Web 测试开发实训教程

V3.0

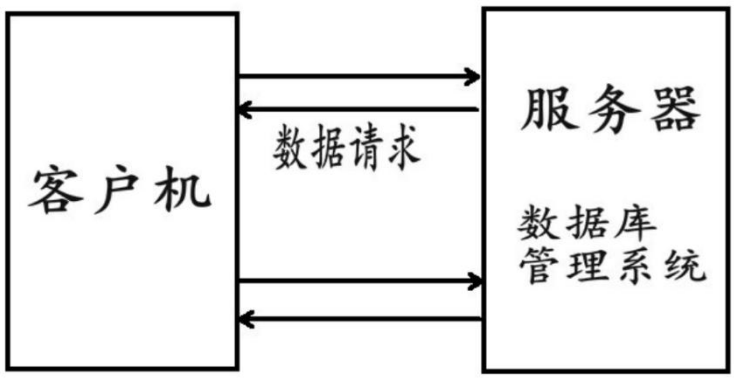
目录

1. Web 基础理论

1.1. Web 系统架构

- **C/S 架构**

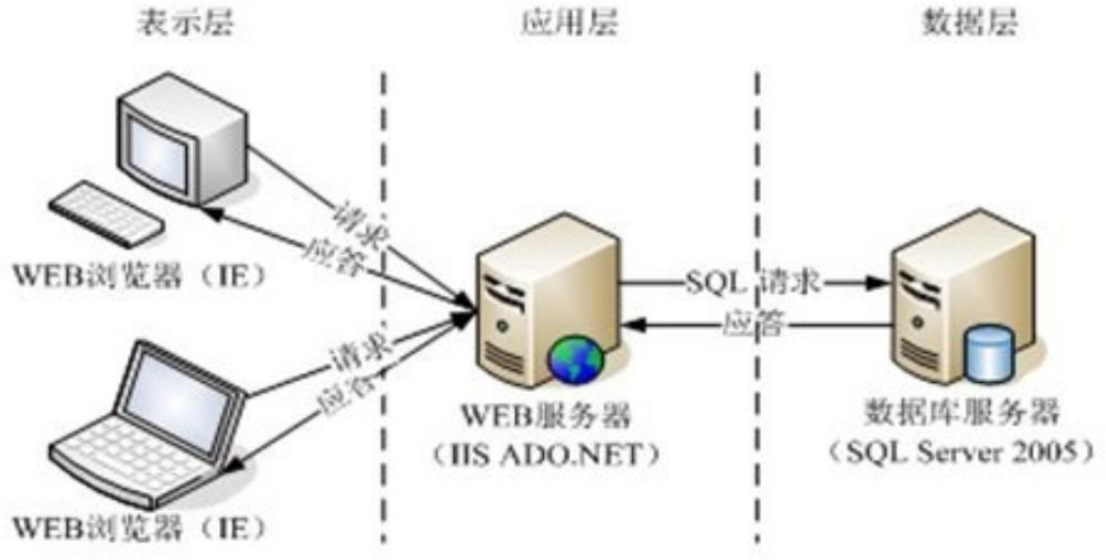
C/S （Client/Server 客户端/服务端）架构也可以看做是胖客户端架构。因为客户端需要实现绝大多数的业务逻辑和界面展示。这种架构中，作为客户端的部分需要承受很大的压力，因为显示逻辑和事务处理都包含在其中，通过与数据库的交互（通常是 SQL 或存储过程的实现）来达到持久化数据，以此满足实际项目的需要。



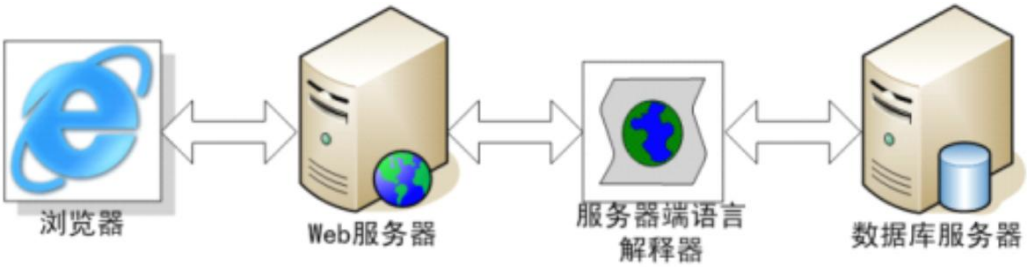
两层C/S架构

- **B/S 架构**

B/S（Browser/Server 浏览器/服务端架构）架构中，显示逻辑交给了 Web 浏览器，事务处理逻辑在放在了 WebApp 上，这样就避免了庞大的胖客户端，减少了客户端的压力。因为客户端包含的逻辑很少，因此也被成为瘦客户端。



B/S 系统架构图



B/S 工作流程示意图

➢ B/S vs C/S

- B/S模式的优缺点：
  - 优点
    - 分布
    - 扩展
    - 维护
  - 缺点
    - 个性化
    - 操作
    - 响应
    - 功能

- C/S模式的优缺点：
  - 优点
    - 响应
    - 界面
    - 业务处理
  - 缺点
    - 分布能力
    - 兼容性
    - 开发成本

1.2. 主流的 Web 应用平台

动态网站应用程序平台的搭建需要使用 Web 服务器发布网页，而 Web 服务器软件又需要安装在操作系统上，并且动态网站都需要使用脚本语言对服务器端进行编程，所以也要在同一个服务器中为 Web 服务器捆绑安装一个应用程序服务器，用于解析服务器端的脚本程序。另外，现在开发的动态网站都是基于数据库的，需要将网站内容存储在数据库中，使用也要为网站选择一款合适的数据库管理软件。这样，一个动态网站服务器平台的最少组合包括：操作系统+Web 服务器+应用服务器+数据库。网站开发平台中的每个组件都有多种可以选择的软件，例如，操作系统可以使用 UNIX、Linux、Windows 等，根据不同的像 ASP、JSP 和 PHP 等脚本语言选择对应的应用服务器，数据库和 Web 服务器更是很多。

➢ **Web 应用程序开发平台对比分析**

目前，网站服务器平台比较常见的有 ASP.NET、JavaEE 和 LAMP 三种：

应用平台	操作系统	Web 服务器	数据库	语言
ASP.NET	Windows	IIS	SQL Server	ASP（C#）
LAMP/LNMP	Linux	Apache/Nginx	MySQL	PHP
JavaEE	Unix	Tomcat	Oracle	JSP

ASP.NET 的服务器端操作系统时使用微软的 Windows，并且需要按照微软的 IIS 网站服务器，数据库管理系统通常是使用微软的 SQL Server，而服务器端编程语言也是使用微软的产品 ASP 技术，就是 ASP.NET 动态网站软件开发平台；JavaEE 的服务器端操作系统使用 UNIX，并在 UNIX 操作系统上安装 Tomcat 或 Webblogic 网站服务器，数据库管理系统使用 Oracle 数据库，服务器端编程语言使用 Oracle 公司的 JSP 技术，就是 JavaEE 动态网站软件开发平

台;LAMP 的服务器端操作系统使用开源的系统 Linux，在 Linux 操作系统上安装自由软件 Apache 网站服务器，数据库管理系统也是采用开源的 MySQL 软件，服务器端脚本编程语言又是使用开源软件 PHP 技术，就是 LAMP 动态网站软件开发平台。

- **ASP.NET 开发平台**

ASP.NET 是 Windows Server+IIS+SQL Server+ASP 组合，所有组成部分都是基于微软的产品。它的优点是兼容性比较好，安装和使用比较方便，不需要太多的配置。ASP.NET 也有很多不足，由于 Windows 操作系统本身存在着问题，ASP.NET 的安全性、稳定性、跨平台都会因为与 Windows 的捆绑而显现出来，因而无法实现跨操作系统的应用，也不能完全实现企业级应用的功能，不适合开发大型系统，而且 Windows 和 SQL Server 软件的价格也不低，平台建设成本比较高。

- **JavaEE 开发平台**

JavaEE 开发架构是 UNIX+Tomcat+Oracle+JSP 的组合，是一个开放的、基于标准的开发和部署的平台，基于 Web 的、以服务端计算为核心的、模块化的企业应用。是一个非常强大的组合，环境搭建比较复杂，同时价格也不菲。Java 的框架利于大型的协同编程开发，系统易维护、可复用性比较好。它特别适合企业级应用系统开发，功能强大，但要难学得多，另外开发速度比较慢，成本也比较高，不适合快速开发和对成本要求比较低的中小型应用系统。

- **LAMP/LNMP 开发平台**

LAMP/LNMP 是 Linux+Apache/Nginx+MySQL+PHP 的标准缩写。Linux 操作系统，网站服务器 Apache、数据库 MySQL 和 PHP 程序模块的连接，形成了一个非常优秀的网站数据库的开发平台，是开源免费的自由软件，与 JavaEE 架构和 ASP.NET 架构形成了三足鼎立的竞争态势，是较受欢迎的开源软件网站开发平台。LAMP 组合具有简便性、低成本、高安全性、开发速度快和执行灵活等特点，使得其在全球发展速度较快，应用较广，越来越多的企业将平台架构在 LAMP/LNMP/LNMPP 之上。

1.3. WEB 的特点

- **图形化，易于导航**

Web 可以在一页上同时显示色彩丰富的图形和文本的性能，并非常易于导航的，只需要从一个连接跳到另一个连接，就可以在各页各站点之间进行浏览了。

- **与平台无关**

无论从 Windows、Linux、UNIX、Mac 平台，还是别的什么平台，都可以通过浏览器访问 WWW 网站。

- **分布式**

大量的图形、音频和视频信息可以放在不同的站点上。使在物理上并不一定在一个站点的信息在逻辑上一体化，然而从用户来看这些信息是一体的。

- **动态**

Web 站点的信息包含站点本身的信息，信息的提供者可以经常对站上的信息进行更新，以保证信息的时间性。

- **交互式**

用户可以向服务器提交请求，服务器可以根据用户的请求返回相应信息.

1.4. WEB 的发展里程

- **单向的 web1.0**

以静态、单向阅读为主，网站内信息可以直接和其他网站信息进行交互，能通过第三方信息平台同时对多家网站信息进行整合使用，用途相当有限，只是简单的信息检索。此时代下的代表站点是一些 BBS、新浪、网易、搜狐三大门户网站等。

- **互动的 web2.0**

以分享为特征的实时网络，用户在互联网上拥有自己的数据，并能在不同的网站上使用。一大特征是社交网络的兴起，具有代表的是开心网、校内网、微博、微信。

- **值得期待的 web3.0**

将以网络化和个性化为特征，提供更多人工智能服务，完全基于 Web，用浏览器即可实现复杂的系统程序才具有的功能。Web3.0 的特征分析：

- 微内容(Widget)的自由整合与有效聚合；
- 适合多种终端平台，实现信息服务的普适性；
- 良好的人性化用户体验，以及基础性的个性化配置；
- 有效和有序的数字新技术；

1.5. Web 工作原理&HTTP 协议

名词解释：

TCP 协议：一种面向连接的、可靠的、基于字节流的传输层通信协议。

http 协议：建立在 tcp 协议基础之上的传输层协议。

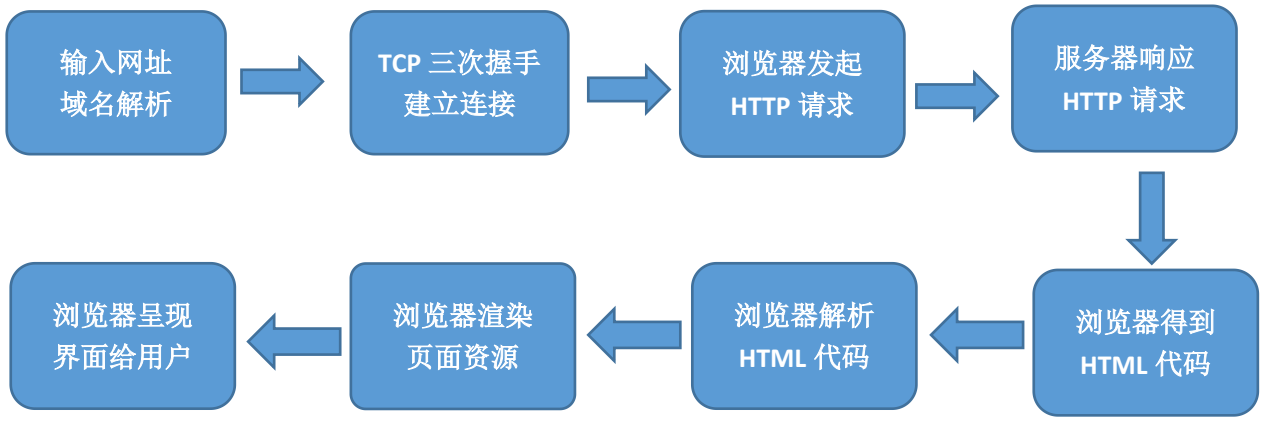
全双工：通讯传输的一个术语。通信允许数据在两个方向上同时传输，它在能力上相当于两个单工通信方式的结合。

Https 协议：HTTPS 就是 http+ssl。ssl 是一个加密协议，http 本身传输内容是明文，Https 是加密传输，传输就是密文。http 端口是 80，https 端口是 443。

HTTP 请求与应答

- **HTTP 请求应答过程**

域名解析 --> 发起 TCP 的 3 次握手 --> 建立 TCP 连接后发起 http 请求 --> 服务器响应 http 请求，浏览器得到 html 代码 --> 浏览器解析 html 代码，并请求 html 代码中的资源（如 js、css、图片等） --> 浏览器对页面进行渲染呈现给用户



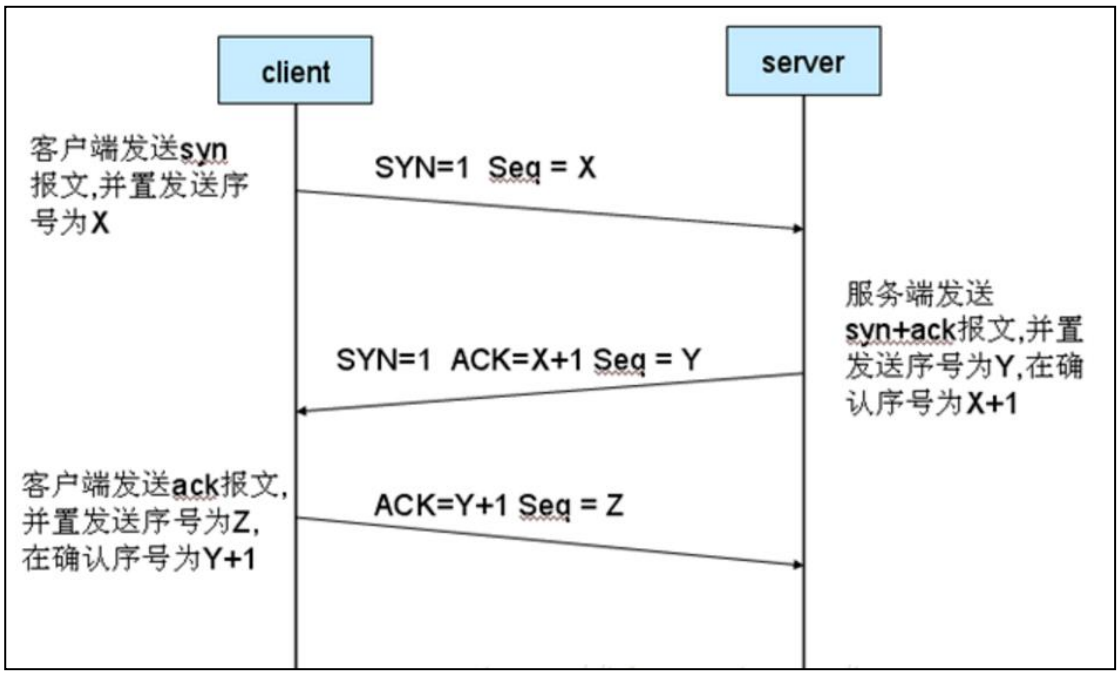
- **Http 请求方法**
  - GET 请求
  - POST 请求
  - GET 与 POST 的区别

1. GE 通过 URL 提交数据，数据在 URL 中可以看到； POST 提交的数据放在 HTTP 包的 Body 中。
2. GET 提交的数据大小有限制（因为浏览器对 URL 的长度有限制）， 而 POST 则没有此限制。
3. 安全性问题：GET 提交数据的时候，参数会显示在地址栏上，而 POST 不会。所以，如果这些数据是中文数据而且是非敏感数据，那么使用 GET；反之使用 POST 为好。
4. 服务器取值方式不一样。GET 方式取值，如 PHP 可以使用$_GET 来取得变量的值，而 POST 方式通过$_POST 来获取变量的值。

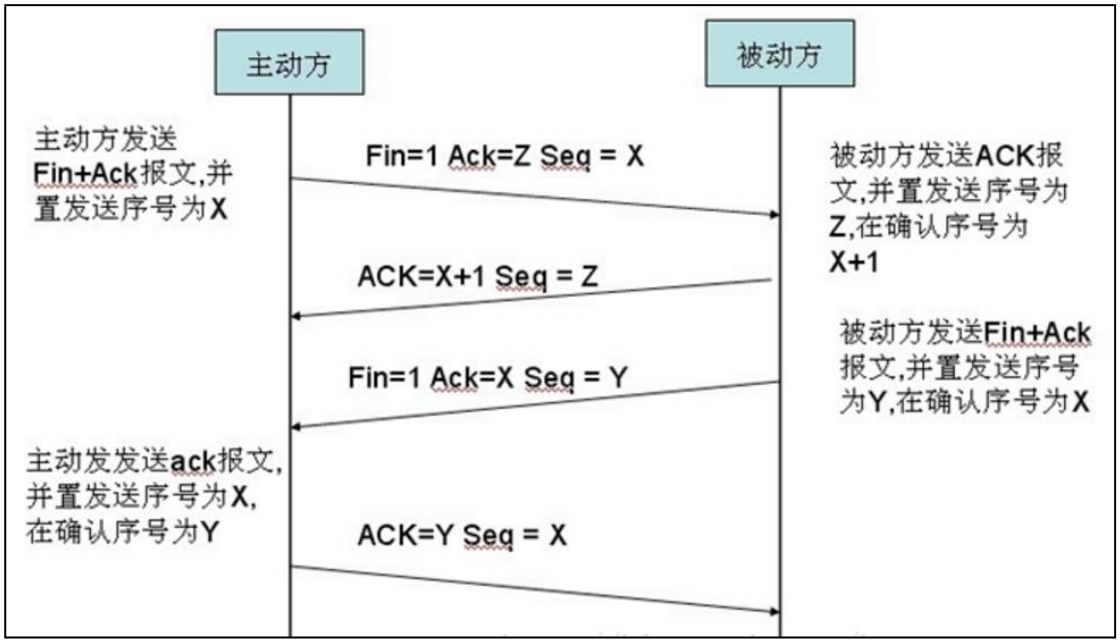
- **应答与状态码**
  - 2XX 系列 成功处理请求
  - 3XX 系列 完成请求需要进一步处理
  - 4XX 系列 请求错误（系统程序错误）
  - 5XX 系列 服务器内部错误
  - 常见状态代码、状态描述的说明如下:

状态码	状态码英文名称	中文描述
200	OK	请求成功。一般用于 GET 与 POST 请求
302	Found	临时移动。与 301 类似。但资源只是临时被移动。 客户端应继续使用原有 URI
403	Forbidden	服务器理解请求客户端的请求，但是拒绝执行此请求
404	Not Found	服务器无法根据客户端的请求找到资源（网页）。 通过此代码，网站设计人员可设置"您所请求的资源无法找到"的个性页面
500	Internal Server Error	服务器内部错误，无法完成请
501	Not Implemented	服务器不支持请求的功能，无法完成请求
502	Bad Gateway	充当网关或代理的服务器，从远端服务器接收到了一个无效的请求

- **TCP 三次握手 — 建立连接**



- TCP 四次挥手 — 关闭连接



1.6. Cookie & Session & Token

Cookie：Cookie 的产生背景是解决 http 协议无状态的问题。我们知道 http 的一个特性就是无状态，一次请求完成后，下一个请求和上一个没有任何联系。所以我们利用 cookie 来解决这个问题，我们可以在浏览器里看到 cookie 设置在 C 盘某个文件目录里，其实就是 txt 文本文件，前面一个 http 请求写到 cookie 里的内容，后面一个 http 请求可以去读取，这样前后就关联起来了。所以解决了 http 请求的无状态特性。cookie 都是保留在用户端电脑。

cookie 实际应用案例：第一：自动登录---很多同学都发现，我们经常登录的网站，并不是每次访问都要输入用户名和密码登录。只需要第一次登录后，后面都不需要再输入用户

名和密码。这里利用的就是 cookie 技术，用户名和密码已经记录到了 cookie，服务器验证没问题就直接进入登录后页面了。如果清除 cookie，又会再次让你登录。第二：广告投放---大家有过这样经历，我们如果浏览或者搜索过一些商品，后面就会连续在网页推荐类似商品，这里也是 cookie 记录了我们的浏览动作。所以给其他网站广告推荐奠定了基础。

Session：提到 cookie 就绕不开 session，session 也叫会话控制。和 cookie 不同，session 是在服务端的。用户第一次访问服务器，会生成一个 session ID 来标识用户并保存信息，相当于是客户端的身份证，再次访问服务器就知道是谁了。这个 session ID 存在 cookie 中，所以访问的时候服务器知道是哪个用户。

Token：token 就是令牌，你拿到 token，就像拿到一个令牌，多用于认证。token 一般就是一长串随机码，而且有时长限制，不可能一直拿这个 token 使用。

比如一个抽奖接口，要求必须是登录后才能进行抽奖。所以抽奖接口要求一个请求参数就是 token。用户在登录后，就会生成一个唯一的 token 作为标识。一般这个 token 会写入 cookie 中。

2. Web 开发技术

2.1. HTML 基本元素

```
<!DOCTYPE  html>
<html  lang="en">
<head>
        <meta  charset="UTF-8">
        <title>Title</title>
</head>
<body>

</body>
</html>
```

- **基本常用标签**
  - 段落标签
  - 标题标签 <h1>...<h6>
  - 换行标签：
  - 图片标签 <img />
  - 超链接标签 <a>
  - 表单标签 <form></form>
  - 输入标签：<input />
  - 选择标签：<select></select>
  - 文本域标签：<textarea></textarea>

```
<!DOCTYPE  html>
<html  lang="en">
<head>
        <meta  charset="UTF-8">
        <title>Title</title>
```

```
</head>
<body>
<p>This is my first paragraph.</p>
<a href="http://www.baidu.com/" target="_blank">百度一下</a>
<img src="logo.png" alt="logo">
</body>
</html>
```

- Table 表格

```
<table>
        <tr><th colspan="2">用户登录</th> </tr>
        <tr><th>用户名</th> <td>harry</td></tr>
        <tr><th>密码</th> <td>123456</td></tr>
</table>
```

- form 标签

action:表单提交的地址；
method：表单提交的方法，主要有 post 和 get 两种；

```
<form method="post" action="login.php">
<table>
        <tr><th colspan="2">用户登录</th></tr>
        <tr><th>用户名
</th> <td><input name="username" type="text" value="" /></td></tr>
        <tr> <th>密码
</th> <td><input name="passwd" type="password" value="" /></td></tr>
        <tr><td colspan="2"><input name="sub" type="submit" value="登录
" /></td></tr>
</table>
</form>
```

- 输入框<input>：

type:text、radio、checkbox、password、button、submit、reset;
类型：文本、单选、多选、密码、按钮、提交、重置；

```
<input   name="username" type="text" value="" >
```

每个输入框都必须要有 name,type,value 三个基本的属性；

- 下拉框<select>:

<option>
默认选择 <option selected="selected" ></option>

- 文本域<textarea>:

```
<form action="register.php" method="post" >
```

```
2.          <table  align="center"  border="1">
3.                  <tr><td  colspan="3">用户注册</td></tr>
4.                  <tr>
5.                          <th>用户名</th>
6.                          <td><input    name="username"  type="text"  value=""></td>
7.                          <td>  <font  color="red"></font></td>
8.                  </tr>
9.                  <tr>
10.                         <th>email</th>
11.                         <td><input  name="email"  type="text"  value=""></td>
12.                         <td>  <font  color="red">  *</font></td>
13.                 </tr>
14.                 <tr>
15.                         <th>性别</th>
16.                         <td  colspan="2">
17.                                 <input  name="sex"  type="radio"  value="男"  >
    男
18.                                 <input  name="sex"  type="radio"  value="女
    "  checked="checked">女
19.                         </td>
20.                 </tr>
21.                 <tr>
22.                         <th>密码</th>
23.                         <td><input  name="passwd"  type="password"  value=""   ></td
    >
24.                         <td>  <font  color="red">  *</font></td>
25.                 </tr>
26.                 <tr>
27.                         <th>MSN</th>
28.                         <td><input  name="MSN"  type="text"  value=""   ></td>
29.                         <td>  <font  color="red">  *</font></td>
30.                 </tr>
31.                 <tr>
32.                         <th>QQ</th>
33.                         <td><input  name="qq"  type="text"  value=""   ></td>
34.                         <td>  <font  color="red">  *</font></td>
35.                 </tr>
36.                 <tr>
37.                         <th>办公电话</th>
38.                         <td><input  name="jobtel"  type="tel"  value=""   ></td>
39.                         <td>  <font  color="red">  *</font></td>
40.                 </tr>
41.                 <tr>
```

```
                    <th>家庭电话</th>
                    <td><input  name="hometel"  type="tel"  value=""  ></td>
                    <td>  <font  color="red">  *</font></td>
                </tr>
                <tr>
                    <th>手机</th>
                    <td><input  name="phonetel"  type="text"  value=""  ></td>
                    <td>  <font  color="red">  *</font></td>
                </tr>
                <tr>
                    <th>兴趣爱好</th>
                    <td  colspan="2">
                        <input    name="aihao"  type="checkbox"  value="旅游"  >旅游
                        <input    name="aihao1"  type="checkbox"  value="书法"  >书法
                    </td>
                <tr>
                    <th>密码提示问题</th>
                    <td>
                        <select  name="mmts"  style="width:  200px">
                            <option  >请选择密码提示问题</option>
                            <option  value="moren">我最好朋友的生日</option>
                            <option  value="moren">我最喜欢的事物</option>
                        </select>
                    </td>
                </tr>
                <tr>
                    <th>密码问题答案</th>
                    <td><input  name="answer"  type="text"  value=""  ></td>
                    <td>  <font  color="red">  *</font></td>
                </tr>
                <tr>
                    <td></td>
                    <td  colspan="2"  style="font-size:  small;  text-align:  left">
                        <input  name="xuanze"  type="checkbox"  value="选择"  checked="checked">我已经看过并接受
                    《<a  href="http://localhost/upload/article.php?cat_id=-1"  target="_blank"  style="text-decoration:  none">用户协议</a>》
```

```
                    </td>
                </tr>
                <tr>
                    <td colspan="3" align="center">
                        <input name="hyzc" type="submit" value="会员注册" id="sub">
                    </td>
                </tr>
            </table>
            <div style="font-size: x-small; text-align: center">
                <a href="http://localhost/upload/user.php?act=qpassword_name" style="text-decoration: none;color: pink" >密码问题找回密码</a>
                <a href="http://localhost/upload/user.php?act=get_password" style="text-decoration: none;color: pink">注册邮件找回密码</a>
            </div>
</form>
```

边框：border：1px solid red;
字体样式 ： font ,font-size,font-weight,font-color
列表样式 ： ul,li {list-style:none;}
RGB 三色体 Red Green Blue
#000000 == #000 黑色
#FFFFFF == #FFF 白色
#ff3022 == #f32
#4e2f89

2.2. CSS 基本样式

- **CSS 层叠样式**

(1)、行内样式表（style 属性）

```
<p style=" color:red;font-size:14px;">hello world</p>
```

(2)、内部嵌入样式表（style 元素）

```
<style type="text/css">
        body {background-color: #FF0000;}
</style>
```

(3)、外部引用样式表（引用一个样式表文件），推荐；

```
<link rel="stylesheet" type="text/css" href="css/style.css" />
```

- **CSS 样式选择器**
  - **ID 选择器：#idName {} 样式唯一；**

```
#sub{    font-size:  24px;  width:  200px;  height:  50px;  background:  aliceblue;}
```

- **类选择器：.className{}　样式共用；**

```
        .ipt{width:  200px;  height:  30px;  }
```

- **标签选择器：html 自带的标签；**

```
body{    background:  red  url('../img/bj.jpg');
         font-size:  20px;  font-family:  微软雅黑,宋体;  font-weight:  bolder;}
```

- 样式优先级：
  ID 选择器>类选择器> 属性选择器
  行内样式 > 内嵌样式 > 外部样式
- CSS 常用样式属性

width　height　border
font　color
background

2.3.　Div+CSS 页面布局

- 外间距: margin
  - margin-left ,margin-right ,margin-top ,margin-bottom
  - margin：上 右 下 左； 4 个值
  - margin：上 左/右 下； 3 个值
  - margin：上/下　左/右；2 个值
  - margin：上/下/左/右； 1 个值
  - div 左右居中　margin：10px　auto;
- 内间距：padding
- 浮动 ： float:left,right;
- 清除浮动　clear:both;

```
<div class="wrap">
    <div class="header"></div>
    <div class="content"></div>
    <div class="footer"></div>
</div>
```

Demo 练习: 模仿百度首页；

2.4.　JavaScript 基础

- **基本用法**
  - 内部调用；

```
<div>
<script  language="javascript">
alert(“hello  world,this  is  my  first  javascript”);
</script>
</div>
```

- 外部引用

```
<script  type="text/javascript"  language="javascript"  src="js/web.js"></script>
```

➢ 变量流程控制语句

- 变量定义：

```
var  name  = "Harry"
var  a  =  1;
var  m=0,n=0;
var  d  =  new  Date();

var  cars=["Audi","BMW","Volvo"];
```

- 流程控制语句：

```
<script  type="text/javascript">
var  age=18;
if(  age  >18
        document.write("已成年");
}else{
        document.write("已成年");
}

for(var  i=1;i<10;  i++){
          document.write(“hello，这是第”+    i  +"次问候"  )
}

var  m=10;
while  (  m>0  ){
          document.write("第"+m+"次循环");
          m-=1;
}

</script>
```

- 弹框
- 警告消息弹框 alert（）

- 确认消息弹框 confirm()

```
<script  type="text/javascript">
var  d=confirm("是否提交？");
if(d==true){
      alert("提交成功");
}else{
      alert("取消提交");
}

</script>
```

- 提示消息弹框

```
<script  type="text/javascript">
var  name  =  prompt("please  input  your  name","Harry");
</script>
```

- 常用事件：

onclick=""//单击事件
ondblclick="" //双击事件
onfocus="" //聚焦事件
onblur="" //失去焦点
onmousemover="" //鼠标悬浮
onmouseout="" //鼠标移开

- **函数**
  - 无参函数

```
<script  type="text/javascript">
function  showPwdMsg()  {
        document.getElementById("pwdId").style.display="block";
}

function  hidePwdMsg()  {
        document.getElementById("pwdId").style.display="none";
}

</script>
```

  - 有参函数

```
<script  type="text/javascript">
function  hideMsg(id)  {
        document.getElementById(id).style.display="none";
}
```

```
5.
6.  function  showAndHideMsg(id,status)  {
7.          document.getElementById(id).style.display=status;
8.  }
9.
10. </script>
```

- **document 获取元素方法**

```
1.  document.getElementById()                    //返回带有指定 ID 的元素
2.  document.getElementsByClassName()          //带有相同类名的所有 HTML 元素
3.  document.getElementsByName()             //返回带有指定名称属性的元素对象集合
4.  document.getElementsByTagName()     //返回带有指定标签名的对象集合
5.
6.  var  value=document.getElementById("search").value;
7.  var  val  =document.getElementsByName("keywords")[0].value;
8.  var  val  =document.getElementsByTagName("input");
9.  var  num  =  val.length;
10. var  keyword  =  "input  标签总数："
11. alert(keyword  +  num);
12.
13. for(var  m=0;m<num;m++){
14.         alert("第"+  m  +  "值为："+val[m].value);
15. }
```

3.Web 测试技术

3.1 Web 界面测试

Web 界面测试的目标：

- Web 界面的实现与设计需求、设计图保持一致，或者符合可接受标准
- 使用恰当的控件，各个控件及其属性符合标准
- 通过浏览测试对象可正确反映业务的功能和需求

如果有不同浏览器兼容性的需求，则需要满足在不同内核浏览器中实现效果相同的目标

Web 界面测试方法

针对 Web 应用的界面测试，可以从以下方面进行用户界面测试：控件测试、多媒体测试、内容测试、容器测试、浏览器兼容性测试、整体界面测试等。

- **控件测试**

Web 应用与其他应用程序一样，也有许多用以实现各种功能或者操作的控件，比如常见的按钮、单选框、复选框、下拉列表框等等。最基本的当然需要考虑每一个控件其功能是否达到使用要求，是否合适的使用。有状态属性的控件在进行多种操作之后，控件状态是否依然能够保持正确，界面信息是否显示正常。

- **多媒体测试**

  - 现如今的 Web 应用中，主流的一些多媒体内容包括图片、GIF 动画、Flash、Silverlight 等。可以通过以下方面进行测试：

- 要确保图形有明确的用途，图片或动画排列有序并且目的明确;
- 图片按钮链接有效，并且链接的属性正确（比如是新建窗口打开还是在当前页面打开）;
- 背景图片应该与字体颜色和前景颜色相搭配;
- 检查图片的大小和质量，一般采用 JPG、GIF、PNG 格式，并且在不影响图片质量的情况下能使图片的大小减小到 30k 以下;
- GIF 动画是否设置了正确的循环模式，其颜色是否显示正常;
- Flash、Silverlight 元素是否显示正常。如果是控件类，功能是否能够实现;

➢ **内容测试**

- 内容测试用来检验 Web 应用系统提供信息的正确性、准确性和相关性。
- 验证所有页面字体的风格是否一致，包括字体，颜色，字号等方面;
- 导航是否直观，Web 应用的主要功能是否可通过主页索引;
- 站点地图和导航功能位置、是否合理;
- Web 页面结构、导航、菜单、超级链接的风格是否一致，比如指向超级链接，点击超级链接，访问后的超级链接是否都进行了处理;
- 背景颜色应该与字体颜色和前景颜色相搭配;
- 验证文字段落、图文排版是否正确，文字内容是否完整显示，图片是否按原有比例显示;
- 检查是否有语法或拼写错误，文字表达是否恰当，超级链接引用是否正确;
- 链接的形式、位置、是否易于理解;

➢ **容器测试**

DIV 和表格在页面布局上的基本作用都是作为一种容器。其中，表格测试分为两个方面，一方面是作为控件，需要检测其是否设置正确，每一栏的宽度是否足够宽，表格里的文字是否都有折行，是否有因为某一格的内容太多，而将整行的内容拉长等；另一方面，表格作为较早的网页布局方式，目前依然有很多的 Web 页使用该方式实现 Web 页设计，此时则需要考虑浏览器窗口尺寸变化、Web 页内容动态增加或者删除对 Web 界面的影响。

DIV+CSS 测试则需要界面符合 W3C 的 Web 标准，W3C 提供了 CSS 验证服务，可以将用 DIV+CSS 布局的网站提交至 W3C,帮助 Web 设计者检查层叠样式表（CSS）。

还需要测试，在调整浏览器窗口大小时，页面在窗口中的显示是否正确、美观，页面元素是否显示正确。

➢ **浏览器兼容性测试**

主要测试在主流浏览器（IE6、IE8、Chrome、Firefox、Opera 等）中 Web 界面是否显示正确，包括页面元素是否显示正确，功能是否能够满足要求。其中很多问题都是非常细致的问题，比如界面元素边框相差 1px,2px 等。

➢ **整体界面测试**

对整体界面的测试过程，其实是一个对最终用户进行调查的过程。可以通过外部人员（与 Web 应用系统开发没有联系或联系很少的人员）的参与，得到最终用户的反馈信息。

➢ **用户体验测试**

用户体验是用户在使用过程中建立起来的一种纯主观感受。例如：网页的重要内容一般放到页面左上部分，因为人的视觉注意力会首先锁定在左上的位置；按钮的按下状态和平常状态一般是不同的，因为用户需要反馈；网页访问的速度一般在 2,3 秒内加载完成，超过这个时间用户会不耐烦。这些都是用户形成的基本合理的主观感受，站在这个角度，我们去审视一个网站功能是否符合人性，响应速度用户是否可接受。

3.2 Web 安全测试

（1）SQL 注入（比如登陆页面）
（2）XSS 跨网站脚本攻击：程序或数据库没有对一些特殊字符进行过滤或处理，导致用户所输入的一些破坏性的脚本语句能够直接写进数据库中，浏览器会直接执行这些脚本语句，破坏网站的正常显示，或网站用户的信息被盗,构造脚本语句时，要保证脚本的完整性。

```
document.write("abc")
<script>alter("abc")</script>
```

（3）URL 地址后面随便输入一些符号，并尽量是动态参数靠后
（4）验证码更新问题
（5）现在的 Web 应用系统基本采用先注册，后登陆的方式。因此，必须测试有效和无效的用户名和密码，要注意到是否大小写敏感，可以试多少次的限制，是否可以不登陆而直接浏览某个页面等。
（6）Web 应用系统是否有超时的限制，也就是说，用户登陆后在一定时间内（例如 15 分钟）没有点击任何页面，是否需要重新登陆才能正常使用。
（7）为了保证 Web 应用系统的安全性，日志文件是至关重要的。需要测试相关信息是否写进了日志文件、是否可追踪。
（8）当使用了安全套接字时，还要测试加密是否正确，检查信息的完整性。
（9）服务器端的脚本常常构成安全漏洞，这些漏洞又常常被黑客利用。所以，还要测试没有经过授权，就不能在服务器端放置和编辑脚本的问题。

3.3 常见输入框测试

- **字符型输入框**

（1）字符型输入框：英文全角、英文半角、数字、空或者空格、特殊字符“~！@#￥%……&*？[]{}”特别要注意单引号和&符号。禁止直接输入特殊字符时，使用“粘贴、拷贝”功能尝试输入。
（2）长度检查：最小长度、最大长度、最小长度-1、最大长度+1、输入超工字符比如把整个文章拷贝过去。
（3）空格检查：输入的字符间有空格、字符前有空格、字符后有空格、字符前后有空格
（4）多行文本框输入：允许回车换行、保存后再显示能够保存输入的格式、仅输入回车换行，检查能否正确保存（若能，检查保存结果，若不能，查看是否有正常提示）、
（5）安全性检查：输入特殊字符串（null,NULL, ,javascript,<script>,</script>,<title>,<html>,<td>）、输入脚本函数(<script>alert("abc")</script>)、doucment.write("abc")、hello）

- **数值型输入框**

（1）边界值：最大值、最小值、最大值+1、最小值-1
（2）位数：最小位数、最大位数、最小位数-1 最大位数+1、输入超长值、输入整数
（3）异常值、特殊字符：输入空白（NULL）、空格或"~!@#$%^&*()_+{}|[]\:"<>?;',./?;:'-=等可能导致系统错误的字符、禁止直接输入特殊字符时，尝试使用粘贴拷贝查看是否能正常提交、word 中的特殊功能，通过剪贴板拷贝到输入框，分页符，分节符类似公式的上下标

等、数值的特殊符号如Σ，log，ln，Π，+，-等；
（4）输入负整数、负小数、分数、输入字母或汉字、小数（小数前 0 点舍去的情况，多个小数点的情况）、首位为 0 的数字如 01、02、科学计数法是否支持 1.0E2、全角数字与半角数字、数字与字母混合、16 进制，8 进制数值、货币型输入（允许小数点后面几位）、
（5）安全性检查：不能直接输入就 copy

- **日期型输入框**

（1）合法性检查：(输入 0 日、1 日、32 日)、月输入[1、3、5、7、8、10、12]、日输入[31]、月输入[4、6、9、11]、日输入[30][31]、输入非闰年，月输入[2]，日期输入[28、29]、输入闰年，月输入[2]、日期输入[29、30]、月输入[0、1、12、13]
(2)异常值、特殊字符：输入空白或 NULL、输入~！@#￥%……&*（）{}[]等可能导致系统错误的字符
（3）安全性检查：不能直接输入，就 copy，是否数据检验出错？
4、信息重复:在一些需要命名,且名字应该唯一的信息输入重复的名字或 ID,看系统有没有处理,会否报错,重名包括是否区分大小写,以及在输入内容的前后输入空格,系统是否作出正确处理.

3.4 搜索功能测试

若查询条件为输入框，则参考输入框对应类型的测试方法

- **功能实现**

（1）如果支持模糊查询，搜索名称中任意一个字符是否能搜索到
（2）比较长的名称是否能查到
（3）输入系统中不存在的与之匹配的条件
（4）用户进行查询操作时，一般情况是不进行查询条件的清空，除非需求特殊说明。

- **组合测试**

（1）不同查询条件之间来回选择，是否出现页面错误（单选框和多选框最容易出错）
（2）测试多个查询条件时，要注意查询条件的组合测试，可能不同组合的测试会报错。

3.5 增删改功能测试

- **增加、修改功能**

- **特殊键：**

  （1） 是否支持 Tab 键 （2）是否支持回车键

- **特殊键**

  不符合要求的地方是否有错误提示

- **唯一性**

  字段唯一的，是否可以重复添加，添加后是否能修改为已存在的字段（字段包括区分大小写以及在输入的内容前后输入空格，保存后，数据是否真的插入到数据库中，注意保存后数据的正确性）

- **数据正确性**

  （1）对编辑页的每个编辑项进行修改，点击保存，是否可以保存成功，检查想关联的数据是否得到更新。
  （2）进行必填项检查（即是否给出提示以及提示后是否依然把数据存到数据库中；是否提示后出现页码错乱等）

（3）是否能够连续添加（针对特殊情况）

（4）在编辑的时候，注意编辑项的长度限制，有时在添加的时候有，在编辑的时候却没有（注意要添加和修改规则是否一致）

（5）对于有图片上传功能的编辑框，若不上传图片，查看编辑页面时是否显示有默认的图片，若上传图片，查看是否显示为上传图片

（6）修改后增加数据后，特别要注意查询页面的数据是否及时更新，特别是在首页时要注意数据的更新。

（7）提交数据时，连续多次点击，查看系统会不会连续增加几条相同的数据或报错。

（8）若结果列表中没有记录或者没选择某条记录，点击修改按钮，系统会抛异常。

- **删除功能**

◆ **提示信息**

（1）不选择任何信息，直接点击删除按钮，是否有提示

（2）删除某条信息时，应该有确认提示

◆ **数据实现**

（1）是否能连续删除多个产品（2）当只有一条数据时，是否可以删除成功 （3）删除一条数据后，是否可以添加相同的数据（4）如系统支持批量删除，注意删除的信息是否正确 （5）如有全选，注意是否把所有的数据删除（6）删除数据时，要注意相应查询页面的数据是否及时更新 （7）如删除的数据与其他业务数据关联，要注意其关联性（如删除部门信息时，部门下游员工，则应该给出提示）（8）如果结果列表中没有记录或没有选择任何一条记录，点击删除按钮系统会报错。

◆ **流程测试**

如：某一功能模块具有最基本的增删改查功能，则需要进行以下测试

单项功能测试（增加、修改、查询、删除）

- 增加——>增加——>增加 （连续增加测试）
- 增加——>删除
- 增加——>删除——>增加 （新增加的内容与删除内容一致）
- 增加——>修改——>删除
- 修改——>修改——>修改 （连续修改测试）
- 修改——>增加（新增加的内容与修改前内容一致）
- 修改——>删除
- 修改——>删除——>增加 （新增加的内容与删除内容一致）
- 删除——>删除——>删除 （连续删除测试）

4. 互联网 web 系统

4.1 URL 访问的过程

我们每天都在访问不同网站，但是大家对于这个过程有了解吗，下面我们来分析一下整个访问过程。

我们访问一个网站，输入的是域名:www.xxx.com，但是在真实网络环境中，域名是无法定位到一个网站的，所以这里有一个 DNS 的服务，叫做域名解析服务。这个服务会把域名转换成 IP 地址，我们每个人的电脑都设置了 IP，可能是固定也可能是动态 IP。所以真实互联网环境中，IP 地址才能够帮助我们找到一台目标服务器，也就是我们要访问的网站。

找到了目标服务器后，还有一个问题，一台服务器开放的服务是很多的，就像我们每个

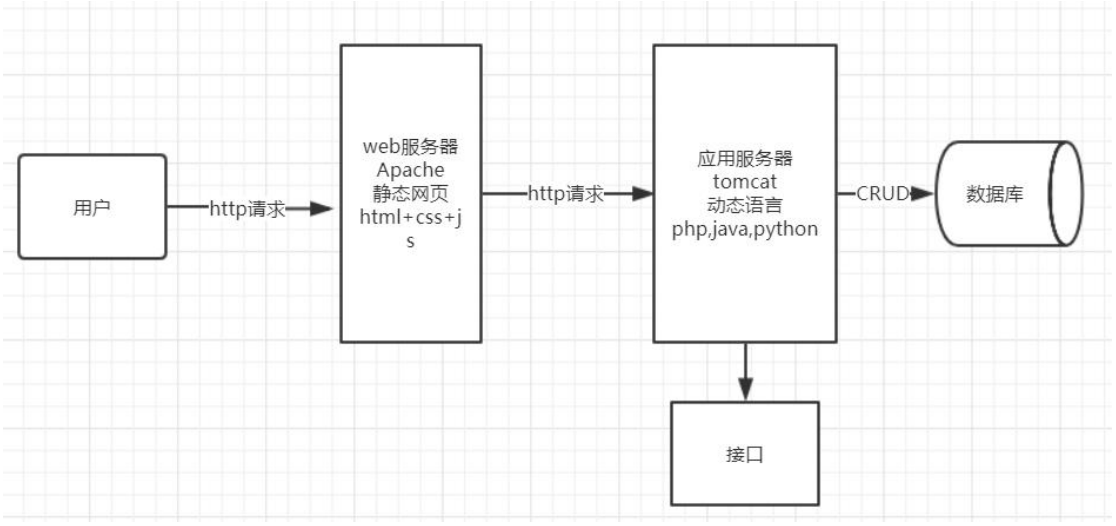
人电脑，可能同时做几件事，比如 编辑文本，放音乐，看电影等。所以我们必须让服务器知道，我们请求的是哪一个服务，这个就是端口来决定的。每种服务都有固定的端口。我们访问网站就是 web 服务，web 服务的端口就是 80，但是我们一般访问不写出来端口，因为这个默认会自己加上去。

总结： IP+端口 就能准确定位到 某一个服务。剩下的事情就是请求进入服务器后，服务器的处理过程和响应了，也就是前面说的 web 服务器和应用服务器的内容。

4.2 Web 前端和后端

学习完前面部分大家对于 web 网站的前后端有了基本的概念。理解了什么是静态网页和动态网页。

下图是一个用户请求的完整过程，请求首先到 web 服务器的 www 目录，这里存放了很多 Html 网页，都是静态网页，但是我们都知道，一个网站数据肯定是要和用户交互的，例如电商网站的搜索功能，用户输入搜索关键词后，网站最终会返回搜索相关的商品。所以最终页面是 静态网页+动态内容。这个动态内容就是根据用户输入关键字返回的商品信息，这一部分就是 应用服务器处理。应用服务器侧重处理后端语言逻辑，也就是动态语言的处理，最主要的就是网站的业务逻辑，和数据库打交道的就是这部分。例如各种 增删改查操作。



4.3 Web 服务器

web 服务器：大家访问 ecshop http://localhost:80/ecshop，这是一个 http 请求，web 服务器也可以叫做 http server，就是专门处理 Http 请求的，lamp 环境搭建大家安装 httpd 后(也就是 apach)，可以看到 /var/www/html 这个目录，这个目录里就是网站代码，如果通过 http://localhost:80/ecshop 这种方式来访问，http 请求就会进入到对应 html 目录下面的 ecshop 网站，对应返回的 Html 页面就是 http 的响应，用户最后看到的就是返回的 Html 网站页面。这个过程中，web 服务器也就是 apach 充当了一个容器的角色，里面放了我们的网站代码，然后让用户可以通过 http 请求找到网站并且返回对应请求给用户。整个过程都是 web 服务器在管理控制。

4.4 应用服务器

应用服务器：上面 web 服务器大家可以看到是侧重于处理网站的静态内容，比如 Html 页

面。但是网站有很多是动态内容，例如 ecshop 里面首页显示的商品，这些肯定是后台配置的，也就是配置了才会出现，这个一般做法就是后端代码提供一个接口给前端页面，这个接口给了商品图片，价格，标题等基础信息，前端代码拿到信息后负责展示。我们经常说的业务逻辑就是后端代码，对于后端代码的管理就是 web 应用服务器。tomcat 是一个经常使用的应用服务器。在我们的 lamp 环境中，其实也安装了 应用服务器，叫做 CGIweb。专门处理后端代码，也就是 PHP 代码。网站的业务逻辑都是集中在后端代码部分，处理完成后返回值通过接口调用返回给前端页面，前端页面拿来显示就行了。所以应用服务器侧重后端代码的管理控制

4.5 架构的理解

系统架构就是系统组成的各个元素和组成方式。就像房屋架构有楼板，墙面，屋顶三种基础结构，如果细分还有承重墙，普通隔墙，吊顶，保温层等等。

架构这个概念首先建立在产品形态的基础上，不同产品架构肯定不同。我们这里就以最通用的 web 网站的架构来讲解。

单体结构：我们第二阶段的实战项目 CRM 就是这种结构，前端+后端（web 容器+应用服务器+数据库），这是非常通用的一种架构模式，很多用户规模较小的网站或者初创公司的业务产品就是这种架构。这种架构业务逻辑集中在后端的应用服务器，耦合度较高。随着用户增长这种方式的弊端越来越突出，因为迭代频繁修改导致问题很多，这种单一模式不能再继续支撑业务的发展。

微服务：因为单体结构的问题，催生了微服务的系统架构。大家需要注意，架构并不纯粹是一个技术问题，技术是服务于业务，国内绝大多数互联网公司都是业务驱动。所以技术说到底是为业务服务的。单体系统在初期因为用户少，开发快，成本低，所以契合了业务快速发展的需求，但是到了某一个用户量或者时间点，单体系统的缺陷就暴露出来了，现在很多公司都在推微服务架构。微服务从技术上来讲，就是把各个业务模块切割开来形成单个进程，独立开发部署上线。例如电商网站的 用户系统，商品系统，订单系统等，每个系统单独开发，部署，上线。每个系统有自己的数据库，这样降低了系统之间的耦合度，很大程度上避免了因为修改某一个小点引发很多 BUG 的问题，每个系统都独立成了不同的进程，所以系统之间通过 协议接口方式交互。耦合度大大降低了。

web 和 app 是否调用一个数据库？

这个问题我们来反推一下，大家在淘宝购买一个商品，不管是从 web 端购买，还是从 app 购买，但是在两端肯定都能查到这个商品。所以从这一点可以证明，都是调用一个数据库。web 和 app 其实就是前端展示不同，后端的业务逻辑没有区别。所以数据库层面交互的都是后端逻辑，自然是同一个数据库。但是 web 和 app 两个端的页面展示不同，导致了数据格式上有差异，所以后端可能会提供 2 个不同接口，一个服务 web 端，一个服务 app 端。数据源一样，但是数据格式不同。

4.6 Nginx 是什么

nginx 是一个高性能的 Http 服务器，作为一个 web 服务器，它和 apache 一样有很多功能。这里我们主要介绍 ngix 在高性能方面的实际应用。

ngix 在大型互联网系统里应用非常广泛，我们都知道，类似淘宝这种大型互联网公司，用户量非常大，后端服务器很多，所以用户请求必须准确合理分发到后端服务器。ngix 作为一个 Http 服务器，用户请求进来首先经过 nginx，nginx 根据现在服务器的负载情况，合理分配请求到具体某一台服务器，这里起到了一个负载均衡的作用。本身 nginx 是 C 语言写成，性能非常好。（负载均衡简单来说就是，不能让 忙的忙死，闲的闲死，合理分配）

4.7 集群和分布式

集群和分布式更多是一个概念上的理解，实际技术上的应用非常多。

这里以云服务为例来解释，近年来云服务非常红火，阿里云，腾讯云，百度云。各大互联网公司都在推自己的云服务。但是什么是云服务，为什么会有云，很多人可能是一头雾水。我们都知道国内云服务最早是阿里云，这个和阿里的实际业务发展是分不开的。阿里每年的双十一用户量非常多，为了应对这种用户规模。阿里准备了足够数量的服务器，单位以万 计，但是一年也就一次双十一，很多时候这些服务器资源是闲置的。所以这就催生了云服务的发展。同样，腾讯用户量也非常大，腾讯在贵州山区建设一个数据中心，上万台服务器放置在山区的山洞里，因为温度较低，缓解了服务器发热的问题。所以云服务的基础是因为大厂的闲置计算资源需要再利用。那物理上这些数量巨大的单台服务器，就在逻辑上形成了一个巨大的集群或者说分布式计算资源。这些云服务可以服务很多中小公司的业务，现在很多小公司都不自己组建运维团队，不买服务器，都使用云服务。如果你进入的公司没有自己的运维，不要惊奇。

总结：不管是什么集群，什么分布式。也就是物理上单独的个体在逻辑上形成一个整体对外提供服务。

4.8 主从服务器

这里以数据库服务器为例，大家试想一个场景：如果 A 用户修改一个数据，B 用户刚好访问这个数据，修改之前和修改之后访问，数据就不同，这里就存在一个同步的问题。

实际应用中，一般是通过主从数据库来解决，一主多从。主库写数据，从库读数据。也就是所有“写”操作，增删改都在主库执行，所有“读”操作，都在从库执行。主库修改的数据 定时同步到从库。这样就解决了并发场景下的数据同步问题。

4.9 缓存的应用

在互联网应用里，缓存是一个非常重要的设计，有大牛说过一句话：缓存一切可以缓存的服务，所以从架构设计来看，缓存设计也是非常重要的一环。

这里我说一下电商公司的缓存设计，缓存因为是内存，所以速度非常快，对于追求用户体验的互联网业务来说，服务响应速度的要求肯定是越快越好。实际业务中，电商网站用户大多数都在看，只有少数人会下单购买，所以这是一个 “读多写少”的业务场景，这样就提出了一个问题。大量的用户只是读，是否不走数据库，因为数据库取数据库是比较慢的，数据库读取是硬盘读取。硬盘读取 IO 有限。这里我们可以用缓存来对商品数据进行存储，现在比较流行的 redis 就是缓存服务器。所以缓存可以对于频繁访问但是又很少变化的数据进行处理。这样大大减少对于数据库的压力。用户如果只是浏览，这些商品信息请求就到缓存取数据。