第 1 章　背景知识简介

本书的重点内容是讲解 Java Web 开发的知识，在本章中，首先简单介绍了 Java 语言的历史和现状，然后对网站运行的基本知识进行了简单的介绍，在本章的最后，对比了各种动态开发语言之间，介绍了各种动态 Web 开发语言的优劣，通过本章知识的学习，读者可以掌握 Java Web 开发所需的基本知识。

1.1　Java 语言简介

Java 是一种跨平台的面向对象语言，是由 Sun 公司于 1995 年推出，自从 Java 语言问世以来，受到越来越多开发者的喜爱， 在 Java 语言出现以前，很难想象在 Window 环境下编写的程序可以不加修改就在 Linux 系统中运行，因为计算机硬件只识别机器指令，而不同操作系统中的机器指令是有所不同的，所以，要把一种平台下的程序迁移到另一个平台，必须要针对目标平台进行修改，如果想要程序运行在不同的操作系统，就要求程序设计语言能够跨平台，可以跨越不同的硬件、软件环境，而 Java 语言就能够满足这种要求。

Java 语言的目标就是为了满足在复杂的网络环境中开发软件，在这种复杂的网络环境中，充满这各种各样的硬件平台和不同的软件环境，而 Java 语言就是针对这种复杂的平台环境设计，使用 Java 语言，可以开发出适应这种复杂网络环境的应用系统。

1.1.1　Java 语言简介

Java 是一种优秀的面向对象语言，在 Java 语言中，有着健壮的安全设计，它的结构是中立的，可以一直到不同的系统平台，优秀的多线程设计也是 Java 语言的一大特色，但是 Java 语言的最大优势还是在于其对多种操作系统平台的支持，这种特性是其他编程语言所无法比拟的。

Java 最初是 Sun 公司为消费电子产品设计的一种通用环境，最初的目的是为了开发一种与平台无关的编程语言，这种技术在开始的时候并没有太大用武之地，在网络出现以后，由于网络软硬件环境的复杂性，常见的编程语言逐渐不能适应这种环境的要求，而 Java 语言平台无关性的特性正好适应网络这个潮流。所以 Java 语言在网络出现以后得到突飞猛进的发展。

目前，Java 语言最大的用途就是 Web 应用的开发，使用 Java 语言可以不用考虑系统平台的差异，在一种系统下开发的应用系统，可以不加任何修改就能运行在另一种不同的系统中，例如在目前的 Web 应用开发中，很多用户会选择使用 Linux 或者是 Unix 作为服务器环境，而开发人员一般情况下会选择在 Windows 平台下开发，因为在 Windows 平台中的开发环境的效率要相对较高点，在这种情况下，就可以使用 Java 语言，由于 Java 语言是跨平台的，所以在 Windows 中开发出的系统，可以直接部署在 Linux 或者是 Unix 的服务器系统中，这就是使用 Java 语言的便利之处。

1.1.2　Java 语言的特性和优势

在目前的软件开发中，尤其是应用系统的开发中，Java 语言成为大部分开发人员的选择，经常会有

用户自己提出要使用 Java 语言进行开发，可见 Java 语言的发展已经是深入人心，Java 语言之所以如此受欢迎，是由其自身的优点决定的，以下简单介绍 Java 语言的特性：

（1） 平台无关性

平台无关性是 Java 语言最大的优势，在 Java 中，并不是直接把源文件编译成硬件可以识别的机器指令，Java 的编译器把 Java 源代码编译为字节码文件，这种字节码文件就是编译 Java 源程序时得到的 class 类文件，Java 语言的跨平台主要是指字节码文件可以在任何软硬件平台上运行，而执行这种类文件的就是 Java 虚拟机，Java 虚拟机是软件模拟出的计算机，可以执行编译 Java 源文件得到的中间码文件，而各种平台的差异就是由 Java 虚拟机来处理的，由 Java 虚拟机把中间码文件解释成目标平台可以识别的机器指令，从而实现了在各种平台中运行 Java 程序的目的，在 Java 语言中针对不同的平台环境提供了不同的 Java 虚拟机，例如在 Sun 的官方网站中就提供了 Windows、Linux 和 Solaris 等各种版本 Java 虚拟机的下载。

（2） 安全性

在 C/C++中，指针的使用是一个高级话题，如果熟练掌握指针可以给程序的开发带来很大的方便，但是对于如果指针使用不当，就有可能带来系统资源泄漏，更严重的是错误的指针操作有可能非法访问的系统文件的地址空间，从而给系统带来灾难性的破坏，所以在 C/C++中，在使用指针的时候，需要非常的小心。

Java 语言放弃了指针操作，在 Java 中，没有显式提供指针的操作，不提供对存储器空间直接访问的方法，所有的存取过程都有 Java 语言自身来处理，这样就可以保证系统的地址空间不会被有意或者无意的破坏。而且经过这样的处理，也可以避免系统资源的泄漏，例如在 C\C++中，如果指针不及时释放，就会占用系统内存空间，大量的指针不及时释放就有可能耗尽可用的内存空间。在 Java 中就不用担心这样的问题，Java 提供了一套有效的资源回收策略，会自动回收不再使用的系统资源。从而保证了系统的安全性和稳定性。

另外，Java 虚拟机在运行字节码文件的时候，会把 Java 程序的代码和数据限制在具体的内存空间内，不允许 Java 程序范围指定内存地址之外的空间，这样就可以保证 Java 程序不会破坏系统的内存空间，从而保证系统的安全性。

（3） 面向对象

面向对象是现在软件开发中的主流技术，在 Java 中同样吸取了各种面向对象语言的优点，从而更加彻底的实现了面向对象的技术，在 Java 程序中，基本所有的操作的都是在对象的基础上实现的，为了实现模块化和信息的隐藏，Java 语言采用了功能代码封装的处理，Java 语言对继承性的实现使功能代码可以重复利用，用户可以把具体的功能代码封装成自定义的类，从而实现对代码的重用。

C++是一种经典的面向对象的语言，Java 语言继承了 C++中面向对象的理论，但是在 Java 中简化了这种面向对象的技术，去掉了一些复杂的技术，例如多继承、运算符的重载等功能。经过这样的处理，Java 中的面向对象技术变得简单容易掌握，同时保留这面向对象中核心的技术，可以是用户方便的享受面向对象技术带来的便利。

（4） 异常处理

在 Java 中，提供异常处理的策略，在 Java 程序的开发中，可以对各种异常和错误进行处理。这些错误包括程序在编译和运行阶段的错误和异常，例如空指针异常、数组越界异常、类型错误等。Java 中的异常处理可以帮助用户定位处理各种错误，从而大大提高了 Java 应用程序的开发周期。而且，这种异常策略，可以扑捉到程序中的所有异常，针对不同的异常用户可以采取具体的处理方法，从而保证了应用程序在用户的控制中运行，从而保证了程序的稳定和健壮。

1.1.3 Java 语言的发展现状

Java 语言并不是为网络环境设计的，用户可以使用 Java 语言来编写独立的桌面应用程序，在桌面应用程序这个领域，Java 已经被各大厂商接受，例如 Oracle 数据库、Borland 的 JBuilder 开发环境，Eclipse 开发环境等工具都是使用 Java 语言编写的，这些软件产品的性能都是非常优秀的，可见使用 Java 同样可以编写出功能强大的应用软件。而且，如果用户需要开发跨平台运行的软件的时候，Java 就成了唯一的选择，跨平台的需要也是各大厂商选择使用 Java 开发桌面应用程序的原因之一。

虽然说 Java 语言并不是为网络环境设计的，但是 Java 语言目前还是主要被用于网络环境中，尤其是在服务器段的程序设计中，Java 语言的地位是其他动态语言所无法替代的。尤其是在 B/S 开发结构盛行的今天，Java 语言的地位更是举足轻重，例如，目前，各种信息管理系统都采用 B/S 进行开发，在 J2EE 中，提供了优秀的 B/S 应用程序的解决方案。再加上 Java 语言跨平台、简单易用等特性，用户自然会选择 Java 语言进行开发。事实上，在服务器端的程序开发中，Java 所占的比例份额是占这绝对优势的。

1.1.4 Java 语言的发展前景

随着网络技术的急速发展，Java 语言必然会取得更大的发展，在这个复杂的网络环境中，Java 语言有着广阔的前景。例如在如下几种开发需求中，Java 语言都有着很大的发展前景：

（1）跨平台的应用软件开发

随着 Linux、Unix 等操作系统逐渐被用户接受，Windows 的地位正面临着巨大的挑战，同时各大软件厂商也必须应对这样的变化，在这种情况下，需要兼顾各种操作系统用户的需要，当然可以选择正对不同操作系统开发出不同的软件版本，但是如果软件产品的规模超大的时候，这样的做法就不太合适，这时候就需要用到 Java，虽然桌面应用软件的开发不是 Java 的强项，但是 Java 语言跨平台的特性弥补了在这方面的不足，软件厂商采用 Java 语言进行开发，只需要开发一个版本就可以运行在不同的操作系统环境中，这就大大降低了重复开发的成本和时间。

所以，Java 语言在跨平台应用软件开发领域的前景还是非常广阔的。

（2）企业信息化解决方案

企业信息化是目前的一大潮流，而且现在的信息化解决方案中，基本上采用的都是 B/S 架构，这样的架构方便应用程序的部署，而且节省了界面程序开发的成本，在客户端需要一个浏览器即可，所有的功能代码都在服务器实现。

J2EE 是 Java 的企业版本，是 Sun 公司针对企业信息化提出的一套技术解决方案，使用这些技术，可以非常方便的实现企业信息化的需求，而且在近几年中，J2EE 正以飞快的速度向前发展，相信在未来几年中，Java 在企业信息化建设中会占到更大的比重。

（3）嵌入设备

J2ME 是 Java 针对嵌入设备，例如手机等设备设计的，在 J2ME 出现之前，在嵌入设置中编程只能选择使用 C/C++，这样底层的编程是相当复杂的，当 J2ME 技术问世以后，就可以使 Java 语言十分方便的开发嵌入设备中的应用软件，目前，J2ME 在手机中使用的比较多，各大手机厂商推出的手机产品中基本都会内置支持 Java 的功能。所以，J2ME 的使用也会越来越多。

Java 语言不断的在发展和完善，现在各大厂商都在努力推动 Java 技术的发展，在这些厂商中间，Sun、IBM、Sybase 等做的都相当出色，而且在 Java 领域中，还有开源力量的支持，例如 Apache、JBoss 等，这些开发力量给 Java 的发展带来巨大的推动作用，很多优秀的 Java 框架都是有这些开源力量开发维护的。

在各方面力量的努力下，Java 语言会越来越趋于完美，使用 Java 开发会给用户带来更大的乐趣和更高的效率。

1.2 Web 应用程序开发基本知识

Java Web 开发也就是基于 B/S 结构的 Java 应用程序开发，在接下来的章节中，将介绍 Java Web 开发最基本的知识，在这里不涉及具体的技术实现，只对 Java Web 开发的基本原理进行介绍。

1.2.1 Web 应用程序的运行原理

在传统的 Web 应用程序开发中，需要同时开发客户端和服务器端的程序，由服务器端的程序提供基本的服务，客户端是提供给用户的访问接口，用户可以通过客户端的软件访问服务器提供的服务，这种 Web 应用程序的开发模式就是传统的 C/S 开发模式，在这种模式中，由服务器端和客户端的共同配合来完成复杂的业务逻辑。例如以前的网络软件中，一般都会采用这种模式，而且现在的网络游戏中，一般还会采用这种 Web 开发模式，在这些 Web 应用程序中，都是需要用户安装客户端才可以使用的。

在目前的 Web 应用程序开发中，一般情况下会采用另一种开发模式，在这种开发模式中，不在单独开发客户端软件，客户端只需要一个浏览器即可，这个浏览器在每个操作系统中都是自带的，软件开发人员只需专注开发服务器端的功能，用户通过浏览器就可以访问服务器端提供的服务，这种开发模式就是当前流行的 B/S 架构，在这种架构中，只需要开发服务器端的程序功能，而无需考虑客户端软件的开发，客户通过一个浏览器就可以访问应用系统提供的功能。这种架构是目前 Web 应用程序的主要开发模式，例如各大门户网站、各种 Web 信息管理系统等，使用 B/S 的架构加快 Web 应用程序开发的速度，提高了开发效率。

1.2.2 Web 服务器汇总

在 C/S 架构的开发模式中，服务器端完全是有开发人员自己提供，开发人员自己制定客户端的访问规则，这时候的服务器就是不仅要提供逻辑功能的服务，还要提供一点的协议支持，通过这样的协议，客户端程序才可以与服务器端进行通信，从而享受服务器端提供的服务。

在 B/S 架构的开发模式中，客户端就是简单的浏览器程序，可以通过 HTTP 协议访问服务器端的应用，在服务器端，与通信相关的处理都是由服务器软件负责，这些服务器软件都是有第三方的软件厂商提供，开发人员只需要把功能代码部署在 Web 服务器中，客户端就可以通过浏览器访问到这些功能代码，从而实现向客户提供的服务，下面简单介绍 B/S 结构中常用的服务器。

- IIS 是微软提供的一种 Web 服务器，提供对 ASP 语言的良好支持，通过插件的安装，也可以提供对 PHP 语言的支持。
- Apache 服务器是由 Apache 基金组织提供的一种 Web 服务器，其特长是处理静态页面，对于静态页面的处理效率非常高。
- Tomcat 也是 Apache 基金组织提供的一种 Web 服务器，提供对 JSP 和 Servlet 的支持，通过插件的安装，同样可以提供对 PHP 语言的支持，但是 Tomcat 只是一个轻量级的 Java Web 容器，像 EJB 这样的服务在 Tomcat 中是不能运行的。
- JBoss 是一个开源的重量级的 Java Web 服务器，在 JBoss 中，提供对 J2EE 各种规范的良好支持，

而且 JBoss 通过了 Sun 公司的 J2EE 认证，是 Sun 公司认可的 J2EE 容器。

- 另外 J2EE 的服务器还有 BEA 的 Weblogic 和 IBM 的 WebSphere 等，这些产品的性能都是非常优秀的，可以提供对 J2EE 的良好支持。用户可以根据自己的需要选择合适的服务器产品。

1.2.3 开发一个 Web 应用程序的简单流程

在传统 Web 应用程序的开发过程中，开发一个应用系统一般情况下需要以下几个步骤：客户端/服务器端软件的开发、服务器端程序的部署、客户端软件的安装，只有完成这几个步骤，用户才可以通过客户端访问服务器提供的服务。

而在基于 B/S 架构的 Web 程序大开发过程中，只需要开发服务器端的功能代码，然后把服务器端的程序部署在 Web 服务器软件中即可，在部署结束之后，启动 Web 服务器，用户就可以通过浏览器访问 Web 应用程序提供的服务。

1.3 Web 应用程序开发

由于技术的进步和网络环境的进化，Web 应用程序开发的技术也在不断的进步，在 Web 应用程序开发的过程中，存在着不少争议，当然，这些争议都是开发人员对各种技术的看法不同造成的，在接下来的内容中，简单介绍这方面的内容，是读者对技术进化过程中的一些问题有所了解。

1.3.1 C/S 与 B/S 之争

在前面的章节中已经介绍过，在 Web 应用程序的开发中，存在这两种开发模式，一种是传统的 C/S 架构，另一种是近些兴起的 B/S 架构。

由于硬件成本的降低，再加上应用系统复杂程度的提高，Web 应用程序的开发逐渐转向到 C/S 架构，所谓的 C/S 架构就是客户端/服务器端的架构形式，在这种架构方式中，多个客户客户端围绕这一个或者多个服务器，这些客户端是安装在客户机上，负责用户端业务逻辑的处理，在服务器端仅仅对重要的过程和数据库进行处理和存储，每个服务器端都分担这服务器的压力，这些客户端可以根据不同的用户的需求进行定制。C/S 这种架构方式的出现大大提高了 Web 应用程序的效率，给软件开发带来革命性的飞跃。

但是，随着时间的推移，C/S 架构的弊端开始慢慢显现，在 C/S 架构中，系统部署的时候需要在每个用户的机器上安装客户端，这样的处理方式带来很大的工作量，而且在 C/S 架构中，软件的升级也是很麻烦的一件事情，哪怕是再小的一点改动，都得把所有的客户端全部修改更新，这些致命的弱点决定了 C/S 结构的命运。在 C/S 架构模式流行一段时间以后，逐渐被另一种 Web 应用系统的架构方式所代替。这种新的 Web 软件架构的模式就是 B/S。

B/S 架构就是浏览器/服务器的架构形式，在这种架构方式中，采取了基于浏览器的策略，简化了客户端的开发工作，在 B/S 架构的客户机中，不用安装客户端软件，只要有通用的浏览器工具，就可以访问服务器端提供的服务。在各种操作系统中，都提供了浏览器中工具，这些浏览器工具都是遵循这相同的协议规范，所以 B/S 的结构的客户端在各种系统环境中都已经实现。而且，在浏览器访问服务器的过程中，使用的 HTTP 协议，所以这种方式非常容易就可以穿过防火墙的限制。

而且在 B/S 结构的服务器端，也不用处理通信相关的问题，这些问题都由 Web 服务器提供，Web

服务器处理用户的 HTTP 请求，开发人员只需要专注开发业务逻辑功能即可，总之，Web 服务器完成了底层的操作，给应用软件的开发提供了最基础的通信服务，从而减轻了开发人员重复开发通信相关的功能，从而提高了开发的效率，降低了 B/S 结构应用程序开发的难度。

使用 B/S 架构，不仅开发减轻了开发的任务，而且软件的部署和升级维护也变得非常简单，只需要把开发的 Web 应用程序部署在 Web 服务器中即可，而客户端根部不需要做任何改动，这是在 C/S 架构中无法实现的。

但是在 B/S 架构中也有自身存在的一些缺点，例如界面元素单调，在 B/S 结构的程序中，失去了桌面应用程序丰富的用户界面，程序在交互性上没有 C/S 架构的人性化。

在 C/S 和 B/S 两种架构之间，并没有严格的界限，两种架构之间没有好坏之分，使用这两种架构都可以实现系统的功能。开发人员可以根据实际的需要进行选择，例如需要丰富的用户体验，那就选择 C/S 架构，例如在目前的网络游戏中，基本都是选择 C/S 架构；如果更偏重的是功能服务方面的实现，就需要选择 B/S 架构，这也正是目前绝大部分管理应用系统采用的软件架构方法。

1.3.2 动态页面语言对比

在互联网发展的最初阶段，所有的网页内容都是静态的 HTML 网页，在这种情况下，网站所能实现的任务仅仅是静态的信息展示，而不能与客户产生互动，当然这样的网站是不能满足用户不同的需要。在现实的生活中，用户的需要总是各种各样的，这就需要网站或者是 Web 应用程序具有收集并处理响应用户需要的功能，而静态的 HTML 是不能满足这种需要的，为了满足这种特殊的需要，就有了后来一系列的动态页面语言的出现。

所谓的动态页面是指可以和用户产生交互，能根据用户的输入信息产生对应的响应，能满足这种需求的语言就可以称之为动态语言。

在最早的时候，动态网页技术主要使用 CGI，现在常用的动态网页技术有 ASP、JSP、PHP 等，下面分别介绍这几种动态语言：

（1）CGI

在互联网发展的早期，动态网页技术主要使用 CGI（共用网关接口），CGI 程序被用来解释处理表单中的输入信息，并在服务器中产生对应的操作处理，或者是把处理结果返回给客户端的浏览器，从而可以给静态的 HTML 网页添加上动态的功能。但是由于 CGI 程序的编程比较困难、效率低下，而且修改维护也比较复杂，所以在一段时间以后，CGI 逐渐被其他新的动态网页技术所替代。

（2）ASP

ASP 是微软公司推出的一种动态网页语言，它可以将用户的 HTTP 请求传入到 ASP 的解释器中，这个解释器对这些 ASP 脚本进行分析和执行，然后从服务器中返回处理的结果，从而实现了与用户交互的功能，ASP 的语法比较简单，对编程基础没有很高的要求，所以很容易上手，而且微软提供的开发环境的功能十分强大，这更是降低了 ASP 程序开发的难度。但是 ASP 也有其自身的缺点。ASP 在本质上还是一种脚本语言，除了使用大量的组件，没有其他办法提高效率，而且 ASP 还只能运行在 Windows 环境中，这样 Windows 自身的一些限制就制约了 ASP 的发挥，这些都是使用 ASP 无法回避的弊端。

（3）JSP

JSP（Java Server Page）是 SUN 公司开发的一种服务器端的脚本语言，自从 1999 年推出以来，逐步发展为开发 Web 应用一项重要技术。JSP 可以嵌套在 HTML 中，而且支持多个操作系统平台，一个用 JSP 开发的 Web 应用系统，不用做什么改动就可以在不同的操作系统中运行。

JSP 本质上就是把 Java 代码嵌套到 HTML 中，然后经过 JSP 容器的编译执行，可以根据这些动态

代码的运行结果生成对应的 HTML 代码，从而可以在客户端的浏览器中正常显示。

由于 JSP 中使用的是 Java 的语法，所以 Java 语言的所有优势都可以在 JSP 中体现出来，尤其是 J2EE 中的强大功能，更是成为 JSP 语言发展的强大后盾。

（4）PHP

PHP 是和 JSP 类似，都是可以嵌套到 HTML 中的语言，不同之处在于，PHP 的语法比较独特，在中混合了 C、Java 等多种语法中的优秀部分，而且 PHP 网页的执行速度要被 CGI 和 ASP 等语言要快很多。在 PHP 中，提供了对常见数据库的支持，例如 SQL Server2000、MySQL、Oracle、Sybase 等，这种内置的方法使 PHP 中的数据库操作变得异常简单。而且 PHP 程序可以在 IIS 和 Apache 中运行，提供对多种操作系统平台的支持。

但是 PHP 也存在一些劣势，PHP 的开发运行环境的配置比较复杂，而且 PHP 是开源的产品，缺乏正规的商业支持。这些因素在一定程度上限制了 PHP 的进一步发展。

总之，各种动态语言都有着自身的优势和劣势，只有根据客户的需求来选择具体的语言。只要能够保证系统的性能和功能，选择什么语言是无关紧要的。

1.3.3 .NET 与 J2EE 之争

自从.NET 和 J2EE 推出以来，对 J2EE 和.NET 的比较已经不是一天两天的事了，钟情于 Windows 的用户会选择.NET，而选择 Unix\Linux 的用户会更钟情于 J2EE，其实这两种技术都有各自的优势和不足，下面简单分析下这两种技术自身的优劣。

1. .NET 的优点

在 Windows 平台的应用程序中，对用户界面的要求比较高，所以.NET 提供了便捷的开发环境和工具，在 Visual Studio 中，用户的界面都可以通过简单的拖拽来完成，这可视化的编程方式在 Java 中还不是很成熟，.NET 的可视化编程环境是得到一些程序员支持的原因之一。

.NET 运行在 Windows 操作平台中，而且和 Windows 一样，都是微软开发的产品，所以，在.NET 中可以访问到操作系统中的各个细节，因此可以调用系统中的各种功能，对于 J2EE 的程序来说，这样的操作就很难实现了。在 Java 中无法访问到操作系统底层的细节的。

.NET 的优点还有很多，在这里不再一一列举，现在介绍使用.NET 的局限性。

2. .NET 的局限

.NET 只能运行在 Windows 平台中，不能跨平台，这是.NET 最大点一个劣势。其次.NET 是微软一个公司的产品，所有的开发设计仅仅局限在一个公司之内，而 Java 则虽然是由 Sun 公司开发，但是在发展的过程中得到了类似 IBM、BEA 这样知名公司的支持，而且还有很多开源力量的支持。这些都是.NET 中不可能拥有的。

3. J2EE 可以弥补.NET 的局限

而在 J2EE 中，可供使用的类库是非常广泛的，这些类库都是非常成熟的，在 Java 发展的十多年中，这些类库的功能经过了大量的检验和测试，已经十分成熟。而且在 Java 语言的跨平台的特性在这十几年的发展中也经受住了考验，在 J2EE 领域中，有很多的开源的资源可供使用，例如 Tomcat、JBoss 这样的 Web 服务器，还有 Spring、Hibernate、Struts 这样的开源框架，这些资源都是 Java 社区中开源力量的贡献。这些资源同样是在.NET 中无法享受的。

.NET 和 J2EE 都是企业级应用系统的解决方案，这两种解决方案都可以很好的实现应用系统的功能，这两种解决方案之间并没有非常明显的优劣区别，关于.NET 和 J2EE 谁好谁坏类似问题的讨论是没有意

义的，在实际的开发过程中，应该根据具体的需要来选择使用哪种技术，例如用户只要求在 Windows 系统中使用，并没有要求跨平台，那选择.NET 和 J2EE 都是可以的，如果用户要求一定在 Linux 平台中部署应用系统，那 J2EE 就是一种很好的选择，所有选择.NET 还是 J2EE 是由需求而定的，两种技术没有优劣之分。

1.4 小结

在本章内容中，对 Java Web 开发中的一些基本知识进行简单的介绍，读者通过本章的学习可以了解开发 Java Web 应用程序的一些基本的概念，而且对于 Java Web 开发中的一些存在争议的问题也有所了解，尤其是一些有争议的问题，读者可以稍加注意，在初学者中，很容易犯这些错误，例如会过多关注具体技术的优劣，期望学到一种最有用的技术，这些想法都是不可取的。技术没有高低分，只有应用场合的不同。所以不要花费太多的精力来考虑这种没有意义的问题。