The New Hacker's Dictionary version 4.2.2

Various editors

The Project Gutenberg Etext of The New Hacker's Dictionary version 4.2.2

Version 4.0.0 of The New Hacker's Dictionary (a.k.a., The Jargon File) was released as etext #817 in February 1997. This file, version 4.2.2, was derived from the public domain source at http://www.tuxedo.org/jargon. This is a plain text version of jarg422h.htm, produced automatically with the Lynx Web browser.

Title: The New Hacker's Dictionary version 4.2.2

Author: Various editors

Release Date: January, 2002 [Etext #3008]
[Yes, we are about one year ahead of schedule]

Edition: 422

Language: English

The Project Gutenberg Etext of The New Hacker's Dictionary version 4.2.2
******This file should be named jarg422.txt or jarg422.zip******

Corrected EDITIONS of our etexts get a new NUMBER, jarg422.txt
VERSIONS based on separate sources get new LETTER, jarg422.txt

Project Gutenberg Etexts are usually created from multiple editions, all of which are in the Public Domain in the United States, unless a copyright notice is included. Therefore, we usually do NOT keep any of these books in compliance with any particular paper edition.

We are now trying to release all our books one year in advance of the official release dates, leaving time for better editing. Please be encouraged to send us error messages even years after the official publication date.

Please note: neither this list nor its contents are final till midnight of the last day of the month of any such announcement. The official release date of all Project Gutenberg Etexts is at Midnight, Central Time, of the last day of the stated month. A preliminary version may often be posted for suggestion, comment and editing by those who wish to do so.

Most people start at our sites at:
http://gutenberg.net
http://promo.net/pg

Those of you who want to download any Etext before announcement can surf to them as follows, and just download by date; this is also a good way to get them instantly upon announcement, as the indexes our cataloguers produce obviously take a while after an announcement goes out in the Project Gutenberg Newsletter.

http://www.ibiblio.org/pub/docs/books/gutenberg/etext02
or
ftp://ftp.ibiblio.org/pub/docs/books/gutenberg/etext02

Or /etext01, 00, 99, 98, 97, 96, 95, 94, 93, 92, 92, 91 or 90

Just search by the first five letters of the filename you want, as it appears in our Newsletters.

Information about Project Gutenberg (one page)

We produce about two million dollars for each hour we work. The time it takes us, a rather conservative estimate, is fifty hours to get any etext selected, entered, proofread, edited, copyright searched and analyzed, the copyright letters written, etc. This projected audience is one hundred million readers. If our value per text is nominally estimated at one dollar then we produce $2 million dollars per hour this year as we release fifty new Etext files per month, or 500 more Etexts in 2000 for a total of 3000+ If they reach just 1-2% of the world's population then the total

should reach over 300 billion Etexts given away by year's end.

The Goal of Project Gutenberg is to Give Away One Trillion Etext Files by December 31, 2001. [10,000 x 100,000,000 = 1 Trillion] This is ten thousand titles each to one hundred million readers, which is only about 4% of the present number of computer users.

At our revised rates of production, we will reach only one-third of that goal by the end of 2001, or about 3,333 Etexts unless we manage to get some real funding.

Something is needed to create a future for Project Gutenberg for the next 100 years.

We need your donations more than ever!

Presently, contributions are only being solicited from people in: Texas, Nevada, Idaho, Montana, Wyoming, Colorado, South Dakota, Iowa, Indiana, and Vermont. As the requirements for other states are met, additions to this list will be made and fund raising will begin in the additional states.

All donations should be made to the Project Gutenberg Literary Archive Foundation and will be tax deductible to the extent permitted by law.

Mail to:

Project Gutenberg Literary Archive Foundation
PMB 113
1739 University Avenue
Oxford, MS 38655 [USA]

We are working with the Project Gutenberg Literary Archive Foundation to build more stable support and ensure the future of Project Gutenberg.

We need your donations more than ever!

You can get up to date donation information at:

http://www.gutenberg.net/donation.html

***

You can always email directly to:

Michael S. Hart <hart@pobox.com>

hart@pobox.com forwards to hart@prairienet.org and archive.org
if your mail bounces from archive.org, I will still see it, if
it bounces from prairienet.org, better resend later on. . . .

We would prefer to send you this information by email.

Example command-line FTP session:

ftp ftp.ibiblio.org
login: anonymous
password: your@login
cd pub/docs/books/gutenberg
cd etext90 through etext99 or etext00 through etext02, etc.
dir [to see files]
get or mget [to get files. . .set bin for zip files]
GET GUTINDEX.?? [to get a year's listing of books, e.g., GUTINDEX.99]
GET GUTINDEX.ALL [to get a listing of ALL books]

The Legal Small Print

(Three Pages)

START**THE SMALL PRINT!**FOR PUBLIC DOMAIN ETEXTS**START
Why is this "Small Print!" statement here? You know: lawyers.
They tell us you might sue us if there is something wrong with
your copy of this etext, even if you got it for free from
someone other than us, and even if what's wrong is not our
fault. So, among other things, this "Small Print!" statement
disclaims most of our liability to you. It also tells you how
you can distribute copies of this etext if you want to.

BEFORE! YOU USE OR READ THIS ETEXT
By using or reading any part of this PROJECT GUTENBERG-tm
etext, you indicate that you understand, agree to and accept
this "Small Print!" statement. If you do not, you can receive
a refund of the money (if any) you paid for this etext by
sending a request within 30 days of receiving it to the person
you got it from. If you received this etext on a physical
medium (such as a disk), you must return it with your request.

ABOUT PROJECT GUTENBERG-TM ETEXTS
This PROJECT GUTENBERG-tm etext, like most PROJECT GUTENBERG-tm etexts,
is a "public domain" work distributed by Professor Michael S. Hart
through the Project Gutenberg Association (the "Project").
Among other things, this means that no one owns a United States copyright
on or for this work, so the Project (and you!) can copy and
distribute it in the United States without permission and
without paying copyright royalties. Special rules, set forth
below, apply if you wish to copy and distribute this etext
under the Project's "PROJECT GUTENBERG" trademark.

Please do not use the "PROJECT GUTENBERG" trademark to market
any commercial products without permission.

To create these etexts, the Project expends considerable
efforts to identify, transcribe and proofread public domain
works. Despite these efforts, the Project's etexts and any
medium they may be on may contain "Defects". Among other
things, Defects may take the form of incomplete, inaccurate or
corrupt data, transcription errors, a copyright or other
intellectual property infringement, a defective or damaged
disk or other etext medium, a computer virus, or computer
codes that damage or cannot be read by your equipment.

LIMITED WARRANTY; DISCLAIMER OF DAMAGES

author, and additional characters may be used to
indicate hypertext links; OR

[*] The etext may be readily converted by the reader at
no expense into plain ASCII, EBCDIC or equivalent
form by the program that displays the etext (as is
the case, for instance, with most word processors);
OR

[*] You provide, or agree to also provide on request at
no additional cost, fee or expense, a copy of the
etext in its original plain ASCII form (or in EBCDIC
or other equivalent proprietary form).

[2] Honor the etext refund and replacement provisions of this
"Small Print!" statement.

[3] Pay a trademark license fee to the Project of 20% of the
gross profits you derive calculated using the method you
already use to calculate your applicable taxes. If you
don't derive profits, no royalty is due. Royalties are
payable to "Project Gutenberg Literary Archive Foundation"
the 60 days following each date you prepare (or were
legally required to prepare) your annual (or equivalent
periodic) tax return. Please contact us beforehand to
let us know your plans and to work out the details.

WHAT IF YOU *WANT* TO SEND MONEY EVEN IF YOU DON'T HAVE TO?
The Project gratefully accepts contributions of money, time,
public domain etexts, and royalty free copyright licenses.
If you are interested in contributing scanning equipment or
software or other items, please contact Michael Hart at:
hart@pobox.com

*END THE SMALL PRINT! FOR PUBLIC DOMAIN ETEXTS*Ver.04.07.00*END*

Node:Top, Next:[2]Introduction, Previous:[3](dir), Up:[4](dir)
#======= THIS IS THE JARGON FILE, VERSION 4.2.2, 20 AUG 2000 =======#

This is the Jargon File, a comprehensive compendium of hacker slang
illuminating many aspects of hackish tradition, folklore, and humor.

This document (the Jargon File) is in the public domain, to be freely
used, shared, and modified. There are (by intention) no legal
restraints on what you can do with it, but there are traditions about
its proper use to which many hackers are quite strongly attached.
Please extend the courtesy of proper citation when you quote the File,
ideally with a version number, as it will change and grow over time.
(Examples of appropriate citation form: "Jargon File 4.2.2" or "The
on-line hacker Jargon File, version 4.2.2, 20 AUG 2000".)

The Jargon File is a common heritage of the hacker culture. Over the
years a number of individuals have volunteered considerable time to
maintaining the File and been recognized by the net at large as

editors of it. Editorial responsibilities include: to collate contributions and suggestions from others; to seek out corroborating information; to cross-reference related entries; to keep the file in a consistent format; and to announce and distribute updated versions periodically. Current volunteer editors include:

Eric Raymond [5]esr@snark.thyrsus.com

Although there is no requirement that you do so, it is considered good form to check with an editor before quoting the File in a published work or commercial product. We may have additional information that would be helpful to you and can assist you in framing your quote to reflect not only the letter of the File but its spirit as well.

All contributions and suggestions about this file sent to a volunteer editor are gratefully received and will be regarded, unless otherwise labelled, as freely given donations for possible use as part of this public-domain file.

From time to time a snapshot of this file has been polished, edited, and formatted for commercial publication with the cooperation of the volunteer editors and the hacker community at large. If you wish to have a bound paper copy of this file, you may find it convenient to purchase one of these. They often contain additional material not found in on-line versions. The two `authorized' editions so far are described in the Revision History section; there may be more in the future.

Node:Introduction, Next:[23]A Few Terms, Previous:[24]Top, Up:[25]Top

Introduction

This document is a collection of slang terms used by various subcultures of computer hackers. Though some technical material is included for background and flavor, it is not a technical dictionary; what we describe here is the language hackers use among themselves for fun, social communication, and technical debate.

The `hacker culture' is actually a loosely networked collection of subcultures that is nevertheless conscious of some important shared experiences, shared roots, and shared values. It has its own myths,

heroes, villains, folk epics, in-jokes, taboos, and dreams. Because hackers as a group are particularly creative people who define themselves partly by rejection of `normal' values and working habits, it has unusually rich and conscious traditions for an intentional culture less than 40 years old.

As usual with slang, the special vocabulary of hackers helps hold their culture together -- it helps hackers recognize each other's places in the community and expresses shared values and experiences. Also as usual, not knowing the slang (or using it inappropriately) defines one as an outsider, a mundane, or (worst of all in hackish vocabulary) possibly even a [26]suit. All human cultures use slang in this threefold way -- as a tool of communication, and of inclusion, and of exclusion.

Among hackers, though, slang has a subtler aspect, paralleled perhaps in the slang of jazz musicians and some kinds of fine artists but hard to detect in most technical or scientific cultures; parts of it are code for shared states of consciousness. There is a whole range of altered states and problem-solving mental stances basic to high-level hacking which don't fit into conventional linguistic reality any better than a Coltrane solo or one of Maurits Escher's `trompe l'oeil' compositions (Escher is a favorite of hackers), and hacker slang encodes these subtleties in many unobvious ways. As a simple example, take the distinction between a [27]kluge and an [28]elegant solution, and the differing connotations attached to each. The distinction is not only of engineering significance; it reaches right back into the nature of the generative processes in program design and asserts something important about two different kinds of relationship between the hacker and the hack. Hacker slang is unusually rich in implications of this kind, of overtones and undertones that illuminate the hackish psyche.

But there is more. Hackers, as a rule, love wordplay and are very conscious and inventive in their use of language. These traits seem to be common in young children, but the conformity-enforcing machine we are pleased to call an educational system bludgeons them out of most of us before adolescence. Thus, linguistic invention in most subcultures of the modern West is a halting and largely unconscious process. Hackers, by contrast, regard slang formation and use as a game to be played for conscious pleasure. Their inventions thus display an almost unique combination of the neotenous enjoyment of language-play with the discrimination of educated and powerful intelligence. Further, the electronic media which knit them together are fluid, `hot' connections, well adapted to both the dissemination of new slang and the ruthless culling of weak and superannuated specimens. The results of this process give us perhaps a uniquely intense and accelerated view of linguistic evolution in action.

Hacker slang also challenges some common linguistic and anthropological assumptions. For example, it has recently become fashionable to speak of `low-context' versus `high-context' communication, and to classify cultures by the preferred context level of their languages and art forms. It is usually claimed that low-context communication (characterized by precision, clarity, and completeness of self-contained utterances) is typical in cultures which value logic, objectivity, individualism, and competition; by contrast, high-context communication (elliptical, emotive, nuance-filled, multi-modal, heavily coded) is associated with cultures

which value subjectivity, consensus, cooperation, and tradition. What then are we to make of hackerdom, which is themed around extremely low-context interaction with computers and exhibits primarily "low-context" values, but cultivates an almost absurdly high-context slang style?

The intensity and consciousness of hackish invention make a compilation of hacker slang a particularly effective window into the surrounding culture -- and, in fact, this one is the latest version of an evolving compilation called the `Jargon File', maintained by hackers themselves for over 15 years. This one (like its ancestors) is primarily a lexicon, but also includes topic entries which collect background or sidelight information on hacker culture that would be awkward to try to subsume under individual slang definitions.

Though the format is that of a reference volume, it is intended that the material be enjoyable to browse. Even a complete outsider should find at least a chuckle on nearly every page, and much that is amusingly thought-provoking. But it is also true that hackers use humorous wordplay to make strong, sometimes combative statements about what they feel. Some of these entries reflect the views of opposing sides in disputes that have been genuinely passionate; this is deliberate. We have not tried to moderate or pretty up these disputes; rather we have attempted to ensure that everyone's sacred cows get gored, impartially. Compromise is not particularly a hackish virtue, but the honest presentation of divergent viewpoints is.

The reader with minimal computer background who finds some references incomprehensibly technical can safely ignore them. We have not felt it either necessary or desirable to eliminate all such; they, too, contribute flavor, and one of this document's major intended audiences -- fledgling hackers already partway inside the culture -- will benefit from them.

A selection of longer items of hacker folklore and humor is included in [29]Appendix A. The `outside' reader's attention is particularly directed to the Portrait of J. Random Hacker in [30]Appendix B. Appendix C, the [31]Bibliography, lists some non-technical works which have either influenced or described the hacker culture.

Because hackerdom is an intentional culture (one each individual must choose by action to join), one should not be surprised that the line between description and influence can become more than a little blurred. Earlier versions of the Jargon File have played a central role in spreading hacker language and the culture that goes with it to successively larger populations, and we hope and expect that this one will do likewise.

Node:A Few Terms, Next:[32]Revision History,
Previous:[33]Introduction, Up:[34]Top

Of Slang, Jargon, and Techspeak

Linguists usually refer to informal language as `slang' and reserve the term `jargon' for the technical vocabularies of various occupations. However, the ancestor of this collection was called the `Jargon File', and hacker slang is traditionally `the jargon'. When talking about the jargon there is therefore no convenient way to

distinguish it from what a linguist would call hackers' jargon -- the formal vocabulary they learn from textbooks, technical papers, and manuals.

To make a confused situation worse, the line between hacker slang and the vocabulary of technical programming and computer science is fuzzy, and shifts over time. Further, this vocabulary is shared with a wider technical culture of programmers, many of whom are not hackers and do not speak or recognize hackish slang.

Accordingly, this lexicon will try to be as precise as the facts of usage permit about the distinctions among three categories:

* `slang': informal language from mainstream English or non-technical subcultures (bikers, rock fans, surfers, etc).
* `jargon': without qualifier, denotes informal `slangy' language peculiar to or predominantly found among hackers -- the subject of this lexicon.
* `techspeak': the formal technical vocabulary of programming, computer science, electronics, and other fields connected to hacking.

This terminology will be consistently used throughout the remainder of this lexicon.

The jargon/techspeak distinction is the delicate one. A lot of techspeak originated as jargon, and there is a steady continuing uptake of jargon into techspeak. On the other hand, a lot of jargon arises from overgeneralization of techspeak terms (there is more about this in the [35]Jargon Construction section below).

In general, we have considered techspeak any term that communicates primarily by a denotation well established in textbooks, technical dictionaries, or standards documents.

A few obviously techspeak terms (names of operating systems, languages, or documents) are listed when they are tied to hacker folklore that isn't covered in formal sources, or sometimes to convey critical historical background necessary to understand other entries to which they are cross-referenced. Some other techspeak senses of jargon words are listed in order to make the jargon senses clear; where the text does not specify that a straight technical sense is under discussion, these are marked with `[techspeak]' as an etymology. Some entries have a primary sense marked this way, with subsequent jargon meanings explained in terms of it.

We have also tried to indicate (where known) the apparent origins of terms. The results are probably the least reliable information in the lexicon, for several reasons. For one thing, it is well known that many hackish usages have been independently reinvented multiple times, even among the more obscure and intricate neologisms. It often seems that the generative processes underlying hackish jargon formation have an internal logic so powerful as to create substantial parallelism across separate cultures and even in different languages! For another, the networks tend to propagate innovations so quickly that `first use' is often impossible to pin down. And, finally, compendia like this one alter what they observe by implicitly stamping cultural approval on terms and widening their use.

Despite these problems, the organized collection of jargon-related

oral history for the new compilations has enabled us to put to rest quite a number of folk etymologies, place credit where credit is due, and illuminate the early history of many important hackerisms such as [36]kluge, [37]cruft, and [38]foo. We believe specialist lexicographers will find many of the historical notes more than casually instructive.

Node:Revision History, Next:[39]Jargon Construction, Previous:[40]A Few Terms, Up:[41]Top

Revision History

The original Jargon File was a collection of hacker jargon from technical cultures including the MIT AI Lab, the Stanford AI lab (SAIL), and others of the old ARPANET AI/LISP/PDP-10 communities including Bolt, Beranek and Newman (BBN), Carnegie-Mellon University (CMU), and Worcester Polytechnic Institute (WPI).

The Jargon File (hereafter referred to as `jargon-1' or `the File') was begun by Raphael Finkel at Stanford in 1975. From this time until the plug was finally pulled on the SAIL computer in 1991, the File was named AIWORD.RF[UP,DOC] there. Some terms in it date back considerably earlier ([42]frob and some senses of [43]moby, for instance, go back to the Tech Model Railroad Club at MIT and are believed to date at least back to the early 1960s). The revisions of jargon-1 were all unnumbered and may be collectively considered `Version 1'.

In 1976, Mark Crispin, having seen an announcement about the File on the SAIL computer, [44]FTPed a copy of the File to MIT. He noticed that it was hardly restricted to `AI words' and so stored the file on his directory as AI:MRC;SAIL JARGON.

The file was quickly renamed JARGON > (the `>' caused versioning under ITS) as a flurry of enhancements were made by Mark Crispin and Guy L. Steele Jr. Unfortunately, amidst all this activity, nobody thought of correcting the term `jargon' to `slang' until the compendium had already become widely known as the Jargon File.

Raphael Finkel dropped out of active participation shortly thereafter and Don Woods became the SAIL contact for the File (which was subsequently kept in duplicate at SAIL and MIT, with periodic resynchronizations).

The File expanded by fits and starts until about 1983; Richard Stallman was prominent among the contributors, adding many MIT and ITS-related coinages.

In Spring 1981, a hacker named Charles Spurgeon got a large chunk of the File published in Stewart Brand's "CoEvolution Quarterly" (issue 29, pages 26-35) with illustrations by Phil Wadler and Guy Steele (including a couple of the Crunchly cartoons). This appears to have been the File's first paper publication.

A late version of jargon-1, expanded with commentary for the mass market, was edited by Guy Steele into a book published in 1983 as "The Hacker's Dictionary" (Harper & Row CN 1082, ISBN 0-06-091082-8). The other jargon-1 editors (Raphael Finkel, Don Woods, and Mark Crispin) contributed to this revision, as did Richard M. Stallman and Geoff

Goodfellow. This book (now out of print) is hereafter referred to as `Steele-1983' and those six as the Steele-1983 coauthors.

Shortly after the publication of Steele-1983, the File effectively stopped growing and changing. Originally, this was due to a desire to freeze the file temporarily to facilitate the production of Steele-1983, but external conditions caused the `temporary' freeze to become permanent.

The AI Lab culture had been hit hard in the late 1970s by funding cuts and the resulting administrative decision to use vendor-supported hardware and software instead of homebrew whenever possible. At MIT, most AI work had turned to dedicated LISP Machines. At the same time, the commercialization of AI technology lured some of the AI Lab's best and brightest away to startups along the Route 128 strip in Massachusetts and out West in Silicon Valley. The startups built LISP machines for MIT; the central MIT-AI computer became a [45]TWENEX system rather than a host for the AI hackers' beloved [46]ITS.

The Stanford AI Lab had effectively ceased to exist by 1980, although the SAIL computer continued as a Computer Science Department resource until 1991. Stanford became a major [47]TWENEX site, at one point operating more than a dozen TOPS-20 systems; but by the mid-1980s most of the interesting software work was being done on the emerging BSD Unix standard.

In April 1983, the PDP-10-centered cultures that had nourished the File were dealt a death-blow by the cancellation of the Jupiter project at Digital Equipment Corporation. The File's compilers, already dispersed, moved on to other things. Steele-1983 was partly a monument to what its authors thought was a dying tradition; no one involved realized at the time just how wide its influence was to be.

By the mid-1980s the File's content was dated, but the legend that had grown up around it never quite died out. The book, and softcopies obtained off the ARPANET, circulated even in cultures far removed from MIT and Stanford; the content exerted a strong and continuing influence on hacker language and humor. Even as the advent of the microcomputer and other trends fueled a tremendous expansion of hackerdom, the File (and related materials such as the [48]Some AI Koans in Appendix A) came to be seen as a sort of sacred epic, a hacker-culture Matter of Britain chronicling the heroic exploits of the Knights of the Lab. The pace of change in hackerdom at large accelerated tremendously -- but the Jargon File, having passed from living document to icon, remained essentially untouched for seven years.

This revision contains nearly the entire text of a late version of jargon-1 (a few obsolete PDP-10-related entries were dropped after careful consultation with the editors of Steele-1983). It merges in about 80% of the Steele-1983 text, omitting some framing material and a very few entries introduced in Steele-1983 that are now also obsolete.

This new version casts a wider net than the old Jargon File; its aim is to cover not just AI or PDP-10 hacker culture but all the technical computing cultures wherein the true hacker-nature is manifested. More than half of the entries now derive from [49]Usenet and represent jargon now current in the C and Unix communities, but special efforts

have been made to collect jargon from other cultures including IBM PC programmers, Amiga fans, Mac enthusiasts, and even the IBM mainframe world.

Eric S. Raymond [50]<esr@snark.thyrsus.com> maintains the new File with assistance from Guy L. Steele Jr. [51]<gls@think.com>; these are the persons primarily reflected in the File's editorial `we', though we take pleasure in acknowledging the special contribution of the other coauthors of Steele-1983. Please email all additions, corrections, and correspondence relating to the Jargon File to [52]jargon@thyrsus.com.

(Warning: other email addresses appear in this file but are not guaranteed to be correct later than the revision date on the first line. Don't email us if an attempt to reach your idol bounces -- we have no magic way of checking addresses or looking up people.)

The 2.9.6 version became the main text of "The New Hacker's Dictionary", by Eric Raymond (ed.), MIT Press 1991, ISBN 0-262-68069-6.

The 3.0.0 version was published in September 1993 as the second edition of "The New Hacker's Dictionary", again from MIT Press (ISBN 0-262-18154-1).

If you want the book, you should be able to find it at any of the major bookstore chains. Failing that, you can order by mail from

The MIT Press 55 Hayward Street Cambridge, MA 02142

or order by phone at (800)-356-0343 or (617)-625-8481.

The maintainers are committed to updating the on-line version of the Jargon File through and beyond paper publication, and will continue to make it available to archives and public-access sites as a trust of the hacker community.

Here is a chronology of the high points in the recent on-line revisions:

Version 2.1.1, Jun 12 1990: the Jargon File comes alive again after a seven-year hiatus. Reorganization and massive additions were by Eric S. Raymond, approved by Guy Steele. Many items of UNIX, C, USENET, and microcomputer-based jargon were added at that time.

Version 2.9.6, Aug 16 1991: corresponds to reproduction copy for book. This version had 18952 lines, 148629 words, 975551 characters, and 1702 entries.

Version 2.9.7, Oct 28 1991: first markup for hypertext browser. This version had 19432 lines, 152132 words, 999595 characters, and 1750 entries.

Version 2.9.8, Jan 01 1992: first public release since the book, including over fifty new entries and numerous corrections/additions to old ones. Packaged with version 1.1 of vh(1) hypertext reader. This version had 19509 lines, 153108 words, 1006023 characters, and 1760 entries.

Version 2.9.9, Apr 01 1992: folded in XEROX PARC lexicon. This version had 20298 lines, 159651 words, 1048909 characters, and 1821 entries.

Version 2.9.10, Jul 01 1992: lots of new historical material. This version had 21349 lines, 168330 words, 1106991 characters, and 1891 entries.

Version 2.9.11, Jan 01 1993: lots of new historical material. This version had 21725 lines, 171169 words, 1125880 characters, and 1922 entries.

Version 2.9.12, May 10 1993: a few new entries & changes, marginal MUD/IRC slang and some borderline techspeak removed, all in preparation for 2nd Edition of TNHD. This version had 22238 lines, 175114 words, 1152467 characters, and 1946 entries.

Version 3.0.0, Jul 27 1993: manuscript freeze for 2nd edition of TNHD. This version had 22548 lines, 177520 words, 1169372 characters, and 1961 entries.

Version 3.1.0, Oct 15 1994: interim release to test WWW conversion. This version had 23197 lines, 181001 words, 1193818 characters, and 1990 entries.

Version 3.2.0, Mar 15 1995: Spring 1995 update. This version had 23822 lines, 185961 words, 1226358 characters, and 2031 entries.

Version 3.3.0, Jan 20 1996: Winter 1996 update. This version had 24055 lines, 187957 words, 1239604 characters, and 2045 entries.

Version 3.3.1, Jan 25 1996: Copy-corrected improvement on 3.3.0 shipped to MIT Press as a step towards TNHD III. This version had 24147 lines, 188728 words, 1244554 characters, and 2050 entries.

Version 3.3.2, Mar 20 1996: A number of new entries pursuant on 3.3.2. This version had 24442 lines, 190867 words, 1262468 characters, and 2061 entries.

Version 3.3.3, Mar 25 1996: Cleanup before TNHD III manuscript freeze. This version had 24584 lines, 191932 words, 1269996 characters, and 2064 entries.

Version 4.0.0, Jul 25 1996: The actual TNHD III version after copy-edit. This version had 24801 lines, 193697 words, 1281402 characters, and 2067 entries.

Version 4.1.0, 8 Apr 1999: The Jargon File rides again after three years. This version had 25777 lines, 206825 words, 1359992 characters, and 2217 entries.

Version 4.1.1, 18 Apr 1999: Corrections for minor errors in 4.1.0, and some new entries. This version had 25921 lines, 208483 words, 1371279 characters, and 2225 entries.

Version 4.1.2, 28 Apr 1999: Moving texi2html out of the production path. This version had 26006 lines, 209479 words, 1377687 characters, and 2225 entries.

Version 4.1.3, 14 Jun 1999: Minor updates and markup fixes. This

version had 26108 lines, 210480 words, 1384546 characters, and 2234 entries.

Version 4.1.4, 17 Jun 1999: Markup fixes for framed HTML. This version had 26117 lines, 210527 words, 1384902 characters, and 2234 entries.

Version 4.2.0, 31 Jan 2000: Fix processing of URLs. This version had 26598 lines, 214639 words, 1412243 characters, and 2267 entries.

Version 4.2.1, 5 Mar 2000: Point release to test new production machinery. This version had 26647 lines, 215040 words, 1414942 characters, and 2269 entries.

Version 4.2.2, 12 Aug 2000: This version had 27171 lines, 219630 words, 1444887 characters, and 2302 entries.

Version numbering: Version numbers should be read as major.minor.revision. Major version 1 is reserved for the `old' (ITS) Jargon File, jargon-1. Major version 2 encompasses revisions by ESR (Eric S. Raymond) with assistance from GLS (Guy L. Steele, Jr.) leading up to and including the second paper edition. From now on, major version number N.00 will probably correspond to the Nth paper edition. Usually later versions will either completely supersede or incorporate earlier versions, so there is generally no point in keeping old versions around.

Our thanks to the coauthors of Steele-1983 for oversight and assistance, and to the hundreds of Usenetters (too many to name here) who contributed entries and encouragement. More thanks go to several of the old-timers on the Usenet group alt.folklore.computers, who contributed much useful commentary and many corrections and valuable historical perspective: Joseph M. Newcomer [53]<jn11+@andrew.cmu.edu>, Bernie Cosell [54]<cosell@bbn.com>, Earl Boebert [55]<boebert@SCTC.com>, and Joe Morris [56]<jcmorris@mwunix.mitre.org>.

We were fortunate enough to have the aid of some accomplished linguists. David Stampe [57]<stampe@hawaii.edu> and Charles Hoequist [58]<hoequist@bnr.ca> contributed valuable criticism; Joe Keane [59]<jgk@osc.osc.com> helped us improve the pronunciation guides.

A few bits of this text quote previous works. We are indebted to Brian A. LaMacchia [60]<bal@zurich.ai.mit.edu> for obtaining permission for us to use material from the "TMRC Dictionary"; also, Don Libes [61]<libes@cme.nist.gov> contributed some appropriate material from his excellent book "Life With UNIX". We thank Per Lindberg [62]<per@front.se>, author of the remarkable Swedish-language 'zine "Hackerbladet", for bringing "FOO!" comics to our attention and smuggling one of the IBM hacker underground's own baby jargon files out to us. Thanks also to Maarten Litmaath for generously allowing the inclusion of the ASCII pronunciation guide he formerly maintained. And our gratitude to Marc Weiser of XEROX PARC [63]<Marc_Weiser.PARC@xerox.com> for securing us permission to quote from PARC's own jargon lexicon and shipping us a copy.

It is a particular pleasure to acknowledge the major contributions of Mark Brader [64]<msb@sq.com> and Steve Summit [65]<scs@eskimo.com> to the File and Dictionary; they have read and reread many drafts, checked facts, caught typos, submitted an amazing number of thoughtful

comments, and done yeoman service in catching typos and minor usage bobbles. Their rare combination of enthusiasm, persistence, wide-ranging technical knowledge, and precisionism in matters of language has been of invaluable help. Indeed, the sustained volume and quality of Mr. Brader's input over several years and several different editions has only allowed him to escape co-editor credit by the slimmest of margins.

Finally, George V. Reilly [66]<georgere@microsoft.com> helped with TeX arcana and painstakingly proofread some 2.7 and 2.8 versions, and Eric Tiedemann [67]<est@thyrsus.com> contributed sage advice throughout on rhetoric, amphigory, and philosophunculism.

Node:Jargon Construction, Next:[68]Hacker Writing Style, Previous:[69]Revision History, Up:[70]Top

How Jargon Works

Jargon Construction

There are some standard methods of jargonification that became established quite early (i.e., before 1970), spreading from such sources as the Tech Model Railroad Club, the PDP-1 SPACEWAR hackers, and John McCarthy's original crew of LISPers. These include verb doubling, soundalike slang, the `-P' convention, overgeneralization, spoken inarticulations, and anthropomorphization. Each is discussed below. We also cover the standard comparatives for design quality.

Of these six, verb doubling, overgeneralization, anthropomorphization, and (especially) spoken inarticulations have become quite general; but soundalike slang is still largely confined to MIT and other large universities, and the `-P' convention is found only where LISPers flourish.

Node:Verb Doubling, Next:[78]Soundalike Slang, Up:[79]Jargon Construction

Verb Doubling

A standard construction in English is to double a verb and use it as an exclamation, such as "Bang, bang!" or "Quack, quack!". Most of these are names for noises. Hackers also double verbs as a concise, sometimes sarcastic comment on what the implied subject does. Also, a doubled verb is often used to terminate a conversation, in the process remarking on the current state of affairs or what the speaker intends to do next. Typical examples involve [80]win, [81]lose, [82]hack, [83]flame, [84]barf, [85]chomp:

"The disk heads just crashed." "Lose, lose."

"Mostly he talked about his latest crock. Flame, flame."
"Boy, what a bagbiter! Chomp, chomp!"

Some verb-doubled constructions have special meanings not immediately obvious from the verb. These have their own listings in the lexicon.

The [86]Usenet culture has one tripling convention unrelated to this; the names of `joke' topic groups often have a tripled last element. The first and paradigmatic example was alt.swedish.chef.bork.bork.bork (a "Muppet Show" reference); other infamous examples have included:

alt.french.captain.borg.borg.borg
alt.wesley.crusher.die.die.die
comp.unix.internals.system.calls.brk.brk.brk
sci.physics.edward.teller.boom.boom.boom
alt.sadistic.dentists.drill.drill.drill

Node:Soundalike Slang, Next:[87]The -P convention, Previous:[88]Verb Doubling, Up:[89]Jargon Construction

Soundalike slang

Hackers will often make rhymes or puns in order to convert an ordinary word or phrase into something more interesting. It is considered particularly [90]flavorful if the phrase is bent so as to include some other jargon word; thus the computer hobbyist magazine "Dr. Dobb's Journal" is almost always referred to among hackers as `Dr. Frob's Journal' or simply `Dr. Frob's'. Terms of this kind that have been in fairly wide use include names for newspapers:

Boston Herald => Horrid (or Harried)
Boston Globe => Boston Glob
Houston (or San Francisco) Chronicle
=> the Crocknicle (or the Comical)
New York Times => New York Slime
Wall Street Journal => Wall Street Urinal

However, terms like these are often made up on the spur of the moment. Standard examples include:

Data General => Dirty Genitals
IBM 360 => IBM Three-Sickly
Government Property --- Do Not Duplicate (on keys)
=> Government Duplicity --- Do Not Propagate
for historical reasons => for hysterical raisins
Margaret Jacks Hall (the CS building at Stanford)
=> Marginal Hacks Hall
Microsoft => Microsloth
Internet Explorer => Internet Exploiter

This is not really similar to the Cockney rhyming slang it has been compared to in the past, because Cockney substitutions are opaque whereas hacker punning jargon is intentionally transparent.

Node:The -P convention, Next:[91]Overgeneralization, Previous:[92]Soundalike Slang, Up:[93]Jargon Construction

The `-P' convention

Turning a word into a question by appending the syllable `P'; from the LISP convention of appending the letter `P' to denote a predicate (a boolean-valued function). The question should expect a yes/no answer, though it needn't. (See [94]T and [95]NIL.)
At dinnertime:

Q: ``Foodp?''

A: ``Yeah, I'm pretty hungry.'' or ``T!''

At any time:

Q: ``State-of-the-world-P?''

A: (Straight) ``I'm about to go home.''

A: (Humorous) ``Yes, the world has a state.''

On the phone to Florida:

Q: ``State-p Florida?''

A: ``Been reading JARGON.TXT again, eh?''

[One of the best of these is a [96]Gosperism. Once, when we were at a Chinese restaurant, Bill Gosper wanted to know whether someone would like to share with him a two-person-sized bowl of soup. His inquiry was: "Split-p soup?" -- GLS]

Node:Overgeneralization, Next:[97]Spoken Inarticulations, Previous:[98]The -P convention, Up:[99]Jargon Construction

Overgeneralization

A very conspicuous feature of jargon is the frequency with which techspeak items such as names of program tools, command language primitives, and even assembler opcodes are applied to contexts outside of computing wherever hackers find amusing analogies to them. Thus (to cite one of the best-known examples) Unix hackers often [100]grep for things rather than searching for them. Many of the lexicon entries are generalizations of exactly this kind.

Hackers enjoy overgeneralization on the grammatical level as well. Many hackers love to take various words and add the wrong endings to them to make nouns and verbs, often by extending a standard rule to nonuniform cases (or vice versa). For example, because

porous => porosity
generous => generosity

hackers happily generalize:

mysterious => mysteriosity
ferrous => ferrosity
obvious => obviosity
dubious => dubiosity

Another class of common construction uses the suffix `-itude' to abstract a quality from just about any adjective or noun. This usage arises especially in cases where mainstream English would perform the same abstraction through `-iness' or `-ingness'. Thus:

win => winnitude (a common exclamation)
loss => lossitude
cruft => cruftitude
lame => lameitude

Some hackers cheerfully reverse this transformation; they argue, for example, that the horizontal degree lines on a globe ought to be called `lats' -- after all, they're measuring latitude!

Also, note that all nouns can be verbed. E.g.: "All nouns can be verbed", "I'll mouse it up", "Hang on while I clipboard it over", "I'm grepping the files". English as a whole is already heading in this direction (towards pure-positional grammar like Chinese); hackers are simply a bit ahead of the curve.

The suffix "-full" can also be applied in generalized and fanciful ways, as in "As soon as you have more than one cachefull of data, the system starts thrashing," or "As soon as I have more than one headfull of ideas, I start writing it all down." A common use is "screenfull", meaning the amount of text that will fit on one screen, usually in text mode where you have no choice as to character size. Another common form is "bufferfull".

However, hackers avoid the unimaginative verb-making techniques characteristic of marketroids, bean-counters, and the Pentagon; a hacker would never, for example, `productize', `prioritize', or `securitize' things. Hackers have a strong aversion to bureaucratic bafflegab and regard those who use it with contempt.

Similarly, all verbs can be nouned. This is only a slight overgeneralization in modern English; in hackish, however, it is good form to mark them in some standard nonstandard way. Thus:

win => winnitude, winnage
disgust => disgustitude
hack => hackification

Further, note the prevalence of certain kinds of nonstandard plural forms. Some of these go back quite a ways; the TMRC Dictionary includes an entry which implies that the plural of `mouse' is [101]meeces, and notes that the defined plural of `caboose' is `cabeese'. This latter has apparently been standard (or at least a standard joke) among railfans (railroad enthusiasts) for many years.

On a similarly Anglo-Saxon note, almost anything ending in `x' may form plurals in `-xen' (see [102]VAXen and [103]boxen in the main text). Even words ending in phonetic /k/ alone are sometimes treated this way; e.g., `soxen' for a bunch of socks. Other funny plurals are `frobbotzim' for the plural of `frobbozz' (see [104]frobnitz) and `Unices' and `Twenices' (rather than `Unixes' and `Twenexes'; see [105]Unix, [106]TWENEX in main text). But note that `Twenexen' was never used, and `Unixen' was not sighted in the wild until the year 2000, thirty years after it might logically have come into use; it has

been suggested that this is because `-ix' and `-ex' are Latin singular endings that attract a Latinate plural. Finally, it has been suggested to general approval that the plural of `mongoose' ought to be `polygoose'.

The pattern here, as with other hackish grammatical quirks, is generalization of an inflectional rule that in English is either an import or a fossil (such as the Hebrew plural ending `-im', or the Anglo-Saxon plural suffix `-en') to cases where it isn't normally considered to apply.

This is not `poor grammar', as hackers are generally quite well aware of what they are doing when they distort the language. It is grammatical creativity, a form of playfulness. It is done not to impress but to amuse, and never at the expense of clarity.

Node:Spoken Inarticulations, Next:[107]Anthropomorphization, Previous:[108]Overgeneralization, Up:[109]Jargon Construction

Spoken inarticulations

Words such as `mumble', `sigh', and `groan' are spoken in places where their referent might more naturally be used. It has been suggested that this usage derives from the impossibility of representing such noises on a comm link or in electronic mail, MUDs, and IRC channels (interestingly, the same sorts of constructions have been showing up with increasing frequency in comic strips). Another expression sometimes heard is "Complain!", meaning "I have a complaint!"

Node:Anthropomorphization, Next:[110]Comparatives, Previous:[111]Spoken Inarticulations, Up:[112]Jargon Construction

Anthropomorphization

Semantically, one rich source of jargon constructions is the hackish tendency to anthropomorphize hardware and software. English purists and academic computer scientists frequently look down on others for anthropomorphizing hardware and software, considering this sort of behavior to be characteristic of naive misunderstanding. But most hackers anthropomorphize freely, frequently describing program behavior in terms of wants and desires.

Thus it is common to hear hardware or software talked about as though it has homunculi talking to each other inside it, with intentions and desires. Thus, one hears "The protocol handler got confused", or that programs "are trying" to do things, or one may say of a routine that "its goal in life is to X". One even hears explanations like "... and its poor little brain couldn't understand X, and it died." Sometimes modelling things this way actually seems to make them easier to understand, perhaps because it's instinctively natural to think of anything with a really complex behavioral repertoire as `like a person' rather than `like a thing'.

At first glance, to anyone who understands how these programs actually work, this seems like an absurdity. As hackers are among the people who know best how these phenomena work, it seems odd that they would use language that seemds to ascribe conciousness to them. The mind-set

behind this tendency thus demands examination.

The key to understanding this kind of usage is that it isn't done in a naive way; hackers don't personalize their stuff in the sense of feeling empathy with it, nor do they mystically believe that the things they work on every day are `alive'. To the contrary: hackers who anthropomorphize are expressing not a vitalistic view of program behavior but a mechanistic view of human behavior.

Almost all hackers subscribe to the mechanistic, materialistic ontology of science (this is in practice true even of most of the minority with contrary religious theories). In this view, people are biological machines - consciousness is an interesting and valuable epiphenomenon, but mind is implemented in machinery which is not fundamentally different in information-processing capacity from computers.

Hackers tend to take this a step further and argue that the difference between a substrate of CHON atoms and water and a substrate of silicon and metal is a relatively unimportant one; what matters, what makes a thing `alive', is information and richness of pattern. This is animism from the flip side; it implies that humans and computers and dolphins and rocks are all machines exhibiting a continuum of modes of `consciousness' according to their information-processing capacity.

Because hackers accept a that a human machine can have intentions, it is therefore easy for them to ascribe consciousness and intention to complex patterned systems such as computers. If consciousness is mechanical, it is neither more or less absurd to say that "The program wants to go into an infinite loop" than it is to say that "I want to go eat some chocolate" - and even defensible to say that "The stone, once dropped, wants to move towards the center of the earth".

This viewpoint has respectable company in academic philosophy. Daniel Dennett organizes explanations of behavior using three stances: the "physical stance" (thing-to-be-explained as a physical object), the "design stance" (thing-to-be-explained as an artifact), and the "intentional stance" (thing-to-be-explained as an agent with desires and intentions). Which stances are appropriate is a matter not of truth but of utility. Hackers typically view simple programs from the design stance, but more complex ones are modelled using the intentional stance.

Node:Comparatives, Previous:[113]Anthropomorphization, Up:[114]Jargon Construction

Comparatives

Finally, note that many words in hacker jargon have to be understood as members of sets of comparatives. This is especially true of the adjectives and nouns used to describe the beauty and functional quality of code. Here is an approximately correct spectrum:

```
monstrosity brain-damage screw bug lose misfeature
crock kluge hack win feature elegance perfection
```

The last is spoken of as a mythical absolute, approximated but never actually attained. Another similar scale is used for describing the

reliability of software:

broken flaky dodgy fragile brittle
solid robust bulletproof armor-plated

Note, however, that `dodgy' is primarily Commonwealth Hackish (it is rare in the U.S.) and may change places with `flaky' for some speakers.

Coinages for describing [115]lossage seem to call forth the very finest in hackish linguistic inventiveness; it has been truly said that hackers have even more words for equipment failures than Yiddish has for obnoxious people.

Node:Hacker Writing Style, Next:[116]Email Quotes, Previous:[117]Jargon Construction, Up:[118]Top

Hacker Writing Style

We've already seen that hackers often coin jargon by overgeneralizing grammatical rules. This is one aspect of a more general fondness for form-versus-content language jokes that shows up particularly in hackish writing. One correspondent reports that he consistently misspells `wrong' as `worng'. Others have been known to criticize glitches in Jargon File drafts by observing (in the mode of Douglas Hofstadter) "This sentence no verb", or "Too repetetetive", or "Bad speling", or "Incorrectspa cing." Similarly, intentional spoonerisms are often made of phrases relating to confusion or things that are confusing; `dain bramage' for `brain damage' is perhaps the most common (similarly, a hacker would be likely to write "Excuse me, I'm cixelsyd today", rather than "I'm dyslexic today"). This sort of thing is quite common and is enjoyed by all concerned.

Hackers tend to use quotes as balanced delimiters like parentheses, much to the dismay of American editors. Thus, if "Jim is going" is a phrase, and so are "Bill runs" and "Spock groks", then hackers generally prefer to write: "Jim is going", "Bill runs", and "Spock groks". This is incorrect according to standard American usage (which would put the continuation commas and the final period inside the string quotes); however, it is counter-intuitive to hackers to mutilate literal strings with characters that don't belong in them. Given the sorts of examples that can come up in discussions of programming, American-style quoting can even be grossly misleading. When communicating command lines or small pieces of code, extra characters can be a real pain in the neck.

Consider, for example, a sentence in a [119]vi tutorial that looks like this:

Then delete a line from the file by typing "dd".

Standard usage would make this

Then delete a line from the file by typing "dd."

but that would be very bad -- because the reader would be prone to type the string d-d-dot, and it happens that in vi(1) dot repeats the last command accepted. The net result would be to delete two lines!

The Jargon File follows hackish usage throughout.

Interestingly, a similar style is now preferred practice in Great Britain, though the older style (which became established for typographical reasons having to do with the aesthetics of comma and quotes in typeset text) is still accepted there. "Hart's Rules" and the "Oxford Dictionary for Writers and Editors" call the hacker-like style `new' or `logical' quoting. This returns British English to the style Latin languages (including Spanish, French, Italian, Catalan) have been using all along.

Another hacker habit is a tendency to distinguish between `scare' quotes and `speech' quotes; that is, to use British-style single quotes for marking and reserve American-style double quotes for actual reports of speech or text included from elsewhere. Interestingly, some authorities describe this as correct general usage, but mainstream American English has gone to using double-quotes indiscriminately enough that hacker usage appears marked [and, in fact, I thought this was a personal quirk of mine until I checked with Usenet --ESR]. One further permutation that is definitely not standard is a hackish tendency to do marking quotes by using apostrophes (single quotes) in pairs; that is, 'like this'. This is modelled on string and character literal syntax in some programming languages (reinforced by the fact that many character-only terminals display the apostrophe in typewriter style, as a vertical single quote).

One quirk that shows up frequently in the [120]email style of Unix hackers in particular is a tendency for some things that are normally all-lowercase (including usernames and the names of commands and C routines) to remain uncapitalized even when they occur at the beginning of sentences. It is clear that, for many hackers, the case of such identifiers becomes a part of their internal representation (the `spelling') and cannot be overridden without mental effort (an appropriate reflex because Unix and C both distinguish cases and confusing them can lead to [121]lossage). A way of escaping this dilemma is simply to avoid using these constructions at the beginning of sentences.

There seems to be a meta-rule behind these nonstandard hackerisms to the effect that precision of expression is more important than conformance to traditional rules; where the latter create ambiguity or lose information they can be discarded without a second thought. It is notable in this respect that other hackish inventions (for example, in vocabulary) also tend to carry very precise shades of meaning even when constructed to appear slangy and loose. In fact, to a hacker, the contrast between `loose' form and `tight' content in jargon is a substantial part of its humor!

Hackers have also developed a number of punctuation and emphasis conventions adapted to single-font all-ASCII communications links, and these are occasionally carried over into written documents even when normal means of font changes, underlining, and the like are available.

One of these is that TEXT IN ALL CAPS IS INTERPRETED AS `LOUD', and this becomes such an ingrained synesthetic reflex that a person who goes to caps-lock while in [122]talk mode may be asked to "stop shouting, please, you're hurting my ears!".

Also, it is common to use bracketing with unusual characters to signify emphasis. The asterisk is most common, as in "What the *hell*?" even though this interferes with the common use of the asterisk suffix as a footnote mark. The underscore is also common, suggesting underlining (this is particularly common with book titles; for example, "It is often alleged that Joe Haldeman wrote _The_Forever_War_ as a rebuttal to Robert Heinlein's earlier novel of the future military, _Starship_Troopers_."). Other forms exemplified by "=hell=", "\hell/", or "/hell/" are occasionally seen (it's claimed that in the last example the first slash pushes the letters over to the right to make them italic, and the second keeps them from falling over). On FidoNet, you might see #bright# and ^dark^ text, which was actually interpreted by some reader software. Finally, words may also be emphasized L I K E T H I S, or by a series of carets (^) under them on the next line of the text.

There is a semantic difference between *emphasis like this* (which emphasizes the phrase as a whole), and *emphasis* *like* *this* (which suggests the writer speaking very slowly and distinctly, as if to a very young child or a mentally impaired person). Bracketing a word with the `*' character may also indicate that the writer wishes readers to consider that an action is taking place or that a sound is being made. Examples: *bang*, *hic*, *ring*, *grin*, *kick*, *stomp*, *mumble*.

One might also see the above sound effects as <bang>, <hic>, <ring>, <grin>, <kick>, <stomp>, <mumble>. This use of angle brackets to mark their contents originally derives from conventions used in [123]BNF, but since about 1993 it has been reinforced by the HTML markup used on the World Wide Web.

Angle-bracket enclosure is also used to indicate that a term stands for some [124]random member of a larger class (this is straight from [125]BNF). Examples like the following are common:

So this <ethnic> walks into a bar one day...

There is also an accepted convention for `writing under erasure'; the text

Be nice to this fool^H^H^H^Hgentleman,
he's visiting from corporate HQ.

reads roughly as "Be nice to this fool, er, gentleman...", with irony emphasized. The digraph ^H is often used as a print representation for a backspace, and was actually very visible on old-style printing terminals. As the text was being composed the characters would be echoed and printed immediately, and when a correction was made the backspace keystrokes would be echoed with the string '^H'. Of course, the final composed text would have no trace of the backspace characters (or the original erroneous text).

This convention parallels (and may have been influenced by) the ironic use of `slashouts' in science-fiction fanzines.

A related habit uses editor commands to signify corrections to previous text. This custom faded in email as more mailers got good editing capabilities, only to tale on new life on IRCs and other line-based chat systems.

I've seen that term used on alt.foobar often.
Send it to Erik for the File.

Oops...s/Erik/Eric/.

The s/Erik/Eric/ says "change Erik to Eric in the preceding". This syntax is borrowed from the Unix editing tools ed and sed, but is widely recognized by non-Unix hackers as well.

In a formula, * signifies multiplication but two asterisks in a row are a shorthand for exponentiation (this derives from FORTRAN). Thus, one might write 2 ** 8 = 256.

Another notation for exponentiation one sees more frequently uses the caret (^, ASCII 1011110); one might write instead 2^8 = 256. This goes all the way back to Algol-60, which used the archaic ASCII `up-arrow' that later became the caret; this was picked up by Kemeny and Kurtz's original BASIC, which in turn influenced the design of the bc(1) and dc(1) Unix tools, which have probably done most to reinforce the convention on Usenet. (TeX math mode also uses ^ for exponention.) The notation is mildly confusing to C programmers, because ^ means bitwise exclusive-or in C. Despite this, it was favored 3:1 over ** in a late-1990 snapshot of Usenet. It is used consistently in this lexicon.

In on-line exchanges, hackers tend to use decimal forms or improper fractions (`3.5' or `7/2') rather than `typewriter style' mixed fractions (`3-1/2'). The major motive here is probably that the former are more readable in a monospaced font, together with a desire to avoid the risk that the latter might be read as `three minus one-half'. The decimal form is definitely preferred for fractions with a terminating decimal representation; there may be some cultural influence here from the high status of scientific notation.

Another on-line convention, used especially for very large or very small numbers, is taken from C (which derived it from FORTRAN). This is a form of `scientific notation' using `e' to replace `*10^'; for example, one year is about 3e7 seconds long.

The tilde (~) is commonly used in a quantifying sense of `approximately'; that is, ~50 means `about fifty'.

On Usenet and in the [126]MUD world, common C boolean, logical, and relational operators such as |, &, ||, &&, !, ==, !=, >, <, >=, and =< are often combined with English. The Pascal not-equals, <>, is also recognized, and occasionally one sees /= for not-equals (from Ada, Common Lisp, and Fortran 90). The use of prefix `!' as a loose synonym for `not-' or `no-' is particularly common; thus, `!clue' is read `no-clue' or `clueless'.

A related practice borrows syntax from preferred programming languages to express ideas in a natural-language text. For example, one might see the following:

In <jrh578689@thudpucker.com> J. R. Hacker wrote:
>I recently had occasion to field-test the Snafu
>Systems 2300E adaptive gonkulator. The price was
>right, and the racing stripe on the case looked
>kind of neat, but its performance left something
>to be desired.

Yeah, I tried one out too.

#ifdef FLAME

Hasn't anyone told those idiots that you can't get
decent bogon suppression with AFJ filters at today's
net volumes?
#endif /* FLAME */

I guess they figured the price premium for true
frame-based semantic analysis was too high.
Unfortunately, it's also the only workable approach.
I wouldn't recommend purchase of this product unless
you're on a *very* tight budget.

#include <disclaimer.h>
--
== Frank Foonly (Fubarco Systems)

In the above, the #ifdef/#endif pair is a conditional compilation syntax from C; here, it implies that the text between (which is a [127]flame) should be evaluated only if you have turned on (or defined on) the switch FLAME. The #include at the end is C for "include standard disclaimer here"; the `standard disclaimer' is understood to read, roughly, "These are my personal opinions and not to be construed as the official position of my employer."

The top section in the example, with > at the left margin, is an example of an inclusion convention we'll discuss below.

More recently, following on the huge popularity of the World Wide Web, pseudo-HTML markup has become popular for similar purposes:

<flame>
Your father was a hamster and your mother smelt of elderberries!
</flame>

You'll even see this with an HTML-style modifier:

<flame intensity="100%">
You seem well-suited for a career in government.
</flame>

Another recent (late 1990s) construction now common on USENET seems to be borrowed from Perl. It consists of using a dollar sign before an uppercased form of a word or acronym to suggest any [128]random member of the class indicated by the word. Thus: `$PHB' means "any random member of the class `Pointy-Haired Boss'".

Hackers also mix letters and numbers more freely than in mainstream usage. In particular, it is good hackish style to write a digit sequence where you intend the reader to understand the text string that names that number in English. So, hackers prefer to write `1970s' rather than `nineteen-seventies' or `1970's' (the latter looks like a possessive).

It should also be noted that hackers exhibit much less reluctance to use multiply-nested parentheses than is normal in English. Part of this is almost certainly due to influence from LISP (which uses deeply nested parentheses (like this (see?)) in its syntax a lot), but it has also been suggested that a more basic hacker trait of enjoying playing with complexity and pushing systems to their limits is in operation.

Finally, it is worth mentioning that many studies of on-line communication have shown that electronic links have a de-inhibiting

defense of the former Superintendent of Finances. He was rewarded for his loyalty with five years in the Bastile. He subsequently regained the royal favor, and became the Historiographer Royal.

Richelieu: (1585-1642) Although he does not appear in The Vicomte de Bragelonne, Armand-Jean du Plessis, Duc de Richelieu, is mentioned several times. He was an admirer of Machiavelli and, under the reign of Louis XIII, he became the most powerful man in France. He greatly strengthened France both at home and abroad, and named Mazarin as his successor shortly before his death. In The Three Musketeers, it is he who lays the snare for Anne of Austria involving the famous diamond studs given to the Duke of Buckingham. D'Artagnan and his three friends rescue the queen from this embarrassing predicament.

Saint-Aignan: (1610-87) Francois de Beauvillier, the Comte de Saint-Aignan, was a former governor of the Touraine. He finally realized his ambition, mentioned by Dumas, of joining the French Academy in 1663. Before becoming First Gentleman to the King's Bedchamber, he was part of Gaston d'Orleans's military household. Though quite a few years Louis XIV's senior, he became the young king's chief purveyor of pleasures.

Saint-Remy: Francoise le Prevot de la Coutelaye became Madame de Saint-Remy following her third marriage. Her first was to a man named Besnard, a councilor of the Parliament at Rennes. Her second marriage was to Laurent de la Baume le Blanc, lord of the manor of La Valliere. He was Louise de la Valliere's father. Laurent died in 1651, and in 1655 she married Jacques Couravel, Marquis de Saint-Remy, First Chamberlain to Gaston d'Orleans. After Gaston's death, they both moved to Paris.

Treville: (1598-1672) Arnaud-Jean du Peyrer, Comte de Troisvilles (written and pronounced Treville) does not appear in The Vicomte de Bragelonne, but he was D'Artagnan's (both the real and fictional) predecessor as Captain of the Musketeers. He was a career soldier and, like D'Artagnan, a Gascon. He was appointed Captain-Lieutenant of the Musketeers in 1634 (the rank of Captain-General was reserved for the king), and was exiled in 1642 for opposing Richelieu. Mazarin disbanded the musketeers in 1646 (an historical fact ignored by Dumas), and Treville retired to Foix as its governor. In The Three Musketeers (which adds about 10 years to the ages of the historical counterparts), it was in Treville's office that the first meeting between D'Artagnan, Athos, Porthos, and Aramis occurred.

Vanel: (1644-1703) Anne-Marguerite Vanel was the daughter of Claude Vanel (a magistrate in the Paris Parliament) and became the wife of Jean Coiffer (a member of the Royal Audit Office) in 1654. Contemporaries described her as a "dainty and extremely pretty young woman with a lively and very witty turn of mind." She was Fouquet's mistress during the 1650s, and later transferred her affections to Colbert. Her high spirits annoyed Colbert, and he passed her off to his brother.

Wardes: (1620-88) Francois-Rene Crespin du Bec was the Marquis de Vardes, and a noted schemer and bold liar. Some women, though, including Madame de Motteville, found him charming. Dumas creates two characters out of the historical De Vardes. The father plays a prominent part in The Three Musketeers and Twenty Years After, and the son in The Vicomte de Bragelonne, though they were, in reality, the same man. He was named Governor of Aigues-Mortes in 1660 and was banished there a few years later following a court scandal. Although a favorite of Louis XIV, he got entangled in a plot by Olympe Mancini (then the Comtesse de Soissons)

to avenge her sister, Marie, whom the king had abandoned in favor of his political marriage to Maria-Theresa of Spain. He remained in Aigues-Mortes for 17 years.

Much of the information for these biographies was taken from the David Coward's editions of the D'Artagnan Romances, published by Oxford World's Classics. Additional material came from the Fireblade Coffeehouse's web page on Alexandre Dumas at www.hoboes.com/html/FireBlade/Dumas/. The quote from Robert Louis Stevenson comes from his A Gossip on a Novel of Dumas's from Memories and Portraits.

End of Project Gutenberg Project Gutenberg Dumas Commentary, by John Bursey