Sun Certified Web Component Developer
SCWCD 考试号310-081

SUN 国际认证

SCWCD 应试指南

北京希望电子出版社　总策划
施　铮　编　著

- 官方国际认证应试必备
- 彻底剖析 SCWCD（考试号310–081）11 项考点
- 各章节 100% 完全对付 SCWCD 考纲主题

Sun Certified 100%

科学出版社
www.sciencep.com

希望
www.bhp.com.cn

Sun Certified Web Component Developer
SCWCD 考试号310-081

SUN 国际认证

SCWCD 应试指南

北京希望电子出版社 总策划
施 铮 编 著

- 官方国际认证应试必备
- 彻底剖析 SCWCD（考试号310–081）11 项考点
- 各章节 100% 完全对付 SCWCD 考纲主题

Sun Certified 100%

科学出版社
www.sciencep.com

内 容 简 介

这是一本 SUN 国际认证 SCWCD（Sun Certified Web Component Developer, Web 组件开发人员认证；考试号 310－081）学习指导书，针对 SCWCD 考试的 11 项知识点，选取典型案例进行详尽探讨，便于有效地提高考生的应试成功率。

全书共分 19 章，分别介绍了 Java Applet 小程序、Servlet 及 JSP 技术、过滤器、会话管理、JavaBean 组件技术、标签库、表达式语言、设计模式和部署描述符等内容。内容详尽，知识讲解与具体实例相结合，实用、实效。

本书既可作为 JSP 知识学习练习，亦可配合 SCWCD 试题使用，同时还可以作为学习 Java 语言和 JSP Web 程序设计指导教材。

图书在版编目（CIP）数据

SUN 国际认证 SCWCD 应试指南 / 施铮编著. —北京：科学出版社，2007

（Java 开发专家）

ISBN 978-7-03-019596-8

Ⅰ. S… Ⅱ. 施… Ⅲ. Java 语言—程序设计—工程技术人员—资格考试—应试指南 Ⅳ. TP·3575

中国版本图书馆 CIP 数据核字（2007）第 123664 号

责任编辑：但明天 / 责任校对：马 君

责任印刷：媛 明 / 封面设计：刘孝琼

科学出版社 出版

北京东黄城根北街 16 号

邮政编码：100717

http://www.sciencep.com

北京媛明刷厂印刷

科学出版社发行 各地新华书店经销

*

2007 年 9 月第 一 版 开本：787×1092 1/16

2007 年 9 月第一次印刷 印张：29

印数：1—3000 字数：674 31

定价：45.00 元

前　言

众所周知，Java 以其独有的开放性、跨平台性和面向网络的交互性席卷全球，并以其安全性、易用性和开发周期短等特点，迅速从最初的编程语言发展成为开发基于互联网应用程序的首选语言。

本书背景

技术是讲究实力的。近年来，认证在就业市场已获得肯定与追捧。根据知名就业网站的调查，与信息技术相关的工作机会中有 62%以提出认证作为录用参考，其中又以需要 Java 相关认证资格者居多。这几年来，Java 大红大紫不仅被微软视为劲敌，而各大企业竞相争取掌握 Java 技术的人才，也使 Java 专业认证成为 IT 人员最想取得的一张证书。特别是随着互联网全面走向应用的今天，全球已经掀起了一股学习 Java 语言开发技术的热潮，而且成为一条非专业人员变成编程高手的快车道。Java 语言自然是转行 IT 的入门首选。

SUN 公司的 SCWCD（Sun Certified Web Component Developer，Web 组件开发人员认证）是业界最广泛认可的 IT 技术认证之一，也是业界最权威、最受尊敬的认证之一。SCWCD 认证由 Sun 公司出题，考试号为 310-081。凡考试合格者，将获得由美国 Sun 公司签发的英文 SCWCD 证书。

拿高薪，是每个人的梦想，但究竟能拿多少钱，得由你的职场身价决定。获得 SUN 公司的 SCWCD 认证不仅仅能证明你的 IT 技术能力，更是你进入职场的敲门砖，也是提高你身价的一个有效捷径。

SCWCD 认证是 Sun 公司 Java 技术在 Web 开发领域应用的基础性认证，通过此项认证即能清楚表明此开发人员掌握了 Servlet 和 JSP 技术的语法和结构，并能使用 Servlet 和 JSP 应用开发接口，创建基于 Web 的应用程序。

本书特色

由于 SCWCD 认证考试内容涉及所有 Servlet 和 JSP 技术相关知识细节、编程概念及开发技巧，相关的书籍非常少，尤其像这样全面介绍基于 SUN 公司新推出对应 J2EE1.4 的 SCWCD 认证考试配套的中文图书，目前市场上几乎没有，这与国内对 Web 开发需求的飞速发展趋势相悖。

本书正是为了解决 SCWCD 考试人员在准备认证考试时无从获得平台相关技术信息而编写的。本书各章节内容的安排遵循从总体到局部，从易到难的原则，安排合理，循序渐进，结构清晰，示例丰富，浅显易懂。即使是没有 Web 开发经验的新手，通过本书的强化学习，也能较快地掌握 SCWCD 认证的内容。为读者顺利掌握 SCWCD 认证知识作好准备，力求做到为读者捅透最后一层窗户纸。

读者对象

本书是广大 SCWCD 认证考试人员必备的参考书，也是热爱 Java 编程的开发者赶上主流 Web 开发技术、学习 J2EE 技术不可多得的一本好书。本书面向 SCWCD 认证考试者、Java Web

软件开发者。同时，本书也可供高等院校相关专业的学生和相关培训机构的学员参考。

本书目标

讲解 SCWCD 认证考试发涉及的各个技术环节，使读者掌握 Java Web 开发的全面技能。

本书组织

根据 SCWCD 认证考试所涉及技术难易和之间的制约关系，全书共分为 19 章。

第 1 章 Java 服务器小程序　介绍 Servlet 技术的工作原理、Servlet 容器、Servlet 和容器之间的关系，以及 Servlet 应用开发接口。

第 2 章 Java 服务器页面　介绍 JSP 技术的工作原理，通过一个简单的 JSP 示例给读者直观的认识，并且对 Servlet 和 JSP 技术做了对比。接着介绍了 JSP 的两种应用架构模型，最后对 JSP 的语法做了简单介绍，为后面的学习做铺垫。

第 3 章 Web 应用程序基础　介绍了 Web 应用程序的概念、构成 Web 应用程序的各个元素以及用于描述 Web 应用程序的部署描述符等。接着对 HTTP 协议的概念以及请求、响应结构做了详细的讲解。

第 4 章 Servlet 模型　介绍了 Servlet 技术对客户端请求和服务器端响应的处理过程，接着讨论了 Servlet 的生命周期，如何使用 Servlet 上下文管理 Servlet 资源，最后对 Servlet 相关的高级技术做了详细讲解。

第 5 章 Web 应用程序结构和部署　介绍了 Web 应用程序的结构，Web 应用程序的部署描述符的构成。

第 6 章 Servlet 容器模型　介绍了封装环境上下文的接口。如何使用部署描述符来配置环境，以及在分布式环境下的 Servlet 和 Servlet 容器的行为表现。

第 7 章 过滤器　介绍了过滤器的概念。如何创建一个过滤器，如何在部署描述符中配置过滤器，并对有关过滤器的高级内容做了讲解。

第 8 章 会话管理　介绍了会话对象及其状态，讨论了 HttpSession 对象以及与会话相关联的监听器。接下来阐述了会话超时的概念，以及如何通过 Cookie 对象和连接地址来实现会话跟踪。

第 9 章 安全的 Web 应用程序　介绍了应用于 Web 的程序的安全有关技术。

第 10 章 JSP 模型基础　介绍了构成 JSP 页面的各个元素，讨论了 JSP 页面的生命周期，最后还对 JSP 的 page 伪指令的语法做了详细讲解。

第 11 章 JSP 模型进阶　介绍了 JSP 生命期的转换阶段，讨论了 JSP 的各个内置对象的使用及 JSP 作用域，最后讲解了 JSP 文档。

第 12 章 Web 组件复用　介绍了 JSP 复用 Web 组件的两种方式——静态包含和动态包含。

第 13 章 表达式语言　介绍了 JSP2.0 引入的新特性——表达式语言。

第 14 章 使用 JavaBean 组件　介绍了 JavaBean 的基本概念，讨论了 JavaBean 组件的具体使用，最后对 JavaBean 的属性做了讲解。

第 15 章 使用定制标签　介绍了自定义标签的相关概念，讨论了如何在 JSP 页面中导入标签库，如何在 JSP 代码中使用各种定制标签，最后对 SUN 公司的 JSTL 做了详细讲解。

第 16 章 标准标签库　介绍了如何创建标签的实现类。
第 17 章 简单标签库　介绍了如何创建简单标签库。
第 18 章 设计模式　介绍了设计模式的历史以及 J2EE 经典模式。
第 19 章 部署描述符　介绍了部署描述符中的各个元素的使用方法。

考试目标

第 1 节：Servlet 技术模型

讲解各种 HTTP 方法（如 GET, POST, HEAD 等）的目的和 HTTP 协议的技术特点，列出导致客户端（通常为一个 Web 浏览器）使用方法的场合，区分对应 HTTP 方法的 HttpServlet 方法。

使用 HttpServletRequest 接口编写代码，从请求中检索 HTML 表单参数，检索 HTTP 请求头信息，或者从请求中检索 Cookie。

使用 HttpServletResponse 接口编写代码，设置 HTTP 响应头，设置响应的 content type，获取响应的文本流，获得响应的二进制流，将 HTTP 请求重定向到另一个 URL 中，或者给响应添加 Cookie。

讲解 servlet 的作用和生命周期中的事件顺序：（1）servlet 类加载，（2）servlet 安装，（3）调用 init 方法，（4）调用服务方法，（5）调用 destroy 方法。

第 2 节：Web 应用的结构和部署

创建 Web 应用的文件和目录结构，可能包括：静态内容、JSP 页面、servlet 类、部署描述符、标签库、JAR 文件以及 Java 类文件。讲解如何保护资源文件的 HTTP 访问。

讲解部署描述符的目的和语法。

创建正确的部署描述符结构。

讲解 WAR 文件的作用，WAR 文件中的内容，以及如何创建 WAR 文件。

第 3 节：Web 容器模型

ServletContext 初始化参数方面：编写 servlet 代码访问初始化参数，创建部署描述元素声明初始化参数。

基本 servlet 属性范围（请求、会话和上下文）方面：编写 servlet 代码添加、检索和删除属性；给定使用情景，识别正确的属性范围，以及各个范围相关的多线程问题。

讲解 Web 容器请求处理模型；编写并配置过滤器，创建请求或响应封包，给定一个设计问题，讲解如何应用过滤器或封包。

讲解 Web 容器生命周期的事件模型请求、会话、Web 应用，为每个范围里的生命周期创建和配置监听器类，创建和配置范围属性监听器类；给定一个情形，识别适用的属性监听器。

讲解 RequestDispatcher 机制，编写 servlet 代码创建请求派遣器，编写 servlet 代码转到或者包含目标资源，识别并讲解容器或目标资源所提供的其他请求范围中的属性。

第 4 节：会话管理

编写 servlet 代码将对象保存到 session 对象中，并从 session 对象中检索出对象。

给定一个情景，描述访问 session 对象的 API，什么时候创建 session 对象。用于销毁 session 对象的机制，以及何时需要销毁。

使用会话监听器编写代码，当把对象添加到 session 时对事件进行响应。编写代码，当 session 对象从一个虚拟机移动到另一个虚拟机时，对事件进行响应。

给定一个情景，讲解 Web 容器所要实现的会话管理机制，如何使用 cookie 来管理会话，如何使用 URL 重写技术来管理会话，并编写 Servlet 代码执行 URL 重写。

第 5 节：Web 应用安全性

基于 Servlet 规范，比较和对比以下安全性机制：认证、授权、数据完整性以及保密性。

在部署描述符中，声明安全性约束、Web 资源、传输保障、登陆配置和安全性角色。

比较和对比认证类型：BASIC, DIGEST, FORM 以及 CLIENT-CERT。这些类型如何工作，给定一个情景，选择合适的类型。

第 6 节：Java 服务器页面（JSP）技术模型

认识、描述或编写下列 JSP 代码：模板文本、脚本（注释、指令、声明、脚本和表达式）、标准动作和自定义动作，以及语言表达式。

编写 JSP 代码，用到命令：page（属性 import, session, contentType 和 isELIgnored），include 和 taglib。

编写 JSP 文档（基于 XML 的文档），使用正确的语法。

讲解 JSP 页面生命周期的目的和事件顺序：（1）JSP 页面翻译，（2）JSP 页面编译，（3）加载类，（4）创建实例，（5）调用 jspInit 方法，（6）调用_jspService 方法以及调用 jspDestroy 方法。

给定一个设计目标，编写 JSP 代码，用适当的隐含对象：request, response, out, session, config, application, page, pageContext 和 exception。

配置部署描述符，声明一个或多个标签库，停用评估语言和脚本语言。给定特定的设计目标，包含另一个页面中的 JSP 代码段。编写 JSP 代码，使用最合适的包含机制（用 include 指令或 jsp:include 标准动作）。

第 7 节：使用表达式语言（EL）创建 JSP 页面

给定一个情景，编写 EL 代码，访问隐含变量：pageScope, requestScope, sessionScope, applicationScope, param, paramValues, header, headerValues, cookie, initParam 和 pageContext。

给定一个情形，编写 EL 代码，使用运算符：属性访问（.运算符）、集合访问（[]运算符）。

给定一个情形，编写 EL 代码，使用运算符：算术运算符、关系运算符和逻辑运算符。

给定一个情形，编写 EL 代码，使用某个函数；编写 EL 函数，在标签库描述符中配置 EL 函数。

第 8 节：用标准动作创建 JSP 页面

给定一个设计目标，使用下列动作创建代码 snippet：jsp:useBean（属性 id, scope, type 和 class），jsp:getProperty 以及 jsp:setProperty（所有属性的组合）。

给定一个设计目标，使用下列动作创建代码 snippet：jsp:include, jsp:forward 和 jsp:param。

第 9 节：使用标签库创建 JSP 页面

针对自定义标签库或标签文件库，为 JSP 页面创建 taglib 指令。

给定一个设计目标，在 JSP 页面中创建自定义标签结构来支持这个目标。

给定一个设计目标，使用核心标签库中适当的 JSP 标准标签库（JSTL v1.1）。

第 10 节：创建一个自定义标签库

讲解经典的自定义标签事件模型执行时的语法（doStartTag, doAfterBody 及 doEndTag），讲解返回值的类型以及各个事件的意义，编写一个标签处理类。

使用 PageContext API 编写标签处理代码，访问 JSP 隐含变量，并访问 Web 应用属性。

给定一个情景，编写标签处理代码，访问父标签和任意标签祖先。

讲解简单的自定义标签事件模型执行时的语法（doTag），编写标签处理类，解释标签中 JSP 内容的约束。

讲解标签文件模型的语义、标签文件 Web 应用结构、标签体中 JSP 内容的限制；编写标签文件。

第 11 节：J2EE 模式

给定一个问题列表描述情景，选定适当的模式解决这些问题。必须了解的模式：截获过滤、模型－视图－控制器、前端控制、服务定位、业务代理和传输对象。

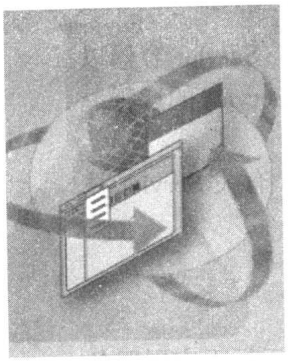
根据给出的优点描述，找出对应的设计模式：截获过滤、模型－视图－控制器、前端控制、服务定位、业务代理和传输对象。

本书由具有丰富的 SCWCD 考试经验的专家编写，是 SCWCD 应试人员的必备教材，同时也是一本 Java Web 开发的优秀参考书，可作为 Java Web 开发人员理论提升的参考。既可作为 Java5 的辅导材料和备考教材，也可用于培训学校教材。

由于作者水平有限，时间紧任务重，难免存在不妥之处，敬请读者批评指正。作者联系电子邮箱 xuanxuan_boys@126.com。

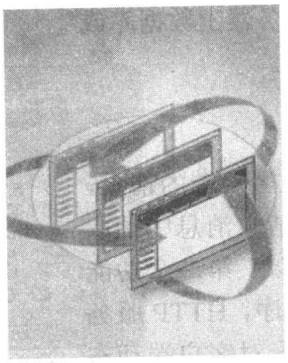
施　铮

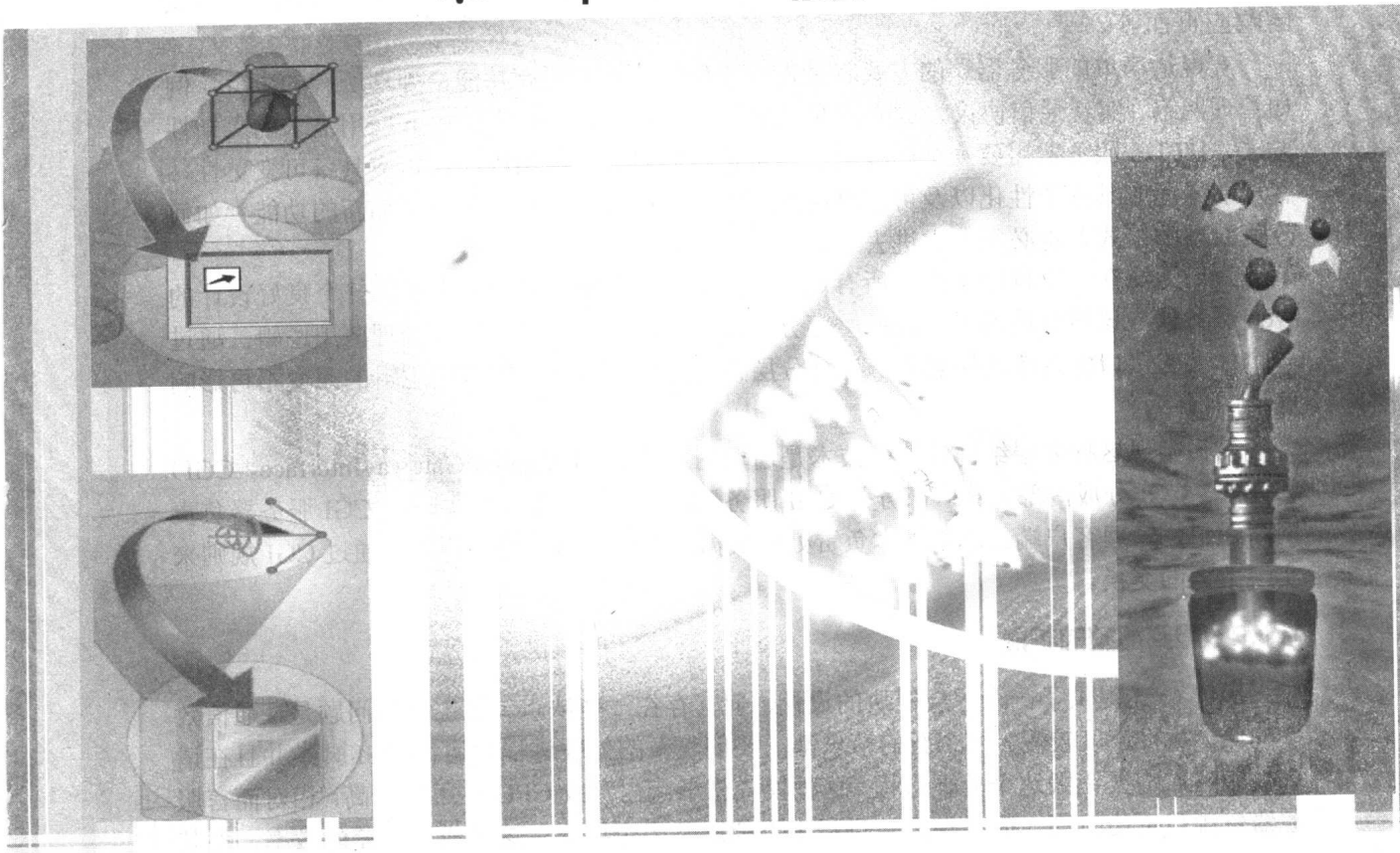
目　录



Chapter 1

Java 服务器小程序





在本章中，首先我们先了解一下 Servlet 技术的工作原理，接着介绍了 Servlet 容器以及 Servlet 和容器之间的关系，之后通过一个简单的示例展示了构建一个 Servlet 的过程，最后介绍了 Servlet 应用开发接口。

本章只作为 Servlet 技术的导入，让读者有一个初步认识和整体概念，有关 Servlet 技术的具体技术细节将在后面章节中详细讲解。

1.1 Servlet 简介

Servlet（Java 服务器小程序）是用 Java 语言编写的服务器端程序，是运行于应用服务器之上，在服务器端调用、执行，并按照 Servlet 规范编写的 Java 类。其是一个中间层，负责连接来自 Web 浏览器或其他 HTTP 客户端程序的请求，以及 HTTP 服务器上数据库或应用程序。为了更好地理解 Servlet，我们来看一下服务器端所扮演的角色。

1.1.1 服务器端的职责

为了给远程客户端提供服务，服务器端主要具备两个功能：一是处理客户端的请求，第二、创建一个返回客户端的响应。第一个功能包括套接字的编程，从请求消息中抽取信息以及实现客户端－服务器间的通信协议，例如 FTP 协议、HTTP 协议。第二个功能包括创建响应、提供不同的服务，例如对基于 HTTP 协议的 Web 应用程序，HTTP 服务器作为 Web 应用程序的入注主机为客户端提供复杂的服务，产生输出。对客户端动态地产生响应，例如从数据库中检索数据，执行商务逻辑组件，为不同用户提供定制格式的响应页面等。

实现最简单的服务器端的方式就是在一个程序中实现所有功能，处理所有的事务，例如管理网络、实现通信协议、定位数据以及形成响应等。但是对于 Web 应用需求的复杂性要求，HTTP 服务器需要一个高度灵活、可扩展的设计。应用程序逻辑应具备可变更性，客户端界面应具备个性化以及可定制商业逻辑处理规则。而且，如何添加新的功能？如何改变数据的格式？也必须考虑进去。

我们无法在一个程序中实现所有这些任务，满足 Web 应用的需求。一个良好设计的 Web 服务器应该被分成两个部分：一部分处理底层的网络，另一部分处理应用逻辑。通过一个标准的接口将两者连接起来，这种分层设计可以使得易变的应用逻辑的改变不再影响底层的网络。

最初实现这种多层结构设计的技术是通用网关接口（Common GatewayInterface，CGI）来完成。CGI 分成两部分：一部分是 CGI 服务器，另一部分是 CGI 脚本。CGI 服务器负责网络通信，管理客户端；CGI 脚本负责处理数据，发送输出。两部分之间通过 CGI 规则来实现数据的交换。

1.1.2 服务器端的扩展

尽管 CGI 提供了一个分层模块的设计，但其存在一些缺点。最主要的问题是，通信效率的低、执行速度较慢。每个新的请求都需要创建新的处理进程来运行 CGI 脚本并且在请求被服务后需要销毁这个处理进程，这种模式将引起内存及 CPU 占用率较高，相对而言服务器资源代价比较高，而且在同一进程中不能服务多个用户，这造成了极差的效率，尤其

是由脚本完成初始化的过程，例如与数据库的连接。此外，CGI 应用开发比较困难，它要求开发人员具备处理参数传递的知识，这不是一种通用的技能。CGI 不可移植，为某一特定平台编写的 CGI 只能应用运行于这一特定环境中。

较理想的服务器设计应该支持可独立执行的模块，这些模块在服务器启动时可以被装载到内存中，并且仅需初始化一次。对每个请求就可以由已在内存中的模块提供服务，并且一个模块可以处理多个请求，这些独立的可执行模块称作服务器端的扩展。基于 Java 平台的服务器端的扩展就是使用 Servlet 应用开发接口（Servlet API）编写而成的，这些扩展模块就称作 Servlet。

和传统的 CGI 及许多类 CGI 技术相比，Java 的 Servlet 执行效率高、易用、强大、易移植、安全、成本低。

1. 效率对比

应用传统的 CGI，针对每个 HTTP 请求都要启动一个新的进程，那么启动进程的开销会占用大部分执行时间。而使用 Servlet，JVM（Java 虚拟机）会一直运行，并用轻量级的 Java 线程处理每个请求，而非重量级的操作系统进程。类似地，应用传统的 CGI 技术，如果存在对同一个 CGI 程序的 N 个请求，那么 CGI 程序的代码会载入内存 N 次。同样的情况，如果使用 Servlet 则会启动 N 个线程，但仅仅载入 Servlet 类的单一副本。这种方式减少了服务器的内存需求，通过实例化更少的对象，从而节省了时间。当 CGI 程序结束对请求的处理之后，程序结束。这种方式难以缓存计算结果，保持数据库连接打开，或是执行依靠持续性数据的其他优化。然而，Servlet 会一直滞留在内存中，即使请求处理完毕也保持在内存中，这样可以直接存储客户端请求之间的任意复杂数据。

2. 便利对比

Servlet 提供了大量的基础方法，可以自动分析、解析、提取 HTML 表单的数据，读取、设置 HTTP 头信息，处理 Cookie、跟踪会话，以及其他此类高级功能。而在 CGI 中，大部分工作都需要开发人员自己来完成。另外，编写 Servlet 代码不需要学习新的编程语言，只要熟悉 Java 语言即可。

3. 功能对比

Servlet 支持传统 CGI 难以实现或根本不能实现的一些功能。Servlet 能够直接与 Web 服务器对话，而传统的 CGI 程序无法做到这一点，至少在不使用服务器专门提供的 API 情况下是这样。例如，与 Web 服务器的通信使得将相对路径转换成具体路径变得更为容易。多个 Servlet 之间还可以共享数据，从而易于实现数据库连接共享和类似地资源共享优化。Servlet 还能维护请求之间的状态、信息，这使得诸如会话跟踪、计算结果缓存等技术更为简单。

4. 移植对比

Servlet 使用 Java 语言编写，并且遵循标准的 Java API。所有主要的 Web 服务器，实际上都直接或间接地通过插件支持 Servlet，因此 Servlet 可以在不同的 Web 服务器间移植，而不需要修改任何 Servlet 代码。

5. 成本对比

对于低容量或中等容量 Web 应用程序的部署，有大量免费的 Web 服务器可供选择。通过使用 Servlet 可以从使用免费的服务器开始，在项目获得初步成功后，再移植到具有更高性能、高级管理工具的商业服务器上，并不需要做出任何更改，有效降低了投入风险。这与传统 CGI 方案形成了鲜明的对比，因为这些 CGI 方案在初期均需要为购买软件包而投入大量资金。由此可以看出，价格与可移植性在某种程度上是相互关联的。

6. 安全对比

传统的 CGI 程序主要漏洞之一，就是 CGI 程序常常由通用的操作系统外壳命令来执行。CGI 程序必须过滤那些可能被外壳命令特殊处理的字符，实现这项预防措施的难度可能超出我们的想象。在广泛应用的 CGI 中，不断发现由这类问题引发的弱点。同时，一些 CGI 程序用非自动检查数组和字符边界的语言编写而成。因而，如果开发人员一旦忘记了执行这项检查，就会将系统暴露在蓄意或偶然的缓冲区溢出攻击之下。Servlet 是不存在这些问题的。即使 Servlet 执行系统调用，激活本地操作系统上的程序，它也不会用到外壳命令来完成任务。而且，数组边界的检查以及其他内存保护特性也是 Java 语言的核心部分。

7. 市场对比

即使软件技术设计很好，但是，如果软件提供商不支持它们，或开发人员很难掌握它们，那么这项技术的优点也无法体现出来。Servlet 技术都得到了服务器提供商的广泛支持，基于 Servlet 技术的应用也遍布各行各业，使得此技术的投入得以很好延续和支持。

1.2 Servlet 容器

什么是 Servlet 容器？有时候也叫做 Servlet 引擎，它是 Web 服务器中专门用于装载、运行 Servlet 的一个模块，是 Web 服务器或应用程序服务器的一部分，用于在发送的请求和响应之上提供网络服务，解码基于 MIME 的请求，格式化基于 MIME 的响应。Servlet 容器在 Servlet 的生命周期内包容和管理 Servlet。

1.2.1 概览

图 1-1 展示了 Web 应用程序中的不同组件。

HTML 文件是存储在服务器文件系统中的静态资源。Servlet 运行于 Servlet 容器中，Servlet 容器是 Servlet 组件运行的环境。DB 是存储持久数据的数据库。客户端浏览器发送请求到 Web 服务器。如果请求的是 HTML 文件，服务器直接将请求的 HTML 文件返回。如果请求的是一个 Servlet，服务器将该请求转发给 Servlet，后者再根据请求使用文件系统，访问数据库，产生返回客户端的响应内容。