第3章

设计方法论

3.1 产品设计

在经过了两章的准备活动后，终于可以开始着手产品的开发工作了。但在正式进入开发阶段之前，还有一个重要的步骤，那就是产品设计。一幢大厦，不能在未绘制蓝图的情况下就开始施工。软件工程，也绝不可在毫无计划的状态下直接编写代码。正所谓“磨刀不误砍柴工”，一个详尽且完善的产品设计过程能够让开发事半功倍。

在一个公司内，设计产品的工作往往由专职的产品经理负责，但也不意味着个人或中小团队就无法完成这部分工作。本章试图以一个电子留言板(BBS)项目为导向，以中小开发团队甚至个人开发者为前提，探讨如何相对高效地完成产品设计流程。

3.1.1 需求分析与用例图

产品的核心价值就是满足人的需求。因此产品从立项伊始就要将需求置于首位，并且贯穿整个项目的始终。需要反复思考这个产品为了什么？产品的定位是什么？主要面向社会上的哪些人群？具体能够解决他们什么样的问题？在什么场景下能够解决这些问题？现有的解决方法是什么？

需求不是空中楼阁，而是社会上个体内心的真实想法和呼声。平时不关注生活、不与人交流，仅凭想当然地闭门造车是无法设计出真正有价值的产品的。因此，在实际的操作中，需求分析前往往还有一个需求采集的过程。

因为本书只做技术讲授，并不涉及产品创意，所以仅以一个简单的 BBS 为例，解决用户在线上围绕某个话题的基本沟通需求。

在确立了产品需求后，可以着手画出产品的用例图(use case diagram)。用例图主要以图示的手段来描述用户、需求以及系统功能之间的关系，是用户所能观察和使用到的系统功

能的模型图。

用例图由四部分构成：参与者、用例、系统边界和关系。

参与者主要代表使用系统的用户身份。可以是人，图形上用一个小人儿表示；也可以是其他系统或硬件之类。

用例是参与者能够感知到的系统服务和功能。用椭圆表示，描述功能或服务的文字写在椭圆内部，而且务必使用动词或者动词＋名词。

系统边界通过一个大方框包住内部的所有用例，确定与其他系统之间的边界。系统名称写在方框内部。

使用箭头表示用例与参与者或用例与用例之间的关系。主要用到的关系有三种：关联、包含和扩展。

关联使用实箭头，用来连接参与者和用例。

包含和扩展都使用虚箭头，用来连接用例与用例。其区别在于：包含相当于将一个大的功能分解为几个小功能，箭头指向小功能；而扩展相当于在一个主要功能的基础上扩展出一个附属功能，箭头指向主要功能。

根据上述规则将 BBS 产品用例图描绘出来，如图 3.1 所示。

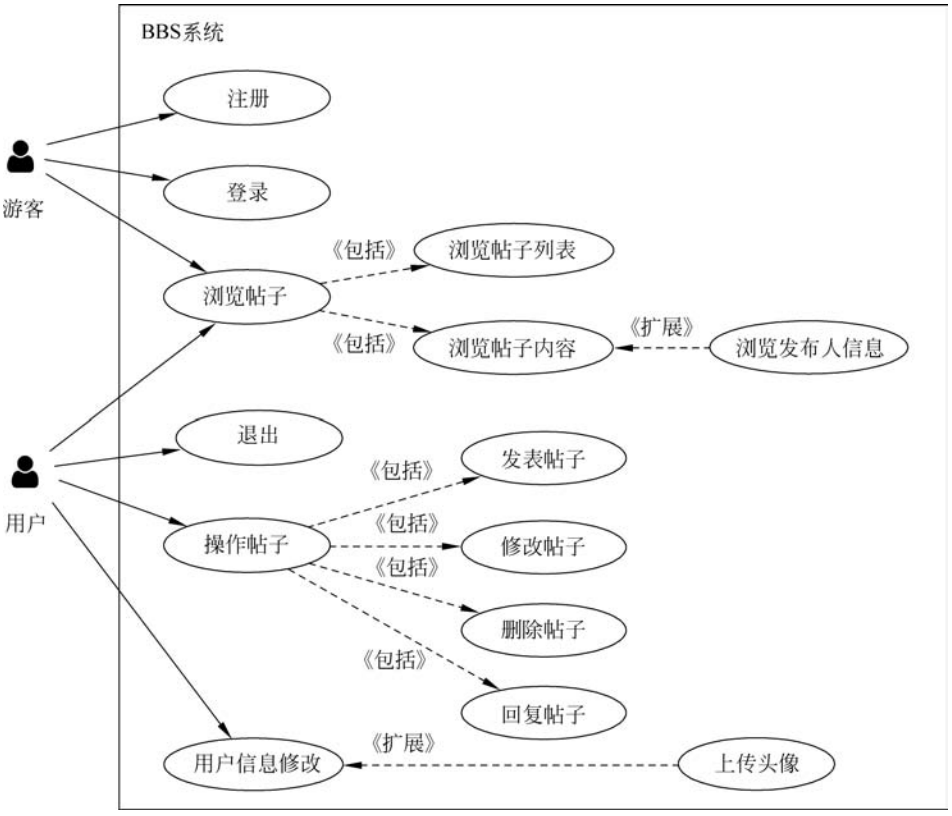


图 3.1　BBS 的产品用例图

用例图根据用户想做什么而描述了利用产品能做什么，作为结果，也就初步规定了产品在功能上会有什么。

3.1.2 DASP 设计模式

一般来说，在基本梳理了用户需求后，实际绘制产品原型前，作为产品经理，需要依次完成功能结构图、信息结构图以及产品结构图。在本书中，笔者并不打算完全遵循这个流程，而是试图结合本书所采用的开发技术与架构，探讨一种更能让前期设计与后期开发直接挂钩的、从项目宏观上更高效的设计模式。

在 1.2 节曾经讲到，本书的基本开发架构就是：①用数据库对现实问题建模，以实现信息持久化存储；②实现对①的增、查、改、删的 RESTful API，以实现全部功能；③在不同客户端上做出不同的 UI 壳，调用②实现的功能，以满足用户需求。

因此，高效的设计过程理应与此基本架构达到高度的统一。①对现实问题建模，完成数据库(database)的设计；②基于①，完成接口(API)的设计；③基于②，完成产品结构(structure)设计；④基于③，完成原型(prototype)设计，将产品结构视觉化为 UI。笔者将这种设计模式称为 DASP 设计模式。

接下来三节将结合 BBS 项目，具体讲解①～③的设计方法。④将单独另开一节详细说明。

3.1.3 数据化设计

所谓数据化设计就是基于用例图，从中提炼出数据实体、相关属性以及相互关系。这可以通过 E-R 图(entity-relationship diagram)，即实体-联系图来实现。

E-R 图由三个要素构成，分别是实体、属性和联系。

实体就是在人的认知上能够相互区分的事物，这个事物可以是现实中的具体事物，也可以是抽象的概念。在 E-R 图中使用矩形框表示。拿 BBS 例子来说，用户、帖子和回复就可以看作是实体，在数据库中将对应一个个表。

属性是实体的特性，也是实体所具备的、可以用数据进行刻画的方面。在 E-R 图中使用椭圆形表示，主属性名下还需要加上下画线。比如，用户的 ID、帖子的内容、回复的发表时间都是属性，在数据库中将对应表的列，也就是“字段”。

联系表示实体之间的关系，也就是以何种方式关联在一起。在 E-R 图中用菱形表示，菱形两边需要通过连线将相关实体连接起来。除此之外，连线上还要标注出实体关联的模式：一对一(1∶1)、一对多(1∶N)还是多对多(M∶N)。在 BBS 例子里，一个帖子会有多个回复，这就是 1∶N 的模式。

综合以上规则将 BBS 项目数据化并用 E-R 图表现出来，效果如图 3.2 所示。同时，也用英文标出将来在数据库建模中会实际使用的实体名和属性名，其中，实体名首字母要大写，属性名保持小写。

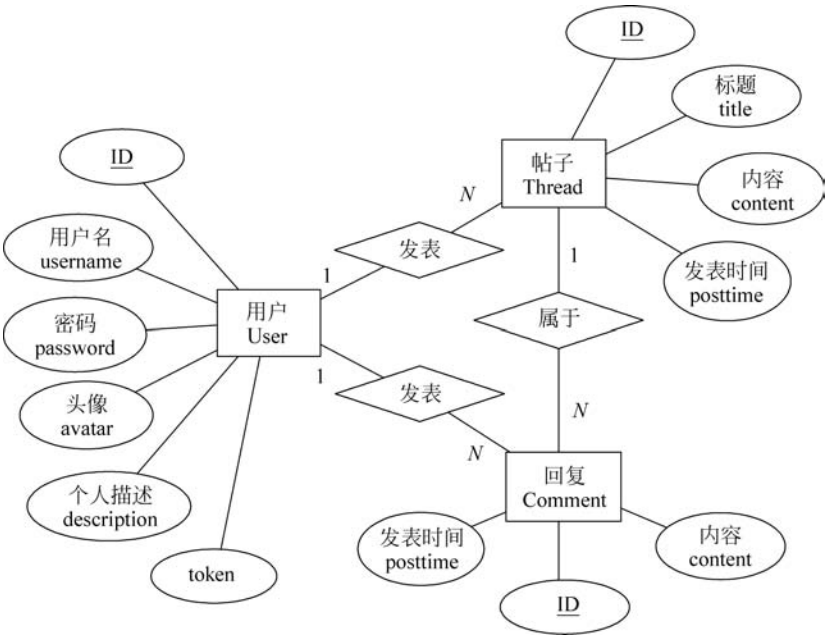


图 3.2 BBS 的 E-R 图

3.1.4 接口化设计

在完成 E-R 图后，就可以进行接口化设计了，目的是结合用例图中的用例以及 E-R 图中的实体与属性设计出清晰合理的 RESTful API。API 将用户的一次次请求尽数化作对数据库的增、查、改、删，是连接需求与信息之间的桥梁。

基于 RESTful API 标准（其具体写法请参见 1.2.2 节），根据产品用例图（见图 3.1），将系统框架内的每个用例都封装成一个 API，规定请求方法和请求 URI：请求方法为 POST、GET、PATCH、DELETE 其中之一，分别对应对数据的增、查、改、删；请求 URI 对应数据库中的实体，像用户 ID、帖子 ID 这样的变量则需要在前面加上一个“:”号。

至于 API 的实际业务逻辑，在现阶段尚无须考虑其具体实现，只需要设计好输入与输出就可以了。对 API 来说，输入就是用户从客户端向服务器发出的请求，输出就是服务器向客户端返回的响应。其中，无论请求体还是响应体都以 JSON 格式承载（参见 1.1.3 节）。

根据 3.1.3 节设计的 E-R 图（见图 3.2）中的三个实体——用户、帖子和回复进行整理分类，就分别成了表 3.1～表 3.3。

表 3.1 BBS 的用户相关接口设计

注册 POST/users				
	字　段	必　选	类　型	说　明
请求体	username	true	string	用户名
	password	true	string	密码
	confirmpass	true	string	确认密码

续表

登录 POST/users/login				
	字 段	必 选	类 型	说 明
请求体	username	true	string	用户名
	password	true	string	密码
响应体	username		string	用户名
	token		string	身份凭证
验证 POST/users/auth				
	字 段	必 选	类 型	说 明
请求体	username	true	string	用户名
	token	true	string	身份凭证
响应体	user		object	用户对象
退出 POST/users/logout				
	字 段	必 选	类 型	说 明
请求体	username	true	string	用户名
	token	true	string	身份凭证
用户信息修改 PATCH/users				
	字 段	必 选	类 型	说 明
请求体	username	true	string	用户名
	token	true	string	身份凭证
	avatar	false		头像图片
	description	false	string	个人描述
浏览发布人信息 GET/users/:username				
	字 段	必 选	类 型	说 明
响应体	username		string	用户名
	avatar		string	头像地址
	description		string	个人描述
	threads		array	帖子列表

表 3.2 BBS 的帖子相关接口设计

浏览帖子列表 GET/threads				
	字 段	必 选	类 型	说 明
响应体	threads		array	帖子列表
浏览帖子内容 GET/threads/:tid				
	字 段	必 选	类 型	说 明
响应体	thread		object	帖子对象

续表

	字　段	必　选	类　型	说　明
发表帖子 POST/threads				
请求体	username	true	string	用户名
	token	true	string	身份凭证
	title	true	string	帖子标题
	content	true	string	帖子内容
修改帖子 PATCH/threads/:tid				
	字　段	必　选	类　型	说　明
请求体	username	true	string	用户名
	token	true	string	身份凭证
	title	true	string	帖子标题
	content	true	string	帖子内容
删除帖子 DELETE/threads/:tid				
	字　段	必　选	类　型	说　明
请求体	username	true	string	用户名
	token	true	string	身份凭证

表 3.3　BBS 的回复相关接口设计

回复帖子 POST/comments/:tid				
	字　段	必　选	类　型	说　明
请求体	username	true	string	用户名
	token	true	string	身份凭证
	content	true	string	回复内容

这三个表都规范了前后端交互将会使用到的字段名（与 E-R 图中的实体属性一致）以及相应的数据类型。为日后的 API 实际开发打下了坚实的基础、提供了可靠的依据。

3.1.5　结构化设计

结构化设计就是以产品的视觉结构为导向，将产品的构造逐级逐层细分，并确定 UI 要素与 API 的对应关系，以实现视觉要素与产品功能的协调统一。

设计结构图首先要从产品的基本模块（或频道）出发，比如 BBS 系统，可以大体分为三个模块：首页、帖子和个人中心，在导航栏中自由切换。

一个模块可能由多个页面构成。所以在第二层，就要具体设计出每个模块所包含的页面，比如帖子模块，就需要一个浏览帖子列表的页面，单击其中的某个帖子后，再进到该帖子内容的页面。这两个页面都和帖子相关，所以归在同一个模块里。

到了第三层，需要将页面再细分为构成元素，这也是日后进行 UI 组件化开发的基础。比如刚才提到的帖子列表页面中，既要包含一个用来浏览的列表，也要包含一个发布帖子的表单。这个列表和表单可以作为组件任务分派给不同的人去实现，最后由项目的负责人组装到一个页面中。

第四层是调用层，这是连接 UI 与 API 的层，也是真正实现产品功能，即用户需求的层。在这一层中所调用的 API 一定要严格遵循 3.1.4 节的接口设计，并确保无遗漏。

最后完成的产品结构图如图 3.3 所示。

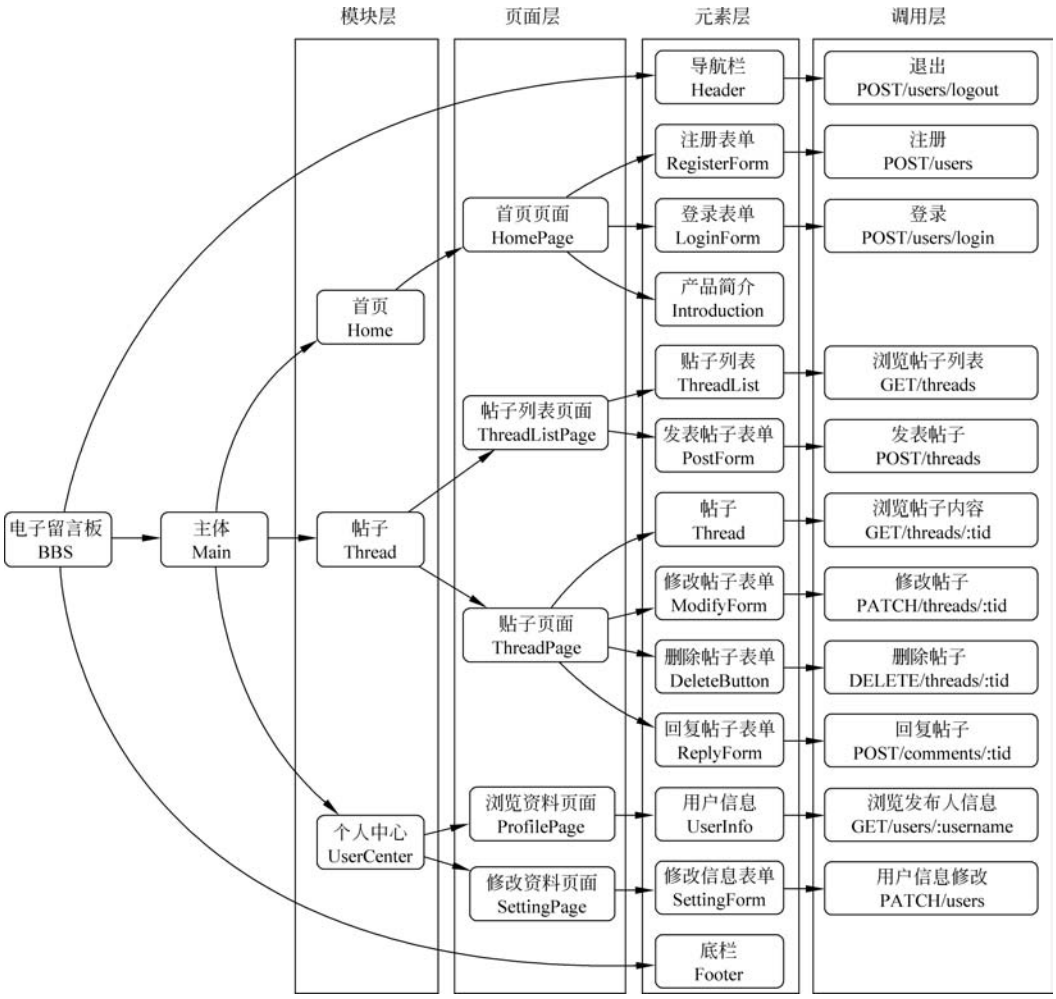


图 3.3　BBS 的产品结构图

为了日后开发方便，每个模块、页面乃至元素都附上了对应的英文名称，将成为日后开发过程中的实际参照，产品结构图中的结构要与日后的实际项目结构保持高度统一。

3.1.6　小结

在本节中，学习了产品设计的基本流程。

产品是为了满足人的需求，所以产品设计也要以实际需求为导向，从需求分析着手。通过绘制用例图梳理出了用户需求，也就完成了产品基本功能的设计。

为了让前期设计与后期开发直接挂钩，从项目宏观上提高效率，笔者提出了一种新的设计模式——DASP。即从数据化设计(D)、接口化设计(A)、结构化设计(S)再到原型设计(P)的流程。

数据化设计中，基于用例图，提炼出实体、属性和关系，完成 E-R 图，成为日后数据库开发的依据。

接口化设计中，基于用例图以及 E-R 图中所确定的实体和属性，设计符合 RESTful API 规范的接口，这成为日后接口开发的参照基准。

结构化设计中，以视觉化为导向将产品构造逐级细分，并在颗粒化到组件级的时候与设计好的接口挂钩，完成视觉要素与产品功能的对接。此处设计好的结构也将与日后的实际项目结构高度统一。

因为结构化的过程完全以视觉化为导向，所以也就成了视觉化设计，即原型设计的重要参照。

3.2 原型设计

所谓原型，就是以图形化的方式表现的产品框架。它可以以一种极为直观的方式展现出成品的样态，既能够在实际投入开发前得到客户的意见反馈、降低返工风险，又是连接产品设计和实际开发的过渡阶段，起到重要的承上启下的作用。

在 3.1.5 节中，完成了产品结构图，这是原型设计的重要参照依据。原型设计的过程也可以看作是将产品结构图彻底视觉化的过程。

3.2.1 原型设计工具

原型设计可以通过手绘，也可以使用现成的原型设计工具。其中，使用范围最广的也是最受产品经理所青睐的工具是 Axure（官方网站是 https://www.axure.com/），这是一款收费软件。要找免费替代品的话，有国产的墨刀（官方网站是 https://modao.cc/）和摹客（官方网站是 https://www.mockplus.cn/）。“墨刀”（Modao）一般使用网页版，属于在线原型设计工具；而摹客（Mockplus）则是离线的桌面端软件。两者都是基础功能免费，高级功能收费。原型设计工具的使用方法大同小异。在本节中，将以摹客为例，讲解原型设计的基本方法。

在官方网站下载并安装后，打开摹客并新建项目，选择项目类型后进入工作界面，其基本区划如图 3.4 所示。

图 3.4 摹客的基本区划

接下来，用一个简单例子介绍摹客的基本使用方法：

(1) 拖动组件。从组件区分别拖动一个“按钮”和一个“单行文字”到工作区。并调整位置。

(2) 调整样式和内容。单击拖入的“单行文字”→在属性区单击“属性”面板→调整字号为 24→双击“单行文字”→将其内容改为“你好”→双击拖入的按钮→将其内容改为“切换”。这样就完成了样式与内容的定制。

(3) 添加交互效果。单击“切换”按钮→按钮右上角出现一个小圆圈→拖动小圆圈到“你好”文本框上→弹出“交互选择”对话框→选中“显示/隐藏”复选框→单击“确定”按钮。

(4) 运行原型。单击工具栏的“演示”按钮→进入到原型的实际运行模式→单击“切换”按钮→“你好”文本框消失→再次单击“切换”按钮→“你好”文本框再次出现。这说明交互效果设置成功。

(5) 页面链接。如果要切换页面，可在项目树新建个页面，按照(3)的做法拖动任意组件的小圆圈到这个页面即可。

基本操作只有这些。通过拖组件、改样式和加交互这三步组合，再复杂的产品结构图也能够轻松转化为产品原型。

3.2.2 产品原型

在 3.1.5 节中完成的产品结构图是以视觉结构为导向，将产品构造逐级细分得到的。因此，产品结构图天然蕴含着原型化的因素。在原型设计过程中，也要严格遵循结构图中所确定的组件结构乃至命名规范，为日后无缝过渡到 UI 开发打好基础。

依据产品结构图，模块应该分成 Home、Thread 和 UserCenter。在项目树中新建三个分组，分别按照这三个模块命名。此为第一层。

在各分组中，继续按照产品结构图添加页面。比如，在 UseCenter 分组中添加 ProfilePage 和 SettingPage 这两个页面，此为第二层。单击左上树状图中的“脑图编辑模式”按钮，可显示脑图，如图 3.5 所示。其构造与结构图完全一致。

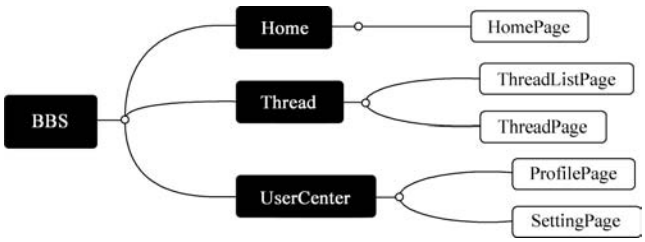


图 3.5 基于摹客的脑图结构

到了第三层，也就是元素/组件层，就需要在各页面中添加元素/组件了。在摹客界面左下角的组件区所给的可拖动组件都是简单的零散组件，而在结构图第三层中所规定的组件都是非常复杂的复合组件。要建立复合组件，只需将摹客所给的简单组件组合起来就可以了。

以登录表单为例：

(1) 确定基底组件。先从组件区的“交互”里拖动一个“面板”组件到工作区。

（2）追加内部元素。双击“面板”组件→进入“编辑”模式→从组件区拖入两个“输入框”、两个“单行文字”，以及一个“按钮”→将两个“单行文字”的内容分别改为“用户名”和“密码”、“按钮”的内容改为“登录”→拖动这几个组件以调整其相对位置。

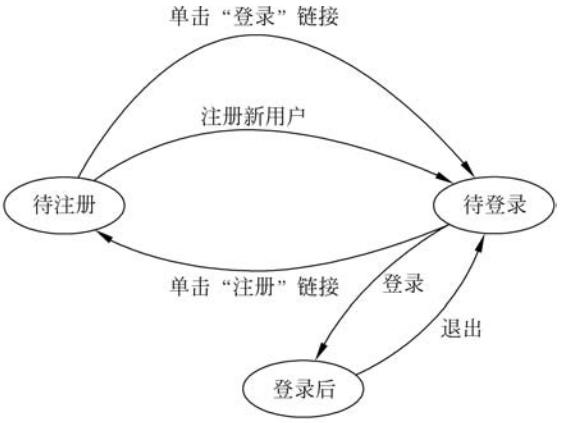
（3）添加调用 API 的备注。按照结构图中的设计，单击“登录”按钮的时候，应该调用对应登录的 API。因此，右击“登录”按钮→选择“备注”选项→输入 POST /users/login→单击“确定”按钮。

（4）重命名。单击“面板”外的工作区→退出“面板”的“编辑”模式→拖动“面板”到任意位置→确认“面板”内的全部组件也随之移动→在右下角的“组件大纲”中将其重命名为 LoginForm，就完成了一个完全符合结构图规定的复合组件。

（5）添加到组件库。右击该组件→选择“添加到我的组件库”选项，以后就可以将刚才做好的 LoginForm 拖动到任意页面的工作区了。特别是对导航栏 Header 和底栏 Footer 这种每个页面都会出现的组件来说，尤为方便。

3.2.3 页面状态切换

一个页面很可能需要根据当前所处状态来显示不同的组件。就拿首页来说，初始状态是待注册状态，显示注册表单；注册新用户或单击“登录”链接后则进入待登录状态，隐藏注册表单，并显示登录表单；此时单击“登录”链接则转为登录后状态，显示产品介绍，并在导航栏右侧显示用户头像和注销链接；在待登录状态下，单击“注册”链接则返回注册表单，隐藏登录表单；在登录后状态下，单击“退出”链接，则隐藏产品介绍表单，显示登录表单，返回待登录状态。以上文字描述用状态迁移图表示出来如图 3.6 所示。



状态	注册表单	登录表单	产品介绍	退出链接
待注册	on	off	off	off
待登录	off	on	off	off
登录后	off	off	on	on

图 3.6 首页的状态迁移图

根据图 3.6 中所示的组件显示/隐藏情况，在摹客中添加组件之间的交互效果即可，这意味着实现了同一页面内的状态切换。单击工具栏的“演示”按钮，实际测试一下原型页面

的状态迁移，看看是否如你所愿，如果有错误的话可退出演示模式重新修正。

其他页面也如法炮制，很快就可以完成整个产品原型。如果还需要在样式上进一步美化，也可以将原型交给专门的设计师继续打磨。

在本节中所完成的原型文件与其他章节的源代码一样，可在 https://github.com/aichatbot/all-plat-dev 下载。下载后的.mp 文件用摹客打开即可。

3.2.4 小结

原型是用图形化方式展现的产品框架，既可以降低返工风险，也是连接产品设计和实际开发的过渡阶段。

根据设计好的产品结构图，可以进一步将其视觉化为产品原型。

设计产品原型时一般使用原型设计工具，Axure 是最受欢迎的原型设计工具，但是免费的墨刀和摹客也能完成绝大多数功能，是非常好的替代品。

在本节中，通过实际的例子，一起学习了包含页面状态切换在内的摹客的基本使用方法，并将在 3.1.5 节所完成的产品结构图彻底转化为了产品原型。至此，设计工作宣告结束。