改，则会触发 StateTriger 中的 updatePage() 函数，立刻更新页面。

主页由三个 List 构成，其元素为问题卡片 Card ，每次更新页面时，对应页面内容的List 会向后端发送请求，更新自己的内容。

1. **针对搜索页面的职责链设计模式**

设计模式的UML职责链图如下所示：



图二 UML职责链图

用户可能会对问题、大学、课程等方面进行搜索，我们认为这是搜索功能的不同职责，所以使用UML的职责链设计模型更加合适。

当用户在搜索栏输入搜索内容，选定搜索种类后，发送请求。我们按照UML图所示，对请求进行处理。例如用户选择 “问题” 后，输入问题并发送。如果后端找到相关问题，则直接返回即可；如果后端没有找到相关问题，则交给 “UnivercityHandler” 处理，返回 “大学” 种类下的相关内容；如果 “大学” 种类下也没有相关问题，则交给 “CourceHandler” 处理。如果都没有找到，则返回未找到即可。

1. **后端消息类的工厂方法模式**

在后端的消息类中我们使用了工厂方法设计模式。该消息类的作用是提供后端向前端返回响应的统一接口。不同的类所需要返回的响应中的payload各不相同。通过这种抽象我们可以增强Message与Controller之间的隔离，使Message的具体实现变得独立，更易于系统的开发及维护。

