Conventions for Writing and Editing MySQL Documentation ("Styleguide")

Conventions for Writing and Editing MySQL Documentation ("Styleguide")

Abstract

This guide details the conventions and writing guidelines for writing MySQL documentation.

Document generated on: 2009-06-02 (revision: 15161)

Copyright © 1997-2008 MySQL AB, 2009 Sun Microsystems, Inc. All rights reserved. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. Sun, Sun Microsystems, the Sun logo, Java, Solaris, StarOffice, MySQL Enterprise Monitor 2.0, MySQL logo™ and MySQL™ are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Copyright © 1997-2008 MySQL AB, 2009 Sun Microsystems, Inc. Tous droits réservés. L'utilisation est soumise aux termes du contrat de licence.Sun, Sun Microsystems, le logo Sun, Java, Solaris, StarOffice, MySQL Enterprise Monitor 2.0, MySQL logo™ et MySQL™ sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exlusivement par X/Open Company, Ltd.

This documentation is NOT distributed under a GPL license. Use of this documentation is subject to the following terms: You may create a printed copy of this documentation solely for your own personal use. Conversion to other formats is allowed as long as the actual content is not altered or edited in any way. You shall not publish or distribute this documentation in any form or on any media, except if you distribute the documentation in a manner similar to how Sun disseminates it (that is, electronically for download on a Web site with the software) or on a CD-ROM or similar medium, provided however that the documentation is disseminated together with the software on the same medium. Any other use, such as any dissemination of printed copies or use of this documentation, in whole or in part, in another publication, requires the prior written consent from an authorized representative of Sun Microsystems, Inc. Sun Microsystems, Inc. and MySQL AB reserve any and all rights to this documentation not expressly granted above.

For more information on the terms of this license, for details on how the MySQL documentation is built and produced, or if you are interested in doing a translation, please contact the Documentation Team.

For additional licensing information, including licenses for libraries used by MySQL, see Preface, Notes, Licenses.

If you want help with using MySQL, please visit either the MySQL Forums or MySQL Mailing Lists where you can discuss your issues with other MySQL users.

For additional documentation on MySQL products, including translations of the documentation into other languages, and downloadable versions in variety of formats, including HTML, CHM, and PDF formats, see MySQL Documentation Library.

Table of Contents

Chapter 1. Principles and Miscellaneous

This section covers general guidelines.

- *Acronyms* should generally be spelled out the first time they appear in a book, as in: “Computer Development Environment (CDE).” After the acronym has been defined, you should generally use the acronym only (not the whole term, unless it makes more sense contextually to use the whole term). Usually, acronyms are defined only one per book. But if you prefer, you can also define them the first time they appear in each chapter.

- *Change notes*: In change notes, references to bug numbers should be written after the sentence describing the bugfix, like this: `This sentence describes the fix. (Bug #nnnn)`, where `nnnn` is the number of the bug assigned in the MySQL bug tracking system. The process that converts the Manual to HTML finds these and transforms them to hyperlinks that take the reader to the proper report in the bug system at http://bugs.mysql.com.

  If there are multiple bugs, write them as `(Bug #mmmm, Bug #nnnn)` rather than as `(Bug #mmmm, #nnnn)`. (The word “Bug” preceding the number is important.)

- *Code lines* should run no longer than 72 characters. This is particularly important for printed books. Code lines should be indented using spaces, not tabs.

- *Commas*: To write a list of items, add commas after all items preceding the last one. Example: `Features, products, and services`, not `Features, products and services`

- *Dates and numbers*:

  - Spell out numbers under 10, unless the same object appears in a sentence with an object 10 or over (five apples; 5 apples and 100 oranges).

  - Use numerals for versions (`Version 5`), if it is an actual value (set `long_query_time` to `2`), for bits (`32-bit`), and for dates (`the 1980s`).

  - Phone numbers should appear in this format, using either spaces or dashes: `+country code city code telephone number`

    Example: `+49 30 82702940` or `+49-30-8270294-0`

- *Ellipsis*: The character entity for an ellipsis (&hellip;) causes problems when transforming a DocBook document to PDF output. Use the literal `...` instead.

- *Footnotes*: Do not use footnotes at all. Footnotes work well for print or PDF, but we use our documentation in many other formats such as HTML, where footnotes require a lot of navigation on the reader's part. It can take some thought to determine how best to incorporate into a passage the text that you might otherwise put in a footnote. Make the effort. Sometimes in doing so you will become aware that you were using a footnote as a means of being lazy, and that having to rethink a passage reveals a much better way of organizing what you are trying to say.

- *Language*: US English, not UK English. (optimize, not optimise; center, not centre; color, not colour; transportation, not transport; etc.)

- *Lists*: We use two types of lists only: numbered lists and bulleted lists. For the former, use the `<orderedlist>`, for the latter, use the `<itemizedlist>` element.

  No period after list items unless one item forms a complete sentence (then use periods for all items within that list, even fragments).

- *Quotes*: For which kind of quotes to use under which circumstances, see Quotes.

- *Spaces, not tabs*: Do not use tab characters at all, use spaces instead.

- *URLs* are considered breakable at punctuation characters. This is how Pearson handles it in printed materials.

  In examples, only use URLs that are either owned by MySQL, or (preferably) URLs that are recommended for documentation in general:

  - `example.com`

  - `example.net`

  - `example.org`

Example:

```
You have reached this web page by typing “example.com”,
“example.net”, or “example.org” into the
address bar of your web browser.
```

These domain names are reserved for use in documentation and are not available for registration. See RFC 2606, Section 3.

- We recommend using binary distributions, rather than source distributions.

  If a section contains instructions for both types of distributions, put those for binary distributions first. If you find a section that still has source instructions first, see if it can be rewritten to put binary instructions first.

- We now have more Windows users than Unix users. So, if a section contains instructions for both platforms, put those for Windows first, then Unix. If you find a section that still has the Unix instructions first, see if it can be rewritten to put Windows first.

  Example of allowable exception: The `--socket` option is used for specifying a Unix domain socket file name or a Windows named pipe name. The name of the option itself obviously has its roots in the Unix usage, so it can make sense to mention its Unix meaning first and then the Windows meaning. Such exceptions tend to be rare.

Chapter 2. Guidelines for Numbering

Do not number elements by hand. The numbering for chapters, appendices, and sections is done automatically when output formats like HTML or PDF are created. This is also true for figures and graphics. If you need numbered lists, you should use the `<orderedlist>` element.

Chapter 3. Guidelines for Capitalization of Terms

This subsection covers the capitalization of words and terms for certain elements, like headings.

- *Figure captions* are initial-capped on first word only, with the exception of proper nouns. There is no period after figure captions. Example: `Switching to configure-service mode`

  Figure captions may end with a period. This makes sense if most figure captions are full sentences. Currently, we don't use periods, but should we start using them we should use periods even for captions that are not full sentences (consistency overrules grammatical correctness).

- *SQL function names* should use all capitals, followed by `()`; for example, `DATABASE()`, and the `<literal>` element. The `<literal>` element must not be used when an SQL function is inside a `<programlisting>` element.

- *SQL keywords and statements* should use all capitals. In paragraph text, enclose the terms within `<literal>`. In program listings (`<programlisting>...</programlisting>`), do not use `<literal>`. `<replaceable>` is allowable within `<literal>` and `<programlisting>`. `<userinput>` and `<replaceable>` are allowable within `<programlisting>`.

- *Titles* of chapters and sections should use initial capitals following these rules: Cap the first letter of each word, with the exception of articles, conjunctions, prepositions of four letters or less, and program names or technical words that are always lowercase. Hyphenated words in titles or captions should both be capped if the second word is a main word, but only the first should be capped if the second word isn't too important (it is a bit of a judgment call). Example: `Big-Endian, Built-in` *Prepositions* with four letters or fewer should be lowercase: `from, with, Within, Between`. Also, the word `to` should be lowercase.

- *Product names* should be capitalized. Sometimes it's not easy to determine whether a term is a product name or not. Here's an example:

  “MySQL Server” is a product name, while “MySQL server” is for phrases like “the MySQL server”. Use of the product name is a concept that gets a lot of love from marketing or something. However, use of it everywhere makes text sound still and stilted. We should use “MySQL Server” when talking about the product in general (for example, discussing features) but “MySQL server” (or just “the server”) when talking about specific uses (for example, “the MySQL server performs character set conversion when necessary”).

Chapter 4. Guidelines for Using Denominations

This section gives advice how names and other denominations should be used.

4.1. Denominations in General

- *CLI*: Do not use CLI as an abbreviation for Command Line Interface. It is a standard abbreviation for Call-Level Interface.
- *Connection*: It is true that connections and sessions are associated, but the word “connection” should not be used when “session” is more accurate. For example: not “a transaction during this connection” but “a transaction during this session.” Indeed, Paul has noted a tendency to use connection, session, client, and thread all to mean roughly the same thing. And with the advent of thread-pooling for handing client connections in MySQL 6.0, it no longer necessarily true that a given connection will be serviced by a given thread. This means the rough synonymy that historically has held true is no longer valid.
- *Cursor*: It is difficult to pin this one down, but it seems that some people might be saying “create a cursor” when they really mean “allocate a statement (stmt).” The word cursor must be reserved for the named object which is the subject of open/fetch/close.
- *Database*: This appears even in the MySQL Reference Manual: “MySQL, the most popular Open Source database ...” Correct is: Database Management System, or DBMS.
- *Derived table*: It is true that, in the following statement, the bit in the `FROM` clause is a “derived table.”

  ```
  SELECT * FROM (SELECT * FROM t) AS X;
  ```

  However, so are many other things. For example, a view is a [named] derived table. Therefore, if you really want to specify that you are using a query in the `FROM` clause, correct is: Query in the `FROM` clause.
- *Field*: People are using “field” when they clearly must mean “column”, as in “a field of a table.” Correct is: Column. This will take a while, since the misuse of “field” occurs in MySQL functions and statements.

  Paul breaks this rule in the following context: Some `SHOW` statements display information about table columns. To discuss this output, it's necessary to refer to table columns and columns in the result set of the `SHOW` statement, which can easily become confusing. It can help to disambiguate this by referring to the fields of the result set and the columns of the table.
- *Key*: The word KEY may refer to a particular value in an index (an “index key”) or to a set of columns which are useful for lookup (a “primary key” or “foreign key”). However, influenced perhaps by MySQL's “KEY(column)” clause in CREATE TABLE, some people are saying “create a key on column1.” They should say “create an index on column1.”
- *MySQL*: Write MySQL without adornment, that is, not as <emphasis>MySQL</emphasis> or <literal>MySQL</literal>.
- *MySQL Reference Manual*: The name of the reference Manual is MySQL Reference Manual. Subsequent references in close proximity can be just Reference Manual or *Manual*, as long as it is clear what the text is referring to.
- *Query*: A query is a question, look in your dictionary. `SELECT` may be called a query, but it is not appropriate to call `UPDATE` a “query.” Also incorrect is: command. Correct is: statement.
- *Relation*: A relation is a set. That is why a relational database is called relational. It isn't because of some “relationship between tables.”
- *ROWID*: The word “rowid” has been used once, by one person, as an apparent synonym for Primary Key. Wrong. We should use rowid only for a physical location.
- *Subselect*: This was okay at one time, but we want to standardize. Correct is: Subquery.
- Write Unix, not UNIX.
- *Unix and Linux*: Unless necessary for some specific reason, we treat Unix and Linux as synonymous, with both being covered by the term “Unix”.
- URLs ending in a domain name or directory should not have a slash (`/`) at the end.

4.2. Denominations With Regards to Collations

This section contains a list of terms and definitions used in the context of collations.

- *allkeys.txt*: An example of a series of collating-table entries as defined by UCA (Unicode Collation Algorithm).

  Actually UCA says `allkeys.txt` is a “collation element table”, that is, it is the part of a collating table which shows collating elements.

- *COLLATING ELEMENT*: The unit which linguistically-aware users perceive as the minimal building block in string comparisons.

  Usually there is a one-to-one relation between characters and collating elements, for example in English there is a character “A” and a collating element for “A”. More rarely there is a many-to-one relation, for example in traditional Spanish the two-character combination “LL” is a single collating element.

  Usually there is a one-to-many relation between collating elements and weights (because there are multiple levels); however, for an ignorable character, one collating element has zero weights.

- *COLLATING TABLE*: A table which describes all the rules for a collation, including Posix-like “Locale” declarations and a list of collating elements.

  Here are entries for collating elements from two sources, ISO 14651 and `allkeys.txt`:

  ```
  [From ISO 14651]
  <U0024> <S2C4>;<BASE>;<MIN>;<U0024> % DOLLAR SIGN
  <UFF04> <S2C4>;<BASE>;<WIDE>;<UFF04> % FULLWIDTH DOLLAR SIGN
  <UFE69> <S2C4>;<BASE>;<SMALL>;<UFE69> % SMALL DOLLAR SIGN
  ```

  ```
  [From allkeys.txt 4.0]
  0024  ; [.0E0F.0020.0002.0024] # DOLLAR SIGN
  FF04  ; [.0E0F.0020.0003.FF04] # FULLWIDTH DOLLAR SIGN; QQK
  FE69  ; [.0E0F.0020.000F.FE69] # SMALL DOLLAR SIGN; QQK
  ```

  Clearly these are the same thing, but ISO 14651 uses names (e.g. “BASE”) where `allkeys.txt` uses numbers (e.g. `0020`). So ISO 14651 had to define earlier in its table `BASE = 0020; MIN = 0002; WIDE = 0003; SMALL= 000F` etc.

- *COLLATION ELEMENT*: Do not use. Use *collating element*.

- *COLLATION TABLE*: Do not use. Use *collating table*.

- *COLLATING TABLE ENTRY*: A line in a collating table, representing one fact.

  Each “line” in `allkeys.txt` (which is a subset of a collating table) is an entry for one collating element.

- *CONTRACTION*: A mapping from *`N`* characters to less-than-*`N`* collation elements.

  Contraction is rare, for example the character “C” has one collation element “C”. But take an example from traditional Spanish: “LL” is a single collation element between “L” and “M”. Contraction also occurs when there has been decomposition. For example here are two collating element entries (from `allkeys.txt` 5.0.0):

  ```
  0622  ; [.15E2.0020.0002.0622] # ARABIC LETTER ALEF WITH MADDA ABOVE
  0627 0653 ; [.15E2.0020.0002.0622] # ARABIC LETTER ALEF WITH MADDA ABOVE
  ```

  Notice that there is one collation element labelled `0627 0653`, which clearly is the result of mapping from two characters `U+0627 ARABIC LETTER ALEF` and `U+0653 ARABIC MADDAH ABOVE`, with the same weights as the composed character `U+0622 ARABIC LETTER ALEF WITH MADDA ABOVE`.

- *EXPANSION*: A one-to-many mapping from collating element to weighting levels.

  For example, German Sharp S may be treated as “ss”, so the `allkeys.txt` entry for collating element `00DF` (Sharp S) is:

  ```
  00DF  ; [.11AF.0020.0004.00DF][.0000.0199.0004.00DF][.11AF.0020.001F.00DF] # LATIN SMALL LETTER SHARP S;
  ```

  The entry for “s” alone is:

  ```
  0073  ; [.11AF.0020.0002.0073] # LATIN SMALL LETTER S
  ```

  Since `0000` means ignorable, two-level weight strings are:

  ```
  11AF 11AF 0020 0199 0020      /* for SHARP S */
  11AF 11AF 0020 0020           /* for 'ss' */
  ```

- *IGNORABLE CHARACTER*: A character which has one collating element which has no significance for comparison. One ignorable character has one collating element, but zero weights at all levels. For example (from allkeys.txt): 0591 ; [.0000.0000.0000.0591] # HEBREW ACCENT ETNAHTA This is ignorable for three levels but not four levels. Therefore it is an “ignorable character” when you produce a weight string for one, two, or three levels. “Ignorable at level 1” means the level-1 weight is ignorable, as represented by 0000 in allkeys.txt. “Fully ignorable” means ignorable for all levels.

- *ISO 14651*: The ISO/IEC 14651 “International String Ordering” standard.

  Draft documents:

  - http://std.dkuug.dk/jtc1/sc22/open/n2933.pdf
  - http://ra.dkuug.dk/jtc1/sc22/wg20/docs/n731-fdis14651.pdf
  - http://www.dulug.dk/JTC1/SC2/WG2/docs/n2691.pdf

- *LEVEL*: A prioritization order for weights.

  Each level has a name “level + number”, for example “level 1”, “level 2”, “level 3”, “level 4”. (Do not use, or rarely use, equivalent terms “primary”, “secondary”, “tertiary”, “quaternary”.) Typically level 1 is the character-differs level for `WHERE` clauses, levels 2 and following are case-differs or accent-differs something-minor-differs levels which might be useful for `ORDER BY` clauses. For example, from `allkeys.txt` 5.0.0:

  ```
  0061  ; [.0FD0.0020.0002.0061] # LATIN SMALL LETTER A
  24D0  ; [.0FD0.0020.0006.24D0] # CIRCLED LATIN SMALL LETTER A;
  0041  ; [.0FD0.0020.0008.0041] # LATIN CAPITAL LETTER A
  ```

  There are four levels here. Level 1 is always `0FD0` for “A”. Level 2 is always `0020`. Level 3 is `0002` for `SMALL`, `0006` for `CIRCLED`, `0008` for `CAPITAL`. Level 4 is the same as the Unicode code point value. Do not confuse “weight level” with “weighting level”.

- *ORDERING KEY*: Do not use. Use “weight string”.

- *SORTKEY*: Do not use. Use “weight string”.

- *SUBKEY*: A sequence of weights for a single level.

- *UCA*: Unicode Collation Algorithm as described in Unicode Technical Standard #10, http://www.unicode.org/reports/tr10.

- *WEIGHT*: A positive numeric value used for comparisons.

  Weights come from collating tables and go to weight strings. Often weight appears as a 4-digit number in collating tables. For example (from `allkeys.txt`):

  ```
  0062  ; [.0FE6.0020.0002.0062] # LATIN SMALL LETTER B
  ```

  This is the entry for collating element `0062`, and there are 4 weights: `0FE6` and `0020` and `0002` and `0062`.

- *WEIGHT STRING*: A binary string, sometimes called a “sortkey” or an “ordering key”, produced by taking a series of weights from a collating table for a certain number of levels, ordering them by position and level, and outputting.

  For example: starting with a character string `ABC`, and knowing that the number of levels is 2, look up the collating elements for `A` and `B` and `C` in `allkeys.txt` 5.0.0:

  ```
  0041  ; [.0FD0.0020.0008.0041] # LATIN CAPITAL LETTER A
  0042  ; [.0FE6.0020.0008.0042] # LATIN CAPITAL LETTER B
  0043  ; [.0FFE.0020.0008.0043] # LATIN CAPITAL LETTER C
  ```

  Result: `0FD0.0FE6.0FFE.0020.0020.0020`. MySQL's `weight_string()` function produces a weight string.

- *WEIGHTING ELEMENT*: A sequence of weights, in ascending order by level.

  For example, from `allkeys.txt` 5.0.0:

  ```
  00DF  ; [.11AF.0020.0004.00DF][.0000.0199.0004.00DF][.11AF.0020.001F.00DF] # LATIN SMALL LETTER SHARP S
  ```

  There are three weighting elements in this example, each is surrounded by square brackets:

  ```
  [.11AF.0020.0004.00DF]
  [.0000.0199.0004.00DF]
  [.11AF.0020.001F.00DF]
  ```

  Often one collating element has only one weighting element (which has many weights), but `SHARP S` is an example of expansion.

- *ZERO WEIGHTS*: The meaning is “an empty sequence of weights” (the ISO 14651 definition), not “weights with value 0000” (the UCA definition).

  For example (from `allkeys.txt`):

  ```
  0591  ; [.0000.0000.0000.0591] # HEBREW ACCENT ETNAHTA
  ```

There are three “empty sequences of weights” here, all of which look like `0000`, which we interpret as code for “empty”.

Chapter 5. Guidelines on Grammar and Style

This section covers a number of common grammar and style issues.

Do not use contructs like this:

```
How to use SELECTs.
```

This is not just bad English, but makes it unnecessarily hard for translators. You could rewrite it like this:

```
How to use SELECT statements.
```

Another problem with constructs such as “SELECTs” is that it involves an ugly within-word font change. Constructs such as non-`NULL` do not have this problem because there is an intervening hyphen.

Under no circumstance allow instance of things like “SELECT's” or “SELECT:s” to go unchanged whenever you encounter them. Writing “SELECT's” is incorrect because it is not a possessive. Writing “SELECT:s” just looks weird. You are most likely to see these things in the Change Notes sections of the Reference Manual, because developers tend to be sloppier in their language there for some reason. As documenters, we have no such luxury. It is required that we expend the thought necessary to figure out how to rewrite the text properly.

Do not use *emoticons* such as :-), ;-), etc.

Do not end sentences with an exclamation mark (!). There can be exceptions, but they are rare. On occasion you might come across a passage consisting of three or four sentences in a row that end with exclamation marks. Invariably, you can change most of these to periods with no damage to the meaning, and the passage no longer will look like it's hyperventilating.

Avoid *expletives* such as “simple”, “easy”, or “just”. Leave it to the reader or user to decide, for example, whether or not something is simple. Here's an example how to not write a sentence:

```
Using this feature is easy; simply click the red button.
```

Make sure you do not confuse `its` with `it's`. This can best be accomplished by following the next rule.

Apostrophes: Avoid using apostrophes. Print publishers don't (uh, do not) like them. Write `it is`, rather than `it's`. Note that this rule may change due to an editor's preference.

Abbreviations: Avoid using abbreviations. Print publishers do not like these, either. Write “and so forth”, rather than “etc.”. Use “that is” and “for example”, rather than “i.e.” or “e.g.” If you absolutely must use i.e. or e.g., do *not* use i.e, ie, e.g, or eg.

The word *data* is problematic. It is commonly used both in plural and in singular form. The Manual uses it as plural, which means you use `data are` rather than `data is`. It is unfortunate that no matter which form we use, it will look incorrect to some people. But we can at least be internally consistent. If you can substitute another word such as “information,” that makes the problem go away entirely. “Metadata” is especially problematic; there is a shortage of good synonyms. Sometimes you can rewrite a sentence to avoid the singluar/plural issue entirely. “The data are stored in the table” and “The data is stored in the table” can be rewritten as “The table stores the data.” (This also has the benefit of changing the sentence from passive to active.)

When reproducing program output, reproduce it exactly, even if it contains typos. Do not “fix” it. (If the output is produced by a MySQL program, then fix the source for the program to the output correctly without the typo, then update the Manual to match.)

To refer to ASCII codes, use `ASCII` *`n`*, not `ASCII(`*`n`*`)`, unless you are referring to the `ASCII()` function, in which case you use `<function>ASCII()</function>` . For example, write `ASCII 13`.

The rules for hyphenated words are:

- Use a hyphen for a combination adjective preceding its noun as in “case-sensitive collation”.
- Do not use a hyphen for a combination adjective following a noun as in “this collation is case sensitive”.
- Do not use a hyphen for a combination noun as in “case sensitivity”.
- There is no hyphen after words ending with “ly”, for example “statically linked”, not “statically-linked”.
- *Do not use hyphens (or spaces) for prefixes* such as “sub”, “pre”, or “re”, unless this is necessary to disambiguate words. For example, it's “subroutine”, “preamble”, or “reorder”, rather than “sub routine”, “pre-amble”, or “re-order”.

  To disambiguate words, though, some people use hyphens, for example to disambiguate “recreation” (leisure) from “re-creation” (creating something anew). Others use hyphens when the stem of a word begins with a vowel. For example, they would use “re-introduced”, rather than “reintroduced”.

- *The usage of hyphens may depend on how naturalized a word is* in English, and (this is related) to what extent it's regarded a proper noun. Here are some examples:
    - Precambrian
    - pre-Columbian

Avoid terms that imply spatial direction when you really mean preceding or following. Use “mentioned previously” instead of “above” when making relative references in your text. Something that is “above” in a single-page presentation format such as a Web page will not necessarily be above in a multiple-page format such as a printed book. Similarly, use “later in this section” instead of “below.”

For a long dash, use `&mdash;`.

Comments: If a `(comment) in parentheses` is at the end of a sentence, start the comment with lowercase and put the period (`.`) after the closing parens (`)`), such as `(like this).`. If a comment is separate, start with uppercase and put the period inside the closing parens, `(Like this.)`.

After a semicolon, don't use uppercase. It is *not* the start of a new sentence.

After a colon, capitalize the next word if the following text is a complete sentence in itself. Do not capitalize if it is not a complete sentence.

Chapter 6. Guidelines on Spelling

This subsection covers a number of common spelling issues.

- 32-bit, not 32 bit or 32 bits. “32 bit result” should be “32-bit result”. However, “the result has 32 bits” has no hyphen.
- 4KB, 4MB, 2GB, 4TB -- not 4 kilobytes, not 4 Kilobytes, not 4 KB; the same is true for MB (not megabytes), GB (not gigabytes), TB (not terabytes).
- Megahertz and gigahertz are MHz, GHz, not Mhz, Ghz.
- B means bytes, b means bits; use appropriately. The usual place for lowercase b is in network rate indicators, such as 100Mb/s to mean 100 megabits per second.
- administer, not administrate
- a lot requires two words; alot is wrong
- application-related, not application related. In titles, use Application-Related, not Application-related.
- authenticate, not authentify
- authentication, not authentification
- authorize, not authorise
- automatically, not automaticly
- backup is a noun or adjective (as in `a backup file`), back up is a verb (as in `to back up a database`)
- backward, not backwards
- backward-compatible, not backward compatible or backwards compatible
- behavior, not behaviour
- byte-swapping, not byte swapping
- cannot, not can not
- client-side, not client side
- client/server, not client-server
- color, not colour
- command-line is an adjective, command line is a noun. Example: “You should use a command-line tool if you prefer entering commands on the command line.”
- compliance, not compliancy
- core dump is a noun or a verb (as in `a core dump file` or `a program core dumps when it fails`). In the latter case, however, it is better to say `a program dumps core when it fails`
- CPU time, not CPU (in phrases like `uses CPU time`; unless you are referring the processor itself.)
- CPU, not cpu
- data file, data set, data type, not datafile, dataset, datatype
- deprecate, not depreciate (depreciate is a word, but not the one you want when you are writing about features that we discourage people from using, and which may be removed in later releases)
- deprecated, not depricated
- different from *`something`*, not different than *`something`*
- dynamically, not dynamicly
- email, not e-mail

- file-size, not file size
- file name, file system, not filename, filesystem
- floating-point, not floating point
- following some, not something [shown] below
- forward, not forwards
- full-text, not fulltext (unless you are referring to a `FULLTEXT` index)
- hand-held, not hand held
- heavy-duty, not heavy duty
- heavy-load production systems (used as an adjective), but used under heavy load (used on its own).
- high-priority *`something`* (when used as an adjective), not high priority
- host name, not hostname
- indexes, not indices; exception: when referring to array elements, use “indices”
- installation, not install (as a noun)
- lettercase, not letter case
- long-awaited, not long awaited
- long-time *`something`* (when used as an adjective), not long time
- lowercase, not lower case
- low-volume *`something`* (when used as an adjective)
- master/slave, not master-slave
- memory-based, not memory based
- multiple CPU, not multiple-CPU
- multi-byte, not multi byte
- multi-thread(ed), not multithread(ed)
- multi-user, not multi user
- natural-language, not natural language
- Net (capitalized), if referring to the Internet in that way
- NetWare, not Netware
- Note:, not NOTE:
- “Note that...” When a sentence begins this way, consider whether the "Note that" is useless noise that can be deleted or whether it is a necessary part of the sentence. If the reader really needs to be brought to a full stop, use a `<note>` element.
- Object-oriented. It is hyphenated.
- okay, not ok or Ok or OK. Exceptions: When describing instructions for a GUI with buttons that say OK, then use OK. That is, use the label that the GUI uses. When showing the output from a program, show the output exactly; do not change ok to okay, and so forth.
- online, not on-line
- onward, not onwards
- Open Source, not open source
- optimize, not optimise

- otherwise is followed by a comma at the start of a sentence
- percent, not per cent
- platform-dependent, not platform dependent
- PostScript, not Postscript
- power-start, not power start
- press Enter, or press the Enter key, not hit Return or hit Enter
- publicly, not publically
- re-issue(ing), not reissue(ing)
- replication-safe, not replication safe
- rewriting, not re-writing
- rollback is a noun or adjective (as in `a rollback operation`), `roll back` is a verb (as in `roll back a transaction`)
- runtime, not run time
- schemas, not schemata
- server-side, not server side
- shown here, not shown below
- single-CPU, not single CPU
- statically, not staticly
- “sub” is a prefix, not a word, so it's subfunction, rather than sub function, or subroutine, rather than sub routine (or sub-routine)
- third-party, not third party
- “this only”, not “only this”
- thread-safe, not thread safe
- toward, not towards
- transaction-safe, not transaction safe
- turnkey, not turn-key
- uncorrelated subquery, not non-correlated subquery
- unstable, not instable
- uppercase, not upper case
- user-defined, not user defined
- user name, not username
- web page, web site, not webpage, website. Note: Pearson prefers Web page, Web site.
- whether, not whether or not
- wildcard, not wild card or wild-card

More words and terms can be found on the O'Reilly Default Stylesheet and Word List (on the bottom of that page).

Chapter 7. Guidelines for GUI Documentation

This subsection covers how to use terms for GUI documentation.

- To indicate that a checkbox is available for the user, use *active*.
- To indicate that a checkbox is unavailable for the user, that is it's greyed out, use *inactive*.
- For checkboxes, terms such as *enable/disable*, *on/off*, or *selected/de-selected* can be used to indicate the value of the control, as determined by what the user does with it.
- When referring to menu actions, use one of the following forms:
  - Menu, menu item (, submenu item, ...), ...; for example: `Select <guimenu>Edit</guimenu>, <guimenu>Options</guimenu>, <guimenu>Connections</guimenu>.`
  - Select ... from menu (item); for example: `Select <guimenu>Connections</guimenu> after choosing the <guimenu>Options</guimenu> item from the <guimenu>Edit</guimenu> menu.` (Obviously, the first form is suited better for longer actions, whereas the second form might be good for shorter actions.)

  Do not use forms like `Select Edit > Options > Connections`, or `Edit => Options => Connections`.