(端口号已在2.1节简要讨论过。它们将在第3章中更为详细地涉及。)

我们通过一个简单的UDP应用程序和一个简单的TCP应用程序来介绍UDP和TCP套接字编程。我们用Python语言来呈现这些简单的TCP和UDP程序。也可以用Java、C或C++来编写这些程序，而我们选择用Python最主要原因是Python清楚地揭示了关键的套接字概念。使用Python，代码的行数更少，并且向新编程人员解释每一行代码不会有太大困难。如果你不熟悉Python，也用不着担心，只要你有过一些用Java、C或C++编程的经验，就应该很容易看得懂下面的代码。

如果读者对用Java进行客户-服务器编程感兴趣，建议你去查看与本书配套的Web网站。事实上，能够在那里找到用Java编写的本节中的所有例子(和相关的实验)。如果读者对用C进行客户-服务器编程感兴趣，有一些优秀参考资料可供使用[Donahoo 2001; Stevens 1997; Frost 1994; Kurose 1996]。我们下面的Python例子具有类似于C的外观和感觉。

2.7.1 UDP套接字编程

在本小节中，我们将编写使用UDP的简单客户-服务器程序；在下一小节中，我们将编写使用TCP的简单程序。

2.1节讲过，运行在不同机器上的进程彼此通过向套接字发送报文来进行通信。我们说过每个进程好比是一座房子，该进程的套接字则好比是一扇门。应用程序位于房子中门的一侧；运输层位于该门朝外的另一侧。应用程序开发者在套接字的应用层一侧可以控制所有东西；然而，它几乎无法控制运输层一侧。

现在我们仔细观察使用UDP套接字的两个通信进程之间的交互。在发送进程能够将数据分组推出套接字之门之前，当使用UDP时，必须先将目的地址附在该分组之上。在该分组传过发送方的套接字之后，因特网将使用该目的地址通过因特网为该分组选路到接收进程的套接字。当分组到达接收套接字时，接收进程将通过该套接字取回分组，进而检查分组的内容并采取适当的动作。

因此你可能现在想知道，附在分组上的目的地址包含了什么？如你所期待的，目的主机的IP地址是目的地址的一部分。通过在分组中包括目的地的IP地址，因特网中的路由器将能够通过因特网将分组选路到目的主机。但是因为一台主机可能运行许多网络应用进程，每个进程具有一个或多个套接字，所以在目的主机指定特定的套接字也是必要的。当生成一个套接字时，就为它分配一个称为**端口号**(port number)的标识符。因此，如你所期待的，分组的目的地址也包括该套接字的端口号。归纳起来，发送进程为分组附上的目的地址是由目的主机的IP地址和目的地套接字的端口号组成的。此外，如我们很快将看到的那样，发送方的源地址也是由源主机的IP地址和源套接字的端口号组成，该源地址也要附在分组之上。然而，将源地址附在分组之上通常并不是由UDP应用程序代码所为，而是由底层操作系统自动完成的。

我们将使用下列简单的客户-服务器应用程序来演示对于UDP和TCP的套接字编程：

1) 客户从其键盘读取一行字符并将数据向服务器发送。
2) 服务器接收该数据并将这些字符转换为大写。
3) 服务器将修改的数据发送给客户。
4) 客户接收修改的数据并在其监视器上将该行显示出来。