实际上，客户端程序的实现者也应该做好应对非预期 100 Continue 响应的准备（这很烦人，但确实如此）。有些出错的 HTTP 应用程序会不合时宜地发送这些代码。

2. 服务器与100 Continue

如果服务器收到了一条带有值为 100 Continue 的 Expect 首部的请求，它会用 100 Continue 响应或一条错误码来进行响应（参见表 3-9）。服务器永远也不应该向没有发送 100 Continue 期望的客户端发送 100 Continue 状态码。但如前所述，有些出错的服务器可能会这么做。

如果出于某种原因，服务器在有机会发送 100 Continue 响应之前就收到了部分（或全部）的实体，就说明客户端已经决定继续发送数据了，这样，服务器就不需要发送这个状态码了。但服务器读完请求之后，还是应该为请求发送一个最终状态码（它可以跳过 100 Continue 状态）。

最后，如果服务器收到了带有 100 Continue 期望的请求，而且它决定在读取实体的主体部分之前（比如，因为出错而）结束请求，就不应该仅仅是发送一条响应并关闭连接，因为这样会妨碍客户端接收响应（参见 4.7.4 节）。

3. 代理与100 Continue

如果代理从客户端收到了一条带有 100 Continue 期望的请求，它需要做几件事情。如果代理知道下一跳服务器（在第 6 章中讨论）是 HTTP/1.1 兼容的，或者并不知道下一跳服务器与哪个版本兼容，它都应该将 Expect 首部放在请求中向下转发。如果它知道下一跳服务器只能与 HTTP/1.1 之前的版本兼容，就应该以 417 Expectation Failed 错误进行响应。[4]

如果代理决定代表与 HTTP/1.0 或之前版本兼容的客户端，在其请求中放入 Expect 首部和 100 Continue 值，那么，（如果它从服务器收到了 100 Continue 响应）它就不应该将 100 Continue 响应转发给客户端，因为客户端可能不知道该拿它怎么办。

代理维护一些有关下一跳服务器及其所支持的 HTTP 版本的状态信息（至少要维护那些最近收到过请求的服务器的相关状态）是有好处的，这样它们就可以更好地处理收到的那些带有 100 Continue 期望的请求了。

3.4.2 200 ~ 299——成功状态码

客户端发起请求时，这些请求通常都是成功的。服务器有一组用来表示成功的状态码，分别对应于不同类型的请求。表 3-7 列出了已定义的成功状态码。

注 4：还有一种合理的方法，是向客户端先返回 100 Continue，在向服务器转发请求时，删掉 Expect 首部。（译者注）