You can install MySQL at the command line or using the GUI through System ▸ Administration ▸ Synaptic Package Manager. If you do install MySQL with `apt-get` at the command line, it is a good idea to update your installer information first with the command:

```
sudo apt-get update
```

Then type this command in a terminal window:

```
sudo apt-get install mysql-server
```

It will ask you a few times if you want to set the `root` password. You can leave it blank for the purposes of this demo and change it later if you wish to make your databases work over the Web.

Working with vector and pixel art in Linux for Chapters 17–19

The open source graphics programs discussed in this book (Inkscape, ImageJ, and GIMP) can be installed on Ubuntu through the Synaptic Package Manager. Select the System menu at the top of the Ubuntu screen, then the Administration submenu, and click on the Synaptic Package Manager. Search for the programs you want to install, mark them for installation, and click the apply button at the top of the screen.

Appendix 2

REGULAR EXPRESSION SEARCH TERMS

Regular expressions—ways to perform adaptive searches and replacements—are described in Chapters 2 and 3. Here we provide a quick reference to some of the more common regular expression terms. This table and the text of the book itself do not encompass the entire range of regular expressions. There are many other useful constructs, for example, embedding miniature scripts into your replacement terms, and searching for A or B in a string using the syntax `(sword|jelly)fish`. If you would like to delve deeper, there are many online references, and there is even an in-depth reference guide built into the Help menu of TextWrangler.

There is some variation in the terms supported from program to program and from language to language. The most widespread terms, which can be used almost anywhere that regular expressions are supported, are the POSIX Extended Regular Expressions. These include `.`, `*`, `+`, `{}`, `()`, `[]`, `[^]`, `^`, `$`, `?`, and `|`. While quite a bit can be accomplished with the POSIX terms, in many implementations the language has been supplemented with some nonstandard terms. Most of these nonstandard terms are based on Perl regular expressions. These include many of the character class wildcards listed in the tables below, such as `\d`, `\w`, and `\n`. These extra wildcards make it easier to write clear regular expressions. Lack of support for Perl-like regular expressions is one of the most common causes of confusion when moving to a new programming context.

If you are using regular expressions in a new context but find that they don't behave as expected, or that they generate errors, check to see which regular expressions are supported by the tool you are using. POSIX does define its own set of wildcards, but the syntax is different from the Perl-style `\w` format that we use in this book. These wildcards include `[:digit:]` in place of `\d` and `[:alpha:]` instead of `\w` that we use in this book (though not including the digits). These POSIX character classes can be used in some contexts where Perl classes aren't available, including SQL queries and the command-line tool `grep`. If you don't want to switch between wildcard types, a more universal solution is to replace character class wildcards with an explicit character range, such as `[0-9]` or `[A-Z]`.

Wildcards

`\w`	Letters, numbers and _
`.`	Any character except `\n` `\r`
`\d`	Numerical digits
`\t`	Tab
`\r`	Return character. Also used as the generic end-of-line character in TextWrangler
`\n`	Line-feed character. Also used as the generic end-of-line character in Notepad++
`\s`	Space, tab, or end of line
`[A-Z]`	A single character of the ranges indicated in square brackets
`[^A-Z]`	A single character including all characters *not* in the brackets. Note that this will include `\n` unless otherwise specified, and may cause you to match across lines
`\`	Used to escape punctuation characters so they are searched for as themselves, not interpreted as wildcards or special symbols
`\\`	The `\` symbol itself, escaped

Boundaries

`^`	Match the start of the line, i.e., the position before the first character
`$`	Match the last position before the end-of-line character

Quantifiers, used in combination with characters and wildcards

`+`	Look for the longest possible match of one or more occurrences of the character, wildcard, or bracketed character range immediately preceding. The match will extend as far as it can while still allowing the entire expression to match.
`*`	As above, matches as many of the previous character to occur, but allows for the character not to occur at all if the match still succeeds
`?`	Modifies greediness of + or * to match the shortest possible match instead of longest
`{}`	Specify a range of numbers to repeat the match of the previous character. For example: `\d{2,4}` matches between 2 and 4 digits in a row `[AC]{4,}` matches 4 or more of the letter A or C in a row

Capturing and replacing

`( )`	Capture the search results between the parentheses for use in the replacement term
`\1` `$1`	Substitute the contents of the matched into the replacement term, in numerical order. Syntax depends on the text editor or language that you are using.