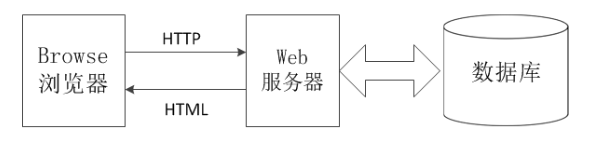
# 第3章 技术介绍

认真阅读本章内容，图、表、公式等，参考此格式进行排版。

## 3.1 B/S体系结构

B/S（Brower/Server,浏览器/服务器）模式又称B/S结构，是Web兴起后的一种网络结构模式。Web浏览器是客户端最主要的应用软件。这种模式统一了客户端，将系统功能实现的核心部分集中到服务器上，简化了系统的开发、维护和使用；客户机上只需要安装一个浏览器，服务器上安装SQL Server, Oracle, MySql等数据库；浏览器通过Web Server同数据库进行数据交互。

B/S架构采取浏览器请求，服务器响应的工作模式。用户可以通过浏览器去访问Internet上由Web服务器产生的文本、数据、图片、动画、视频点播和声音等信息；而每一个Web服务器又可以通过各种方式与数据库服务器连接，大量的数据实际存放在数据库服务器中；从Web服务器上下载程序到本地来执行，在下载过程中若遇到与数据库有关的指令，由Web服务器交给数据库服务器来解释执行，并返回给Web服务器，Web服务器又返回给用户。在这种结构中，将许许多多的网连接到一块，形成一个巨大的网，即全球网。而各个企业可以在此结构的基础上建立自己的Internet。工作原理如图：3.1



## 3.2vue.js框架

### 3.2.1 vue.js介绍

Vue.js是一个轻巧、高性能、可组件化的MVVM库，同时拥有非常容易上手的API； Vue.js是一个构建数据驱动的Web界面的库。Vue.js是一套构建用户界面的 渐进式框架。与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，并且非常容易学习，非常容易与其它库或已有项目整合。另一方面，Vue 完全有能力驱动采用单文件组件和 Vue 生态系统支持的库开发的复杂单页应用。数据驱动+组件化的前端开发。简而言之：Vue.js是一个构建数据驱动的 web 界面的渐进式框架。Vue.js 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。核心是一个响应的数据绑定系统。

### 3.2.2 vue.js数据双向绑定原理

vue数据双向绑定是通过数据劫持结合发布者-订阅者模式的方式来实现的。  
我们已经知道实现数据的双向绑定，首先要对数据进行劫持监听，所以我们需要设置一个监听器Observer，用来监听所有属性。如果属性发上变化了，就需要告诉订阅者Watcher看是否需要更新。因为订阅者是有很多个，所以我们需要有一个消息订阅器Dep来专门收集这些订阅者，然后在监听器Observer和订阅者Watcher之间进行统一管理的。接着，我们还需要有一个指令解析器Compile，对每个节点元素进行扫描和解析，将相关指令（如v-model，v-on）对应初始化成一个订阅者Watcher，并替换模板数据或者绑定相应的函数，此时当订阅者Watcher接收到相应属性的变化，就会执行对应的更新函数，从而更新视图。因此接下去我们执行以下3个步骤，实现数据的双向绑定：

1.实现一个监听器Observer，用来劫持并监听所有属性，如果有变动的，就通知订阅者。

2.实现一个订阅者Watcher，每一个Watcher都绑定一个更新函数，watcher可以收到属性的变化通知并执行相应的函数，从而更新视图。

3.实现一个解析器Compile，可以扫描和解析每个节点的相关指令（v-model，v-on等指令），如果节点存在v-model，v-on等指令，则解析器Compile初始化这类节点的模板数据，使之可以显示在视图上，然后初始化相应的订阅者（Watcher）。

## 3.3Webpack介绍

Webpack是当下最热门的前端资源模块化管理和打包工具。它可以将许多松散的模块按照依赖和规则打包成符合生产环境部署的前端资源。还可以将按需加载的模块进行代码隔离。通过loader的转换，任何形式的资源都可以视作模块，比如CommonJS模块、AMD模块、ES6模块、CSS模块、图片、JSON、Coffeescript、LESS等。简单的来说，Webpack就是一个模块打包器，它对模块的依赖关系进行静态分析，然后把这些模块按照相应的规则放在不同的文件夹里assets，如图3.2

