但这种简单协议无法提供更多的灵活性，也无法实现本书中描述的大部分 HTTP 特性和应用。这里对其进行简要的描述，是因为仍然有一些客户端、服务器和其他应用程序在使用这个协议，应用程序的编写者应该清楚它的局限性。

3.3 方法

现在，我们对前面在表 3-1 中列出的一些基本 HTTP 方法进行更为深入的讨论。注意，并不是每个服务器都实现了所有的方法。如果一台服务器要与 HTTP 1.1 兼容，那么只要为其资源实现 GET 方法和 HEAD 方法就可以了。

即使服务器实现了所有这些方法，这些方法的使用很可能也是受限的。例如，支持 DELETE 方法或 PUT 方法（本节稍后介绍）的服务器可能并不希望任何人都能够删除或存储资源。这些限制通常都是在服务器的配置中进行设置的，因此会随着站点和服务器的不同而有所不同。

3.3.1 安全方法

HTTP 定义了一组被称为安全方法的方法。GET 方法和 HEAD 方法都被认为是安全的，这就意味着使用 GET 或 HEAD 方法的 HTTP 请求都不会产生什么动作。

不产生动作，在这里意味着 HTTP 请求不会在服务器上产生什么结果。例如，你在 Joe 的五金商店购物时，点击了“提交购买”按钮。点击按钮时会提交一个带有信用卡信息的 POST 请求（稍后讨论），那么在服务器上，就会为你执行一个动作。在这种情况下，为购买行为支付信用卡就是所执行的动作。

安全方法并不一定是什么动作都不执行的（实际上，这是由 Web 开发者决定的）。使用安全方法的目的就是允许 HTTP 应用程序开发者通知用户，什么时候会使用某个可能引发某些动作的不安全方法。在 Joe 的五金商店的例子中，你的 Web 浏览器可能会弹出一条警告消息，说明你正在用不安全的方法发起请求，这样可能会在服务器上引发一些事件（比如用你的信用卡支付费用）。

3.3.2 GET

GET 是最常用的方法。通常用于请求服务器发送某个资源。HTTP/1.1 要求服务器实现此方法。图 3-7 显示了一个例子，在这个例子中，客户端用 GET 方法发起了一
53 次 HTTP 请求。