

まえがき

本書はとても乱暴かつイイカゲンな書籍である。なぜなら

- 後々、リファレンスとして使用できるようなかっちりした技術解説が皆無
- 四の五の言わずに動かしてみろ！、課題をこなしてみろ！というだけの内容

だからである。文句があるなら市販のしっかりした本や Web 上の情報を探索されたい。

何でこんな乱暴な本を書いたのか？ それは書いた奴がバカだからとか、書いた奴の知識がイイカゲンで収拾がつかなくなったからとか、書いた奴の性格が強制不可能なほどねじ曲がっているからとか、さまざまな理由があるが、一つだけあげるとすれば、既存のテキストには満足できるものがなかったからである (著者が)。

著者 (幸谷) が、静岡理工科大学にて「UNIX」の講義を受け持つようになったのは 2005 年 (?) からだが、既にその頃から UNIX なるものと、受講する学生さんとの関係が非常に希薄になっていた。

無理もない。

普段使っている PC では Windows が動いているし、メールはケータイで済ますのが普通、Windows の「コマンドプロンプト」の存在すら知らず、CUI ベースの操作なんて無縁に過ごしてきたのだ。当然、この講義の単位を取得したとしても、その後に UNIX を使う機会は、自ら求めない限りあるはずがない。卒業後、自分の仕事として UNIX サーバをメンテし続けるなど、千人に一人、いや、万人に一人がいいところであろう。

そんな学生さんに対して、やれ「ls のオプション」がどーだの、「ファイルパーミッション」がこーだの言ったところですぐに忘れてしまうのが関の山。いや、大学の講義内容なんて覚えている方が珍しいってのは確かだが、せめて「UNIX に触った」だけではなく、「UNIX ではこんなことができた」と、具体的なモノづくりの記憶が残ってほしい・・・と、非力な教員の切なる思いの乱雑な結晶が本書なのである。

では UNIX を使ってモノづくりとは何か・・・、そう、Internet 上のサーバ OS として、21 世紀の今でも過半のシェアを抑えている UNIX は、Web アプリケーションの堅牢な土台ではないか。ならば

Web プログラミング

を最終目標にし、そこに関連する物事だけを抜粋して UNIX のテキスト面をさせてしまえばよい・・・ときわめて乱暴なコンセプトの下に執筆されたのがこの「自習書」なのである。従って、本書の後半の内容を理解するための準備をその前の章で行っている、という構成になっている。全てはデータベースを使った Web アプリケーションを作るため「だけ」に本書は生まれたのだ。

それ故に本書は、大学の講義全 15 回で使用するために 1 講義の内容をコンパクトに詰め込み、課題を自力でこなすことができればオッケーという構成になっている。実習環境は、自分の Windows マシンにターミナルソフト TeraTerm をインストールし、そこから CentOS 5 の Linux サーバにリモートログインするというものを想定している。最近では Windows 上に仮想 PC 環境を作って CentOS をインストールすることも簡単にできるので、それを最初から使うということも考えたが、Web アプリケーションを作る醍醐味は、個別にものを作って自己満足に浸るのではなく、他人が作ったものをすぐに閲覧でき、比較検討ができるというところにある。そのためには誰もが閲覧できる Linux サーバを、たとえば vi というとんでもなく使いづらいエディタしか利用できなくても、常に使うようにしておいた方がいい。ハードウェアが故障しない限り、日々のソフトウェア的メンテナンスはリモートログインして行うというのは、UNIX でも Windows Server でも普通のことである。それがどのようなものか、半年ぐらいは苦しみながら経験するというのも悪いものではないと、著者は信じるものである。

ということで、本書は書いてあることを実際に動かしつつ、何をやっているかをインタプリタのように自分の頭で理解していけば、自ずと「本日の課題」がこなせるようになっているはずである。逆に言えば、本書さえあれば教員がいなくても環境さえあれば最後の「名簿データベース」ぐらいは作れるようになっているはずなのだ。もし出来なれば

- 本書の記述に間違いがある
- 自分の理解の仕方に間違いがある

のどちらかである（両方かもしれないが）。前者ならば表紙のアドレスまでお知らせ頂きたい。後者ならば、あなたの履修態度を何とかして頂きたい。普通に使っていれば UNIX が壊れることはないので、本書の記述をもとにあれこれ試して冒険をしてもらえば、自分がキーボードから入力したコマンドがどのような意味を持ち、どのような結果をもたらすのかは理解できるようになるはずだ。

成績評価と本書の使い方

本書のような「自習書」を執筆した理由は縷々書いてきたが、若年者の学習意欲の低下が叫ばれて久しい昨今において単なる根性論を説いても仕方ない、という事情が大きい。根本的に「学習」という行為を誤解している向きも増えてきたようなので、まず著者が想定する「学習」というものを、図1に示す GUDA プロセス^{*1}で示しておく。

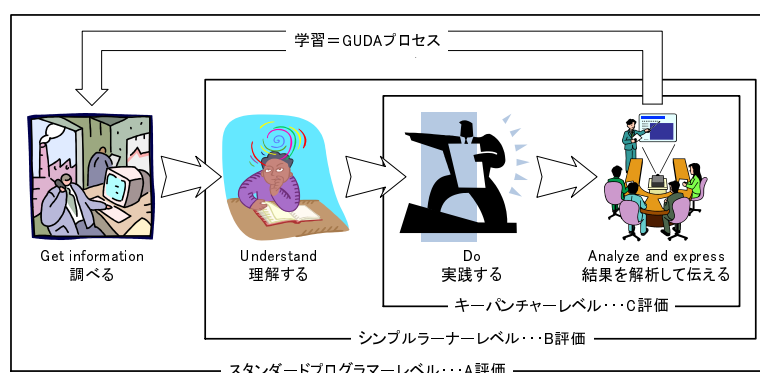


図1 学習の GUDA プロセス

本書で行うべき学習は、「情報を得る (Get information)」→「(情報を) 理解する (Understand)」→「(CUI 操作やプログラミングを) 実践する (Do)」→「(実践結果を) 解析し公表する (Analyze and express)」という GUDA、この4つのプロセス全体を繰り返す行為を意味する。どれか一つが欠けていると、このプロセスがうまく回らなくなり機能不全を引き起こす。成績評価はこの GUDA プロセスに基づいて行う。GUDA の各項目をもう少し本書狙いにつきあわせて解説すると次のようになる。

- (G) **情報を得る** 情報技術は古い時代からの積み上げによって成立した層の厚いものになっており、本書で扱う CUI 操作やプログラミングはそのごく一部をかいつまんて利用しているに過ぎない。「このコマンドのオプションは他にどんなものがあるのだろうか?」「このスクリプト言語でもっと複雑なことはできないのか?」という疑問が沸いたら是非とも情報を検索して欲しい。
- (U) **理解する** 摂取した食料は消化しなければ自分の栄養にならない。収集した情報も自分の脳でかみ砕いて理解しなければ単なるデータに過ぎない。「このコマンドの役割は?」「このスクリプト内のこの記述の意味は?」という疑問に対して解答することができ、既に得た知識と組み合わせて応用展開が図れるようになること、それ

^{*1} 所謂「PDCA サイクル」の真似をして著者が創作した学習サイクルモデル名。

が「理解」の深さをはかる指標となる。

(D) **実践する** 頭で理解できただけの知識は空理空論に陥りがちであり、実際に自分の頭と体を使って実践することで初めて生きた「知恵」に昇華する。「眼高手低」という言葉に象徴される頭でっかちの人間にならないよう、理解した知識を適用して実践することを心がけて欲しい。CUIのコマンドオプションは悩む前に打ち込んでみる、スクリプトの文法は実際にアルゴリズムを記述するために使って動作させてみる、ということは他人から強制される前に、自分から進んで行うものである。

(A) **解析と表現** 実践した結果について、単純に「成功」「失敗」と決めつけるだけでなく、より深い考察を行ってその理由を吟味する必要がある。「なぜこのコマンドはエラーメッセージを出すのか?」「なぜこのスクリプトは期待通りの実行結果を出力しないのか?」という失敗理由の探索だけでなく、成功した場合でも「このコマンドを使うより他のコマンドを使った方がいいのではないか?」「もっと高速な処理が可能ないようにスクリプトや処理系を変更できないか?」という更なる改善を志向することも重要な学習サイクルの要素である。こうして吟味した結果、第三者にその結果について説明し、深い理解を得られるようになる。

上記のサイクルのうち、DA部分のみを真面目にこなせるレベルを「キーパンチャー(打ち込み係)」と称し、単位認定の最低基準とする。UDAまでは何とかできた、というレベルを「シンプルラーナー(単純学習者)」と称して平均レベルとし、本書で解説を行っていない情報を自分で探し出して完成度の高い課題の回答を導くことのできる、GUDAプロセスを完全に習得できているレベルを最高の「スタンダードプログラマー(標準プログラム開発者)」とする。

学習がうまくいっているかどうかは、本書の各章最後に記した「学習チェックリスト」の項目をチェックすれば自ずと分かるはずである。十分理解できていればチェック欄(□)に印を付け、理解していなければ自助努力して資料を探るか、本書の欠点として著者に報告する等のアクションを起こして頂きたい。

まえがきの学習チェックリスト

- 本書が「自習書」と称している理由を理解し、第三者に説明できる。
- 学習という行為がどういうプロセスであるかを理解し、第三者に説明できる。