Part I
Context

1

Philosophy: Philosophy Matters

> Those who do not understand Unix are condemned to reinvent it, poorly.
>
> *Usenet signature, November 1987*
> —Henry Spencer

1.1 Culture? What Culture?

This is a book about Unix programming, but in it we're going to toss around the words 'culture', 'art', and 'philosophy' a lot. If you are not a programmer, or you are a programmer who has had little contact with the Unix world, this may seem strange. Senior engineers develop huge bodies of implicit knowledge, which they pass to their juniors by (as Zen Buddhists put it) "a special transmission, outside the scriptures".

> Robert Metcalf [the inventor of Ethernet] says that if something comes along to replace Ethernet, it will be called "Ethernet", so therefore Ethernet will never die[1]. Unix has already undergone several such transformations.
>
> —Ken Thompson

[1] In fact, Ethernet has already been replaced by a different technology with the same name—twice. Once when coax was replaced with twisted pair, and a a second time when gigabit Ethernet came in.

Table 1.1: Regular-expression examples.

Regexp	*Matches*
`"x.y"`	`x` followed by any character followed by `y`.
`"x\.y"`	`x` followed by a literal period followed by `y`.
`"xz?y"`	`x` followed by at most one `z` followed by `y`; thus, `"xy"` or `"xzy"` but not `"xz"` or `"xdy"`.
`"xz*y"`	`x` followed by any number of instances of `z`, followed by `y`; thus, `"xy"` or `"xzy"` or `"xzzzy"` but not `"xz"` or `"xdy"`.
`"xz+y"`	`x` followed by one or more instances of `z`, followed by `y`; thus, `"xzy"` or `"xzzy"` but not `"xy"` or `"xz"` or `"xdy"`.

Outsiders have frequently dismissed Unix as an academic toy or a hacker's sandbox. One well-known polemic, the *Unix Hater's Handbook* [Garfinkel], follows an antagonistic line nearly as old as Unix itself in writing its devotees off as a cult religion of freaks and losers.

But the cost of this approach is that when the user *can* set policy, the user *must* set policy. Non-technical end-users frequently find Unix's profusion of options and interface styles overwhelming and retreat to systems that at least pretend to offer them simplicity.

First, we'll look at an example of transparency in UI design. It is *audacity*, an open-source editor for sound files that runs on Unix systems, Mac OS X, and Windows. Sources, downloadable binaries, documentation, and screen shots are available at the project site (`http://audacity.sourceforge.net/`).

```
#SNG: This is a synthetic SNG test file

# Our first test is a paletted (type 3) image.
IHDR: {
 width: 16;
 height: 19;
 bitdepth: 8;
 using color: palette;
 with interlace;
}

# Sample bit depth chunk
sBIT: {
  red: 8;
  green: 8;
  blue: 8;
}
```

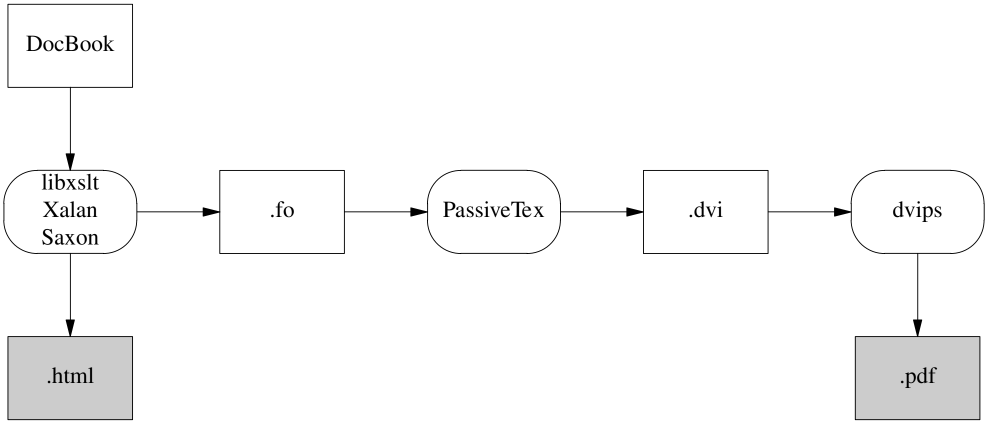


Figure 1.1: Present-day XML-DocBook toolchain.

In the short term, Unix's laissez-faire approach may lose it a good many nontechnical users. In the long term, however, it may turn out that this 'mistake' confers a critical advantage—because policy tends to have a short lifetime, mechanism a long one. Today's fashion in interface look-and-feel too often becomes tomorrow's evolutionary dead end (as people using obsolete X toolkits will tell you with some feeling!). So the flip side of the flip side is that the "mechanism, not policy" philosophy may enable Unix to renew its relevance long after competitors more tied to one set of policy or interface choices have faded from view[2].

1.1.1 Case Study: `.newsrc` Format

There are not many operating systems that anyone has ever described as 'fun'. Indeed, the friction and labor of development under most other environments has been aptly compared to kicking a dead whale down the beach[3].

[2] Jim Gettys, one of the architects of X (and a contributor to this book), has meditated in depth on how X's laissez-faire style might be productively carried forward in *The Two-Edged Sword* [Gettys]. This essay is well worth reading, both for its specific proposals and for its expression of the the Unix mindset.

[3] This was originally said of the IBM MVS TSO facility by Stephen C. Johnson, perhaps better known as the author of *yacc*.

> (i) Make each program do one thing well. To do a new job, build afresh rather than complicate old programs by adding new features.
>
> (ii) Expect the output of every program to become the input to another, as yet unknown, program. Don't clutter output with extraneous information. Avoid stringently columnar or binary input formats. Don't insist on interactive input.
>
> (iii) Fancy algorithms are slow when `n` is small, and `n` is usually small. Fancy algorithms have big constants. Until you know that `n` is frequently going to be big, don't get fancy. (Even if `n` does get big, use Rule 2 first.)

More of the Unix philosophy was implied not by what these elders said but by what they did and the example Unix itself set. Looking at the whole, we can abstract the following ideas:

1. Rule of Modularity: Write simple parts connected by clean interfaces.
2. Rule of Clarity: Clarity is better than cleverness.
3. Rule of Composition: Design programs to be connected to other programs.

All the philosophy really boils down to one iron law, the hallowed 'KISS principle' of master engineers everywhere:

K.I.S.S.

Keep It Simple, Stupid!

Almost all Usenet news readers understand the `.newsrc` file, which records which Usenet messages have been seen by the calling user. Though it is named like a run-control file, it is not only read at startup but typically updated at the end of the newsreader run. The `.newsrc` format has been fixed since the first newsreaders around 1980. Example 5.2 is a representative section from a `.newsrc` file.

- Everything that can be a source- and destination-independent filter *should* be one.

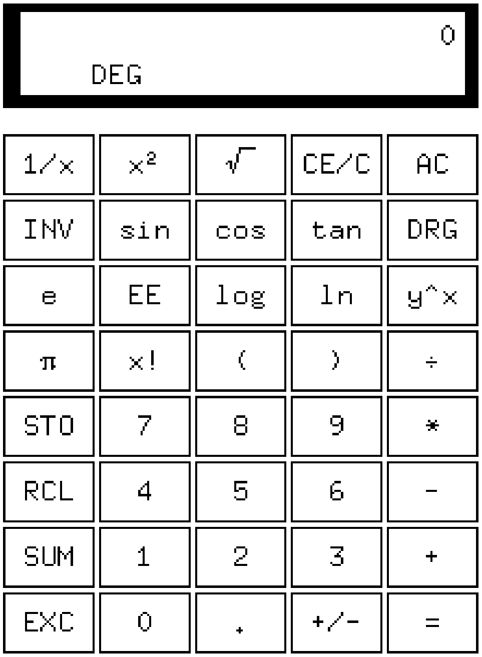


Figure 1.2: The xcalc GUI.

- Data streams should if at all possible be textual (so they can be viewed and filtered with standard tools).
- Database layouts and application protocols should if at all possible be textual (human-readable and human-editable).

1.2 Separating Complexity Control from Performance Tuning

First, though, we need to dispose of a few red herrings. Our discussion is *not* going to be about using concurrency to improve performance. Putting that concern before developing a clean architecture that minimizes global complexity is premature optimization, the root of all evil (see Chapter 12 for further discussion).

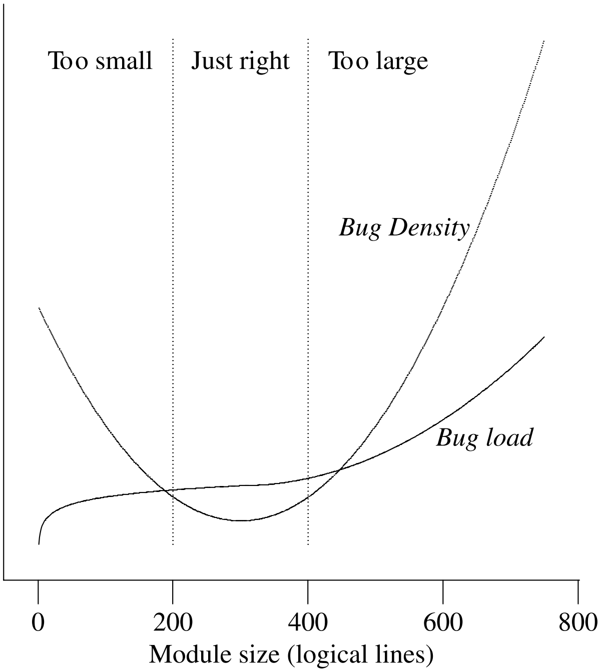


Figure 1.3: Qualitative plot of defect count and density vs. module size.

> One favorite pipeline was "**`bc | speak`**" — a talking desk calculator. It knew number names up to a vigintillion.
>
> —Doug McIlroy

1.2.1 *yacc* and *lex*

On many operating systems, the per-user data required to validate logins and start a user's session is an opaque binary database. Under Unix, by contrast, it's a text file with records one per line and colon-separated fields.

Example 5.7 is an example transaction in SMTP (Simple Mail Transfer Protocol), which is described by RFC 2821. In the example below, *C:* lines are sent by a mail transport agent (MTA) sending mail, and *S:* lines are returned by the MTA receiving it. Text *emphasized like this* is comments, not part of the actual transaction.

This program supports cutting, pasting, and editing of audio samples. It supports multitrack editing and mixing. The UI is superbly simple; the sound waveforms are shown in the *audacity* window. The image of the waveform can be cut and pasted;

Example 1.1: An SMTP session example.

```
C: <client connects to service port 25>
C: HELO snark.thyrsus.com                  sending host identifies self
S: 250 OK Hello snark, glad to meet you receiver acknowledges
C: MAIL FROM: <esr@thyrsus.com>            identify sending user
S: 250 <esr@thyrsus.com>... Sender ok      receiver acknowledges
C: RCPT TO: cor@cpmy.com                   identify target user
```

operations on that image are directly reflected in the audio sample as soon as they are performed.

In syntactic style, SNG resembles CSS (Cascading Style Sheets), another language for controlling presentation of graphics; this makes at least a gesture in the direction of the Rule of Least Surprise. Here is a test example.

1.2.2 Standard Open-Source Licenses

Here are the standard open-source license terms you are likely to encounter. The abbreviations listed here are in general use.

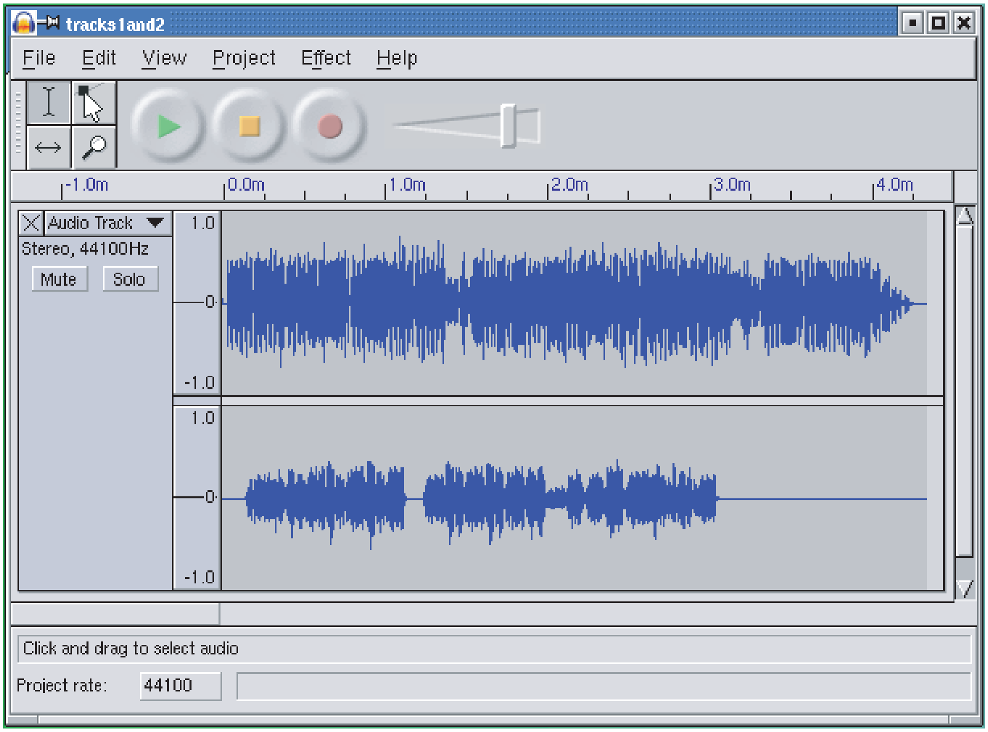
MIT (`http://www.opensource.org/licenses/mit-license.html`**)**
MIT X Consortium license (like BSD's but with no advertising requirement)

BSD (`http://www.opensource.org/licenses/bsd-license.html`**)**
University of California at Berkeley Regents copyright (used on BSD code)

Artistic License (`http://www.opensource.org/licenses/artistic-license.html`**)**
Same terms as Perl Artistic License

GPL (`http://www.gnu.org/copyleft.html`**)**
GNU General Public License

For the purposes of this chapter, the only important distinction among them is whether they are infectious or not. A license is *infectious* if it requires that any derivative work of the licensed software also be placed under its terms.



Click and drag to select audio

Project rate: 44100

Figure 1.4: Screen shot of audacity.

1.2.2.1 Case Study: FreeNet

Freenet is a peer-to-peer networking project that is intended to make censorship and content suppression impossible[4]. Freenet developers envision the following applications:

- Uncensorable dissemination of controversial information: Freenet protects freedom of speech by enabling anonymous and uncensorable publication of material ranging from grassroots alternative journalism to banned exposês.

- Efficient distribution of high-bandwidth content: Freenet's adaptive caching and mirroring is being used to distribute Debian Linux software updates.

[4] There is a Freenet project website (`http://freenetproject.org`)

Figure 1.5: The PDP-7.

- Universal personal publishing: Freenet enables anyone to have a website, without space restrictions or compulsory advertising, even if the would-be webmaster doesn't own a computer.

There are a number of minor variants of regexp notation.

1.2.2.2 Case study: *make* for Document-File Translation

In the fetchmail makefile, for example, you'll see three productions that relate files named `FAQ`, `FEATURES`, and `NOTES` to HTML sources `fetchmail-FAQ.html`, `fetchmail-features.html`, and `design-notes.html`.

Consider also Postel's Prescription[5]: "Be liberal in what you accept, and conservative in what you send." Postel was speaking of network service programs, but the underlying idea is more general.

[5] Jonathan Postel was the first editor of the Internet RFC series of standards, and one of the principal architects of the Internet. A tribute page (`http://www.postel.org/jonpostel.html`) is maintained by the Postel Center for Experimental Networking.

1.2.2.3 *Imake*

Imake was written in an attempt to mechanize makefile generation for the X window system. It builds on *makedepend* to tackle both the dependency-derivation and portability problems.

1.2.2.3.1 Case Study: PostgreSQL

Another excellent example of this pattern is the `xcdroast/cdrtools` combination. The cdrtools distribution provides a program *cdrecord*(1) with a command-line interface. The *cdrecord* code specializes in knowing everything about talking to CD-ROM hardware. *xcdroast* is a GUI; it specializes in providing a pleasant user experience. The *xcdroast*(1) program calls *cdrecord*(1) to do most of its work.

Example 1.2: An SMTP session example.

```
C: <client connects to service port 25>
C: HELO snark.thyrsus.com               sending host identifies self
S: 250 OK Hello snark, glad to meet you receiver acknowledges
C: MAIL FROM: <esr@thyrsus.com>         identify sending user
S: 250 <esr@thyrsus.com>... Sender ok   receiver acknowledges
C: RCPT TO: cor@cpmy.com                identify target user
S: 250 root... Recipient ok             receiver acknowledges
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Scratch called.  He wants to share
C: a room with us at Balticon.
C: .                                    end of multiline send
S: 250 WAA01865 Message accepted for delivery
C: QUIT                                 sender signs off
S: 221 cpmy.com closing connection      receiver disconnects
C: <client hangs up>
```

We can get mathematically specific about this. It is almost never worth doing optimizations that reduce resource use by merely a constant factor; it’s smarter to concentrate effort on cases in which you can reduce average-case running time or space

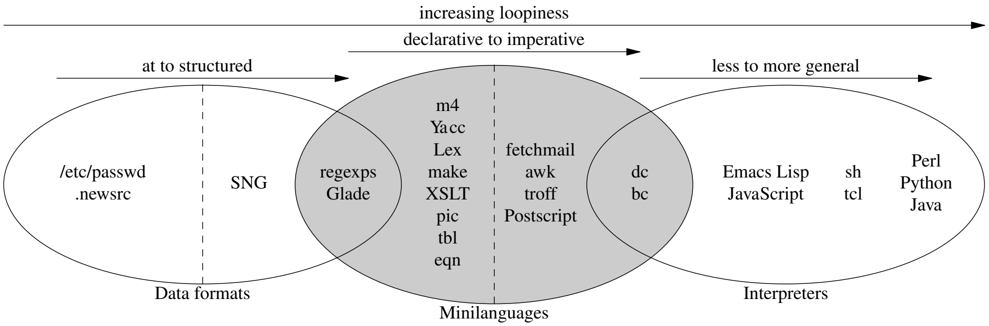


Figure 1.6: Taxonomy of languages.

use from O(n^2) to O(n) or O(n log n),[6] or similarly reduce from a higher order. Linear performance gains tend to be rapidly swamped by Moore's Law.[7]

1.2.2.3.2 Case Study: Freeciv

In Chapter 6, we introduced Freeciv as an example of transparent data formats. But more critical to the way it supports multi-player gaming is the client/server partitioning of the code. This is a representative example of a program in which the application needs to be distributed over a wide-area network and handles communication through TCP/IP sockets.

1.2.3 Case Study: *audacity*

Example 5.7 is an example transaction in SMTP (Simple Mail Transfer Protocol), which is described by RFC 2821. In the example below, *C:* lines are sent by a mail

[6] For readers unfamiliar with O notation, it is a way of indicating how the average running time of an algorithm changes with the size of its inputs. An O(1) algorithm runs in constant time. An O(n) algorithm runs in a time that is predicted by `An + C`, where `A` is some unknown constant of proportionality and `C` is an unknown constant representing setup time. Linear search of a list for a specified value is O(n). An O(n^2) algorithm runs in time `An`^{2} plus lower-order terms (which might be linear, or logarithmic, of any other function lower than a quadratic). Checking a list for duplicate values (by the naïve method, not sorting it) is O(n^2).

[7] The eighteen-month doubling time usually quoted for Moore's Law implies that you can collect a 26% performance gain just by buying new hardware in six months.

transport agent (MTA) sending mail, and *S:* lines are returned by the MTA receiving it. Text *emphasized like this* is comments, not part of the actual transaction.

Another theme that emerges from these examples is the value of programs that flip a problem out of a domain in which transparency is hard into one in which it is easy. Audacity, *sng*(1) and the *tic*(1)/*infocmp*(1) pair all have this property. The objects they manipulate are not readily conformable to the hand and eye; audio files are not visual objects, and although images expressed in PNG format are visual, the complexities of PNG annotation chunks are not. All three applications turn manipulation of their binary file formats into a problem to which human beings can more readily apply intuition and competences gained from everyday experience.

Called without arguments, *ascii* generates a usage screen that looks like this:

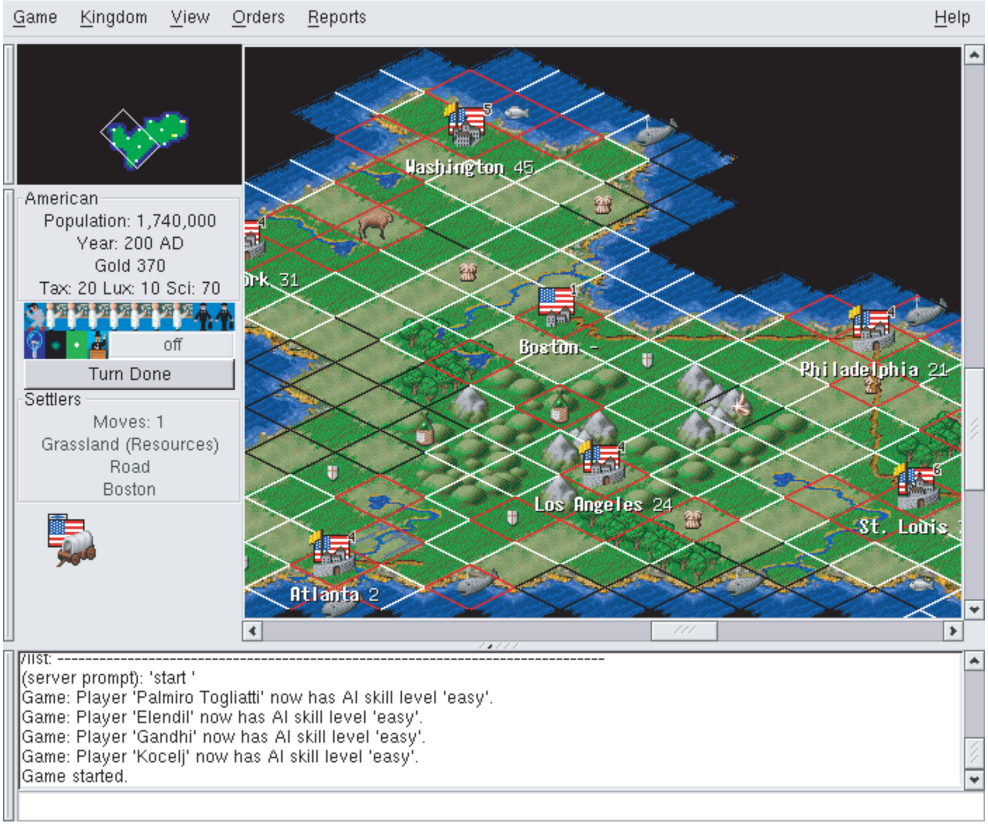
```
Usage: ascii [-dxohv] [-t] [char-alias...]
   -t = one-line output  -d = Decimal table  -o = octal table  -x = hex table
   -h = This help screen -v = version information
Prints all aliases of an ASCII character. Args may be chars, C \-escapes,
English names, ^-escapes, ASCII mnemonics, or numerics in decimal/octal/hex.

Dec Hex    Dec Hex    Dec Hex  Dec Hex  Dec Hex  Dec Hex   Dec Hex   Dec Hex
  7 07 BEL  23 17 ETB  39 27 '  55 37 7  71 47 G  87 57 W  103 67 g  119 77 w
  8 08 BS   24 18 CAN  40 28 (  56 38 8  72 48 H  88 58 X  104 68 h  120 78 x
  9 09 HT   25 19 EM   41 29 )  57 39 9  73 49 I  89 59 Y  105 69 i  121 79 y
 12 0C FF   28 1C FS   44 2C ,  60 3C <  76 4C L  92 5C \  108 6C l  124 7C |
 13 0D CR   29 1D GS   45 2D -  61 3D =  77 4D M  93 5D ]  109 6D m  125 7D }
 14 0E SO   30 1E RS   46 2E .  62 3E >  78 4E N  94 5E ^  110 6E n  126 7E ~
 15 0F SI   31 1F US   47 2F /  63 3F ?  79 4F O  95 5F _  111 6F o  127 7F DEL
```

The first Unix of which it can be said that essentially all of it would be recognizable to a modern Unix programmer was the Version 7 release in 1979.[8]

1.2.4 Case Study: *fetchmail*'s `-v` option

When `-v` is on, *fetchmail* dumps each one of its POP, IMAP, and SMTP transactions to standard output as they happen. A developer can actually see the code doing protocol with remote mailservers and the mail transport program it forwards to, in real time. Users can send session transcripts with their bug reports.

[8] The version 7 manuals can be browsed on-line at `http://plan9.bell-labs.com/7thEdMan/index.html`.



American
Population: 1,740,000
Year: 200 AD
Gold 370
Tax: 20 Lux: 10 Sci: 70

Settlers
Moves: 1
Grassland (Resources)
Road
Boston

```
/list: -----------------------------------------------------------------------
(server prompt): 'start '
Game: Player 'Palmiro Togliatti' now has AI skill level 'easy'.
Game: Player 'Elendil' now has AI skill level 'easy'.
Game: Player 'Gandhi' now has AI skill level 'easy'.
Game: Player 'Kocelj' now has AI skill level 'easy'.
Game started.
```

Figure 1.7: Main window of a Freeciv game.

1.3 Case Study: Generating Code for the *ascii* Displays

There are a number of well-known environment variables you can expect to find defined on startup of a program from the Unix shell. These (especially `HOME`) will often need to be evaluated *before* you read a local dotfile.

`USER`
Login name of the account under which this session is logged in (BSD convention).

`HOME`
Home directory of the user running this session.