

WORD

2010.11 From College of Information Science

読者アンケート
書かなイカ?

GRな日々。VJやないか?

雑見沢村に行ってきてゲソ

ICT 中間報告会 2010 でゲソ

use Perl::Object qw(GESO);

WiFi暗号がやばい件についてでゲソ

電子の歌姫はアイドルの夢を見るか-APPEND TRACK-でゲソ

RVMを使っていくつかのRuby実装を使ってみなイカ?

革命的で魔法のようなラクダ。じやないか?

新型MacBook Air購入レポートじやないか?

最近のポケモン事情~育成編~でゲソ

侵略されたいでゲソ号

16

Vol.

目 次

- p. 3 新型MacBookAir購入レポート
- p. 7 電子の歌姫はアイドルの夢をみるか
—APPEND TRACK—
- p. 13 GRな日々。 V
- p. 17 RVMを使っていくつかのRuby実装を
使ってみよう
- p. 25 use Perl::Object;
- p. 31 革命的で魔法のようなラクダ。
- p. 34 最近のポケモン事情～育成編～
- p. 47 ICT中間報告会 2010
- p. 51 雛見沢村に行ってきた
- p. 55 読者アンケート

新型MacBook Air購入レポート

文 編集部 えねりっく

はじめに

こんにちは。みなさん Mac は好きですか？僕は好きです。筑波大学の計算機室では iMac が多く導入され、授業等で Mac を扱うことが多いので、洗脳された好きになってきた方もおられることがあります。中には「Windows?ああゲーム機ね」という方もおられることでしょう。

そんな Mac 好きなみなさんであれば先日行われたイベントである、Back to the Mac はもちろん中継を見たのではないでしょうか。僕も iPhone3GS から見ました。i ナントカの説明のところで寝てしまつたなんてことはありません。当然ながら起きたら MacBook Air の発表が終わっていたなんてこともありません。

まあ、今回のイベントの目玉の一つは先程述べた新型 MacBook Air の発表でした。イベントの以前からリーク情報が挙がり始め、もっと軽い MacBook が欲しい！と祈願していた人たちにとて待望の発表でした。

かくいう僕も新型 MacBook Air に多大なる期待を寄せていました人の一人です。どの程度期待していたかというと、発表があったら即購入しようと思っていた程度には期待していました。

購入経緯

僕は大学に持っていくマシンとして、以前は高校時代に購入したネットブックを使っていました。ネットブックでも Ubuntu なら^{*1} 起動も早く、ファンの音がうるさいことと解像度が低いことを除けばそれほど不満もなくやっていました。

ところが先日、そのネットブックを自宅の床に放置置いていたところ、不注意不慮の事故により液晶が割れてしまいました。その状態で一応頑張って使っていたのですが、元々解像度が低いにも関わらず、さらに液晶割れを放置していたせいで日に日に可視範囲が狭まってゆくというストレスに、最早発狂寸前でした。そんな時に飛び込んできたのが新型 MacBook Air のリーク情報だったわけです。即購入決定だったわけです。

注文

思いっきり「俺が人柱になってやんよ！！！」と宣言していた手前、初期ロットにつきものである不具合の不安などは度外視するしかなく、21日の明け方には AOC^{*2} を利用して購入しました。

僕が今回購入した構成は、

- 11.6inch
- Core 2 Duo 1.4GHz
- RAM 4GB
- フラッシュストレージ 128GB
- US キーボード

という感じです。お値段 108,357 円。約 1,919 リラです。

*1 Windows なんてものはネットブックには存在しなかったのだ……

*2 Apple on Campus の略。この制度を利用すると、Apple Store で通常の学生割引よりも更に割安で Mac のマシン等を購入できる。

新型 MacBook Air

第一印象

段ボール箱の中に、緩衝剤とともに MacBook Air の箱が入っているという形で送られてきました。段ボール自体それほど大きくなかったです。当然 MacBook Air の箱はそれよりもさらに一回り小さく、なかなかコンパクトな印象でした。

箱を開けるといきなり MacBook Air の天板がお出迎えしてくれます。これは iPhone を購入したことがある人ならイメージしやすいと思います。あんな感じです。

実際手に取るとやはり薄いなーという印象を受けました。薄さの割に重さはそれなりにある感じです。本体の剛性はナカナカのものがあると感じました。この薄さを最初に見た人の中には、「(耐久的な意味で) こんな薄さで大丈夫か?」と思った人もいると思いますが、とりあえず大丈夫のようですし、問題もなさそうな感じです。しかしヒンジ部分は 135°までしか開かないようになっているので、人の話を聞かずに無理矢理開こうとすると、ヒンジが破壊されて工場送りになりそうです*3。

最初に MacBook Air を起動すると、ジャーンというやかましい起動音が鳴り、おなじみの(らしい) カッコイイムービーが流れます。

あと、余談ですが付属の USB フラッシュメモリ等が封入されている紙製のケースのようなものを開けると Hello. とあいさつされます。



再インストールはこれで行う

説明書の表紙は爽やかなあいさつ



本体レビュー

● 筐体

MacBook Pro と同じく、アルミ製のユニボディを採用しています。第一印象でも述べたように、本体は意外にしっかりした作りになっています。MacBook Pro などにある、LED のインジケーター類はありません。なのでスリープ状態なのか電源オフ状態なのか見た目にはわかりません。電池残量も起動しないとわかりません。

あと、残念ながら箸としても包丁としても使い勝手はあまりよくないと思います。

● バッテリー

ジョブズ閣下曰くワイヤレス環境下で 5 時間保つそうです。「ん?あれ?某ニュースサイトの直

*3 「神は言っている……ここで死ぬ定めではないと (涙目)」

前リーク情報では 8～10 時間もの連続使用を実現！って書いてあった気が s（r y）と思った鋭い方もおられると思いますが、気にしたら負けです。もう一度言いますが、前モデルの最長連続駆動時間も 5 時間であることなんて気にしたら負けです⁴。

実際自分が使ってみた感じだと正確に検証した訳ではありませんが、画面輝度を 3,4 目盛りくらいに下げ、タブをいくつか開いたブラウザ（Chrome）と Twitter クライアント（YoruFukurou）などを立ち上げている程度では大体ジョブズ閣下が言ったくらい保つかなー、といった具合です。

しかしながら、Firefox を数百タブ開き、Skype や Twitter クライアント、その他のソフトを開いている状態がデフォだろ！なんていってしまうヘヴィなユーザの方は持って 3 時間程度だと思います。

● 使用感

CPU が少し弱いから多少もっさりするかなーと思っていましたが、そんなことよりもフラッシュストレージの恩恵の方が動作には影響が大きいようで、非常にサクサク動いてくれます。起動も終了も Windows で HDD な僕としてはちゃんと終了したのか不安になるレベルのスピードでやってくれます。

バッグにもすっきり入ってくれるので、持ち運びにも最適です。僕が持っている電子辞書より MacBook Air の方が薄かったです。びっくり。動作音もほぼ無音なので、チキンな僕も講義中に「ファン音周りの人の迷惑じゃないかな」などとガクブルする生活から解放されました。大学へ持ってくるマシンとしてもってこいですね！

キーボードも、MacBook Pro のキーボードとほぼ同じサイズのキーなので非常に打ちやすいです。トラックパッドに関しても、流石の出来で全く文句なしの使いやすさです。マルチタッチの認識の精度などは本当にすばらしいと思います。

● 言い訳（Apple 擁護）

リーク情報では Core i 世代の搭載が囁かれていただけに、発表後には「えーマジ Core 2 Duo!? ダサイ、Core 2 Duo なんて時代遅れ CPU 使っていいのは小学生までだよねー キャハハハハ」みたいな声が噴出している様子をネットでよく見かけたので、これを読んでいる方の中にもきっとそう思っている方もいることでしょう。

しかし待ってください。Core i 世代の CPU は超低電圧版モデル（Core i3 330UM）でも TDP⁵ が 18W もあるんですね。今回 MacBook Air に搭載されている Core 2 Duo (SU9400 or 9600) の TDP は 10W です。それに加え、Arrandale⁶ は GPU を内蔵するので、CPU 負荷が低い状態でも、GPU 部分がそれなりに動作しているせいで CPU の消費電力を下げるのが難しいという話も聞きます。ですからこれから先は知りませんが、少なくとも今は消費電力的に考えて Core 2 Duo を採用するしかなかったんです！！多分！！

まとめ

Apple 製品にしてもなんにしても初期ロットというものには不具合が多いと思っていたので、

*4 こ、今回のはワイヤレス環境下での駆動時間だから前モデルとは違うんです！！

*5 設計上想定されるマイクロプロセッサの最大放熱量。「熱設計電力」とも訳される。インテルなどの CPU メーカーはこの値を最大作動時の消費電力とほぼ等しい値であると公表している。

*6 Core 2 の後継マイクロアーキテクチャとして開発されている Nehalem マイクロアーキテクチャのプロセスシーリング版である Westmere をベースとした CPU のコードネームの一つ。

新型 MacBook Air

人柱になるつもりで腹を括り、不具合程度は覚悟していました。しかし、そんな心配は今回の MacBook Air には無用だったようで一安心です。MacBook Air が届いてまだあまり日が経っていませんが、すっかり Apple 信者です ^ ^

ジョブズ万歳！
MacBook Air 最高！

そんな Apple 教に入信した僕ですが、家では Windows 7 マシンを使っています。
Windows もやっぱり必要ですよね！みんな仲良くやっていきましょう！



神々しい

電子の歌姫はアイドルの夢を見るか -APPEND TRACK-

文 編集部 Genyakun

はじめに

前回の記事*を見た方も見てない方もこんにちは。この記事は SEGA から発売されている、「初音ミク -Project DIVA- 2nd」(以下 DIVA 2nd)の攻略記事となっており、前回の記事の追補版にあたります。一応、前回の記事を見なくても出来るだけわかるように説明しますが、ゲームの概要を詳しく知らない方や、ネタバレを嫌う方、前回の記事をご覧になつてない方は WORD 編集部のサイトにある前回の記事を読んだり、ここから数ページを袋とじにしたり、記事の画像だけ見て VOCALOID キャラクター達の可愛らしさを堪能してください。

追加楽曲について

さて、前回の記事でダウンロードコンテンツ(以下 DLC)について少し触ましたが、このゲームには追加コンテンツが用意されています。追加コンテンツには、モジュール(衣装)や新曲、アイテムなどがあり、これらを専用のストアからダウンロードしてゲーム内で使用することができます。ちなみに、原稿執筆時点では次の表にあるようなコンテンツが配信されています。

カテゴリ	名称	価格	配信日
アイテム	発売記念ルームアイテム 1 発売記念ルームアイテム 2	無料	7月 29日
アイテム	ミク生誕記念ルームテーマ ミク生誕記念ルームアイテム(床) ミク生誕記念ルームアイテム(棚) ミク生誕記念ルームアイテム(壁)	各 100 円	8月 26 日
モジュール	初音ミク アpend	300 円	8月 31 日
楽曲	初音ミクの消失 StargazeR	各 400 円	9月 30 日
セット	アイドルマスター コラボセット 1 アイドルマスター コラボセット 2	各 1000 円	10月 28 日
モジュール & アイテム	メイコ 大正浪漫 (MEIKO 親衛隊ハッピ セット)	300 円	11月 4 日
ルームテーマ	メイコ メモリアルブース	100 円	
アイテム	メイコ バースデーケーキ(床) i CUP ドリンク(棚)	各 100 円	

この中で「アイドルマスター コラボセット」と書いてある物で「おや?」と思った方は恐らくアイマスファンの方でしょう。

実は去る 7 月に「アイドルマスター SP」とこのゲームのコラボレーション企画を行うことが発表されていて、このセットはそのコラボの一環として行われた物です。ちなみにセットの内訳は 楽曲・モジュール・アイテムなどで、コラボセット 1 の楽曲は「GO MY WAY!!」、モジュールは「初音ミク 春香 Style」、コラボセット 2 の楽曲は「relations」で、モジュールは「鏡音リン 亜美・真美 Style」と「巡音ルカ 千早 Style」となっています。実はこれ以外にもルームテーマや小物

*1 WORD15 号(非実存 WORD 編集部号)掲載、「電子の歌姫はアイドルの夢を見るか」

Project DIVA 攻略記事第二弾

のアイテムがセットには含まれていますが、紙面上の都合で割愛します。

攻略の前に

さて、実は前回の記事の配布とほぼ同時くらいに追加楽曲が来てしまったため、追加楽曲の攻略を書くことが出来ませんでした。そこで、今記事では追加楽曲の攻略を紹介します。なお、各難易度はゲーム内に表示されている物を使用し、難易度の★は最大で9つまでです。難易度の参考にどうぞ。

9月配信曲の攻略

9月30日に配信されたDLC第一弾は「StargazeR」と「初音ミクの消失」でした。

詳しい攻略は以下をご覧ください。

StargazeR (BPM 195)

本ゲームのアーケード版とも言える「初音ミク -Project DIVA- Arcade」から移植された曲ですが、譜面は新しい物になっているので、Arcadeでやり尽くして飽きた方も楽しめるとお思います。

肝心の難易度は左のようになっていて、標準で入っていた曲の難易度がEXTREMEでも大半をクリア出来る人は何度かやればEXCELLENTが出るはずです。

攻略のポイントとしては、序盤で下図のような「△□□×(長²)」のような三連続ノート³の後数拍開けて来る短い長押し(HOLD)⁴が続くゾーンがあります。最初はここでステージが終了する⁵悲しい展開になることが多いですが、数回かやるとタイミングがつかめるのであきらめずに何度もやってみましょう。



ここを抜けた後は割と簡単な譜面です。さくさくと進めていけるはずなので、後半に備えてソソ

*2 長押しの意味。

*3 流れてくるマーカーのこと。

*4 ボタンを押して、タイミングを合わせて離す。

*5 ソングエナジーゲージという物があり、それが無くなると強制的にプレイが終了しリザルトが表示される。

グエナジーをためておきましょう。

後半では「←← ↑↑ ↓↓ →→」のような同時押し^{*6}が来ますが、この部分については、PSPのスピーカでは聞き取りにくいものの、微妙に音が聞こえるのでそれを参考にしてやるとうまく抜けることが出来ます。また、CHANCE TIME^{*7}の後に、下図のような通常押し(後半は同時押し)とHOLDが組み合わさったゾーンがあり厄介です。最後まで気を抜かないようにしましょう。



筆者がプレイした感想としては、全体的にあらかじめ入っている難易度 EXTREME の「From Y to Y」や「こっち向いて Baby」などにある通常押しと HOLD のコンボや、「サイハテ」のような高度な譜面の読み切り能力が問われる譜面だと思います。

初音ミクの消失 (BPM 240)

二曲目ですが、前作である「初音ミク -Project DIVA-」(以下 DIVA 1st)で幾多もの人の親指を疲労へと追いやった楽曲がついに 2nd へ移植されました。まずは難易度から見てみましょう。

左の表を見てわかる通り、全難易度において標準曲にある難易度の最も高い「初音ミクの激唱」

と同じ難易度となっていて、大変攻略が難しい楽曲になっています。ちなみに、先ほどから引き合いに出している「初音ミクの激唱」は同時押しやホールドをほとんど使用していない連打だけの譜面でしたが、この楽曲は BPM も高い上に 2nd で追加された機能をフル活用した楽曲のため鬼のような譜面となっています。そのため、連打の「初音ミクの激唱」とテクニックの「初音ミクの消失」でツートップの

ボス曲とも言えるでしょう。

さて、実際に攻略に入りたいと思うのですが……実は筆者も EXTREME は未だにクリアが出来ていません。そこで譜面確認用に用意されていると思われるアイテム^{*8}を使ってプレイしました。

*6 矢印ボタンと連動するボタンを同時に押す。

*7 この間はプレイが強制終了されず、ミス数に連動したボーナススコアが入る。

*8 「アシストシステム」というアイテムで、何回でも入力ミス(WORST 判定)を出しても入力ずれ(SAFE 判定)してくれるが、クリア失敗扱いになる。

Project DIVA 攻略記事第二弾

プレイが始まると早速の CHANCE TIME で早めの連打と同時押しながら歓迎してくれます。



ここではソングエナジーゲージは減らないものの、成績やスコアには大きく関わるので、気を抜かないようにプレイします。ここを抜けると譜面がすこし落ち着くので、その部分でソングエナジーを稼ぎましょう。

落ち着いた区間が終わると、また連打に突入します。この連打は単純な連打ではなく同時押しや唐突に別のキーを指示されたりするので、より注意が必要な上に「初音ミクの激唱」の EXTREME ほどではないにしても「初音ミクの激唱」HARD ないしは DIVA 1st の「初音ミクの消失」 HARD 程度の連打速度が要求されます。



筆者がプレイした感想としては、「初音ミクの激唱」よりは難易度は低いものの⁹、未だに高い難易度を誇るボス曲という感じです。しかし、回数を重ねれば「初音ミクの激唱」よりクリアは簡単のようにも思えます。

*9 少なくとも人間の理解出来る範囲の譜面であるという意味

10月配信曲の攻略

10月29日に配信されたDLC第二弾は先ほども説明しましたが、かの有名なアイドルマスターのPSP版である「アイドルマスター SP」^{*10}とのコラボレーション企画として夏頃に発表された「GO MY WAY!!(初音ミク カバー)」と「relations(鏡音リン・巡音ルカ カバー)」の2曲でした。

2曲ともダンス・オケはほぼ原作であるアイドルマスターそのままで、歌い方もかなり自然で納得のいく物でした。~~アイマス信者の視点から見ると、これで1000円なら安い方~~(以下略)

GO MY WAY!!(BPM 180)

EASY	★
NORMAL	★★★
HARD	★★★★★
EXTREME	★★★★★★★★

難易度は左の通りですが、個人的にはEXTREME譜面の難易度は逆詐欺^{*11}で、★7つくらいが妥当じゃないかなと……。

攻略については、前半は(EXTREMEをプレイしたことがあるような人ならば)普通にプレイが出来ると思いますが、CHANCE TIMEに入った直後に突然少々難しいノートが配置されているので、その部分を注意して攻略すれば割合早くEXCELLENTになるでしょう^{*12}。



relations (BPM 142)

EASY	★
NORMAL	★★★
HARD	★★★★★
EXTREME	★★★★★★★★

難易度は左の通りで、「GO MY WAY!!」と同じですが譜面は難易度相当、人によってはそれ以上の出来になっています。

この譜面は最初の方だけ見ると一見簡単そうに思えますが、後半に行くにつれて難易度が上昇していき、最終的にはCHANCE TIMEに入ると次ページの図のような標準収録曲の「サイハテ」EXTREMEにある同時押しと通常押しが混じった譜面へと変化します。

*10 バンダイナムコゲームス(NBGI、元ナムコ)製作

*11 表示されている難易度より体感の難易度が低く感じること

*12 最悪この部分を落としても他がそこそこ出来ていればEXCELLENTを取れることもあります。

Project DIVA 攻略記事第二弾



さらにとどめを刺すためにアウトロ^{*13}の最後の方は明らかに我々をつぶしに来ているような「わけのわからない」発狂譜面へと変化します。



ここさえ（ソングエナジーゲージを使ってでも）抜けてしまえばこちらの勝利です。

おわりに

若干駆け足で曲の概要と攻略のポイントを紹介しましたが、いかがだったでしょうか。全体の傾向としては、どの DLC も標準で収録されている楽曲よりも若干難易度が高かつたり特徴的な譜面が多いため、全曲クリアして飽きてきた人には是非お勧めをしたいと思います。また、本ゲームには DLC 以外にも自分で譜面を作ったり、他のユーザが作った譜面を使ってプレイする「エディットモード」も存在します。こちらには DLC 配信されていない曲などが配布されているサイトがあるため、DLC に飽きたらこちらに手を出すのも悪くはないでしょう。

なお、本記事に掲載したすべての画像の著作権は SEGA 様と Crypton Future Media, Inc. 様、(C) NBGI 様に帰属します。すばらしいゲーム・ソフトウェアを作った各社に心から感謝。

*13 イントロの反対語

GRな日々。V

文 編集部 葡萄酒

SPな日々

さて、連載も五回目となったこの「GR な日々。」、今回は趣向を変えて別のカメラを紹介したいと思う。今回登場するカメラは、この機体だ。



ASAHI PENTAX SP
旭光学工業株式会社
(1964年)

このカメラは、現 PENTAX^{*1} の前身である旭光学から 1964 年に発売された一眼レフカメラである。1957 年に発売された ASAHI PENTAX から派生したシリーズの後継に当たる機種であり、高級機として新規に設計された本機は全世界で約 180 万台を売り上げるというベストセラーアイテムとなった。1960 年に開催されたフォトキナ^{*2} に試作機"ASAHI PENTAX Spotmatic"として発表された際、世界初^{*3} の TTL 露出計^{*4} 搭載一眼レフカメラとして高く評価され、業界に衝撃を与えた。

*1 以降 PENTAX という語が複数使われておりややこしいので、ここで整理しておくことにする。当初旭光学が"ASAHI PENTAX"という機種名のシリーズを売り出し、これが世界的に大ヒットしたため、以降は同社で開発された製品のブランド名を"PENTAX"とした。その後 2002 年に社名を「PENTAX 株式会社」と改名し、2008 年に HOYA 株式会社に吸収合併され、現在では同社の一事業部門として PENTAX というブランド名でカメラなどの光学機器を生産・販売している。こうした複雑な経緯のため、機種名としての ASAHI PENTAX に関しては区別しやすいように AP と呼ばれることが多い。

*2 Photokina - 2 年に一度、ドイツで開催される映像機器の見本市。各メーカーが技術の粋を凝らした新製品を発表する場として 50 年以上前から続いている。

*3 世界で最初に製品化されたのは本機ではなく、東京光学(現トプロン)のトプロン RE スーパー。

*4 Through The Lens の略。明るさを検知して自動的に適切な露光に絞りを調整する機構だが、実際にレンズの中を通った光を測定するためレンズによる露光測定の誤差をなくすことができる。

GR Days V

えた。シャッター速度は1～1/1000、自動絞り時のフィルムはASA 20～1600に対応している。なお、本体の刻印は"SPOTMATIC"となっているが、製品として発売された SP はスポット測光ではなく平均測光となっている。これはスポット測光の扱いが難しいため、あまりにプロ向けに傾倒してしまうのを避けたかったようだ。機種名を試作機の Spotmatic から取って"SP"としたため、フォトキナで一世を風靡した筐体左側"SPOTMATIC"の刻印は外したくなかったのだろう。

発売後も、洗練されたデザインと堅牢な設計から来る実用性を兼ね備えた名機として人気は続き、およそ10年に渡って生産が続けられた。このため現在は中古市場でも容易に入手することができるのも魅力である。実はこの SP、TTL が壊れやすいと言う欠点があるのだが、壊れた機体は中古の中でも「ジャンク」という扱いになるため、さらに安価入手することができる。おそらくレンズ付きで5000円台から、高くて15000円程度で(TTL以外は)完動品が購入できるだろう。私は3年前にこのカメラを祖父から譲り受けたが、製造から40年近く経った現在でも問題無く^{*5}動作している。



AOCO (Asahi Optical Co.) の刻印



標準レンズ Super-Takumar 1:1.8/55

勿論 TTL が壊れた機体で自動絞りはできないため、撮影時には手動で絞りを調節する必要があるが、それくらいはご愛嬌。コンパクトな筐体ながらもズシリと手に来る機械の重みと、ミラーを跳ね上げる感触、デジタルカメラでは味わえないシャッター音を楽しむことができる。ちなみに TTL 以外の機構はすべて機械式のため、電池など無くても撮影が可能。たとえ電子機器の使用が許されない離陸中の機内でも、問題無く撮影が行えるのだ。逆に TTL を動作させる場合は電池入手する必要がある。本来使われていた型の電池

は生産が終了しているので多少工夫する必要があるのだが、その辺は各自で調べて欲しい。

標準レンズは M42 マウント Super-Takmar の 55mm 単焦点、絞りは f1.8～16^{*6}と癖も無く扱いやすいレンズである。GRD2 の 28mm に慣れた私にとって 55mm の視界は別世界なので、たまに SP を手に取るとまた違った世界が見えて面白い。現行のマウントと互換性は無いとは言え M42 マ

*5 例の如く TTL は壊れているが。

*6 上位のグレードに f1.4～16 のレンズもあつたらしい。

ウントのレンズは大量に生産されていたようなので、いくらでも中古で手に入る。かのツァイスも M42 マウントのレンズを生産していたので、ヤフオクなど探せば欲しいレンズが見つかる筈だ。私は標準の 55mm と 200mm 単焦点を一本持っているが、大体この二本があれば困ることは無いだろう。たまに 35mm 位の広角寄りが欲しくなる程度である。

さて、実際にこのカメラを使う際に難しいのがフィルムの装填である。今の一眼レフは自動巻きが主流であるから、そちらに慣れていると手順を誤り悲しいことになるかもしれない、しつかり押さえておこう。

まずは裏蓋の開け方から。現在フィルムが装填されていないのを確認^{*7}したら、巻き戻しハンドルを上に引くと裏蓋を開けられる。フィルムの突起がある方を下に持ち、巻き上げレバーの下

にある軸のスリットにフィルムの先端を差し込む。このとき、あらかじめ少しフィルムを引き出しておくと入れやすいだろう。ローラーの突起にフィルム端の穴を引っ掛けるようにして左に引いて、巻き戻しハンドル下の窪みに押し込み、ハンドルを下へ押し下げる。このときフィルムをしっかりと奥まで押し込まずに、斜めになってしまふと悲惨なことになってしまうので注意しよう。具体的にはフィルムが正しく送られずカメラの中で謎の力が働き、巻き上げ時にズタズタに裂けてしまった。上手くフィルムが入らない場合は、少しハンドルを引き上げてみると入れやすいだろう。上手くフィルムが装填できたら一度だけ巻き上げて正しく動作することを確認し、裏蓋を閉じれば準備完了だ。

本体左上、ロットナンバーと SP の刻印

撮影に関しては、デジタルカメラとそうは変わらない。強いて挙げるとすれば、撮った写真がその場で確認できないことと、ISO 感度が途中で変更できない点くらいだろうか。実際に撮影してみると分かるのだが、その場で出来の善し悪しを確認できないことが写真に対するセンスを研ぎすませてくれる。多いフィルムでも撮影できるのはせいぜい 36 枚、今のデジカメに比べると圧倒的に少ないため失敗は許されない。露出は合っているか、余計な写り込みは無いか、構図は決まっているか——シャッターを切る寸前に思考が脳裏を駆け巡る。何度も撮り直せるデジカメでは感じられない、何物にも代え難い一瞬である。

一通り撮影が終わったらフィルムを取り出そう。本体底面のスイッチを押してフィルムの固定

*7 確認と言っても裏蓋に小窓など無いので、撮影枚数ダイヤルと記憶に頼るしかない。不安ならば巻いてしまうのも手か。

GR Days V

を解除した後、巻き取りハンドルを起こして矢印の方向に回す。回しているうちに手応えが無くなるので、そこまで巻いたらハンドルを上に引いて裏蓋を開け、フィルムを取り出す。あとは店に持つて行って現像してもらえば良い。もちろん家に暗室があるという人は自分で現像するのも一興だろう。何はともあれ、ここまで来れば一段落。あとは写真の仕上がりを楽しみに待つだけだ。

ここまで SP の使い方を見てきたが、確かに銀塩^{*8} カメラにはデジカメに比べて手間や制約が多い。しかし、銀塩でなければ得られない楽しみは間違いなく存在する。現像に出したネガをワクワクしながら光に透かす喜び、写真好きであれば一度味わってみるのも悪くはないだろう。



*8 フィルムのこと。

RVM を使っていくつかの Ruby 実装を使ってみよう

文 編集部 いのひろ

こんにちは

みなさん Ruby を使っていますか？ Ruby と言えば、情報科学類の前身、情報学類を卒業されたまつもとゆきひろ氏が開発し（始め）たプログラミング言語として有名ですが、ご存じだったでしょうか。

いわゆる Ruby と言えば、C で実装された CRuby (MRI; Matz Ruby Implementation) が有名ですが、Java で実装された JRuby、Ruby で実装された Rubinius、C#で実装された IronRuby などいくつかの実装が存在します。Mac 上で動作する MacRuby と呼ばれる実装もあり、Objective-C の API を備えているため、iPhone 上でインタプリタを動かしてみたという事例もあります。

これらは、各言語における膨大なライブラリ (Java や Microsoft .NET Framework) やライブラリ化された既存の資源（コード）を Ruby のシンタックスで利用したり、CRuby よりも高速に処理することを前提に開発されたりしています。

例えば JRuby は Java がクロスプラットフォーム^{*1} であるため、完全に Java で書かれた JRuby も「(Java が動けば) どこでも動くよ」を実現しています。

近年の Ruby には MRI に YARV (Yet Another Ruby VM) と呼ばれるバイトコードインタプリタを導入し処理系の高速化を図っている実装があります。これが Ruby 1.9 系です。Ruby 1.9 の最新安定版 1.9.2 は Ruby 1.8.7 と同じ「Ruby の実行可能な仕様書 (RubySpec^{*2})」を 99% 超クリアしており、本格的な 1.8 系から 1.9 系への移行が進み始めると見えます。

我々もどんどん 1.9 系列を使うべきであります！が、やはり安心と信頼の 1.8 系が欲しい！

本記事は、そのような方々に、rvm をお勧めする記事です。

rvm とは

rvm^{*3} は「Ruby Version Manager」のことと、様々な Ruby 実装を簡単に切り替えて使うことを支援するツールです。

rvm を導入すれば、インストールした Ruby 1.9.2 を使うときは、「rvm use 1.9.2」と打つだけです。また Ruby 1.8.7 を使いたいときは、「rvm use 1.8.7」で切り替えることができます。同じように、JRuby を利用したいときは、「rvm use jruby」とすればすぐに切り替えて使うことができます。

*1 Write once, ~~debug~~ run anywhere

*2 <http://rubyspec.org/>

*3 <http://rvm.beginrescueend.com/>

Ruby Version Manager

導入

今回は COINS ユーザーなら誰でも簡単に利用できるように、計算機室の Mac (Mac OS X 10.6)^{*4} でやってみたいと思います。

rvm をインストールすると、ホームディレクトリ以下に「.rvm」というディレクトリが作成されます。私たちは「rvm install」を使って Ruby 実装をインストールしますが、やることはこのコマンドを打つだけです。rvm がダウンロード、configure、make を実行してくれます^{*5}。生成されたバイナリは、「~/.rvm」ディレクトリ以下に置かれ、「rvm use」を使って任意の実装に切り替えることができます。

早速 rvm を導入してみましょう。Ruby のパッケージを管理する RubyGems からインストールすることができます。

```
$ gem install rvm
```

rvm のインストールは gem コマンドを打っただけでは終わりません。ホームディレクトリ以下の「.rvm」ディレクトリなどを作るために、rvm-install コマンドを実行します。

```
$ echo $PATH
```

して PATH に、「~/.gem/ruby/1.8/bin」が含まれていなかつたら、.bashrc を編集して PATH を追加します。

```
$ emacs ~/.bashrc
```

下記、太字部分を追加します。

```
~~ 省略 ~~  
export PATH=$PATH:~/.gem/ruby/1.8/bin/
```

^{*4} \$ uname -a =>Darwin acacia16 10.4.0 Darwin Kernel Version 10.4.0: Fri Apr 23 18:28:53 PDT 2010;
root:xnu-1504.7.4~1/RELEASE_I386 i386

^{*5} Ruby のインストールは夜中にやるとよいと思います。

保存したら再度読み込み、echo で確認してみましょう。

```
$ source ~/.bashrc  
$ echo $PATH
```

出力に PATH が正しく含まれているか確認します。確認できたら、

```
$ rvm-install
```

を実行して、rvm のインストールを行います。するとバーツと出力されるので読んでみましょう。

まず上のあたり、

```
Installing RVM to /home/ugrad/nm/s0000000/.rvm/  
# 省略
```

Notes for Darwin (Mac OS X)

省略

If you intend on installing MacRuby you must install LLVM first.

If you intend on installing JRuby you must install the JDK.

If you intend on installing IronRuby you must install Mono (version 2.6 or greater is recommended).

MacRuby を使うなら先に LLVM をインストールしなさい、JRuby を使うなら JDK をインストールしなさい、IronRuby を使うなら Mono (2.6 以上推奨) をインストールしなさいとのことです。
次にセットアップの内容が載っています。

- 1) Place the following line at the end of your shell's loading files (.bashrc or .bash_profile for bash and .zshrc for zsh), after all PATH/variable settings:

```
[[ -s "$HOME/.rvm/scripts/rvm" ]] && source "$HOME/.rvm/scripts/rvm"
```

You only need to add this line the first time you install rvm.

Ruby Version Manager

.bashrc、PATHなどの環境変数を設定するスクリプトを追記します。これはすべての PATH の設定が終わった所に追記します。.bashrc に return が含まれている場合（特に Ubuntu など）は、2) に書いてあるように、return を除去し、fi で if 文を閉じた後に追記します。

- 2) Ensure that there is no 'return' from inside the ~/.bashrc file, otherwise rvm may be prevented from working properly.

This means that if you see something like:

```
'[ -z "$PS1" ] && return'
```

then you change this line to:

```
if [[ -n "$PS1" ]] ; then
```

```
# ... original content that was below the '&& return' line ...
```

```
fi # <= be sure to close the if at the end of the .bashrc.
```

```
# This is a good place to source rvm v v v
[[ -s "$HOME/.rvm/scripts/rvm" ]] && source "$HOME/.rvm/scripts/rvm" # This loads RVM
into a shell session.
```

```
EOF - This marks the end of the .bashrc file
```

編集した.bashrc ファイルを閉じ、再度読み込みます。

```
$ source ~/.bashrc
```

これで導入は終わりです。これらは .bashrc が .bash_profile から読み込まれる為に設定が必要ですが、COINS の Mac では既にそのスクリプトが書かれているので、.bashrc の編集だけで問題ありません。

rvm version と打つと、インストールされた rvm のバージョンが表示されます。

```
$ rvm version
```

```
rvm 1.0.18 by Wayne E. Seguin (wayneeseguin@gmail.com) [http://rvm.beginrescueend.com/]
```

いくつかの Ruby 実装のインストール

現在 rvm から利用できる Ruby 実装を、いくつかインストールしてみましょう。まずは 1.8 系の最新バージョンをインストールしてみます。

```
$ rvm install 1.8.7
```

#complete と表示されればインストール完了です。とても簡単ですね。現在手元に導入されている Ruby 実装の一覧は、rvm list で表示することができます。2010 年 11 月現在の最新版がインストールされました。

```
$ rvm list  
rvm rubies  
  
ruby-1.8.7-p302 [ x86_64 ]
```

続いて、Ruby 1.9.2 をインストールしてみることにします。

```
$ rvm install 1.9.2
```

#complete が表示されれば Ruby 1.9 系の安定最新版がインストール完了です。2010 年 11 月現在では、2010 年 08 月にリリースされた 1.9.2-p0 がインストールされます。

次に、Java で書かれた JRuby、Ruby で書かれた Rubinius (rbx) をインストールしてみます。JRuby のインストールには「sun-java6-jdk」がインストールされている必要がありますが、COINS の Mac では既にインストールされていました⁶。

```
$ rvm install jruby  
$ rvm install rbx
```

#complete が表示されたら rvm list して確認してみましょう。

⁶ Ubuntu などで apt-get (aptitude) を利用して sun-java6-jdk をインストールする必要のある方は、筆者ブログ 「<http://d.hatena.ne.jp/InoHiro/20100909/1284012593>」 を参照してください。

Ruby Version Manager

```
$ rvm list  
rvm rubies  
  
ruby-1.8.7-p302 [ x86_64 ]  
ruby-1.9.2-p0 [ x86_64 ]  
jruby-1.5.3 [ x86_64-java ]  
rbx-1.1.0-20100923 [ x86_64 ]
```

JRuby や Rubunius を使いたいときは、「rvm use jruby」や「rvm use rbx」を実行します。またデフォルトで使う実装を設定する事もできます。この場合は、

```
$ rvm use 実装名 --default
```

とります。例えば、COINS の Ruby は「ruby 1.8.7 (2009-06-12 patchlevel 174) [universal-darwin10.0]」ですが、デフォルトで 1.9.2 を設定すれば、再度ログインしたときも rvm use 1.9.2 をいちいち実行する必要はありません。

Ruby のバージョン毎に管理する gem が異なることに注意してください。例えば、Ruby 1.8.7 では Rails 2.3.9 のインストールをしても、Ruby 1.9.2 に切り替えたときに rails コマンドを実行することができません。

```
$ rvm use 1.8.7  
$ gem list  
*** LOCAL GEMS ***  
actionmailer (2.3.9)  
actionpack (2.3.9)  
activerecord (2.3.9)  
activeresource (2.3.9)  
activesupport (2.3.9)  
rack (1.1.0)  
rails (2.3.9)  
rake (0.8.7)  
  
$ rvm use 1.9.2  
$ gem list  
*** LOCAL GEMS ***  
rake (0.8.7)
```

逆に、Ruby のバージョンで使い分けることができると言えます（Ruby 1.8.7 では Rails 2.3 系、Ruby 1.9.2 では Rails 3 系など）。

せっかくなのでベンチマーク

せっかくいくつかの Ruby 実装を導入してみたので、ベンチマークをとってみました。0 から 40 までのフィボナッチ数を求めるコードを実行し、その計算にかかった時間を計ってみます。

```
require 'benchmark'

def fib(n) (0..1).include?(n) ? n : fib(n-2) + fib(n-1); end

puts Benchmark.measure {
  (0..40).each do |i| puts( "#{i.to_s}:#{fib(i)}" ) end
}
```

これを、fib.rb という名前で保存し、実行してみます。以下が結果です。上から順に「Ruby 1.8.7-p1302」「Ruby 1.9.2-p10」「JRuby 1.5.3」「Rubinius 1.1.0」で、結果の単位は「秒」です。

```
$ ruby -v
ruby 1.8.7 (2010-08-16 patchlevel 302) [i686-darwin10.4.0]
$ ruby fib.rb
      user      system       total        real
1.8.7    198.590000    0.040000 198.630000 (198.629436)
```

```
$ ruby -v
ruby 1.9.2p0 (2010-08-18) [x86_64-darwin10.4.0]
$ ruby fib.rb
      user      system       total        real
1.9.2    83.250000    0.040000  83.290000 ( 83.247478)
```

```
$ ruby -v
jruby 1.5.3 (ruby 1.8.7 patchlevel 249) (2010-09-28 7ca06d7) (Java HotSpot(TM) 64-Bit Server VM 1.6.0_22) [x86_64-java]
$ ruby fib.rb
      user      system       total        real
1.8.7    64.420000    0.000000  64.420000 ( 64.369000)
```

Ruby Version Manager

```
$ ruby -v  
rubinius 1.1.0 (1.8.7 release 2010-09-23 JI) [x86_64-apple-darwin10.4.0]  
$ ruby fib.rb  
user      system      total      real  
1.8.7    226.271742  0.054350  226.326092 (226.239436)
```

Ruby 1.9.2 が最速かと思いきや、JRuby が最速でした⁷。Rubinius が Ruby 1.9 系よりも高速であるという記事がありますが⁸、これの通りになりませんでした。

JRuby はコンパイルして Java バイトコードにすることもできるので、それを行えばさらに高速に実行できそうです。

MacRuby も導入してみようとしたが、「rvm install macruby」のスクリプトの中に sudo を実行しているところがあり、一般的な COINS ユーザーだと導入できませんでした。

ヒント : COINS-Admin⁹。

まとめ

rvm を使うと、複数の Ruby 実装の切り替えを超簡単に実現できることがおわかりいただけたと思います。実装ごとのベンチマークや、動作の確認などに非常に便利です。

導入時の設定ファイルを編集するところが面倒ですが、これだけやれば後は簡単です。ぜひ使ってみてください。

*7 ちなみに「user (ユーザータイム)」が単純に「計算時間」。

「system (システムタイム)」が「入出力時間」。「total」は user と system の合計です。

「real (実時間)」は「コマンドを実行してから終了まで」の時間です。

*8 <http://d.hatena.ne.jp/kwatch/20100602/1275459256>

*9 注: COINS-Admin に参加しただけでは sudo 権限はもらえません。

use Perl::Object;

文 編集部 mitty

TIMTOWTDI^{*1}

やり方は一つじゃないよね。

ってことで Perl です。Perl です。大事なことなので（ry
なんで今更 Perl なのかと問われても一言では語れないわけですが、まあ強いて言えばそこに Perl
があるから^{*2}、でしょうか。

対象年齢

```
1.  #! /usr/bin/perl -w
2.
3.  use strict;
4.  use warnings;
5.
6.  print for <>;                                # mycat.pl
```

これを見て何をしているのか^{*3} 理解できることを前提に……するのは読む人を選びすぎている
気がするので、

- プログラミングについての基礎知識を持ち、
- Perl に触ったことは無くても \$ とか ; とか @_ とか変な記号が多いことを知っていて、
- オブジェクト指向の概念はあるけれども実際にコード書くときに使い込んだことは無いよ
ぐらいの方でもなるべく読めるように書いていくつもりです。というか私自身 Perl でオブジェク

*1 "There's more than one way to do it." のことで Perl のモットーとして知られる。

最近拡張され、"There's more than one way to do it, but sometimes consistency is not a bad thing either (TIMTOWTDIBSCINABTE)." になつたらしい。略が長すぎて困る人は"Tim Toady Bicarbonate"つて発音してね、ってウィキペたんが言ってた。

*2 「可愛いは正義」……じゃなかった、「好きだからは真理」はいい言葉だと思います。言ってる
の主に私だけだけど。

例：「男の娘になる理由は何があるんでしょうか？女の子じやダメなんでしょうか？」「だって、
男の娘とか好きだから！」「わあい！」

*3 このコードを「mycat」とか名前を付けて、chmod +x すると*nix での実際の cat っぽくなります。
といつても、本家とは違いファイル名以外の引数を取ることはできませんが。

use Perl::Object;

ト指向プログラミングってあんまりやったことが無いので、自分自身のレベルアップ^{*4}的な意味も込めて。

なので、プログラミングは大学の授業(情報科学類の「プログラミング入門 I」とか)が初めて、という方はちょっと辛いかもです。

準備するもの

今回使用するPerlはPerl 5です。オブジェクト指向の無いPerl 4のこと^{*5}は可及的速やかに忘れてやって下さい。現在絶賛開発中のPerl 6 version 1(^{*})^{*6}については編集部の葡萄酒氏が**素敵な記事**を書いてくれているはず^{*7}なのでそちらを参照のこと。

Windowsユーザの方は、手元にPerlの実行環境なんか無いよという人が大多数だと思いますが、その場合はActiveStateからActivePerlを入手^{*8}してきましょう。Windows上では基本的に

```
D:¥>perl -w hoge.pl
```

*4 「Perl プログラマのレベル 10」なるものがある(<http://d.hatena.ne.jp/naoya/20050809/1123563794>)のですが、これでレベル 2 ~ 3 と判定される方でも、「大丈夫だ、問題ない。」と言える内容を心がけて書くつもりです。というか私自身もせいぜい 6 ~ 7 なので。しかし、「さらにレベル数を大きくしていくと、すぐに該当者が Larry Wall (筆者注:Perl の創始者)だけになります。」というのはあれだな、「本当のPerl処理系は Larry Wall の頭の中にしか無い」ってやつですね。(違)

*5 筆者がPerlに触るようになった2000年頃、Web上でCGIに使われる言語はほとんどがPerlというのが実態でした。もっとも、プロバイダ提供のサーバスペースを含め、レンタルサーバ上で選べるCGI用の言語がPerlぐらいしか無かった、という背景がありますが(無料レンタルサーバだと**CGI機能**そのものが無いことが多い)。そして、オブジェクト指向が導入されたPerl 5系列が初めてリリースされた1994年からかなり経った2000年頃でさえ、CGI用Perlプログラムでオブジェクト指向の初步としてモジュールを「use」しているものすらほとんどありませんでした。というか、当時の一般的な個人サイトでのCGIプログラムの大半は、掲示板やカウンタなどの比較的簡単な目的しか持たなかったためオブジェクト指向を用いるまでもなく、その結果Web上で一番使われる層のPerlのコードは全くオブジェクト指向を用いていないものばかりでした。そのせいか、筆者くらいの年齢層(1980年代前半生まれ、とより上の世代)だと、Perlでオブジェクト指向プログラミングをすると変な(なにそれこわい、的な)顔をされることがままあったのは良い思い出です。

*6 (ほぼ)上位互換だったPerl 4 → 5とは違い、Perl 5のコードは基本的にPerl 6では動かないそうです。伝聞系なのは筆者自身はPerl 6に全く触れていないから。その代わり、Perl 5でオブジェクト指向が後付けなせいで色々とコードがフリーダムだったのが6ではかなり綺麗な書き方になったとのこと。でも、**綺麗なPerl**ってそれってもうPerlじゃないし、そもそも非互換だからいつそ新しい言語として「Perl 6 version 1」とか呼ぼうぜ、と葡萄酒氏と馬鹿な話をしてました。まあ、それでもPerlであるからにはきっと綺麗という言葉の解釈が一般とは違うという落ちなんだろうけれど。

*7え? 葡萄酒なら私の隣(にあるソファの上)で寝てるよ?

……そんなWORDで大丈夫か?(縮め切り的な意味で)

*8 <http://www.activestate.com/activeperl/downloads>

というような形で Perl インタプリタ^{*9} にソースコードを渡して実行することになる^{*10} ので、環境変数 PATH に含まれるところであればどこにインストールしても特に問題は無い^{*11} でしょう。多くの場合 C:\Perl または C:\Perl64^{*12} 以下に展開されます。

Windows 以外の環境、特に Linux や MacOSX などの*nix 系はデフォルトで perl がインストールされていることと思います。

筆者は以下の環境でテストしています^{*13}。

- ActivePerl 5.12.2 build 1202 x86 および x64 on Windows 7 x64
- perl 5.10.1 x64 on Ubuntu 10.04 LTS (Lucid Lynx) x64

x86 と x64 の間では基本的な使い方で差異は無いはずですが、CPAN^{*14} からモジュールを持ってくる際^{*15} などに問題が起る場合があるので、それについては適宜記載します。

目次

予定は未定ですが、とりあえず下記を想定しています。

1. use による既存モジュールの活用と package 宣言
2. Perl でのクラス・インスタンス・メソッド
3. 既存モジュールの拡張・継承
4. POE: Perl Object Environment の紹介
5. POE を使ったマルチタスキング
6. POE によるイベントドリブン生活
7. 並列クローラの作成

途中脇道に逸れたりバグに嵌ったり新たな章が追加されたりタイトルが変わったり内容が変わったり記事が無かったことになったりするとは思いますが、なるべく完結させるつもり^{*16} ですのでどうぞお付き合い下さい。

*9 実際には Perl は実行時にコードをコンパイルする JIT コンパイラ形式になっています。

*10 pl2bat という、Perl のソースコードにヘッダとフッタを付け加え Windows バッチファイル形式に変換するコマンドが ActivePerl には付いてくるので、それを利用して*nix における chmod +x のようなことが擬似的に実現できたりはします。あるいは、ActivePerl のデフォルトのインストールオプションでは拡張子 pl のファイルが Perl インタプリタに関連付けされるため、pl ファイルを開くと自動的にインタプリタが起動する環境にすることもできます。

*11 というより、デフォルトのインストールオプションではインストール先を環境変数 PATH に追加してくれます。

*12 x64 版。

*13 2010/10/31 現在。今後 update があればできるだけ追従していきます。

*14 <http://www.cpan.org/> "Comprehensive Perl Archive Network"の略。Perl にまつわるライブラリ・モジュールその他いろんなコードの巨大なアーカイブサイトです。方向性はちょっと違いますが ruby でいうと RubyGems とかが近いですかね。

*15 特に最新のソースを使って tar.gz から make したりする場合に。

*16 まあ、最初から「打ち切り」にするつもりで書き(or 描き)始める作者は少数派でしょうから。

…… mitty 先生の次回作にご期待下さい！！！ 1

use Perl::Object;

蛇足

と、本来は本記事は「近日連載開始」的な告知ではなくて内容の伴った記事^{*17}にするつもりだった^{*18} ですし、このまま終わるのも寂しいので最初のページに載せた mycat.pl の解説をしながら終わりたいと思います。

まず理解してほしいのが、コードには記述されていない特殊変数を Perl は暗黙的に多用する、ということです。具体的には「\$_」および今回は出てきませんが、「@_」です^{*19}。また、スクリプト起動時に渡されるコマンドライン引数が格納される「@ARGV」も隠れています。これらを省略しない形で mycat.pl を書き直し、また foreach のエイリアスである for を元に戻す^{*20} と以下のようになります。

```
1.  #! /usr/bin/perl -w
2.
3.  use strict;
4.  use warnings;
5.
6.  if (! @_ARGV) {
7.      @_ARGV = ('-');
8.  }
9.  foreach $_ (@ARGV) {
10.     open FileHandle, $_;
11.     foreach $_ (<FileHandle>) {
12.         print $_;
13.     }
14. }
```

mycat2.pl

まず、Perl では制御構文^{*21} が後置できます。このため、元の mycat.pl の 6 行目は print 文に for 文を後置したと解釈されます。if 文や unless 文などの条件文の後置くらいだと他の言語でも結構

*17 そもそも、元々は目次予定の 4 ~ 5 章あたりをまとめた単体記事にするつもりでした。しかし、「『Perl でオブジェクト指向とか無いわー』とか言われないように入門も付けるか」→「クラスとかの説明もしないといけなくね?」→「分量的に連載になるよな」→「いっそ Perl のオブジェクト指向入門記事ってことにした方が(自分にとっても)勉強になっていい気がする」→(ry と膨らんだ結果がごらんの有様だよ!!!

*18 締め切り三日前になつていきなり記事を書くことを思い立った時点で間に合わない予感は十二分だったわけですけどね。

*19 ドル記号 or アットマーク+アンダーバー。これを書いているときに「%_」も暗黙的に定義されていることに気づきましたが、どのように使われているのかが分かりません。「%_」が何に使われているのかご存じな方は是非教えて下さい。

*20 C 言語などの for 文と同じ認識で Perl のループを捉えようと大変なことになりますよ？

*21 if 文や unless 文、foreach 文(for 文)、while 文、until 文などのこと。

見かけるますが、foreach 文や while 文などの後置^{*22} は Perl くらいでしか見かけない気がします。そして、後置した場合のみ、制御構文の条件節の()が省略できます。ここで、ループが一重から二重になってるだろ、とか突っ込みが来そうですが、この程度のことは Perl では良くあることなので諦めて下さい^{*23}。次に「\$_」ですが、Perl では組み込み関数^{*24} での引数の省略時やループ構文などでのイテレータの省略時に、デフォルトの対象変数として「\$_」を暗黙的に使うというお約束があります。しかも、「\$_」に何が入っているかはその時点での文脈による^{*25} ので、実行時にならないと分からぬといふことが良くあります。

この「\$_」が曲者で、Perl のソースコードが Perler^{*26} 以外の人にとってスパゲティコードにしか見えない理由の大半^{*27} はこいつのせいだと個人的には思っています。逆に Perler が Perl から離れられない理由でもあるでしょう。

さて、mycat2.pl のコードを一行ずつ見ていきましょう。1 行目は shebang です。3 行目と 4 行目の「use」はモジュールの読み込みでも使われますが、今回のものは C コンパイラでのコンパイルオプションのようなもので、インタプリタの挙動を変更する「プラグマ」と呼ばれています。「use strict;」は文法などが厳格に適用^{*28} されるようになり、変数のスコープなどを明確に宣言しないといけなくなります。「use warnings;」は文字通り怪しい構文や未定義の変数使用などを警告してくれるようになります。

6 行目以降がコード本体ですが、まず、mycat.pl には存在していなかった if 文が出現している理由を説明しましょう。mycat.pl の 6 行目で使われている「<>」というダイヤモンドみたいなものは「山括弧演算子^{*29}」と呼ばれ、ファイルハンドルを引数にしてファイルの内容を一行ずつ読み

*22 do {} while 構文や do {} until 構文は、もちろん while 文あるいは until 文の後置とは別に扱われます。というより、Perl では do {} while 構文は制御構文ではなく、{}によって囲まれた複数の文を引数とする do 演算子に while 文が後置されると解釈されます。つまり do 演算子に if 文や unless 文が後置された、do {} if 構文あるいは do {} unless 構文なども正しいコードとなります。また、通常の制御構文である while 文などで使われるループ制御のための next 演算子や last 演算子は、do 演算子が制御構文ではないため、do {} while 構文や do {} until 構文において使うことができません。

*23 そういう極端な省略も Perl では普通です。(キリッ)

*24 組み込み命令、と捉えた方が使い方の実態に近い気もします。たとえば「print」は関数なので「print "hoge";」ではなく「print("hoge");」という書き方もできますが、あまりやりません。そもそも「print 文」という呼び方をしますし。

*25 要するに Perl の気分次第です。慣れるしかないね。むしろ慣れてくると快感に(r y)

*26 日常に Perl でコードを書く人たちの総称。

*27 理由の大半、と書こうとしてやめたのは、私くらいでも何とか説明できそうな「\$_」の扱われ方程度では Perl のやばさは全然書き切れないと思ったからです。

*28 とはいっても、Perl の文法のフリーダムとしてみれば不可算無限集合が加算無限集合になった程度の差しかないと思われます。いや、その差ってかなり大きいですけどね。

*29 あるいは「行入力演算子」とも呼ばれます。実際には「readline」関数を呼んでいるので字義通りの「演算子」ではないのですが、「readline(FileHandle)」と記述することはあまり一般的ではなくたいてい「<FileHandle>」のような書き方をされるので「演算子」と言われた方がしっくり来ますね。

use Perl::Object;

込んで返すという演算子^{*30} です。mycat.pl では対象となるファイルハンドルが省略されているため、暗黙的に「open」済みの「ARGV」という特別なファイルハンドルが対象となります。「ARGV」ファイルハンドルは「コマンドラインに指定されたファイル名のリスト@ARGV^{*31} を順にオープンして読み込むための特殊ファイルハンドル」^{*32} として用意されています。そして、「@ARGV」が空、つまりコマンドライン引数が無い場合、暗黙的に標準入力すなわち「-」(ハイフン)が対象となります。従って、mycat2.pl にあるように if 文による「@ARGV」の判定が必要となります。

9 行目の foreach ループでは、「@ARGV」から要素を一つずつ取り出し「\$_」に代入します。そして渡された引数をファイル名文字列として「open」し、「FileHandle^{*33}」ハンドラに結びつけます。

11 行目の内側の foreach ループでは、「山括弧演算子」により読み込まれた一行をまたも「\$_」に代入し、ループ内の「print」関数で出力しています。

最後の行の行末にある「#」以下の部分はコメントです。

以上が例として挙げた mycat.pl および mycat2.pl の簡単な説明です。いかがでしょうか、コードを 10 分の 1 近くまで圧縮できる Perl のシンプルさ^{*34} を分かっていただけたことだと思います。そんな Perl で楽しいオブジェクト指向プログラミングを目指して連載していきたいと思います。

最後に

本記事の内容は、引用部分や素材等、著作権その他の権利が筆者に帰属しない物、或いは個別に但書きされている物を除き、Perl そのものと同じ「Artistic License」^{*35} に従って再利用できます。

また、記事タイトルの「Perl::Object」というモジュールは実際に存在しているわけではありませんのでご注意下さい。

参考文献

- 「プログラミング Perl 第 3 版 VOLUME 1,2」(オライリー・ジャパン)
- 「PERL HACKS プロが教えるテクニック & ツール 101 選」(オライリー・ジャパン)
- <http://www.perl.org/>
- perldoc コマンド

*30 一行ずつではなくファイル全体を一気に読み込む構文も書けますが、今回は省略します。

*31 '@ARGV' に格納されているコマンドライン引数をファイル名が渡されたと見なすわけです。

*32 「プログラミング Perl 第 3 版」 p.780

*33 「FileHandle」は特殊変数でもなんでもない、単なるファイルハンドルです。

*34 from <http://slashdot.jp/developers/comments.pl?sid=429722&cid=1468207>

> シンプルに書かれた Perl

同じ処理をするととして

- ・ 単純な処理を明快に積み重ねて 20 行で書かれた処理(理解するのに 2 分)

と

- ・ 変態じみた正規表現を駆使して 2 行で書かれた処理(理解するのに 20 分)

だと

後者を「シンプル」だと思ってしまうのが Perl 教徒の恐ろしいところ

*35 <http://dev.perl.org/licenses/artistic.html>

革命的で魔法のようなラクダ。

文 編集部 葡萄酒

はじめに

今年の夏、私が愛して止まない Perl の新しいメジャーバージョン、Perl6 がリリースされました。Stable リリースではなく Early Adopters (いわゆる人柱版) という扱いですが、それなりに実用もできるということで紹介したいと思います。

何故 Perl6 なのか

よく「Perl は汚い」と言われるのを耳にします。モダンな Perl の作法は読みやすいだとか、イディオムを知らないから読み辛く感じるのだとか、反論は色々できるのですが、何故 Perl が「汚い」と見られるのかを謙虚に考えてみましょう。

やはり挙げられるのは Perl 独特の「特殊変数」の扱い、あまりに自由すぎる「リファレンス」の存在、初心者には理解し辛い「コンテキスト」などでしょう。今まで Perl は過去の互換性を考慮しつつ場当たり的に拡張を繰り返してきた節があり、リファレンスを駆使したオブジェクト指向周りは本当にカオスになっています。この無駄に複雑なシステムのおかげで色々と黒魔術があったりして楽しいのですが、そのせいで一般的な用途に用いるには抵抗があるのも仕方ない話なのかもしれません。

そこで、そんな状況を打破するために、痛みを伴う改革として互換性を切り捨てた Perl6 の開発が始まったのです。バージョン 6 は Perl の持つ自由さを保ちつつも、既存のバージョン 5 のように後付け感満載なオブジェクト指向ではなく、Ruby のようなクラスベースの完全なオブジェクト指向を目指して設計されました。より簡潔なコードで明瞭に処理が書ける —— そのために根本から文法を見直し、パラダイムを一新し、完全に新しい処理系として Perl は生まれ変わりました。

Perl6 の導入

Perl6 の実装はいくつかあるのですが、本記事では開発が活発な Rakudo Star を用いることにしました。Rakudo Star は Parrot という仮想マシン上で動作する Perl6 の実装です。Java が JavaVM の上で動作する様子を思い浮かべると理解しやすいでしょう。これから、この Rakudo Star の環境を作る手順を説明していきます。

まずは Rakudo Star の配布ページ^{*1} からソースコードをダウンロードしましょう。このアーカイブを開封すると幾つかのファイルとディレクトリがあります。このうち./docs ディレクトリには "Using Perl6" という Perl6 の文法を纏めた解説書が pdf で同梱されています。内容は英語ですが、分かりやすく文法と記法が纏められているので、読む価値は大きいでしょう。

実際の Perl6 本体のソースコードは./rakudo ディレクトリの中にあります。make には Subversion が必要になるので、予め準備しておいてください。実際の make のコマンドは以下の通りです。yyyy.mm はリリースされた年と月^{*2} に置き換えてください。

*1 <http://github.com/rakudo/star/downloads>

*2 本記事執筆時点では 2010.10 が最新。毎月 25 日前後にリリースされることが多いようです。

Learning Perl 6

```
$ tar zxvf rakudo-star-yyyy.mm.tar.gz  
$ cd ./rakudo-star-yyyy.mm/rakudo  
$ perl ./Configure.pl --gen-parrot  
$ make all
```

make が正常に終了すると "perl6" という名前の実行ファイルが生成されている筈です。 -v オプションを付けて実行してみましょう。2010.10 の場合は以下のようなバージョンが表示されれば成功です。

```
$ ./perl6 -v  
  
This is Rakudo Perl 6, version 2010.10 built on parrot 2.9.1 r49584  
  
Copyright 2008-2010, The Perl Foundation
```

それでは、実際に Perl6 を使ってみましょう。オプションや引数無しで Perl6 を実行すると対話的なインタプリタが起動するので、適当な式を入力してみてください。

```
> my Str $test = "hoge"  
hoge  
> $test.chars  
4  
> $test.WHAT  
Str()
```

いかがでしょうか？ Perl5 の文法とは大きく違うので戸惑うかもしれません、残念ながら紙面と時間の都合で文法の説明は割愛します³。

Perl6 のこれから

最初にも書いた通り、まだ Rakudo Star は安定版ではありません。Perl6 自体の起動が非常に遅いために、速度が重要な処理に用いるのもまだ難しい段階です。しかし処理系のバグ自体はリリースを追うごとに減ってきていますし、速度が必要でないシステムであれば十分に実用的だと思います。ただユーザの数が少ないので、コードを書いていて詰まった時に知識の共有が難しいのも現状です。もし Perl6 に興味を持った読者の方がおられたら、是非導入を考えてみてください。

³ もしもこの記事が続いたら、Perl6 入門として文法の解説などやるかもしれませんが……。

オマケ

Perl の得意なワンライナー、文法が充実した Perl6 ではさらに楽しめます。以下に私の書いた拙いワンライナーをオマケとして掲載しておきます。実行時 (-e オプションでも実行できます) に引数にとったファイル(CSV 形式で整数のリストを記述)を読み込み、ソートした結果を同じく CSV 形式で標準出力に吐き出します。ちなみに組み込みのソート関数は使っていません^{*4}。

```
say (repeat {@a = (@a or lines.split(/\n/)) .pick (*)} until [<=] my @a and @a) .join(",")
```

用いたアルゴリズム^{*5} の問題で実行は激しく遅いですが、これだけの記述でファイルオープンから出力まで可能な Perl6 の実力がご覧頂けたかと思います。

参考サイト

以下に Perl6 を学ぶ上で有用なサイトを列挙しておきます。

* <http://perl6.org> - Perl6 公式サイト

* <http://rakudo.org> - Rakudo 公式サイト

* <http://try.rakudo.org> - ブラウザ上で動作する Perl6 のインタプリタ

* <http://perlcabal.org> - Perl6 に関する仕様書や資料などを集めたサイト

* <http://perl6.wikia.com> - 有志による Perl6 の情報をまとめた wiki

*4もちろん、ソート関数を使えばもっと短く高速なワンライナーが書けます。

*5 実装が単純なので、悪名高いボゴソートを使っています。なんと $O(n*n!)$ という驚異的な遅さ。

最近のポケモン事情～育成編～

文 編集部 IX

■はじめに

皆さん、こんにちは。この記事は、『ポケットモンスター廃人日記～厳選編～』の続編となっています。初めての方には難解な用語が出てくると思いますが、記事の最後に専門用語集を用意しましたので、そちらを参照してください。

『ポケットモンスターブラック・ホワイト(BW)』が発売されてから早二ヶ月弱、いかがお過ごしでしょうか。

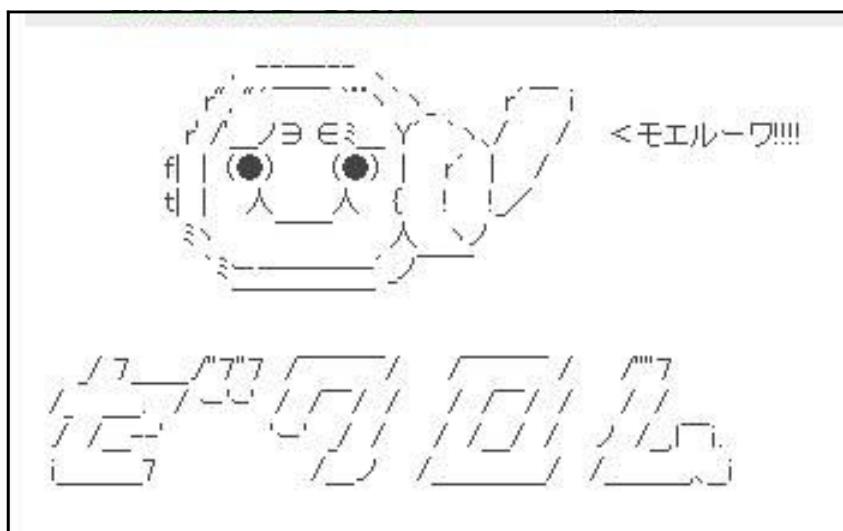
DS の十字キーを左右させる毎日を送っていますか？

地下鉄^{*}に乗って BP を稼ぐ毎日ですか？

ドレディア[†]ちゃんとチュッチュしてますか？

エルフーン[‡]ちゃんとモフモフしてますか？

何はともあれ、今回はポケモンの育成の仕方について、具体例を挙げつつ話していくこうと思います。私、たいした実力もない新米トレーナーであります故、過度の期待はしないでくださいね(はあと)。



オタマロ[§] 氏からの有り難いお言葉。

*1 地下鉄：バトルサブウェイのこと。特定のルールの下、トレーナー達と戦い、バトルポイント(BP)を稼ぐことでアイテムやわざマシン入手できる施設。相手トレーナーの発言がひどいことで有名。

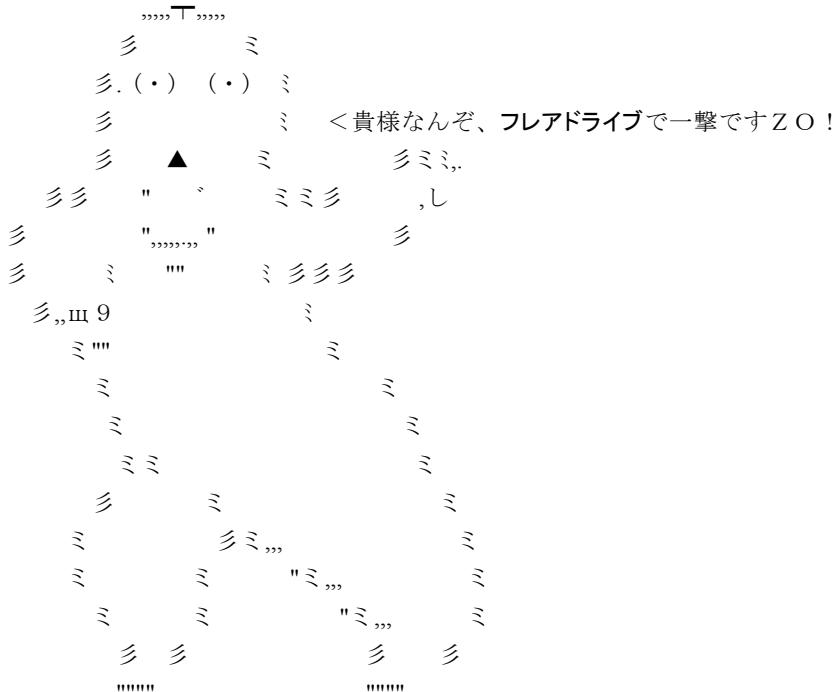
「ゴタクハ イラナイヨ…… タダ ブツツブスノミ……」 by メイドのテツ

*2 ドレディア：はなかざりポケモン。＼ドレディアちゃんマジお姫様！／

*3 エルフーン：かぜがくれポケモン。褐色もこもこもふもふなヒツジっ娘。

*4 オタマロ：おたまポケモン。非常にアレな顔つきをしている。

■育成例1：ヒヒダルマ



== ヒヒダルマ (基本情報)

ヒヒダルマ

タイプ：ほのお

特性^{*5}：ちからずく(追加効果が有る技の追加効果が無くなり威力が1.3倍あがる)

種族値^{*6}：HP:105 攻撃:140 防御:55 特攻:30 特防:55 素早さ:95

かなり初期の段階でグラフィックが公開されていたえんじょうポケモン。

いざ蓋を開けてみれば、攻撃種族値 **140** から特性ちからずくで強化された炎物理^{*7} 最強技を叩き込んで敵をバッタバッタ蹴散らす化けものでしたとさ。

進化前のダルマッカが、ムック^{*8}さんに見えたり見えなかつたり。また、同じ砂漠で出現するズルッグの進化系、ズルズキン^{*9}がガチャピンに見えたり見えなかつたり。なにこの砂漠こわい。

育てやすいうえに強力だから初心者にもオススメなポケモン。

個体値^{*10}も攻撃と素早さだけ粘れば良いし、レベルアップとわざマシンで良い技を覚えられる。

*5 特性：ポケモンが種族ごとに持っている特殊能力のようなもの。詳しくは専門用語集にて。

*6 種族値：ポケモンの種族ごとに決まっている値のこと。詳しくは専門用語集にて。

*7 物理：物理技のこと。ポケモンの攻撃技は、物理技と特殊技に分類され、それぞれ攻撃と特攻のステータスの善し悪しで相手に与えるダメージが決まる。

*8 ムック：「ひらけ！ポンキッキ」のキャラクターですぞ。

*9 ズルズキン：あくとうポケモン。世界末救世主伝説に出てきそうなモヒカン。

*10 個体値：ポケモンの個体ごとに決まっている値のこと。詳しくは専門用語集にて。

BW の噂

== 育成方針 =====
性別：♂

個体値：HP:31 攻撃:31 防御:20~21 特攻:10~13 特防:06~07 素早さ:31

努力値^{*11}：攻撃 252 素早さ 252 HP4

性格^{*12}：いじっぱり(攻撃上昇、特攻下降)

特性：ちからずく

技構成：フレアドライブ じしん いわなだれ とんぼがえり

持ち物：いのちのたま(技の威力を 1.3 倍にする代わりに反動ダメージ) or
こだわりスカーフ(素早さを 1.5 倍する代わりに最初に選んだ技しか繰り出せなくなる)

一致^{*13} ちからずく補正付き “フレアドライブ(炎物理、威力120)” の破壊力は異常。反動ダメージ(与えたダメージの 1/3) も異常。しかも、耐久^{*14} が進化前のポケモンと同程度で無いに等しいとハイリスクハイリターンなポケモン。

今回は典型的な、攻撃素早さ振りの物理アタッカーのヒヒダルマを育成しました。攻撃種族値 140 は全ポケモン中 10 位、伝説のポケモンを除くと 4 位と非常に高く、第五世代^{*15} のパワーインフレを象徴しています。

攻撃に振るのは当然のことですが、先制^{*16} をとれないと何もできずにやられる可能性が十分あります。紙耐久なので、素早さにも振ります。余った努力値は申し訳程度に HP に。

性格は、攻撃が伸びやすく特攻が伸びにくい、いじっぱり。

ちからずくは、使った攻撃技に追加効果があった場合、その追加効果を無効にして威力を 1.3 倍にする特性です。ただ、特性の効果を受けるのは、一定の割合で自分の能力アップ、一定の割合で相手の能力ダウン、一定の確率で異常状態や怯み等を与えるの三つになります。例をあげるならば、10%の確率で火傷の追加効果を与える “かえんほうしや(炎特殊、威力95)” 対象内で攻撃後に自分の特攻を 2 段階下げる “オーバーヒート(炎特殊、威力140)” は対象外です。また、ちからずくが発動すると、いのちのたまによる反動ダメージがなくなる仕様^{*17} があり、技の威力を最大で 1.69 倍まで上げることができます。

*11 努力値：ステータスに作用する経験値のようなもの。詳しくは専門用語集にて。

*12 性格：ステータス画面で確認することができます。性格によって特定の能力が伸びやすかったり、伸びにくかったりします。詳しくは専門用語集にて。

*13 一致：タイプ一致のこと。使うポケモンのタイプと技のタイプが一致すると、技の威力が 1.5 倍になる。

*14 耐久：やられにくさのこと。HP、防御、特防の種族値が高いほど耐久は高くなる。

*15 第五世代：BW のこと。ポケモンの数が増える度に世代が増えていく。

*16 先制：先制攻撃を行うこと。素早さが早いポケモンほど先制しやすくなる。ただし、技の優先度によっては一概に言えない。

*17 いのちのたまによる反動ダメージがなくなる仕様：バグの可能性が高いと言われている。よって、対人戦の際には採用しない方が無難。

今回の技構成でちからずくの対象になるのは、フレアドライブといわなだれの二つ。

フレアドライブは、本来、威力 120 の炎物理技ですが、特性等の補正を考慮すると……

$$120 * 1.5 (\text{タイプ一致による補正}) * 1.3 (\text{特性による補正}) = 234$$

となり、圧倒的破壊力を誇ります。

また、いわなだれも、本来、威力 75、命中率 90% の岩物理技ですが、

$$75 * 1.3 (\text{特性による補正}) = 97.5 \text{ (小数点以下切り捨て)}$$

となり、本来いわなだれより威力の高い**ストーンエッジ**(岩物理、威力100、命中率80%)よりも使いやすい技になります。

また、ほのおタイプの技を半減するほのおタイプといわタイプ対策にじめんタイプの**じしん**(地面物理、威力 100)。どうしようもない相手から逃げるための“**とんぼがえり**(虫物理、威力 70、攻撃後控えのポケモンと交代)”を採用。

持ち物は、ちからずくと組み合わせて爆発的な火力を得られるいのちのたま、もしくは、立ち回りが制限されるが素早さが 1.5 倍になる**こだわりスカーフ**、のどちらかが候補になるでしょう。

■育成例 2：ブルンゲル

まずは、ブルンゲル(♀)のことがよく分かるかもしれないコピペをご覧いただこう。

アタシの名前はブルンゲル。モテカワポッチャリちょすい体質ののろわれボディ♪

アタシがつるんでる友達は合唱団をやってるガマガル^{*18}。子供にナイショで秘伝要員として働いてるスワンナ^{*19}。訳あってパンツをはいていないヒヤッキー^{*20}。友達がいてもやっぱり野生はタイクツ。今日もヒヤッキーとちょっとしたことで口喧嘩になった。水タイプ同士だとこんなこともあるからストレスが溜まるよね☆そんな時アタシは一人で4番道路を泳ぐことしている。

のろわれた自分へのご褒美ってやつ？自分らしさの演出とも言うかな！

「あー肩重い」・・。そんなことをつぶやきながらしつこいトレーナーを軽くあしらう。

「カノジョー、ちょっとボールに入ってくれない？」どいつもこいつも同じようなボールしか投げてこない。

トレーナーの男は海パンはいてるけどなんか汗臭くてキライだ。もっとゴーストタイプのアタシを見て欲しい。

「やせいの ブルンゲル があらわれた！」・・・またか、とちょすいなアタシは思った。じこさいせいするつもりだったけど、

チラッとトレーナーの男の顔を見た。

「・・・！」

・・・チガウ・・・今までのトレーナーとはなにかが決定的に違う。ギガインパクト^{*21}な感覚がアタシのカラダを

駆け巡った・・。「・・・(カッコイイ・・・！！！・・・これって運命・・・?)」

男は主人公だった。努力値稼ぎ^{*22}を利用された。「キャーやめて！」あまごいをきめた。

「かみなり！バリバリダー^{*23}！ガッシ！ボカッ！」アタシは死んだ。しめりけ（笑）

*18 ガマガル：しんどうポケモン。オタマロの進化形。バナ○マン○村に似ている。

*19 スワンナ：しらとりポケモン。座んな。南斗水鳥拳伝承者ではない。

*20 ヒヤッキー：ほうすいポケモン。川^ω^川シ こんな感じ。たぶんはいてない。

*21 ギガインパクト：ノーマル物理、威力 150。ただし、使った次のターンは反動で動けない。ここでは、運命的な出会いを暗示していると思われる。

*22 努力値稼ぎ：文字通り、努力値を稼ぐこと。詳しくは専門用語集にて。

*23 バリバリダー：ホワイトのパッケージを飾る伝説のポケモン、ゼクロムの鳴き声。ちなみに、ブラックのパッケージを飾る伝説のポケモン、レシラムの鳴き声はモエルーワ。

== ブルンゲル (基本情報) =====

ブルンゲル

タイプ : みず/ゴースト

特性 : のろわれボディ(攻撃を受けた際に 30%の確率でかなしばり状態にする)

ちよすい(水タイプの攻撃を受けるとその攻撃を無効化し、自分は HP1/4 回復)

種族値 : HP:100 攻撃:60 防御:70 特攻:85 特防:105 素早さ:60

BWにおいて、旧作のメノクラゲ系のポジションをつとめる、ふゆうポケモン。進化前のブルリルは海上でわんさか出現します、というかブルリルしか出現しません。

他のゴーストタイプと同様と図鑑の説明が無駄に怖いことで有名です。

■■ポケットモンスターブラック ■■

ブルンゲルの じゅうきょに まよいこんだ ふねは しづめられて のりくみいんの
いのちは すいとられて しまうのだ。

□□ポケットモンスターホワイト□□

からだに すいこんだ かいすいを いきおいよく ふきだして すすむ。
せいめいエネルギーが だいこうぶつ。

また、♂の外見と♀の外見がかけ離れていることでも有名。♂が青く、王様のような外見をしている一方で、♀は全体的にピンクがかったり、なんかこう……膨れあがったケバい化粧をした魔女みたいな見てくれで好みが分かれる感じになっております。

== 育成方針 =====

性別 : ♂

個体値 : HP:31 攻撃:28~39 防御:31 特攻:12~13 特防:31 素早さ:25~26

努力値 : HP252 防御 252 素早さ 4

性格 : ずぶとい(防御上昇、攻撃下降)

特性 : のろわれボディ

技構成 : なみのり おにび ナイトヘッド じこさいせい

持ち物 : ゴツゴツメット

八つの耐性 (半減 : 炎、水、氷、鋼、毒、虫 無効 : ノーマル、格闘 (ちよすいの場合は水も))
を持ち、高い耐久性能を誇るが、一方で弱点が四つ (草、電気、ゴースト、悪) が多いのが特徴。

今回は、物理受け^{*24} のブルンゲルを育成してみました。種族値を見る限り、防御く特防なので、特殊受けに特化したほうがいいと思う方もいるかと思います。

しかし、このブルンゲルは HP と特防の種族値が高めなので努力値を振らなくとも特殊耐久は十分といえるのです。なので、防御に努力値を振り物理耐久を強化することでより居座りやすくなるのです。まあ、おだやか(特防上昇、攻撃下降)HP 特防 V^{*25} の個体を粘っていたときに偶然入手できたからというのもあります。

まあ、物理アッカーが多いタイプに耐性があるので、十分実用的であると言えると思います。

のろわれボディはなかなか強力な特性で、**攻撃を受けた際に 30%の確率でかなしばり状態^{*26} にする**といったものになっています。大抵のアッカーはブルンゲルに対する有効打を一つしか持っていないことが多く、相手をまったく打つ手がない状態にさせることができます。

技構成は、**おにび**、**じこさいせい**、**ナイトヘッド**、**なみのり**。

まず、**おにび**は相手を火傷状態にする技です。火傷状態になったポケモンは攻撃のステータスが元の値の半分になり、毎ターン終了時に最大 HP の 1/8 のダメージを受けるようになります。相手の攻撃を半減させることによって、物理耐久がより硬くなります。また、攻撃面に努力値を振らないので、毎ターン固定ダメージが入るのはうれしい。ただし、**おにび**は命中率が 75%と少し低いのが不安なところ。更に、**こんじょう^{*27}** という特性を持つポケモンを火傷状態にすると、攻撃のステータスダウンを無効にしつつ攻撃が 1.5 倍になってしまないので注意が必要です。つぎに、**じこさいせい**。この技は文字通り、最大 HP の半分を回復する技です。相性にもありますが、“相手の攻撃を耐える→じこさいせいで回復→のろわれボディ発動→相手有効打なし”となればしめたもんです。

残りの二つは攻撃技になります。**ナイトヘッド**は自分のレベル分のダメージを与えるゴーストタイプの技で、**おにび**同様貴重なダメージソースになります。**なみのり**はタイプ一致の特殊技で、火傷状態にならないほのおタイプに有効な技になります。

持ち物には、相手が接触技^{*28} をしてきた際にダメージを与える**ゴツゴツメット**を採用。やはり、攻撃面のステータスに関係なくダメージを与えられるのは便利なのです。

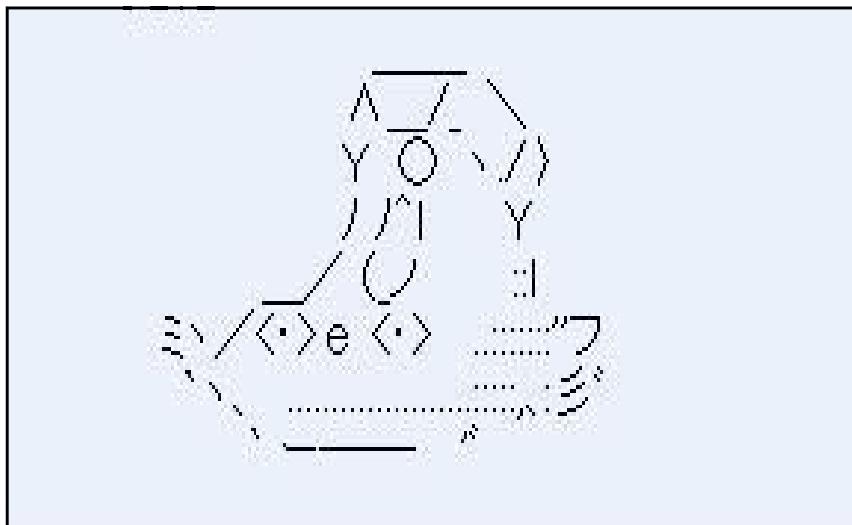
*24 受け：守備寄りのポケモンおよび型のこと。物理技で攻めるポケモンを高い HP と防御で対処とするポケモンを物理受けといい、特殊技で攻めるポケモンを高い HP と特防(特殊防御)で対処とするポケモンを特殊受けという。ただし、どちらか一方のステータスが低くても努力値の振り方次第で十分に受けは可能です。

*25 V：個体値(0~31)MAX のこと。31 を 32 進数で表すと V となる。

*26 かなしばり状態：相手が最後に使った技を 4 ターン封じる技、かなしばりを食らった状態のこと。

*27 こんじょう：異常状態になると攻撃のステータスが 1.5 倍になる特性。元々攻撃の高いポケモンが持っていることが多い、発動されると手に負えない。

*28 接触技：直接攻撃のこと。相手に直接触れる技が該当する。ちなみに、のろわれボディは直接攻撃でなくとも発動する。

■育成例 3：マッギョ

マッギョ 「君は卑怯者だな」

== マッギョ (基本情報) =====

マッギョ

タイプ：でんき/じめん

特性：じゅうなん(麻痺状態にならない)

せいでんき(触れた相手を麻痺させことがある)

種族値：HP:109 攻撃:66 防御:84 特攻:81 特防:99 素早さ:32

BWにおいて、流星のごとく現れたなんとも言えないシュールな見た目のトラップポケモン。図鑑の説明によると、相撲取りに踏まれてももろともせず、放電する際は笑顔になるとのこと。

その某さくらももこ原作漫画の某玉ねぎ頭に似ている容姿から「なが○わ」と NN をつけるトレーナーが後を絶たないとか。あと「マツ○ DX」とか。

某掲示板には専用スレが立ち、GTS²⁹に流せば光の速さで交換が成立する人気っぷり。

マッギョを好んで使用するトレーナーたちはマギョラーと呼ばれ、日夜、マッギョを愛で、研究し、GTSに放流し続けているのである。

*29 GTS：グローバルトレードステーションのこと。ニンテンドー Wi-Fi コネクションを使って世界中の様々な地域の人とポケモンを通信交換できる施設。

BW の噂

== 育成方針 ==

性別：♂

個体値：HP:31 攻撃:31 防御:31 特攻:31 特防:28 素早さ:27

努力値：HP192 攻撃 252 防御 60 素早さ 4

性格：いじっぱり(攻撃上昇、特攻下降)

特性：じゅうなん

技構成：でんじは じわれ いわなだれ メロメロ

持ち物：たべのこし

電気と地面を組み合わせたまったく新しい複合タイプの持ち主なのだが、弱点(地面、水、草、氷)が有名どころばかりなのがつらいところ。

しかし、耐性(半減：飛行、毒、岩、鋼 無効：電気)も悪くなく、電気タイプの中では屈指の耐久を誇り、受けに必要な技もそろっています。

しかも、攻撃範囲が広く、基本的に特殊メインだが、両刀^{*30}も可能となかなか多才。

今回育成したのは、受けではなく、俗にまひるみと呼ばれる博打型。でんじはで相手を麻痺状態^{*31}にし行動を制限。そして、攻撃後 30%の確率で相手をひるませるいわなだれでずっと俺のターンをやるのが理想。はまれば、相手になにもさせることなく倒すことができるわけです。

本来、特性はせいでんきの方がいいとされるのですが、麻痺をまくこと前提のこの型では、でんじはを選択したターンに、相手が接触技で麻痺ってしまうと無駄なターンが発生してしまうためじゅうなんでもよいでしょう。言えない、4Vの個体が出たから妥協したなんて口が裂けても言えない。

努力値の振り方は、攻撃に全振りし、HP を 16n 調整^{*32}、素早さをマヒ状態最速 130 族抜き調整、残りを防御といった配分。

いわなだれを主体にするため、攻撃に 252、たべのこしを持たせた際の回復量を増やすために HP を 16n 調整の 192、素早さの個体値が 27 だったので、麻痺状態の最速 130 族を抜ける 51 になるように 4 振ります。余った努力値 60 を防御に振ることで “HP:防御:特防 = 2:1:1” となるようにしバランスの良い耐久に。

*30 両刀：物理アタッカー、特殊アタッカーを同時にこなすこと。エロい意味は無い。ちなみに、物理受け、特殊受けを同時にこなすことを両受けをいう。もちろん、エロい意味はない。

*31 麻痺状態：麻痺状態になったポケモンは素早さが 1/4 になり、1/4 の確率で行動が出来なくなります。

*32 16n 調整：たべのこしの回復量が最大 HP の 1/16 なので、HP を 16 の倍数(今回は、208)になると回復量が増え、お得です。

技構成は、まひるみを行うためにでんじはといわなだれ。また、命中率 30%の一撃必殺技じわれを自力習得することが可能なので搭載。ロマン度がさらにアップ。そして、どうしようもできない相手にはメロメロ^{*33}で魅了し、その隙に交替するなりする。

持ち物は、前述したとおり、たべのこし。まひるみ型ははまれば、結構なターン数居座れるので、たべのこしとの相性はなかなかよいのです。

■終わりに

いかがでしたか。わけがわからなかつたでしょう。大丈夫、私もよくわかりません。ただ、一つだけ言えます。こっちは来るのにはあまりおすすめできない、と。

今現在、BW の発売から三ヶ月も経っておらず、戦法等は手探りの状態です。対戦環境も十分に整っているともいえず、バトレボ^{*34}の最新作が出るのを待つばかりです。

今回、具体例にあげた三体は、地下鉄で絶賛活躍中です。先鋒のマッギョが麻痺を撒きつつ相手を攪乱し、後続を高火力アタッカーのヒヒダルマや物理受けブルンゲルが処理する。実に安定して16BP^{*35}稼ぐことができます。

さて、私はこれからドレディアちゃんとキャッキヤうふふする系の仕事があるんで失礼しますね。

*33 メロメロ：自分と相手の性別が異なっていた場合、相手を 1/2 の確率で行動不能にするメロメロ状態にする技。マッギョがやると相手への精神ダメージはすごいことになる……と思う。

*34 バトレボ：ポケモンバトルレボリューションのこと。ポケモンスタジアムの進化版

*35 16BP：バトルサブウェイで最も簡単とされるシングルトレインで 21 連勝した際に手に入る BP の総数。

■ ■ 専門用語集 ■ ■

ポケモンのステータスは、**種族値**、**個体値**、**努力値**、**性格**の四つの要素で決まっています。ここでは、その四要素の解説を行っています。

また、性格以外の要素はゲーム中では確認できないので、少なからず google 先生のお世話になることとおもいます。

■種族値

「種族値」とは「種族ごとに決まった値」のことを指します。たとえば、イワーク^{*36} の攻撃はポッポ^{*37} と一緒にとかイワークの素早さはパルシェン^{*38} と同速といったポケモンごとの特徴は、この値で決まっています。

例としてイワークの種族値は、

HP:35 攻撃:45 防御:160 特攻:30 特防:45 素早さ:70

となっています。

これより、イワークは「HP、攻撃、特攻、特防は低いが、素早さはそこそこで、防御が非常に高いポケモン」ということができます。

■個体値

「個体値」とは「個体ごとの値」のことを指します。要は「才能」です。六つのステータスすべてに存在していて、基本的にこの値が高いほどそのポケモンは強いという扱いになります。

■努力値

ポケモンは、戦闘で相手のポケモンを倒した時に得られる経験値によってレベルがあがり、成長していきます。実はこのとき、経験値以外に努力値と呼ばれる値を得ているのです。

「努力値」とは、ステータス版の経験値のようなものです。ステータスの伸び方に作用する値で、努力値を稼いだステータスは稼がなかった場合に比べて大幅に上昇します。

ただし、努力値には上限が存在し、一律 510 までとなっています。さらに一つのステータスにつき稼ぐことのできる努力値の上限は 255 までとなっています。

この流れで行くと、二つのステータスに 255 ずつ振れば良いように思えます。しかし、努力値は 4 の倍数ごとにステータスの伸びに作用するため、252 が実質の最大値になります。

つまり、二つのステータスの 252 ずつ振り、残りの 6 をそれ以外の能力に振る^{*39}のが、一番単純な振り方になります。これは、**極振り(厨振り)**と呼ばれます。他にも○○の攻撃を耐え、さらにこちらの攻撃で□□を一回で倒せるように振るといったような振り方もあります。これは、**調整**と呼ばれます。

*36 イワーク：いわへびポケモン。全ポケモン中最大だったのも今や遠い昔の話。

*37 ポッポ：ことりポケモン。内閣総理大臣だったのも今や遠い昔の話。

*38 パルシェン：2まいがいポケモン。BW にて覚醒。今や対戦では引っ張りだこ。

*39 振る：努力値を振り分けること。

倒したことによって得られる努力値はポケモンごとに決まっており、素早さの努力値を 1 持っているコイキング^{*40} を倒し続ければ、素早さのステータスの伸びが良くなるといった感じです。

また、努力値を稼ぐのを補助するアイテムが存在します。以下の三つになっています。

・ドーピング系アイテム

マックスアップ、タウリン、プロムヘキシン、リゾチウム、キトサン、インドメタシンの六つを指します。それぞれ、HP、攻撃、防御、特攻、特防、素早さの努力値を、一つにつき 10 稼ぐことができます。

しかし、一つのステータスにつき最大で 10 個までしか使えないという制限があります。このため、ドーピング系アイテムだけで努力値を稼ぎきることはできない場合が多いです。

また、これらのアイテムが一つあたり 9,800 円と高価であることも注意が必要です。まあ、所持金の上限が 999,999 から 9,999,999 に変更された今作なら問題ないかもしれません。

・きょうせいギプス

このアイテムはポケモンに持たせることによって、戦闘で相手を倒したときに得られる努力値を倍にすることができるアイテムです。ただし、持たせたポケモンの戦闘中のすばやさが下がってしまうので注意が必要です。

・パワー系アイテム

厳選の際に個体値遺伝用に用いたこのアイテムですが、きょうせいギプスの上位互換にあたる存在でもあります。このアイテムはポケモンに持たせることによって、戦闘で相手を倒したときに得られる努力値に+4 した値を得ることができるアイテムです。

ただし、きょうせいギプス同様、持たせたポケモンの戦闘中のすばやさがさがってしまうので注意が必要です。

さらに、低確率ですがポケモンがポケルスに感染することがあります。この状態になると、デメリットなしで得られる努力値が倍になります。ちなみに、ポケルス完治後でも得られる努力値はデメリットなしで倍になります。

つまり、ポケルスに感染したポケモンがパワー系アイテムを持って努力値 1 のポケモンを倒した場合、

$$(1+4) * 2 = 10$$

一気に、努力値を 10 稼ぐことができるわけです。

*40 コイキング：さかなポケモン。BW では初代同様 500 円で売られている。

BW の噂

■努力値稼ぎ

文字通り、努力値を稼ぐことです。どのステータスの努力値を稼ぎたいかによって、稼ぎ場所が異なります。

== おもな稼ぎ場所一覧 ==

HP :	リバティガーデン(ビクティニ)、8番道路水上(マッギョ)
攻撃 :	1番道路(ミネズミ、ヨーテリー)、10番道路(ワシボン、バップロン、ハーデリア)
防御 :	ヤグルマの森(クルミル、フシデ)
特攻 :	タワーオブヘブン(ヒトモシ、リグレー)
特防 :	17番水道(プルリル)
素早さ :	1番道路水上(バスラオ)

■性格

性格は、つよさを見る画面で確認することができます。全部で 25 種類存在します。例えば、このイワークはわんぱくな性格だから防御が伸びやすく、特攻が伸びにくいとか、このポップはおくびょうな性格だから素早さが伸びやすく、攻撃が伸びにくいといったように、HP 以外のステータスの伸びやすさ及び伸びにくさを知ることができます。

ICT 中間報告会 2010

文 編集部 はるべり

写真 編集部 葡萄酒

はじめに

皆さん、ICT とは何の略かご存知ですか？

ご存知、無いのですか！？

ICT とは Information and Communication Technology の略で、日本語で言うと情報通信技術を表す言葉です。そして今回皆さんにお伝えするのは、コンピュータサイエンス専攻^{*2}で実施されている「ICT ソリューションアーキテクト育成プログラムの中間報告会」の様子です。

この中間報告会は 10 月 10 日に大学会館のとなりにある国際交流会館で行われました。何がめでたいのか、この日は休日だというのに、学内は屋台やステージなどで賑わっていて、それはもう人が ■■■ のようでした。

プロジェクトの一覧

ペット検出のための Web 画像を用いた学習サンプルの効率的な収集
未来のキッチンプロジェクト”Kitchen of the Future”～親子間のコミュニケーションと学びの場としてのキッチン環境の構築～
モータ駆動型ハイブリッド移動システムの制御検証環境の開発
GPGPU 向け前処理の開発
照明条件の能動的制御に基づく指文字認識の高精度化
ネットワークを経由した FPGA の動的部分再構成とその応用
バネモデルを用いたポップアップカード設計支援ツールの開発
関数クロージャと継続の永続化機構の実装
マルチタッチ可能なタッチパネルに適した直観的な数式入力機能と数式変換システムの実装
GPU による多倍長精度 BLAS の開発
超並列計算向け行列固有値解法の汎用ライブラリの開発
物体認知に関する統合的シミュレーション環境開発
検索キーワード非依存型生成手法によるスニペットのパーソナライズとその表示機構
拡張現実感インターフェースを用いた評判情報可視化システムの開発
構造型集約多重署名方式の開発

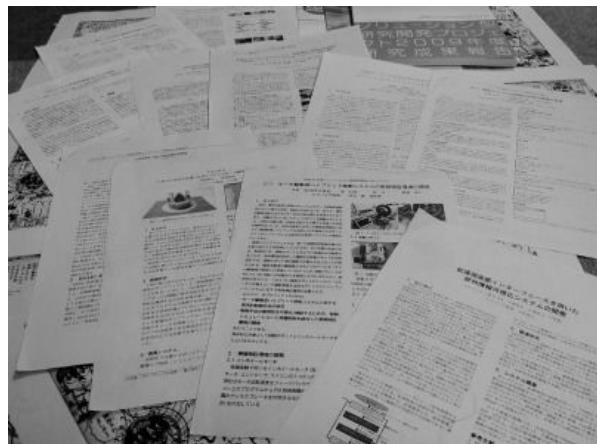
*1 マクロス F の名ゼリフらしい。ICT 記事のテンプレです。

*2 筑波大学大学院 システム情報工学研究科 コンピュータサイエンス専攻

本編

前項の表にプロジェクトの一覧をまとめました。ここではいくつかのプロジェクトについて簡単にご紹介します。

プロジェクトごとに研究内容をまとめた資料が配布されていましたが、~~その資料も~~ 正直何言ってんのかわからない部分があるので^{*3} 紙面の都合もあるので、ここではその一部を簡単に紹介させていただきます。



拡張現実感^{*4} インターフェースを用いた評判情報可視化システムの開発

評判分析というものがあります。これは特定の商品がネット上でどのような評判を受けているかを分析する技術です。これを使えば気になる商品の評判を簡単に調べることができます。

この研究では、携帯電話のカメラを使ってお店の看板や商品名などの文字列を読み取り^{*5}、Twitter 上での評判を調べて表示するシステムを開発しています。キーボードを使わないので入力がスムーズで、簡単に情報を得ることができます。現在は試作品として iPhone アプリで実装されています。

GPU による多倍長精度 BLAS^{*6} の開発

PGP^{*7} で多倍長演算を行う数値計算ライブラリです。GPU はもともと画像処理の演算のために作られたプロセッサなので、これを用いることにより、行列・ベクトル演算を大幅に高速化することができます。

このライブラリでは 256 ビット浮動小数点数演算や、可変長任意精度演算などをサポートするようです。また、GMP^{*8} のデータ型を扱えるようにするらしいので、既存のライブラリとの親和性もあります。

*3 筆者は線形代数 I を習ってる最中なのに近似逆行列がどうこうとか言われても難しかった。筆者にも分かるようにスカイガールズで例えてほしかった。

*4 拡張現実感(AR)とは、簡単に言えば携帯電話などのコンピュータを通して、実世界に詳しい情報を付加する技術です。もっと簡単に言えば電腦コイル。

*5 OCR (Optical Character Recognition) スキャナなどで読み取った画像からテキストを抽出する技術。

*6 Basic Linear Algebra Subprograms の略。ベクトル・行列の計算を行うライブラリ。

*7 General-Purpose computing on Graphics Processing Units の略。

*8 GNU Multi-Precision Library の略。GNU プロジェクトで開発されている多倍長演算ライブラリ。

検索キーワード非依存型生成手法によるスニペットのパーソナライズとその表示機構

ユーザひとりひとりに合ったスニペット^{*9}を生成するための研究です。ユーザのブックマークや検索履歴、さらにはソーシャルブックマークサービス^{*10}の情報からユーザの趣向を分析し、そのユーザにとって重要な文を優先的に引用してスニペットを作ります。

Google 検索^{*11}が生成するスニペットは検索ワードのみに依存しますが、この方法を使えばユーザひとりひとりに最適化されたスニペットを出力することができるので、より効率の良い検索が期待できます。

現在の実装としては Firefox のプラグインと、スニペットを生成するためのサーバを用いて実現されています。Google による検索結果から、スニペットの部分を前のものに置換しているようでした。

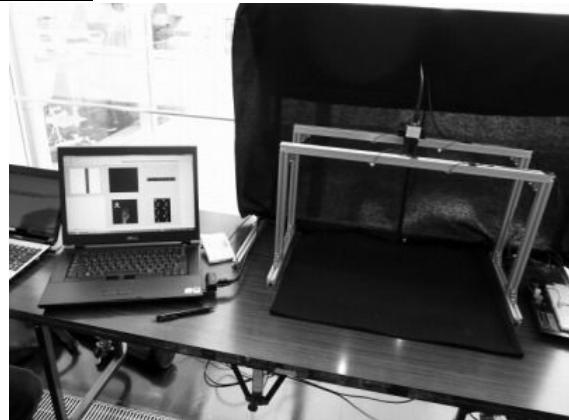
照明条件の能動的制御に基づく指文字認識の高精度化

指文字^{*12}をカメラで読み取る研究です。

1枚の画像から手の形状を識別するは難しいので、様々な角度(照明条件)から照明を当てることによって手の三次元形状を識別します。識別に必要な照明条件は省くようになっています。

用途としては手話を使う人たちが入力に使うというよりも、指文字の練習などに応用できるそうです。

右の写真の装置に手をかざすと認識してくれます。わりと早い周期で白色 LED がチカチカしていました。



ペット^{*13}検出のための WEB 画像を用いた学習サンプルの効率的な収集

人間を対象とした顔認識などの研究は活発に行われていてデジカメや顔認証などで実用化されています。しかし、ぬこを対象としたものというのはそう多くはありません。実際にはぬこ検出を行うデジカメも商用化されていますが、ぬこ画像の需要に対して、技術が追いついていないのが現状です。

この研究はぬこの検出・状態認識を目的としているそうで、現時点ではぬこを検出するところまでできているようでした。最初は Flickr^{*14}からぬこ画像を人手で集めてきて機械に学習させらしいです。

*9 ググったときに出てくる、ページごとのダイジェストです。

*10 要するに「はてなブックマーク」です。

*11 有名な検索サイトです。詳細は g g r k s。

*12 手話の五十音です。人名や外来語など、手話には無い単語を表現するときに使われます。

*13 ペットと言うよりもヌコです。プロジェクト名にはペットと書いてありますが、ヌコです。発表を見る限り犬の話は一切出てきませんでした。聞いてみたところ、ここのプロジェクトの方はヌコがお好きなようです。筆者もヌコは好きです。学業もないし、納税義務もないし……。

*14 画像共有サイト <http://www.flickr.com/>

ICT 中間報告会 2010

検出の手法としては人間の顔検出に利用されているものに似た手法を開発したそうですが、ぬこの顔だけではなく、ネコミミも検出のための手がかりに利用することによって、誤検出をほぼ半数に減らすことが出来たそうです^{*15}。

資料によると、「今後は、学習した画像分類機や検出器をダウンロードした様々な Web 画像へ適用し、様々なネコの画像データを収集することを行う」らしいです^{*16}。

終わりに

ここでは誌面の都合上、すべてのプロジェクトをご紹介することができませんでしたが、他にも多くのプロジェクトがありました。大学院ではどのような研究を行われているのか、直接お話を聞くことができる良い機会です。興味を持たれた方は、ぜひ最終報告会^{*17}に行ってみてはいかがでしょうか。

報告会の開催予定などのお知らせについては、ICT ソリューション・アーキテクト育成プログラムのページをご覧下さい。今までの報告会で展示したポスターや報告書も閲覧できます。

ICT ソリューション・アーキテクト育成プログラム

<http://abelia.cs.tsukuba.ac.jp/ICT>

ここまで読んでくださった読者の皆様、そして我々のしつこい質問に対応してくださった発表者の皆様、ありがとうございました。

*15 ということはネコミミがぬこの本体なんじゃないか？ネコミミが取れたぬこ型ロボットなんて、もはやタヌキですらない気がする。今後の研究に期待せざるを得ない。

*16 筆者としてはもっと違う画像を収集するべきだと思う。訓練データにする画像ならいっぱい提供させていただきますのでぜひ……。

*17 昨年は 2 月 20 日に行われ、WORD ではこの報告会の取材も行いました。詳しくは WORD14 号「ICT ソリューション最終報告会」をご覧下さい。

雛見沢村に行ってきた

文 編集部 ミレトス

お久しぶりです。

最近いきなり寒くなってしまったので、ユニクロに暖かい服を買いに行ったら、自分に似合う服が全く解らない事に気づいて嫌な汗を流しながら店内をうろうろしていたミレトスです。みんな服どうやって選ぶんだろうね。教えてください（必死）。

いきさつ

今回は大人気「ひぐらしのなく頃に」の舞台、雛見沢村に行ってきました。「雛見沢ってマジであるの！？」と思ったあなた、残念！

雛見沢村は実際には存在しません。実際は岐阜県にある白川郷がモデルとなっています。白川郷は合掌造りが有名で、1995年にユネスコの世界遺産に登録されています。

なぜ白川郷に行くことになったのかというと、母親が「白川郷行きたい！うーうー！」というので、祖父母宅へ帰省するついでに行くことになったのです。

以下は帰省する前日の友人との会話

ミレ「明日帰省するんだ」

友人「お土産は神戸チョコレートな」

ミレ「その前にしらなんとかって所に寄るらしいんだけど、興味ないんだよねー」

友人「白川郷？」

ミレ「そうそれ」

友人「ちょ、そこひぐらしの聖地だよ！」

ミレ「なん…だと…やる気出てきた！！！」

とまあこういう経緯で、偶然ながらも雛見沢見物へと行くことになったのでした。

白川郷へ

出発は深夜12時頃でした。いつも帰省する時は東海道を使って行くのですが、今回は白川郷に寄るということで、関越道、上信越道、北陸道、東海北陸道を使っていきました。普段帰省する時とは違う風景や涼しい気温、日本海の近くを通るなどとても新鮮で楽しかったです。

そして白川郷へ到着。この間には「ガソリンスタンド消失事件」「消えた靴下」「踏みつけられた妹」という様々な事件が起こっていたのですが、それはまた別の話。

到着したのは朝の7時頃でした。白川郷の案内所みたいなところでとりあえず休憩をするために外に出たのですが、涼しかったです。山に囲まれているだけあって空気が美味しい！そして、まだそんなに人もいなかったのでここで少しゆっくりして家族が起きるのを待ってから白川郷の観光に行きました。

雛見沢探検



少し見づらいですが白川郷の文字が写っています。

まず高台へ行って全体の風景を撮影しました。携帯で撮ったので画像がかなり粗いですが……。
夜通し起きてたのでちょっと眠かったですが、風が気持ちよかったです。



全体図。それっぽいです

そこから少し坂を下ると、小さい祠みたいなものがありました。ここからも全体が見渡せます。
こここの近くから坂が下まで続いており、そこから村へ行けるようになっていました。



ここで合掌造りについて少し説明します。合掌造りは日本の建築様式の一つで、屋根の形が合掌した時の手の形に似ていることから名付けられました。この地方では雪がよく降るので、雪が屋根に積もらないようにする為に急な傾斜になっているのです。昔は日本の民家で広く見られた様式でしたが、今では少なくなりました。屋根裏では蚕を飼っていたらしく、それに使っていた道具なども展示されていました。



合掌造りの家

古手神社らしきものも見つけました。ここは道路の近くにあったので、見つけやすかったです。



古手神社（推定）

そうこうして一通りぐるぐる見た後で、ひぐらしの園崎家のモデルとなっている和田家の家屋に入りました。合掌造りの家は見学料が 300 円かかりますが、入って中を歩きまわることが出来ます。天井裏などにも行くことができ、割と自由に歩き回ることができます。この日は結構暑かったのですが、縁側は日が当たらなくて、冷房がなくても涼しかったです。残念ながら中は見てまわるのに夢中で撮影するのを忘れていました。ミレトス、不覚…！

雛見沢探検

そろそろいい時間になったので、帰ることに。駐車場から白川郷の見物場所までは少し距離があって、その間に橋を通ります。この橋が中々いい感じでした。



橋 자체はしっかりとしていました。



橋から見下ろした図。ドラマの撮影に使えそう。

最初はそんなに興味がなかったのですが、ひぐらしと関係があるということから始まって色々なところを回っているうちに結構楽しんでいることに気づきました。

きっかけはアニメや漫画でも、こういう所を楽しく見て回れるというのはとても良い事ですね。

この後昼食を食べ、祖父母宅へと向かいました。とにかく山に囲まれているので、トンネルがめちゃくちゃ多かったです。

帰ってきた後でググって画像を見ていたら、雪が積もってる写真を見つけてしいました。凄い趣があるって、冬も行ってみたいなと感じました。これから冬の季節、興味がある方は是非行ってみてはいかがでしょうか。冬に行く場合は**チェーン必須**です。また、雪が降ると交通規制がかかって道路が封鎖されてしまう場合もあるので、行く時は交通情報もしっかりとチェックしてくださいね。行く場合は筆者も連れていってください！

情報科学類誌 WORD 読者アンケート

本文・題字 編集部 ふあい

■プロローグ(WORD サイド)

時は 2010 年。筑波大学某所にて、ふあいは悩んでいた。WORD の最新号を配ったはいいが、数十部ほど余ってしまった。そもそも今までには余っていなかつたのかというと、発行した後は計算機室の前に積んでそれっきりなので、その後の行方やどれくらい読まれているかはあまり把握していない。把握してゐる事と言つたら、計算機室に転がっているのを良く見かける事くらいだ。あと、この間トイレに駆け込んだら読みかけの WORD が置いてあつた。

ともかく、どれくらい読まれているかを把握するには、どうすればいいだろうか？

そうだ、アンケートを採ろう！そのアンケートの結果を記事にすれば記事のネタにも困らないぞ！がつはつはつはワシって頭いいー！

■プロローグ(カオス研サイド)

時は 2010 年。筑波大学某所にて、21 世紀の太陽^{*1}・我が惑星の守護神^{*2}・人類が生んだ傑出した英雄^{*3}・全世界が崇め奉る英明な指導者^{*4}・百戦百勝の鋼鉄の靈将^{*5}・百勝の作戦家^{*6}・人徳で天下を動かす絶世の偉人^{*7}・無敵必勝の象徴^{*8}であらせられる徳永隆治先生は悩んでいた。Twitter 上で行った COJT^{*} ルームのデザインアンケート用の景品である COJT グッズが余ってしまった。この余ったグッズを体良く処理する方法はないだろうか？

そうだ！元はアンケート用のグッズなのだから、どこかのアンケート用の景品として渡してしまうのが良いのではないか！そうと決まれば WORD と MAST^{*10} に渡してしまおう！がつはつはつはワシって頭いいー！

上の 2 つのプロローグは、ほぼ同時期に平行して発生した出来事です。これを分かりやすく図にまとめると、次ページのようになります。

*1 将軍様の呼称

*2 将軍様の呼称

*3 将軍様の呼称

*4 将軍様の呼称

*5 将軍様の呼称

*6 将軍様の呼称

*7 将軍様の呼称

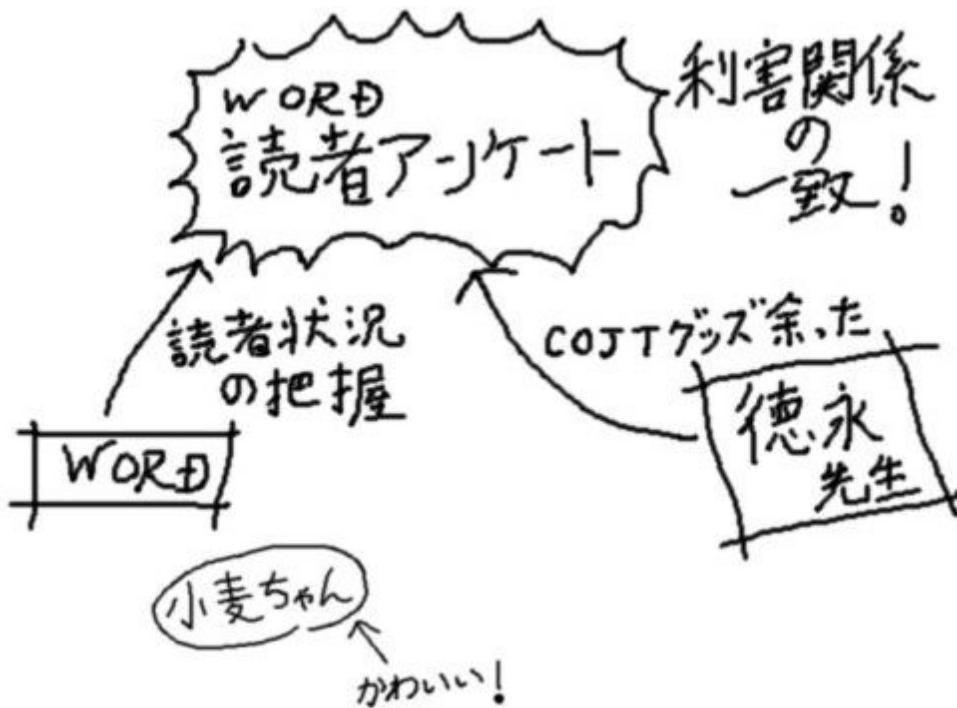
*8 将軍様の呼称

*9 情報科学類・情報メディア創成学類 3 年生向けの授業「組み込み技術キャンパス OJT」の事。

興味のある方は徳永先生まで！Twitter 軟式アカウントは@cojt_tsukuba です。

*10 情報メディア創成学類の学類誌

WORD 読者アンケート



■アンケートの概要

- ・次のページからのアンケートにご協力下さい

アンケートの結果は今後の誌面作りの参考にさせていただきます。また、**集計結果をまとめた記事を次号に掲載する予定です。**意見・感想などの自由記述欄を晒されたくない方は、アンケート最後の欄にチェックをお願いします。

- ・景品があります

アンケートの景品として、21世紀の太陽・我が惑星の中略徳永先生より、COJTオリジナルゴ入りのクリアファイル5枚&テープのりセットを5名分いただいております。

誠に勝手ながら、先着順とさせていただきます。

- #### ・アンケートの提出について

WORD 編集部(3C212)にて直接手渡しか、同室前に設置してあるアンケート回収 BOX にて回収しております。アンケートの景品をご要望の方は手渡しでお願いします。(景品の在庫管理のため。)

- ・小麦ちゃんはかわいい

あああああああ小麦ちゃんかわいいよおおおおおおおおお！！！天使たちが聖なる泉に降り立つが如く眩いばかりの輝きを後光としパラソルピカルクロホルム！！＄%！！！小生は感激のあまり涙がちょちぎれてきました！ 1 1 1 ！！！ オホウ！！！ イエス！！！

小麦ちゃんのナース服で瀧過した空気しか吸いたくない！！！！！

WORD読者アンケート

情報科学類誌 WORD をご愛読いただき、ありがとうございます。以下のアンケートにご協力を
お願ひいたします。提出の際は、お手数ですがこのページを切り離してご提出下さい。

Q1:所属を教えて下さい。

1:学生(_____学類) 2:院生(_____専攻) 3:筑波大学職員・教員 4:その他

Q2:性別を教えて下さい。

1:男 2:女 3:その他(_____)

Q3:WORD の公式 Web サイト「WORD Press」(<http://www.word-ac.net/>) はご存じでしたか？

- 1:ぼく知ってるよ。日本で 30 番目に IPv6 に対応したサイトだよね。頻繁にチェックしてるよ。
- 2:数回見たことはある。
- 3:知ってるけど見た事はない。
- 4:ああ、ブログ/CMS プラットフォームの……
- 5:知らんな。

◆今号のタイトル一覧

- | | | | |
|---------------------|------------------|--------------------------------|--------------------|
| 1:表紙 | 2:目次 | 3:新型 MacBook Air 購入レポート | 4:電子の歌姫はアイドルの夢を見るか |
| -APPEND TRACK- | 5:GR な日々。V | 6:RVM を使っていくつかの Ruby 実装を使ってみよう | |
| 7:use Perl::Object; | 8:革命的で魔法のようなラクダ。 | 9:最近のポケモン事情～育成編～ | |
| 10:ICT 中間報告会 2010 | 11:雛見沢村に行ってきた | 12:情報科学類誌 WORD 読者アンケート | |
| 13:編集後記 | 14:裏表紙 | | |

Q4:今号の記事で、良かったと思う記事があれば、上の表から番号またはタイトルをご記入下さい。
また、理由・感想なども教えて下さい。

Q5:今号の記事で、良くなかったと思う記事があれば、上の表から番号またはタイトルをご記入下さい。
また、理由・感想なども教えて下さい。

◆次ページに続きます。

WORD 読者アンケート

Q6:過去の記事に関して感想等がありましたら、以下の欄にご記入下さい。

Q7:以下は自由記述欄です。WORD に関する要望や意見、おすすめのアニメ、嫁に対する熱い想い等、何でもご記入下さい。(日本語でなくても構いません。むしろ文字でなくても構いません。)

氏名(PN 可) : _____

- ◆意見・感想等の自由記述欄を載せられたくない方は、右の欄にチェックをお願いします。
- ◆ご協力ありがとうございました。

情報科学類誌

WORD

From College of Information Science

WORD 侵略されたいでゲソ号

発行者

情報科学類長

編集長

石川 陽一

制作・編集

筑波大学情報学群

情報科学類WORD編集部

(第三エリアC棟212号室)

2010年11月11日 初版第一刷発行
(5 1 2部)