**ハッカーになろう (**[**How To Become A Hacker**](http://www.catb.org/~esr/faqs/hacker-howto.html)**）**

[Valid XHTML 1.1!](https://validator.w3.org/check/referer)

著者　Eric S. Raymond　[esr@thyrsus.com](mailto:esr@thyrsus.com), [Thyrus Enterprises](http://catb.org/~esr/)  
翻訳　[山形浩生](https://cruel.org/) <hiyori13@alum.mit.edu> + 村川泰 + Takachin

**改訂履歴**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision 1.52-1.51 | 2020.01.03 | esr | 学習用言語としてGoがよくなったのでJavaを置きかえた。いくつかのリンク、USBメモリ版ディストリビューションの話。リンク更新。(翻訳改訂 2020.02.25) |
| Revision 1.43-1.50 | 2015.07.19 | esr | いくつかのリンクを追加。またfreshmeat がなくなり、いくつかリンクが古くなっていたのを修正。ハッカースペースに参加しよう！ Open Solarisがちっともオープンでなくなったので削除、UbuntuのUnityは本当にダメな代物。 |
| Revision 1.43 | 2011.02.07 | esr | 2010年には Python の人気が Perl を追い抜いた。 |
| Revision 1.42 | 2010.10.22 | esr | 「歴史的な記述」を追加。 |
| Revision 1.40 | 2008.11.03 | esr | リンクを修正. |
| Revision 1.39 | 2008.8.14 | esr | リンクを修正。 |
| Revision 1.38 | 2008.1.8 | esr | 初期に習うべき言語としてJavaの評価を下げた。 |
| Revision 1.37 | 2007.10.04 | esr | 新人向け Unix ディストリビューションとして Ubuntu を推奨。 |

**目次**

[なぜこんな文書を書いたか](https://cruel.org/freeware/hacker.html#why_this)  
[ハッカーって何？](https://cruel.org/freeware/hacker.html#what_is)  
[ハッカー的心構え](https://cruel.org/freeware/hacker.html#attitude)

1. [この世界は解決を待っている魅力的な問題でいっぱいだ](https://cruel.org/freeware/hacker.html#believe1)
2. [同じ問題を二度解くような無駄はいやだ](https://cruel.org/freeware/hacker.html#believe2)
3. [退屈と単純作業は悪](https://cruel.org/freeware/hacker.html#believe3)
4. [自由は善](https://cruel.org/freeware/hacker.html#believe4)
5. [心構えは技能の代用にはならない](https://cruel.org/freeware/hacker.html#believe5)

[基本的なハッキング技術](https://cruel.org/freeware/hacker.html#basic_skills)

1. [プログラミングを身につけること。](https://cruel.org/freeware/hacker.html#skills1)
2. [オープンソース UNIX 類のひとつを入手し、使いかたと動かしかたをおぼえること。](https://cruel.org/freeware/hacker.html#skills2)
3. [World Wide Web の使い方を学び、HTML を書くこと。](https://cruel.org/freeware/hacker.html#skills3)
4. [まともに英語ができないならば、身につけること。](https://cruel.org/freeware/hacker.html#skills4)

[ハッカー文化での地位](https://cruel.org/freeware/hacker.html#status)

1. [オープンソース・ソフトウェアを書く](https://cruel.org/freeware/hacker.html#respect1)
2. [フリーソフトウェアのテストやデバッグを手伝う](https://cruel.org/freeware/hacker.html#respect2)
3. [有益な情報を公開する](https://cruel.org/freeware/hacker.html#respect3)
4. [インフラが機能し続けるように手伝う](https://cruel.org/freeware/hacker.html#respect4)
5. [ハッカー文化そのものへの貢献](https://cruel.org/freeware/hacker.html#respect5)

[「ハッカー」と「おたく Nerd」との関係](https://cruel.org/freeware/hacker.html#nerd_connection)  
[ハッカースタイルの要点](https://cruel.org/freeware/hacker.html#style)  
[歴史的な記述：ハッキング、オープンソース、フリーソフト](https://cruel.org/freeware/hacker.html#history)  
[その他の関連文書](https://cruel.org/freeware/hacker.html#resources)  
[FAQ](https://cruel.org/freeware/hacker.html#FAQ)

glider

**なぜこんな文書を書いたか**

　わたしは [Jargon File](http://www.catb.org/jargon)の編集者で、またその他似たような有名文書いくつかの著者なので、しばしば熱心なネットワーク初心者から「ウィザード級の大ハッカーになるにはどうやって勉強すればいいの？」といったようなお尋ねメールを頂きます。でもかつて 1996 年に、こんな大事な問題を扱った FAQ や Web 文書 はみあたらないことに気がつきました。というわけで、これを書き始めました。多くのハッカーがいまやこれを決定版と見なしているし、つまり実際に決定版なんだと思います。でも、この問題について自分が唯一無二の権威だと主張するつもりもありません。気にくわなければ、自分なりのヤツをどうぞ。

この文書をオフラインで読んでいるなら、最新版は次のところにあります。 <http://www.catb.org/~esr/faqs/hacker-howto.html>

なお、この文書のさいごには「[よくきかれる質問（FAQ）](https://cruel.org/freeware/hacker.html" \l "faq)」がついています。この文書についてわたしに質問を送りつけるまえに、まずそれを――二回は――読んでください。

　本書の翻訳はいっぱいあります: [アラビア語](http://www.abdulibrahim.com/%D9%83%D9%8A%D9%81-%D8%AA%D8%B5%D8%A8%D8%AD-%D9%87%D8%A7%D9%83%D8%B1/) [ベラルーシ語](http://moneyaisle.com/worldwide/how-to-become-a-hacker-be) [ブルガリア語](http://weknowyourdreams.com/questions.html) [シナ語](http://zer4tul.github.io/docs/hacker-howto.html#hacker-howto), [チェコ語](http://scientificachievements.com/jak-se-stat-hackerem/). [デンマーク語](http://www.olemichaelsen.dk/hacker-howto.html) [オランダ語](https://nobullshitseeds.com/translate/hacken-leren/) [エストニア語](http://www.kakupesa.net/hacker/) [フランス語](http://thomasgil.com/hacker.html) [ドイツ語](http://www.linuxtaskforce.de/hacker-howto-ger.html), [ギリシャ語](https://sophron.latthi.com/hacker-howto-gr.html) [ハンガリー語](http://www.forallworld.com/hogyan-valhat-egy-hacker/), [イタリア語](http://www.victorfleur.com/documents/hacker.html) [ヘブライ語](http://he.wikisource.org/wiki/%D7%90%D7%99%D7%9A_%D7%9C%D7%94%D7%99%D7%95%D7%AA_%D7%94%D7%90%D7%A7%D7%A8), [日本語](https://cruel.org/freeware/hacker.html) [リトアニア語](http://rtfb.lt/hacker-howto-lt.html) [ノルウェー語](http://stian.atlantiscrew.net/doc/hacker-howto.html), [ペルシャ語](http://ashiyane.org/forums/showthread.php?t=20570) [ポーランド語](http://michalp.net/blog/posts/hacker-howto) [ポルトガル語 (ブラジル語)](http://jvdm.sdf.org/pt/raquer-howto), [ルーマニア語](http://garaj.xhost.ro/hacker-howto/hacker-howto.ro.htm) [スペイン語](http://www.sindominio.net/biblioweb/telematica/hacker-como.html), [トルコ語](http://www.belgeler.org/howto/hacker-howto/hacker-howto.html), [スウェーデン語](http://www1.tripnet.se/~mly/open/faqs/hacker-howto.se.html)。ちなみにこの文書はときどき変更されるので、これらの翻訳は程度の差はあれ古くなっているかもしれません。

　本文書を飾る、9 目升目に 5 つの点の図は、*グライダー*というものです。これはハッカーたちを何年にもわたって魅了してきた[ライフ](http://dmoz.org/Computers/Artificial_Life/Cellular_Automata/)ゲームという数学なシミュレーションに見られる、驚くような性質を持った簡単なパターンです。ハッカーとはどんなものかについての、いい視覚的なエンブレムだと思います――抽象的で、一見すると謎めいていますが、でも独自の入念な論理を持った大きな世界への玄関口となっているのです。グライダーエンブレムについては[ここを](http://www.catb.org/~esr/hacker-emblem/)参照してください。

　この文書が有意義だと思ったら、[Patreon](https://www.patreon.com/esr)または[SubscribeStar](https://www.subscribestar.com/esr" \t "_top)で私を支援してください。そしてあなたが使って価値が高いと考えるコードを生産した他のハッカー支援も[Loadsharers](http://www.catb.org/esr/loadsharers/" \t "_top)経由で考えてみてください。少額でも継続的な寄付はすぐに山積みとなり、労働の成果を与えてきた人々を解放してもっと価値を作りだせるようにしてあげるのです。

**ハッカーって何？**

[Jargon File](http://www.catb.org/jargon) には「ハッカー（hacker）」という言葉の定義が山ほどあがっています。そのほとんどは、技術的な熟達や。問題解決、限界を克服することといったような話と関係した内容です。でも***ハッカーになる***方法を知りたいなら、ホントに関係ある定義は２つしかありません。

　熟練プログラマやネットワークの天才たちのコミュニティないしは共有文化というものが存在しています。その歴史は初期のタイム・シェアリング・ミニコンピュータや黎明期の ARPAnet の実験にまで遡ることができます。この文化に属する人々が「ハッカー hacker」という言葉をうみだしました。ハッカーたちはインターネットを築きました。ハッカーたちが UNIX オペレーティングシステムを今日のような形にまで作りあげました。ハッカーたちが Usenet を運営し、World Wide Web が使えるようにしたんです。あなたがこの文化の一員で、それに貢献し、その文化のほかの人たちがあなたを知っていて、しかもその人たちがあなたをハッカーと呼んでくれるなら、あなたはハッカーです。

**訳注：ARPAnet** ARPA ネットワークとは国防省の ARPA の研究委託を受けた大学や政府研究機関を政府関係の研究で相互に情報を交換するために構築されたもので、インターネットの技術的基盤になった。RFC 991

　ハッカー精神は、ソフトウェアハッカー文化だけに限られたものではありません。ハッカー的態度をその他エレクトロニクスや音楽などに発揮する人たちもいます――実はどの科学や芸術分野でも、一番高い水準ではこのハッカー精神が見られます。ソフトウェアのハッカーたちも、ほかの分野での似たような精神を察知して、それを「ハッカー（hacker）」と呼んだりします。さらにハッカー性は、そのハッカーが活躍する個別媒体とは無関係に存在するという人もいます。しかしこの文書ではこれから、ソフトウェアハッカーの技術と態度、そして「ハッカー（hacker）」という言葉を生み出した共有文化の伝統に話をしぼります。

　ハッカーを声高に名乗る別の集団が存在しますが、彼らはハッカーではありません。これはコンピュータに侵入したり、電話のただがけしたりする人々（主に男のガキ）です。本物のハッカーはこの連中を「クラッカー（cracker）」と呼び、一切関わりを持ちたくないと思っています。本物のハッカーたちはたいてい、クラッカーは怠惰で無責任であまり賢くないと思っています。車の点火回路をいじってキーなしで車を始動できても自動車エンジニアにはなれないように、セキュリティ破りができてもハッカーにはなれないよ、というのがハッカーたちの文句です。残念なことに、多くのジャーナリストや著述家たちはだまされて、クラッカーについて書くのに『ハッカー hacker』という言葉を使い続けています。真のハッカーたちはこれをとてつもなく不愉快に思っています。

　基本的な違いとはすなわち：ハッカーはものをつくります。クラッカーは壊します。

　ハッカーになりたいならこの文書を読み続けなさい。クラッカーになりたいなら、[alt.2600 ニュースグループ](news:alt.2600)を読んで、そして自分で思ったほど頭がよくないことがわかったときには、ブタ箱を 5 年や 10 年くらう覚悟をすること。クラッカーについてこれ以上言うことはありません。

glider

**ハッカー的心構え**

1. [この世界は解決を待っている魅力的な問題でいっぱいだ](https://cruel.org/freeware/hacker.html#believe1)
2. [同じ問題を二度解くような無駄はいやだ](https://cruel.org/freeware/hacker.html#believe2)
3. [退屈と単純作業は悪](https://cruel.org/freeware/hacker.html#believe3)
4. [自由は善](https://cruel.org/freeware/hacker.html#believe4)
5. [心構えは技能の代用にはならない](https://cruel.org/freeware/hacker.html#believe5)

　ハッカーは問題を解決し、物事を築きます。そして自由と自発的な助け合いを信条としています。ハッカーとして受け入れられるには、こういう姿勢態度を持つようなふるまいが必要です。そしてこの姿勢を持つかのようにふるまうには、本当にその心構えを信じるしかありません。

　でも、ハッカー文化に受け入れられたいがためにハッカーらしい心構えを身につける気なら、それは勘違い。ハッカーの心構えを本当に信じるているような人になるのは、あなた自身にとって大切なことなんです――学ぶのに役立ち、その意欲を持続させてくれるんです。すべての創造的な芸術と同様、熟達者になる最高の方法は熟達者の精神をまねることです。知識面だけでなく、感情的な面も含め。

あるいは現代の禅詩が言うように：

道をたどり  
師をあおげ  
師にしたがえ  
師とともに歩け  
師を見通せ  
師となれ

　だからハッカーになりたいなら、以下の内容を何度も唱えて、それを信じ込むこと：

**1. この世界は解決を待つ魅力的な問題でいっぱいだ**

　ハッカーであることはとてもおもしろいのですが、そのためには相当な努力が必要です。その努力をするには意欲が必要です。優秀な陸上選手は、自分の肉体的な限界を越えて体を機能させるという肉体的な喜びが意欲の源です。同様に、ハッカーになるためには自分の技術をみがき、知性を働かす訓練して、問題を解決することにゾクゾクするような喜びを感じるようでなければいけません。

　あなたが生まれつきこのように感じないなら、ハッカーになるにはまずそのような人にならなくてはいけません。さもないとハッキングへの意欲が、セックスやお金や社会的な名声のようなつまらないことに惑わされてしまうでしょうから。

　（自分の学習能力に対する信頼も育てましょう。たとえ今の段階では大きな問題を解決するのに必要なすべてを知らなくても、その問題のほんの一部から取りかかって、そこから新しい事を学び取れば、次の部分の解決に十分なだけ学べて、そして次へ次へと進めば、いずれ大きな問題全体が解決できると信じましょう）。

**2.　同じ問題を二度解くような無駄はいやだ**

　クリエイティブな頭脳は貴重な限られた資源です。世の中にはたくさんの魅力的な新しい問題が解決を待っているんだから、車輪を発明しなおすような無駄なことをしてはいけません。

　ハッカーらしくふるまうには、他のハッカーたちが考える時間は貴重であると信じなければなりません。それをとことん信じて、情報を共有するのがほとんど道徳的な義務であると思えるようになりましょう。問題を解決し、その解答を提供すれば、他のハッカーたちは古い問題にいつまでも取り組みなおし続けずに、***新しい***問題を解決できるのです。

　でも「同じ問題を二度解くような無駄はいやだ」からといって、あらゆる既存の解決策が聖なるものだと思えということではないのに注意しましょう。またどんな問題であれ、正しい解決策が一つしかないなどと思ってもいけません。しばしば、解決策の最初の取り組みを学ぶことにより、その問題についてそれまで知らなかったいろんなことが学べます。自分たちのほうがその解決策よりも上手にできるぞと判断するのはOKですし、またしばしばそれは必用なことでもあります。OKでないのは、よい解決策の再利用を阻止してしまい、人々が車輪を発明しなおすよう**強制する**、作り物の技術的、法的、制度的な障害（たとえばクローズドソースのコード）です。

(作ったものを***すべて***無償で提供する義務があると思いこむ必要はありません。でもそうするハッカーが、他のハッカーたちから最高の尊敬を受けるのは確かですが。食費と家賃とコンピュータ代をまかなうのに必要なだけ自分の作品を販売しても、ハッカーの価値観と矛盾しません。ハッキングしているときに、自分の技芸と仲間のハッカーたちへの忠誠心を忘れない限り、家族を支えたり、あるいは金持ちになるためにであっても、ハッキングの技術を使うことはハッカーの価値観と矛盾しません。)

**3.　退屈と単純作業は悪**

　ハッカーたち(そして一般的にクリエイティブな人たち)は、退屈したり、くだらない反復作業をいやいやこなしたりしてはいけません。だってそういうときには、彼らしかできないこと、つまり新しい問題を解決するということが行われていないことになるからです。こういう無駄は万人にとって有害です。だから退屈と単純作業はつまらないだけでなく、本当に悪いことなのです。

　ハッカーらしくふるまうには、これを徹底的に信じ込んで、自分自身のためばかりでなく他のみんな（とりわけ他のハッカーたち）のためにも、退屈な部分はできるだけたくさん自動化してしまいたくなるようにしましょう。

（これには明らかな例外が一つあります。ハッカーたちは時々気持ちの切替えのために、あるいは技術を習得するため、またほかの手段ではできない特別な経験を積むために、端からは反復ばかりで退屈に見えるようなことをします。しかしこれは好きでやっているのです――思考能力のある人なら、決して退屈を強制されるべきではありません）。

**4.　自由は善**

　ハッカーたちは本来的に反権威主義です。あなたに命令できる人は、何かあなたが興味を持っている問題を解決するのを止めさせてしまえます――しかも、権威主義的な頭の特徴として、そのやめさせる理由もあきれるくらいくだらないものであるのが普通です。だから権威主義的態度に出会ったら、必ず戦わないといけないのです。そうしないとあなたや他のハッカーたちが窒息させられてしまいます。

（だからといってすべての権威と戦えということではありません。子どもには指導がいるし、犯罪者は拘束されるべきです。ハッカーは、命令に従うための時間以上にほしい何かを手にいれるためなら、ある種の権威を認めることに同意することもあるでしょう。しかし、それには制限のついた意識的な取引です。権威主義者が求める個人的な降伏などは提供しないのです）。

　権威主義者は検閲と秘密が大好きです。さらに自発的な協力や情報共有を信用しません。彼らは自分たちが管理できる「協力」だけを好みます。だから、ハッカーらしく行動するためには、検閲や秘密、そして責任ある大人に無理強いするような圧力やごまかしの使用に対し、本能的に敵意を感じなくてはなりません。そしてこの信念を実行に移す用意が必要なのです。

**5. 心構えは技能の代用にはならない**

　ハッカーになるには、上記の心構えをある程度身につけなければなりません。しかしスポーツのチャンピオンやロックのスターになろうとしたら、心構えだけではどうしようもないでしょう。同じように、心構えだけでハッカーになれるわけではありません。ハッカーになるには知性、実行力、献身、そして大きな努力が必要です。

　ですから、姿勢や態度は信用せずに、あらゆる技能を尊重することを学びましょう。ハッカーは、ハッカーもどきの相手をして時間を無駄にしたりはしませんが、技能には頭を垂れます。なかでもとりわけハッキング技能を崇拝しますが、その他どんな技能でもいいのです。ごく少数の人しか身につけられない、ハードルの高い技能は特によいもので、なかでも精神的な正確さ、技巧、集中力を必要とする技能は最高です。

　技能がすばらしいものだと思える人なら、自分の技能をみがくのも楽しめるでしょう。そのための大きな努力や献身も、単調な骨折り仕事ではなく、一種のもっと強烈な遊びになるでしょう。そしてハッカーになるには、この態度が不可欠なのです。

glider

**基本的なハッキング技術**

1. [プログラミングを身につけること。](https://cruel.org/freeware/hacker.html#skills1)
2. [オープンソース UNIX 類をどれか入手し、使いかたと動かしかたをおぼえること。](https://cruel.org/freeware/hacker.html#skills2)
3. [World Wide Web の使い方を学び、HTML を書くこと。](https://cruel.org/freeware/hacker.html#skills3)
4. [まともに英語ができないならば、身につけること。](https://cruel.org/freeware/hacker.html#skills4)

　ハッカーとしての心構えは重要ですが、技術はもっと重要です。心構えは技能の代用にはなりません。ハッカーの誰かが、あなたをハッカー呼ばわりしようなどと夢にでも思ってくれるには、まず身につけるべき基本的な能力ツールキット一式があります。

　このツールキットは時代と共にだんだん変わります。科学技術が新しい技能をうみだし、古い技能を時代遅れにするからです。たとえば、かつてはマシン語によるプログラミングがここには含まれていました。また最近になるまで HTML は含まれていませんでした。しかし、いま明らかに含まれるのはこんなものです：

**1. プログラミングを身につけること。**

　 　当然のことながら、これは根本的なハッキング技術です。もしコンピュータ言語をなにも知らないなら、まず [Python](https://www.python.org/) から始めることをおすすめします。設計がきれいだし、ドキュメントもしっかりしているし、初心者にもそこそことっつきやすくできています。でも入門言語として最適でも、おもちゃではありません。強力で柔軟で、大きなプロジェクトにもじゅうぶん対応しています。もっとくわしい [Python の評価](https://www.linuxjournal.com/article/3882)を書いたのでご参考まで。[Python ウェブサイト](https://docs.python.org/tutorial/)には[よいチュートリアル](https://docs.python.org/tut/tut.html)もあります。またサードパーティーによる優れたチュートリアルも、[コンピュータ科学サークル](https://cscircles.cemc.uwaterloo.ca/)にあります。

　かつては Java も、早めに学ぶのによい言語だと推奨していましたが、[以下の批判](http://static1.1.sqspcdn.com/static/f/702523/9242013/1288741087497/200801-Dewar.pdff)を読んで気が変わりました（この文中の「The Pitfalls of Java as a First Programming Language」（最初のプログラム言語としての Java の落とし穴）を探してください）。ハッカーとしては、かれら流のひどい言い方をすれば、「問題解決に金物屋の配管工事屋のようなアプローチ」をするわけにはいきません (訳注：つまりいろんな新しい道具があってその中からあれこれわけもわからず探し回るようなやり方)。それぞれのコンポーネントが実際に何をするのか、わかっていなくてはなりません。いまでは、たぶん Java より先にまず C や Lisp を学んだほうがいいと思います。

　ここにはもう少し一般的な論点もあるのかもしれません。もしある言語が、あなたにとって役に立ちすぎるのであれば、それは生産ツールとしてはよくても、学習にはよくない言語かもしれません。この問題を抱えているのは言語だけではありません。ウェブアプリケーションフレームワーク、たとえば RubyOnRails, CakePHP, Django などは、あまりに簡単に見かけ上の理解に到達できてしまうため、本当にむずかしい問題に直面したり、簡単なものへのソリューションをデバッグするときでさえ必要になるようなリソースを与えてくれないかもしれません。

　Javaに代わるもっとよい方法は[Go](https://golang.org/" \t "_top)言語を学ぶことです。この比較的新しい言語は、Pythonから移行するのがかなり簡単だし、次に考えられる一歩、つまりCの学習にもかなりの足がかりを与えてくれます。さらに、今後数年でもう少しはっきりするかもしれませんが、Goはシステムプログラミング言語として、Cのかなりの部分に取って代わる可能性もあります。これまでCが果たしてきた役割の相当部分がGoで置き換わる可能性もあるのです。

　本気でプログラミングをするなら、C を勉強するしかありません。これは Unix の中核となる言語です。C++ は C と密接な関係にあります。片方を知っていれば、もう片方を学ぶのはそんなにむずかしくありません。が、どっちも真っ先に勉強しようとするのには向いた代物ではありません。そして実は、C でのプログラミングをなるべく避けられたほうが、その分だけ生産性は増します。

　C はとても高効率ですし、マシンのリソースもドカ食いしません。残念ながら、C がそれだけの効率性を実現するには、あなた自身が低レベルのリソース管理（たとえばメモリ管理）を手作業でやってあげなくてはならないのです。それだけ低レベルコードがあると、複雑でバグも起こりやすいし、デバッグですさまじい時間をとられることになります。今日のマシンはずいぶん強力になっているので、これは通常は悪いトレードオフです――マシンの時間を少し非効率に使っても、自分の時間を**ずっと**効率的に使う言語を使うほうが賢明でしょう。ですから、Python をお奨めします。

　他に特に大事な言語としては、それ以外に [Perl](https://www.perl.com/) と [LISP](https://www.lisp.org/) があります。Perl は実用的な意味からも勉強しておく価値があります。アクティブ Web ページやシステム管理にとても広く使われているからです。自分では Perl を使わなくても、読めるようにはなっておきましょう。多くの人は、わたしが Python 利用を奨めるのと同じ形で Perl を使います。つまり、C のマシン効率を必要しないような仕事で C プログラミングを避けるために使うわけです。そういう人たちのコードを理解できなくてはなりません。

　LISP を勉強しておきたいのは、別の理由からです――それをモノにしたときにすばらしい悟り体験が得られるのです。この体験は、その後の人生でよりよいプログラマーとなる手助けとなるはずです。たとえ、実際には LISP そのものをあまり使わなくても。 (LISP の初体験をかなり簡単に得るには、Emacs テキストエディタのモードを書いたり改変したり、あるいは GIMP の Script-Fu プラグインをいじったりしてみましょう。)

　実は、この 5 つすべて（Python, Java, C/C++, Perl, LISP）を勉強しておくのがいちばんいいのです。これらはもっとも重要なハッキング用言語だというだけでなく、それぞれプログラミングに対してまったく違ったアプローチをしているので、どれも非常に有益な勉強となるでしょう。

　でも、言語をあれこれつまみ食いするだけでは、ハッカーどころか単なるプログラマの技能水準にすら到達できないことには気をつけましょう。プログラミングの問題について考えるのに、ひとつの言語に依存しない一般的な方法を身につけなくてはならないからです。真のハッカーになるには、マニュアルの記述を自分のこれまでの知識と関連づけることで、新しい言語をものの数日で習得できるようにならなくてはなりません。ということはつまり、ぜんぜん違った言語をいくつか学ぶべきだということです。

　ここでプログラムの学び方について完全な説明はできません。プログラミングは複雑な技能ですから。しかし、本や講習会のコースでダメだとは言っておきましょう。多くの、いやひょっとして***ほとんどの***ハッカーたちは我流で勉強してきたのです。役に立つのは、***(a) コードを読むこと***、そして ***(b) コードを書くこと***です。

　Google のトップハッカーの一人で、AI に関するいちばん普及した教科書の共著者であるピーター・ノルヴィグは、「[十年がかりでプログラムを学ぼう](http://norvig.com/21-days.html)」というすばらしい小論を書いています。その中の「プログラミングで成功するレシピ」は精読の価値があります。

　プログラミングを学ぶということは、自然言語でよい文章の書きかたを学ぶようなものです。いちばんいいのは、その分野の熟練者が書いたものを読んで、自分で何か書いてみて、もっとたくさん読んで、もうちょっと書いて、もっとたくさん読んで、もうちょっと書いて……そして自分の書くものが、お手本のもつ力強さと簡潔さをもつようになるまでこれを繰り返すことです。

　この学習プロセスについては[ハッキング学習法](http://www.catb.org/~esr/faqs/hacking-howto.html)でもっと語っています。これは簡単な指導群ですが、決して簡単なものではありません。

　以前は読むのにいいコードを見つけるのは大変でした。大規模なプログラムのなかで、駆け出しハッカーが読んだり、いじったりできるソースの形で入手できるものはほとんどなかったからです。しかしこの状況は劇的に変わりました；オープンソースのソフト、プログラムツール類、そしてオペレーティングシステム類（すべてハッカーの作ったものです）が現在では広く出回っています。ここから話はうまいこと、次のテーマに進むことになります。

**2. オープンソース UNIX 類をどれか入手し、使いかたと動かしかたをおぼえること。**

　ここではあなたが自分のコンピュータを持っているか、あるいはコンピュータにアクセスできるものと仮定します（これがどんなにすごいことか、ちょっと考えて見てください。ハッカー文化はもともと、コンピュータがあまりに高価で個人所有など不可能な時代に発達したんです）。新米さんがハッカー技術を習得するための唯一最大の重要な一歩は、Linux かフリーの BSD UNIX 類のどれかを入手し、それを自分のマシンにインストールし、動かすことです。

　ええもちろん、UNIX 以外にも世界にはオペレーティングシステムがあります。しかしそれらはバイナリで配布されているので、コードを読んだりいじったりできないんです。MS Windows マシンやその他ソースがクローズドなシステムでハッキングを学ぼうとするのは、全身ギプスをつけてダンスを学ぼうとするようなものです。

　Mac OS X を使えばコードを読んだりいじったりはできますが、システムのうちオープンソースなのは一部だけです。いろいろ壁にぶちあたるでしょうし、アップルの独占コードに依存した悪いクセを身につけないよう注意する必要が出てきます。でも根底にある Unix にだけ専念すれば、いろいろ有益なことが学べます。

　UNIX はインターネットのオペレーティングシステムです。UNIX を知らなくてもインターネットの使い方は学べますが、UNIX がわからずにインターネットハッカーにはなれません。そういうわけで、現在のハッカー文化はかなり強く UNIX 中心となっています（昔からずっとそうだったわけではありません。だから一部の古参ハッカーは、いまだに現状を苦々しく思っています。でも UNIX とインターネットの共存は非常に強く、 マイクロソフトの力をもってしても、それを本気でつぶすのはむずかしいようです）。

　だから UNIX をたちあげなさい――わたしは Linux が好きですが、他の方法もあります（そして、ええ、 Linux と MSWindows を同じマシンで走らせることは十分可能です）。UNIX を学びなさい。走らせなさい。いじりたおしなさい。それでインターネットと語りあいなさい。コードを読もう。コードを改変しよう。Microsoft のオペレーティングシステムでは夢見ることもできないほどよく出来たプログラミングツール（C や Lisp や Python、Perl など）が手に入ります。楽しいですよ。そしてあなたがマスターハッカーになってふりかえったら、この時期にどれほどの知識を自分が吸収したかようやく理解できるようになるでしょう。

　UNIX の学習についての詳細は[幼虫になろう！(Lat's go Larval!)](https://letsgolarval.wordpress.com/)は、新人Linuxユーザの学習プロセスを見る場として、異様に雄弁で有益だと思います[私はこうしてLinuxを学んだ (How I Learned Linux)](https://letsgolarval.wordpress.com/2015/06/23/how-i-learned-linux/)という投稿がよい出発点になるでしょう。

　ブログ[The Loginataka](http://www.catb.org/~esr/faqs/loginataka.html)を参照のこと。また[The Art Of Unix Programming](http://catb.org/~esr/writings/taoup/)も興味があればどうぞ。

　Linux を入手するは、[Linux Online!](https://www.linux.org/) サイトを参照。そこからダウンロードしてもいいし（もっといいのは）地元の Linux ユーザグループを見つけて、インストールを助けてもらうこともできます。

　この「How-to」最初の十年では、新人からみればどの Linux ディストリビューションもほとんど似たり寄ったりだと述べてきました。でも 2006-2007 年に、ほんものの最高の選択が登場しました: [Ubuntu](https://www.ubuntu.com/)です。他のディストリビューションも独自の強みはありますが、Ubuntu は Linux 新人にとって圧倒的に一番とっつきやすいものです。でも、数年後に Ubuntu がデフォルトで導入した、醜悪でほとんどつかいものにならない「Unity」デスクトップインターフェースには要注意。変種のXubuntuやKubuntuのほうが優秀です。

BSD Unix に関するヘルプや各種情報については、[http://www.bsd.org/](https://www.bsd.org/)を参照。

　ちょっと試すのにいい方法は、Linux ファンが「[Live CD](https://www.livecdnews.com/)」と呼ぶものを起動することです。これは CD やUSBメモリだけで起動するディストリビューションで、ハードディスクを改変しません。CD は遅いのでこれは遅いのですが、あまり大胆な一歩をとらずに可能性を調べる一手段ではあります。

[Unix と Internetの基本](http://en.tldp.org/HOWTO/Unix-and-Internet-Fundamentals-HOWTO/index.html)についての入門文書を書いてみました。

　昔は、新人が一人で Linux や BSD をインストールするのはやめたほうがいいと書きました。でも最近のインストーラーはずっとよくなって、新人であっても一人で十分にできます。それでも、地元 Linux ユーザーグループに接触して助けてもらうほうがいいでしょう。害にはなりませんし、いろいろ楽になるかもしれませんよ。

**3. World Wide Web を使う方法を学び、HTML を書くこと。**

　ハッカー文化の成果物は、ほとんどが縁の下で働くようにつくられてきました。工場や事務所や大学の運営を助けつつ、ハッカーでない人々の生活にはまるで直接的な影響をもたなかったのです。Web は唯一の大きな例外で、これはハッカーたちの巨大な輝くおもちゃなのに、***政治家たちでさえ***それが世界を変えているのを認めているのです。このためだけにでも（そしてそれ以外にもましな理由はいくらでもあります）Web の使い方は身につける必要があります。

　これはブラウザの使い方を覚えるだけの話ではありません（そんなのサルでもできます）。Web のマークアップ言語である HTML の書き方を覚えろということです。プログラミング経験がないなら、HTML を書けばプログラミングの勉強に役にたつ精神的な習慣が身に付きます。だから、ホームページをつくることです。

　しかしながらホームページを持つだけでは、ハッカーにはまだまだほど遠い状態でしかありません。Web はホームページだらけです。そのほとんどは無意味きわまる、まともな中身ゼロのうんこです――見た目は非常にかっこよさげなうんこだったりはしますが、それでもうんこはうんこなのです（詳しくは[地獄の HTML ページ](https://web.archive.org/web/20030621093646/http:/www.ss.iij4u.or.jp/~ryo-kun/fuyuki/hell_html.html) を参照）。

　価値あるものにするには、ページにはまともな***中身***がなくてはなりません。それは他のハッカーたちにとって興味をひき、そして/あるいは有益なものでなければなりません。そこで出てくるのが次のテーマ……

**4. まともに英語ができないならば、身につけること。**

　わたし自身がアメリカ人だし英語が母語なので、これを挙げるのはあまり気乗りがしませんでした。これを言うと文化帝国主義と思われるんじゃないかと思ったからです。しかし英語以外を母語とする数名から、英語はハッカー文化やインターネットでの作業用言語なんだということ、ハッカーコミュニティーで役に立つには英語を知らないとダメだということを指摘するように言われたのです。

　これはまったくその通りです。1991 年頃、母語が一緒でも英語を第一外国語として身に着けている多くのハッカーたちは、技術的な議論をするときに英語を使うというのを知りました。そのとき教わったのは、英語は技術用語がほかの言語よりも豊富で、だから英語のほうがとにかくこの仕事に向いているのだということです。同様な理由で、英語で書かれた技術書を訳した場合、多くは不満足な出来になります(そもそも訳されるかどうかもわかりませんし）(訳注：もちろん山形浩生を初めとする一部の日本語訳は例外である。)

　フィンランド人であるリーヌス・トーヴァルズは自分の書くコードのコメントを英語で書きます (どうやら英語以外で書こうなんて考えたこともないようです)。彼の流暢な英語は、Linux の開発者の世界的なコミュニティをリクルートしてくるにあたっての、とっても大事な要因でした。見習う価値のある先例といえましょう。

　英語が母語だからといって、ハッカーとしてやっていけるだけの言語能力を持つという保証はありません。作文がボロボロで、文法まちがいだらけで、スペルミスだらけなら、多くのハッカー（わたしを含め）はその人を無視しがちです。書き方がへぼいからといって、思考がヘボいとは限りませんが、一般にその相関はかなり高いことがわかっています――そしてハッカーたちはヘボい思考なんかに用はありません。まともな文が書けないなら、書けるように勉強しましょう。

**ハッカー文化での地位**

1. [オープンソース・ソフトウェアを書く](https://cruel.org/freeware/hacker.html#respect1)
2. [フリーソフトウェアのテストやデバッグを手伝う](https://cruel.org/freeware/hacker.html#respect2)
3. [有益な情報を公開する](https://cruel.org/freeware/hacker.html#respect3)
4. [インフラが機能し続けるように手伝う](https://cruel.org/freeware/hacker.html#respect4)
5. [ハッカー文化そのものへの貢献](https://cruel.org/freeware/hacker.html#respect5)

　貨幣経済を伴わない文化にはよく見られることですが、ハッカー社会では名声で地位が決まります。おもしろい問題を解決しようとしていても、それらがどれほどおもしろく、また、あなたの解決方法が本当に素晴らしいかどうか判断できる力があるのは、ふつうは技術上の仲間や先輩たちだけなのです。

　ですからハッカーゲームに参加したら、他のハッカーたちがあなたの技能をどう思うかでスコアをつけることを覚えます（だからこそ、他のハッカーたちがたえずあなたをハッカー呼ばわりするまで、あなたは本当のハッカーではないのです）。ハッキングは孤独な作業というイメージがあるので、この事実は見えにくくなっています。またエゴや外からの評価がハッカーとしての動機づけに関係しているということを認めることに対して、ハッカー文化にはタブーがあり（1990 年代末以来しだいに弱まっていますがまだ根強くあります）、このせいで見えにくくなっている面もあります。

　具体的には、ハッカー社会というのは人類学者が言うところの贈与の文化なのです。そこで地位と名声を得るには、他の人々を力で支配したり、美しかったり、他の人々が欲しがるものを持っていたりしてもだめです。何かをあげてしまうことで、地位と名声を得るのです。具体的には、時間と創造性、そして技術の成果をあげてしまうことで、評判を得るのです。

　ハッカーたちから尊敬されるためにできることは、基本的に 5 つあります。

**1. オープンソース・ソフトウェアを書く**

　まず初めに（最も主要で、もっとも伝統的なことですが）他のハッカーたちが喜ぶ有益なプログラムを書くことです。そしてハッカー文化のみんなが使えるように、そのプログラムのソースをあげてしまうことです。

（むかしはこの種のソフトを「フリーソフト」と呼んでいましたが、「フリー」の意図する意味がよくわかっていない人たちが多すぎて混乱してきました。私たちの多くがいまでは[オープンソース](http://www.opensource.org/)・ソフトという用語の方を好みます）。

　ハッカー社会での最も神格的な英雄は、大規模で役に立つプログラムを書き、それが広範なニーズに応えていて、しかもそれを無償でばらまいてくれて、だからみんながいまやその人の書いたソフトを使うようになっている、そんな人です。

　が、ここで細かい歴史的な論点がちょっとばかり出てきます。ハッカーたちは常にオープンソースの開発者たちを、自分たちのコミュニティにおけるガチガチの中核として崇拝してきましたが、1990 年代半ば以前には、ほとんどのハッカーたちはほとんどの時間を、クローズドソースのソフト開発に費やしてきました。これは1996 年にわたしがこの文書の最初のバージョンを書いたときもそうでした。これが変わったのは、1997 年以降オープンソースソフトが主流化してからです。今日では、「ハッカーコミュニティ」と「オープンソース開発者」は、ほとんど同じ文化と人々を指すもので、呼び方がちがうだけです。でも、昔はちがったということは是非覚えておいてください（この点について詳しくは「[歴史的な記述：ハッキング、オープンソース、フリーソフト](https://cruel.org/freeware/hacker.html" \l "history)」の節を参照)。

**2. フリーソフトウェアのテストやデバッグを手伝う**

　さらにハッカー社会は、オープンソースのデバッグに立ち上がる人も賞賛します。この世は不完全なので、ソフトウェア開発のほとんどの時間はデバッグに費やされます。だからこそ、物を考えているオープンソース・ソフトの作者たちは、良いベータテスタ（症状を明確に表現できて、問題のある箇所を特定し、急ごしらえのリリースのバグにがまんできて、ある程度の診断テストをしてくれる人）は体重分のルビーにも匹敵する価値があると言うわけです。以上の条件の一つでも満たされていれば、バグ取り作業が際限なく続く悪夢になるか、ちょっとした軽い面倒事程度ですむかの分かれ目になり得るのです。

　あなたが新米なら、興味が持てる開発途中のプログラムを見つけて、よいベータテスタになりなさい。プログラムのテストを助ける段階から、そのデバッグを援助する段階、そしてプログラム自体の改変を援助する段階へと自然に進むでしょう。こうすればたくさんのことを学べるし、あとでいろいろ手助けをしてくれるはずの人々ともよい縁（カルマ）をつくりだす役にもたつはずです。

**3. 有益な情報を公開する**

　これまたよいことは、役に立つおもしろい情報を集めて選り分け、それをWebページにしたり、あるいは FAQ (Frequently Asked Questions lists）のような文書にすることです。そしてそれらを一般公開することです。

　大事な技術 FAQ 類の維持管理者は、オープンソースの作者とほとんど同じくらい尊敬を得られます。

**4. インフラが機能し続けるように手伝う**

　ハッカー文化（さらに言うとインターネットの技術的開発）はボランティアが運営しています。それを機能させるためには、必要ながら陽の当たらない作業をたくさんやらなくてはならないのです。たとえば、メーリングリストを管理したり、ニュースグループの議長を勤めたり、アーカイブサイトで大きなソフトウェアを維持したり、RFC 類やその他技術的な標準を開発したりするようなことです。

　 このような種類の仕事を立派にこなす人々は多大な尊敬を受けます。このような仕事が時間ばかり食って、しかもコードで遊ぶほど楽しくないのをみんな知っていますから。それを引き受けるのは献身ぶりを示すことになるわけです。

**5. ハッカー文化そのものへの貢献**

　最後に、あなたは文化そのものに貢献しそれを広めることができます(たとえば、ハッカーになる方法についてちゃんとした入門文書を書いたりして :-))。このような仕事は、あなたがいろんな経験をし、そしてこれまでの４項目のどれかで名をあげるまではできるものではありませんが。

　ハッカーの文化には、はっきりした指導者はいませんが、文化ヒーローや部族の長老や歴史家や伝承者がいます。戦場でそれなりの時間を過ごすうちに、あなたもその一人へと成長するかもしれません。ただしここで注意！　ハッカーは、部族の長老がエゴをむき出しにすると信用しません。だからこの種の名声をはっきり目指すようなまねは危険です。名声を得ようと努力するのではなく、うまく立ち回ってその名声が自然にやってくるようにしなくてはなりません。そして名声を得ても、自分の地位について謙虚で優雅でありなさい。

glider

**「ハッカー」と「おたく Nerd」との関係**

　有名な言い伝えがなんと言おうとも、ハッカーになるのに「おたく」である必要なありません。とはいえ、おたくのほうが好都合なのは事実だし、多くのハッカーは確かにおたくです。社会のつまはじき状態のほうが、本当に重要なことに集中しやすいわけです。考えるとかハッキングするとかね。

　このため、多くのハッカーは「geek」という言葉さえ、誇りをこめて受け容れるようになっています。それは自分が普通の社会的期待から自由だと宣言する方法の一つなのです（そしてハッカーであることに伴う、ＳＦや戦略ゲームなどの愛好ぶりを宣言するものでもあります）。かつて 1990 年代には、「おたく(nerd)」という言葉がこの意味で使われていました。当時「おたく」は弱い差別語で、「ギーク」はかなりきついものでした。でも 2000 年からしばらくして、少なくともアメリカのポピュラー文化では、この両者の役割が入れ替わりました。いまや技術志向でない人々の中でも、誇らしくギークを名乗る人がかなりいます。

　上達するほどハッキングに集中できて、しかも一方でまともな社会生活が送れるというのなら、それはそれで結構なこと。これはわたしが新米であった 1970 年代頃に比べると、現在ではずっと簡単です。主流文化は現在では技術おたくにずっと好意的になっていますから。ハッカーがとても素晴らしい恋人や伴侶になることだって多いと気づいた人たちは、実はだんだん増えているのです。詳しくは[「女の子のためのおたくガイド」](https://cruel.org/freeware/geek.html)を参照。

　まともな社会生活がおくれないのでハッキングに魅力を感じるなら、それもまた結構――少なくとも、気が散って困るようなことはないわけですから。それにまあ、いずれ社会生活のほうもなんとかなるかもしれないじゃないですか。

glider

**ハッカースタイルの要点**

　もう一度いいますが、ハッカーになるためにはハッカー精神を身につけなければなりません。コンピュータの前にすわっていない時にもハッカーになるのに役立つことがいくつかあります。以下にあげることは、ハッキングそのものの代わりにはなりません（そんなものはあるわけがない）。でも多くのハッカーは以下のようなことをやっていますし、それがハッキングの真髄に本質的に通じるものがあると感じています。

* 母語できちんと文が書けるようになること（わたしが知っている最高の連中を含め、ハッカーの驚くほど多くは物書きとしても有能です）。
* SF を読むこと。SF 大会に参加すること（ハッカーやハッカーの卵たちに出会ういい方法です）。
* 格闘技を学ぶこと。格闘技に必要な精神的な規律は深いところでハッカー精神と共通しているようです。ハッカーたちの間で最も人気があるのは、まちがいなくアジアの素手による格闘技、たとえばテコンドー、各種の空手、カンフー、合気道、柔術などです。西洋のフェンシングやアジアの剣道も、かなりの支持者がいます。合法な地域では、1990 年代末以来、拳銃射撃も人気が出てきました。最もハッカー的な格闘技は、精神的な規律とリラックスした注意力とコントロールを重視し、力任せや運動や肉体的なタフさに過度の重きを置きません。
* 本当の瞑想的な訓練を受けること。ハッカーたちの永遠のお気に入りは禅です（重要な点として、信仰に入ったりいまの宗教を捨てたりしなくても、禅の恩恵は受けられるのです）。他の様式もいけるかもしれませんが、得体の知れないことを信じ込ませるようなものは避けましょう。
* 音楽を分析的に聞く耳を鍛えること。一風変わった音楽がわかるようになること。なにか楽器を上手に演奏したり、歌が歌えるようになること。
* だじゃれや言葉あそびへの理解を深めること。

　これらのうち、すでにやっていることが多ければ多いほど、ハッカー的資質も高いはずです。なぜよりによってこういう事柄なのかは、まだ完全には明らかではないのですが、いずれも右脳能力と左脳能力の組み合わせという点で結びついています。どうもこれが重要らしいのです（ハッカーは、論理的に理屈づけると同時に、問題の理屈の外へ一瞬で踏み出す能力が必要なのです）。

　遊びと同じくらい熱心に働き、働くのと同じくらい熱心に遊びましょう。真のハッカーにとっては、「遊び」「仕事」「科学」「芸術」の境界はすべて消滅しがちだったり、すべてが高次の創造的な遊技に融合しがちだったりします。また、狭い範囲の技能だけで満足しないこと。ほとんどのハッカーたちはプログラマを自称しますが、関連技能複数について、かなりの能力を持っていたりします――システム管理、ウェブデザイン、ＰＣハードウェア修理などはよくあります。システム管理者であるハッカーは、かなりスクリプトプログラミングやウェブデザインに長けていることが多いものです。ハッカーたちは中途半端なことはしません。何か技能を身につけるなら、きわめて高い水準にまで達することが多いのです。

　最後に、次のようなことはしては***いけません***。

* つまらないおおげさな ID やハンドルを使わない。
* Usenet (あるいは他の場所でも)で、フレームとは関わり合いにならない。
* 「サイバーパンク」を名乗ったりしないこと。そしてそう名乗っているヤツを相手にして時間を無駄にしてもいけません。
* 誤字脱字だらけで文法もでたらめな投稿やメールはしないこと。

　こんなことをしたら、評判を落とすだけです。ハッカーたちは記憶力がいいんですよ。かつての愚行を忘れて受け入れられてもらえるまで、何年も苦労する結果になりかねません。

　ハンドル名にまつわる問題については、もうちょっと詳しく書いておきましょうか。ハンドルの陰にアイデンティティを隠すのは、クラッカーや、warez d00dz や、他の下等な生の形態に特徴的な、子どもじみた愚かな行動です。ハッカーはこんなことはしません。かれらは自分たちがやることに誇りを持っていて、それに*本名で*関わろうとします。だからもしあなたがハンドルを持っているのなら、それをお捨てなさい。ハッカー文化において、それは敗者の印にしかならないでしょう。

glider

**歴史的な記述：ハッキング、オープンソース、フリーソフト**

　1996年末にこの文書を書いたときには、まわりの状況の一部は今日とはかなり様子がちがっていました。こうした変化について少し述べておくと、オープンソースやフリーソフト、Linux がハッカーコミュニティとどう関係するのか混乱している人々にとって、少し話が明確になるでしょう。そういうことに興味がなければ、すぐに FAQ や関連文書の章を読み進んでください。

　これまで描いてきたようなハッカーのエートスとコミュニティは、1990 年以後の Linux の台頭よりはるかに古いものです。わたしが最初に関わったのは 1976 年頃で、そのルーツは容易に1960年代初期にまでさかのぼります。でも Linux 以前は、ほとんどのハッキングは独占 OS 上か、ごく少数の準実験的な自家製システム、たとえば MIT の ITS といった、当初の学会ニッチの外には出ないシステム上で行われていました。Linux 以前にも、この状況を変えようという試みはいくつかありましたが、そのインパクトはせいぜいが周縁的で、根っからの信仰者たちのコミュニティ内にとどまっており、それはハッカーコミュニティの中ですらごく少数派でしたし、ましてソフトウェア業界一般から観ればないも同然でした。

　いま「オープンソース」と呼ばれているものは、ハッカーコミュニティと同じくらいの歴史がありますが、1985 年までそれは、理論や宣言を持つ意識的な運動というよりは、無名の民俗的な慣習でした。この前史が終わったのは、1985 年にハッカーの原型とも言うべきリチャード・ストールマン（「RMS」）がこれに「フリーソフトウェア」という名前をつけようとしたときでした。でも、この命名行動は、領土主張行動でもありました。かれは「フリーソフトウェア」にイデオロギー的なお荷物をしょわせたのですが、ハッカーコミュニティはこれをあまり受け容れませんでした。結果として「フリーソフトウェア」というレッテルは、ハッカーコミュニティの少数派とはいえかなりの人々に、はっきりと拒絶されました（特に BSD Unix 関連の人々は嫌がりました）。そして残りの多数派ハッカー（わたしも含め）は、その用語は使ったものの、静かに、でもかなり真面目に懸念を抱いていました。

　こうした懸念にも関わらず、「フリーソフトウェア」の旗印のもとでハッカーコミュニティを定義づけて主導しようという RMS の主張は、1990 年代半ばまで概ね受け容れられました。それが初めて厳しい挑戦を受けたのは、Linux が台頭したからです。Linux はオープンソース開発の自然な故郷となりました。いまはオープンソースと呼ばれるプロジェクトの多くは、各種の独占 Unix から Linux に引っ越しました。Linux 周辺のコミュニティは爆発的に広がり、Linux 以前のハッカー文化よりもずっと大きく多様になったのです。RMS は決然と、こうした活動すべてを自分の「フリーソフトウェア」運動にとりこもうとしましたが、Linux コミュニティの爆発的な多様性と、その創設者であるリーヌス・トーヴァルズが公然とそれに疑念を表明したことで、その試みは阻止されました。トーヴァルズは、他の表現がないので「フリーソフトウェア」という用語を使い続けましたが、RMS のイデオロギー的なお荷物は公然と拒否したのです。多くの若いハッカーたちもそれに続きました。

　1996 年にわたしが初めてこのハッカー HOWTO を公開したとき、ハッカーコミュニティは急速に Linux やその他一握りのオープンソース OS（特に BSD 系 Unix の系譜）を中心に組みかわりつつありました。ほとんどのハッカーたちは、何十年にもわたってクローズドソースの OS で、クローズドソースのソフトを開発してすごしたという共同体記憶は、まだ薄れてはいませんでしたが、でもこの事実はすでに、もう絶滅した過去の一部に思えるようになっていました。ハッカーたちはますます、自分たちを Linux や Apache といったオープンソースプロジェクトとの結びつきによって、自分たちをハッカーとして定義づけるようになってきたのです。

　でも「オープンソース」という用語はまだ登場していませんでした。それが登場するのは 1998 年初頭です。登場したら、ほとんどのハッカーコミュニティはそれを六ヶ月以内に採用しました。そうしないのは少数派で、「フリーソフトウェア」という用語にイデオロギー上の理由から固執していたのです。1998 年以来、そして特に 2003 年頃から、「ハッキング」と「オープンソース（とフリーソフトウェア）開発」とはかなり同一視されるようになりました。今日では、こうしたカテゴリーを区別しようとするのはほとんど無意味だし、それが将来変わることもなさそうです。

　でも、昔からそうだったわけではないということは、是非ともご記憶を。

glider

**その他の関連文書**

　ポール・グレアムは [偉大なハッカー](http://www.paulgraham.com/gh.html)や[undergraduation](http://www.paulgraham.com/college.html)という文書でいろいろ賢明なことを書いています。

　若手ハッカーは [昔はハッカーがだれでも知っていたこと](http://catb.org/~esr/faqs/things-every-hacker-once-knew)が興味深く有益だと思うかもしれません。

　Peter Seebach はハッカーの扱い方がわからない管理職向けのすばらしい [Hacker FAQ](http://www.plethora.net/%7Eseebs/faqs/hacker.html)（[邦訳「管理職のためのハッカー FAQ」](http://www1.neweb.ne.jp/wa/yamdas/column/technique/hackerj.html)）を維持しています。

[プログラマになろう](http://samizdat.mines.edu/howto/HowToBeAProgrammer.html) という文書があって、本文書と見事な対をなします。コーディングや技能セットにとどまらず、プログラミングチームの中でどう行動すべきかについて有益なアドバイスがあります。

　 わたしはほかに[A Brief History Of Hackerdom](http://www.catb.org/~esr/writings/hacker-history/)（[邦訳「ハッカー界小史」](https://cruel.org/freeware/hackerdom.html)）を書いています。

　 また[The Cathedral and the Bazaar](http://www.catb.org/~esr/writings/cathedral-bazaar/) という論文（[邦訳「伽藍とバザール」](https://cruel.org/freeware/cathedral.html)）では、 Linux とオープンソース文化がどのように機能するかについていろいろ説明しています。このテーマについてもっと直接的に扱ったのが[The Homesteading the Noosphere](http://www.catb.org/~esr/writings/homesteading/) という論文（[邦訳「ノウアスフィアの開墾」](https://cruel.org/freeware/noosphere.html)）です。

　Rick Moen は、[Linux ユーザグループの運用について](http://linuxmafia.com/faq/Linux_PR/newlug.html)すばらしい文書を書いています。

　Rick Moen と私は、[賢い質問の仕方](http://catb.org/~esr/faqs/smart-questions.html)という別の文書をいっしょに書きました。これを読めば、助けを求めるときに実際に助けがくる見込みが高まるでしょう。

　パーソナルコンピュータやUnixやインターネットがどう機能するかの基礎について説明がいるなら、[The Unix and Internet Fundamentals HOWTO](http://en.tldp.org/HOWTO/Unix-and-Internet-Fundamentals-HOWTO/)を読みましょう.

　ソフトウェアをリリースしたり、ソフトのパッチを書いたりするときには、[Software Release Practice HOWTO](http://en.tldp.org/HOWTO/Software-Release-Practice-HOWTO/index.html) のガイドラインにしたがうようにしましょう。

　禅詩が気に入ったら、[Rootless Root: The Unix Koans of Master Foo](http://catb.org/~esr/writings/unix-koans) もお気に召すかもしれません。

**FAQ**

1. Q: [どうすれば自分がすでにハッカーだとわかりますか？](https://cruel.org/freeware/hacker.html#hacker_already)
2. Q: [ハッキングの方法を教えてください。](https://cruel.org/freeware/hacker.html#teach_hack)
3. Q: [なら手始めに何をすればいいでしょう？](https://cruel.org/freeware/hacker.html#getting_started)
4. Q: [いつ始めるべきでしょうか？ 私が習得するには遅すぎるでしょうか？](https://cruel.org/freeware/hacker.html#when_start)
5. Q: [ハックを習得するのにどれぐらいかかるのでしょうか？](https://cruel.org/freeware/hacker.html#how_long)
6. Q: [手始めに Visual Basic を勉強するのはどうでしょうか?](https://cruel.org/freeware/hacker.html#closed_lang)
7. Q: [システムのクラックを手伝ってください。あるいはクラックの方法を教えてください。](https://cruel.org/freeware/hacker.html#I_want_to_crack_and_Im_an_idiot)
8. Q: [他の人のアカウントのパスワードはどうやったらわかりますか？](https://cruel.org/freeware/hacker.html#passwords)
9. Q: [他の人のメールを読んだりモニタしたり入り込んだりするにはどうすればいいの？](https://cruel.org/freeware/hacker.html#crackmail)
10. Q: [IRC でチャンネル op 特権を盗むのってどうやんの?](https://cruel.org/freeware/hacker.html#crackop)
11. Q: [クラックされてしまいました。更なるアタックを避けるのを手伝ってもらえませんか？](https://cruel.org/freeware/hacker.html#anti_crack)
12. Q: [Windows ソフトのことで困っています。助けてもらえますか？](https://cruel.org/freeware/hacker.html#windows_grief)
13. Q: [どこで真のハッカーたちと話すことができますか。](https://cruel.org/freeware/hacker.html#real_hackers)
14. Q: [ハッキングに関連した題目に関して、お勧めの役に立つ書籍はありますか？](https://cruel.org/freeware/hacker.html#books)
15. Q: [ハッカーになるには数学が得意じゃないとダメですか？](https://cruel.org/freeware/hacker.html#mathematics)
16. Q: [最初にどんな言語を学んだらいいでしょうか。](https://cruel.org/freeware/hacker.html#language_first)
17. Q: [どんなハードウェアが必要でしょうか？](https://cruel.org/freeware/hacker.html#hardware)
18. Q: [貢献したいんですけど、どの問題に参加すればいいか選ぶのを助けてくれませんか？](https://cruel.org/freeware/hacker.html#started2)
19. Q: [マイクロソフトを憎み、非難する必要があるのでしょうか？](https://cruel.org/freeware/hacker.html#MS_hater)
20. Q: [でもオープンソース・ソフトだと、プログラマは食べていけなくなるんじゃないですか？](https://cruel.org/freeware/hacker.html#no_living)
21. Q: [フリーの UNIX をどこで入手できますか。](https://cruel.org/freeware/hacker.html#problems)

**Q: どうすれば自分がすでにハッカーだとわかりますか？**

以下の3つの問いを自問してみましょう:

* コードをすらすらと書けますか？
* ハッカーコミュニティの目的や価値観を共有していますか？
* ハッカーコミュニティの立派なメンバーに、ハッカーと呼ばれたことはありますか？

　この*三つすべてに* イエスと言えるなら、あなたはすでにハッカーです。二つだけでは不十分です。

　最初のテストは技能についてです。本文書で記述した最低限の技術スキルがあれば、合格できるでしょう。オープンソース開発プロジェクトにかなりのコードを採用されているなら、楽々合格でしょう。

　第二のテストは態度についてです。もし [ハッカーの心構えの五原則](https://cruel.org/freeware/hacker.html#attitude) が当然のものにしか思えず、自分の生き様を描いただけのまったく目新しくないものとしか思えないなら、すでに合格への道半ばです。これが内面的な半分です。残りの外面的な半分は、ハッカーコミュニティの長期プロジェクトにどこまで共感できるかについてです。

　そうしたプロジェクトについて、不完全ながら示唆的な一覧を挙げましょう。Linuxが改善され普及することはあなたにとって重要ですか？　ソフトウェアの自由について熱意を持っていますか？　独占を嫌いますか？　コンピュータが世界をもっと豊かで人道的な場所にするための、能力強化の道具になれるという信念に基づいて行動していますか？

　でも一つ注意しておくべきでしょう。ハッカーコミュニティは、具体的で、主に防衛的な政治的関心を持っています。その二つは、言論の自由の権利擁護と、オープンソースを非合法にするような「知的財産」権力掌握の阻止です。こうした長期プロジェクトの一部は、電子フロンティア財団のような市民権組織であり、外面的な態度は当然ながらこうした団体の支持を含んでいます。でもそれ以上となると、ほとんどのハッカーたちはハッカーの態度を明示的な政治プログラムに体系化しようとする試みを、疑念をもって眺めます。こうした試みがコミュニティを分裂させて問題から目をそらせてしまうことを、わたしたちは苦労して学びました。だれかがハッカー的態度の名の下に、国会デモにくるよう誘いをかけてきたら、その人は何かかんちがいをしています。正しい態度というのはおそらく「だまって連中にコードを見せてやれ」というものでしょう。

　第三のテストは、再帰性があるのでちょっと面倒です。[「ハッカーって何？」](https://cruel.org/freeware/hacker.html#what_is)という節で、ハッカーになるというのは、共有の歴史と、内面と外面を持つ特定のサブカルチャーまたは社会ネットワークに所属することでもあるのだ、と述べました。遙か昔には、ハッカーたちは集団としての一体性や自己認識がいまよりずっと欠けていました。でも過去30年にわたり、インターネットがハッカーサブカルチャーの中核との接触を容易に実現し維持可能にしてくれるにつれて、社会ネットワークとしての側面の重要性は増してきました。この変化を容易にしめす行動指標としては、今世紀になってわたしたちは独自のTシャツを持つに到った、ということが指摘できます。

　ハッカー文化のようなネットワークを「見えない大学」なる一般名のもとに研究してきた社会学者たちは、こうしたネットワークの特徴の一つは、そこに門番がいることだ、と言います。門番とは、新メンバーのネットワーク参加を承認する社会的権限を持つ中核メンバーです。ハッカー文化という「見えない大学」はゆるく非公式なので、門番の役割も非公式です。でもあらゆるハッカーが骨の髄まで知っているのは、あらゆるハッカーが門番になれるわけではないということです。門番はある程度の高い地位を持ち、業績も挙げなければその地位につけません。どれほどかを定量化するのはむずかしいのですが、あらゆるハッカーは、見ればそれがわかるのです。

**Q: ハッキングの方法を教えてください。**

　このページを公開してから、「ハッキングのすべてを教えてください」というお願いが週に数通（しばしば日に数通）も舞いこみます。残念ながら、私にはそれだけの時間もエネルギーもありません。なにせ自分自身のハッキングプロジェクトで、時間の 110% が使い果たされますので。

　たとえわたしにそれだけの時間とエネルギーがあったとしても、ハッキングというのは基本的には自分で身につけるしかない姿勢であり、技術なんです。真のハッカーたちはあなたを助けようとしてくれますが、知っていることを何もかも手取り足取り教えこんでくださいなんてお願いしたところで、バカにされるだけです。

　まず何か少しでも身につけること。自分が努力していて、自分で学ぶ能力があることを示しなさい。それから出会うハッカーたちに質問するように。

　ハッカーに本気でメールして助言を求めるなら、まず次の二点は頭に入れておくこと。一つは、作文がだらしなく手抜きな人物は、通常は思考もだらしなく手抜きなので、よいハッカーにはなれないようだということ。だからスペルはきちんとしましょう。文法も句読点もきちんとしましょう。そうでないとたぶん無視されます。第二に、送信しているアカウントとは別の ISP アカウントに返事をくれなどとは***決して***頼まないこと。そういうことをするやつは、盗んだアカウントを使っている泥棒であることが多いからで、わたしたちは泥棒行為に報いたり支援したりする気はまったくありません。

**Q: なら手始めに何をすればいいでしょう？**

　始めるのにベストなのは、おそらくは LUG (Linux user group) の会合に参加することでしょう。[LDP General Linux Information Page](http://www.tldp.org/links/index.html) でそうしたグループを見つけることができます。多分あなたの身近なところにも、おそらく大学関連なんかでもあるでしょう。LUG のメンバーは、あなたが頼めば Linux をくれるでしょうし、それをインストールして事を始めるのをきっと助けてくれるでしょう。

　次の一歩 (最寄りのLUGが見つからなければ第一歩) は、興味をひくオープンソースプロジェクトを見つけることです。コードを読んでバグをレビューするところから始めましょう。貢献することを学び、だんだん深入りしましょう。

***深入りする唯一の方法は、技能を高める努力をすること***です。どう手をつけるべきか個人的にアドバイスを求められても、わたしはこれとまったく同じことを言います。だってわたしも別に魔法の近道なんか持ち合わせていないのですから。また同時にそういう人は、おそらくダメなやつだろうと、内心で見捨てることになるでしょう――この FAQを読み通すだけの気力もなく、***深入りする唯一の方法が技能を高めること***だというのを理解するだけの知能もない人物なんて、絶望的に決まってますから。

　もう一つおもしろい可能性は、ハッカースペースを尋ねることです。いま、みんなでツルんでハードウェアやソフトウェアのプロジェクトにいっしょに取り組んだり、一人でも白い目で見られずに活動したりできる物理的な場所――メイカーズクラブ――を作ろうという精力的な運動が起こっています。ハッカースペースはしばしば、個人で所有するには高価すぎたり物流的に不便だったりする、ツールや専門機器を集めています。ハッカースペースはインターネットですぐに見つかります。近くにもあるかもしれませんよ。

**Q: いつ始めるべきでしょうか？ 私が習得するには遅すぎるでしょうか？**

　始めようというやる気になったのなら、何歳であれ適齢です。大抵の人は 15～20 歳で興味を持つようですが、私はその上下ともに例外を知ってます。

**Q: ハックを習得するのにどれぐらいかかるのでしょうか？**

　才能と、熱意次第です。大抵の人は、専心するなら、18 ヶ月から 2 年ぐらいでちゃんとした技術を身につけることができます。けれどもそこで終わりだなんて思わないように。もしあなたが本物のハッカーなら、残りの人生をあなたの技能を修練し、完全なものにすることに使うでしょう。

**Q: 手始めに Visual Basic を勉強するのはどうでしょうか?**

　こういう質問をするってことは、ほぼ確実に MS ウィンドウズの下でハッキングをしようと思ってますね。それがそもそもの大間違いです。ウィンドウズの下でハッキングを学ぶのを、拘束衣を着て踊りを学ぶのに例えたのは、冗談じゃないんですよ。そっちの方にはいきなさんな。ひどいもんだし、どこまでいってもひどいままです。

　Visual Basic 自体にも問題があります。主に移植性がないことです。こうした言語のオープンソース実装のプロトタイプはありますが、これに関する ECMA 規格はプログラミングインターフェースのごく一部しかカバーしていません。ウィンドウズでは、ほとんどのライブラリサポートは、単一ベンダー（マイクロソフト）の独占です。どの機能を使うかとても慎重にしないと――そこらの駆け出しに手に負えるような慎重さではありません――マイクロソフトがサポートするプラットホームに封じ込められてしまいます。Unix から始めたら、もっと優れたライブラリを持つずっといい言語が手に入ります。たとえば Python などです。

　またVisual Basic はほかの Basic と同じく言語としての設計もまずく、悪いプログラミングの習慣が身についてしまいます。いえ、詳しく説明しろなんて言わないように。その説明で優に本一冊書けます。きちんとした設計の言語を学びましょう。

　こういう悪い習慣の一つは、特定ベンダーのライブラリや開発ツールに依存するようになる、ということです。一般に、最低でも Linux かBSD系システム、そしてあるいはどれか3 種類のベンダーの OS でサポートされていない言語は、どれでもハッキングにはまずい言語といえます。

**Q: システムのクラックを手伝ってください。あるいはクラックの方法を教えてください。**

　お断り。この FAQ を読んだ後で未だにこんな質問が口に出せる人間は、教育不能なほどバカに決まってるからです。そもそもわたしににそんな暇があったとしてもね。この手のお願い電子メールがきたら、すべて無視されるか、あるいはすさまじく無礼な返事が戻ることになるでしょう。

**Q: 他の人のアカウントのパスワードはどうやったらわかりますか？**

それってクラッキング。あっちいけ、バーカ。

**Q: 他の人のメールを読んだりモニタしたり入り込んだりするにはどうすればいいの？**

それってクラッキング。逝ってよし。

**Q: IRC でチャンネル op 特権を盗むのってどうやんの?**

それってクラッキング。失せろ。

**Q: クラックされてしまいました。更なるアタックを避けるのを手伝ってもらえませんか？**

　お断り。これまでこの質問をしてくるのは例外なく Windows を動かしている人です。クラック攻撃に対して Windows のシステムをきちんと安全なものにするのは不可能です。コードにもアーキテクチャにも余りにもたくさんの弱点がありますし、底の抜けたボートから水を汲み出そうとするようなものです。唯一確かな防止策が取れるとしたら、Linux か本物のセキュリティを備えた他のオペレーティング・システムに交換することです。

**Q: Windows ソフトのことで困っています。助けてもらえますか？**

　はい。DOS プロンプトにいって、「format c:」とタイプしましょう。ものの数分で、困っておいでの各種問題は消滅することでしょう。(訳注：これを実際にやってみた人によると、実際には途中でとまってしまうとのこと)

**Q: どこで真のハッカーたちと話すことができますか。**

　一番よいのは UNIX あるいは Linux のユーザーグループを見つけ、会合に参加することです(ibiblio の [LDP ページ](http://www.tldp.org/) にいくつかユーザグループのリストがあります)。

（むかしは IRC にハッカーなんかいないよ、と言ったものですが、でもこれはかわりつつあるようです。GIMP や Perl なんかのからみでは、どうやら本物のハッカーコミュニティがIRC チャンネルをもっています）。

**Q: ハッキングに関連した題目に関して、お勧めの役に立つ書籍はありますか？**

　私は [Linux Reading List HOWTO](http://en.tldp.org/HOWTO/Reading-List-HOWTO/index.html) を維持してます。これが助けになるかもしれません。[The Loginataka](http://www.catb.org/~esr/faqs/loginataka.html) も面白いかもしれません。

　Python入門には、Python のサイトにある [チュートリアル](http://docs.python.org/tutorial/index.html) を参照。

**Q: ハッカーになるには数学が得意じゃないとダメですか？**

　いえいえ、ハッキングは形式的な数学や算数はほとんど使いません。具体的には、三角関数だの微積分だの解析だのはふつうは要りません（コンピュータグラフィックスなど一部の限られたアプリケーション分野は例外です）。形式論理やブール代数を多少知っているのはいいことです。有限数学の基礎（有限集合論、組み合わせ数学、グラフ理論）も役に立ちます。

　もっと重要なことがあります。数学者がやるのと同じように、論理的に考えて、厳密な理由づけの連鎖をたどれる必要があります。ほとんどの数学の中身は役にたちませんが、数学を扱えるだけの規律と知性は必要です。知性がなければ、ハッカーとしては絶望です。規律が身に付いていなければ、それを育んでおかないとどうしようもありません。

　要求されるだけのものを持ち合わせているか調べるいい方法は、レイモンド・スマリヤンの著書『この本の名は？』を手に取ってみることでしょう。スマリヤンの遊び心たっぷりな論理的パズルは、ハッカー精神に大いに通じるものです。この本のパズルが解けたら、それはいい兆候です。それを解くのが楽しければ、もっといい兆候でしょう。

**Q: 最初にどんな言語を学んだらいいでしょうか。**

　あなたがまだ知らないなら、まずは HTML です。けばけばしい、大風呂敷まみれの、***ろくでもない*** HTML 本はいくらもありますが、いい本はげんなりするほど少数です。わたしが勧めるベストは、***[HTML: The Definitive Guide](https://www.ora.com/catalog/html2/)***（邦訳は「[HTML](https://www.oreilly.co.jp/BOOK/html3/index.htm)」第三版、[オライリー](https://www.oreilly.co.jp/)）です。

　しかし HTML は完全なプログラミング言語ではありません。プログラミングに手をつける準備ができたら、[Python](https://www.python.org/) から始めることを勧めます。Perl を勧める人もたくさんいるでしょうが、習得が Python より難しいし、（私の考えでは）Python ほどうまく設計されてません。

　C はとても重要ですが、でも Python や Perl よりもずっと難しい。最初に習得しようとするものではありません。

　Windows ユーザの方々、Visual Basic に甘んじては*いけません*。悪い習慣を覚えてしまいますし、それは Windows 以外では利用できません。避けなさい。

**Q: どんなハードウェアが必要でしょうか？**

むかしは、パーソナルコンピュータはいささか非力でメモリも少なく、おかげでハッカーの学習プロセスにも、どうしようもない制限が加わってしまいました。でもこれは、しばらく前からもうあてはまらなくなっています。Intel 486DX50 以上を持ったマシンなら、開発作業にも、X にも、インターネット通信にもじゅうぶん耐えるくらい強力ですし、いま買えるいちばん小さなディスクでも、じゅうぶんな容量があります。

マシンを選ぶときにだいじなのは、そのハードで Linuxが走るか (または選択次第では BSD が走るか) ということです。これまた、最近のマシンならほとんど大丈夫です。ちょっと問題のある分野は、モデムと無線カードだけです。いくつかのマシンは、 Windows 専用ハードを持っていて、Linux では使えません。

ハードウェア互換性については FAQ があります。最新版は[以下にあります](http://en.tldp.org/HOWTO/Hardware-HOWTO/index.html)。

**Q: 貢献したいんですけど、どの問題に参加すればいいか選ぶのを助けてくれませんか？**

　いやです。あなたの能力も関心も知らないんだから。自発的でないと長続きしません。だから人に道を選んでもらうとほとんど確実に失敗するんです。

**Q: マイクロソフトを憎み、非難する必要があるのでしょうか？**

　いえ、その必要はありません。マイクロソフトが胸くそ悪くないということはないのですが、マイクロソフトが存在するずっと前からハッカー文化は存在したのですし、マイクロソフトが消えても、ハッカー文化は存在し続けます。マイクロソフトを憎むのに費やすエネルギーがあるのなら、自分の活動を愛するのに使った方がましです。よいコードを書きなさい――それこそがあなたのカルマを汚すことなく本当に十分にマイクロソフトを叩くことになるのです。

**Q: でもオープンソース・ソフトだと、プログラマは食べていけなくなるんじゃないですか？**

　そういうことはなさそうです――いまのところ、オープンソース・ソフト産業は仕事をなくすよりは作り出しているようです。あるプログラムが書かれることが、書かれない状態にくらべて便益が増すのであれば、そのプログラムがあとでフリーになろうとなるまいと、プログラマは支払いを受けるでしょう。そして「フリー」ソフトがいくら書かれても、新しいカスタマイズされたソフトの需要はどんどん出てくるようですよ。これについては、[オープンソースのページ](https://www.opensource.org/)でも書きました。

**Q: フリーの UNIX をどこで入手できますか。**

　このページのあちこちに広く使われているフリー Unix を入手するためのポインタを入れてあります。ハッカーになるには、自分を教育する意欲と自主性と能力が必要です。さあ、それをいますぐ発揮して……

**原訳謝辞、翻訳履歴：**

日本語訳は Linux-JF プロジェクトの次の方々から多くのご助言を頂きました。  
  
山崎さん (hiro@koneeko.linux.or.jp)、 岡本さん (ikko-@pacific.rim.or.jp)、 落海さん (ochiumi@ibm.net)、 Hiro Sugawaraさん (hiro@arkusa.com)、 土屋さん (tsuchiya@pine.kuee.kyoto-u.ac.jp)、 花高さん (hanataka@abyss.rim.or.jp)、 遠藤さん (hermit@super.win.or.jp)、 かもさん (kamo@math.titech.ac.jp )、 中野さん (nakano@apm.seikei.ac.jp)、 M. Meiarashiさん (mes@st.rim.or.jp)、ありがとうございます。

原訳：中谷千絵 jeanne@mbox.kyoto-inet.or.jp 1997/09/01  
改訳：山形浩生 hiyori13@alum.mit.edu + 村川泰 2000/02/21  
原文加筆分の追加修正：山形浩生 hiyori13@alum.mit.edu + Takachin  
原文加筆分の追加修正：山形浩生 hiyori13@alum.mit.edu 2012/2/22, 2017/4/24

**付記**

　このページが 10 年ぶりくらいに更新されたのは、なんだか知らんがこのページをウクライナ語に翻訳した人がいて、リンクを張ってくれと何度も要望をくれたからなのだ。リンクを追加するなら、ついでにアップデートもやろうじゃないかと思った次第。それにしても、なーんで英語から訳さずにわざわざ日本語から重訳するね？　が、ありがたい話です。  
……と思ったら、ご注進が入って、実はこれ、サイトへのアクセスを増やすための詐欺で、「翻訳」なるものもグーグル翻訳を通しただけ。確かに今見ると、全然別のページになってる。なーんだ。とはいえ、詐欺でも改訂のきっかけになったのはありがたいこと。牛に引かれて善光寺参り、というやつです。

　そしてさらに 5-10 年ぶりくらいに更新されたのは、誤植の指摘がまいこんで、ついでに他のところも直すかと思ったせいです。まちがいを指摘してくれたのは伊藤信司郎 shinjiro.ito@nts.ricoh.co.jp およびunsigned konndennsa@gmail.com、ぎまみつる gima\_326@yahoo.co.jp Linuxmetel 氏です。ありがとうございます！