

# WORD

2011.2

From College of Information Science

型制約の謎に迫る！  
in Perl6  
WiFi 暗号がやばい件について

Mozc カレー

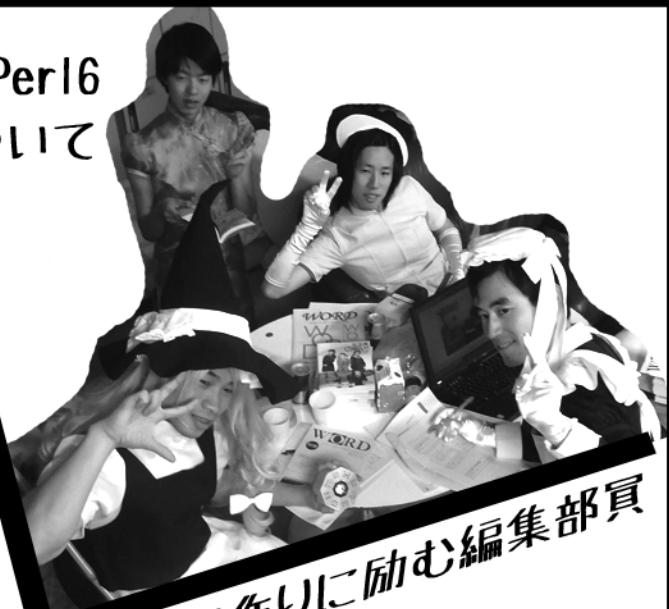
use Perl::Object 0;

降臨！偶像大師

CX & GR な日々。

PCをゲームコントローラーで操作しよう！  
この一歩から、未来が始まる。 -LTE レビュー-

☆プレゼントが貰える  
アンケートもあるよ☆



記事作りに励む編集部員

vol. 17

# おせち特盛号

## 新約 禁書目録

p.03 CX & GR な日々。～サンライズウサギ島～

p.20 Mozc カレー

p.24 型制約の謎に迫る！ in Perl6

p.27 PC をゲームコントローラーで操作しよう！

p.35 WiFi 暗号がやばい件について

p.39 降臨！偶像大師

p.43 お詫びと訂正

p.48 この一歩から、未来が始まる。～LTE レビュー～

p.56 use Perl::Object 0;

p.67 情報科学類誌 WORD 読者アンケート

とじ込み付録 アンケート用紙

# CX & GRな日々。

～サンライズウサギ島～

文 編集部 ふあい 葡萄酒 mitty

## ■あいさつ

明けましておめでとうございます。2011年、卯年が始まりました。今回の「GRな日々。」は、ウサギ島という愛称で知られる大久野島への旅行記(のようなもの)をお届けします。複数人で書いているため、途中で文体がガラリと変わりますが、ご容赦ください。

ちなみにタイトルのCXとは、ふあいが使っているデジタルカメラ「CX1」(RICOH製)よりとっています。秋葉原のヨドバシカメラで「バカでも操作が覚えられるデジカメ下さい！！」と言ったらこれがでてきました。今では満足しています。

あと、記事の内容がカメラと何の関係も無くなっているのは公然の秘密ということでお願いします。

## ■サンライズ出雲号に乗ろう

ここでは筑波大学を出発して、ウサギ島へのフェリー乗り場最寄り駅である忠海駅<sup>\*</sup>までの経路について、淡々と鉄オタ視点で語ります。

### ◆経路

(12月26日) 第三エリア前 20:21 発

↓ 関東鉄道バス 土浦駅行き

つくばセンター 20:32 着

↓ 徒歩

つくば 20:37 発

↓ つくばエクスプレス 区間快速 | 秋葉原行き

秋葉原 21:30 着 21:35 発

↓ 山手線 外回り

東京 21:40 着 22:00 発

↓ 東海道本線・山陽本線 寝台特急サンライズ出雲 | 出雲市行き

(12月27日) 倉敷 6:56 着 7:08 発

↓ 山陽本線 普通 | 三原行き

三原 8:18 着 8:29 発

↓ 呉線 普通 | 広行き

忠海 8:50 着

この行程の目玉はなんといっても関鉄バスが定刻通り運転したこと……ではなく、寝台特急サンライズ出雲号です。サンライズ出雲を使った理由は2つあります。まず、朝早くから行動できる事です。仮にTXの始発に乗って新幹線を使った場合、忠海駅到着は11:40になります。それにTXの始発に乗るために、バスが動いていない時間に家を出て、バス以外の手段でつくば

\*1 忠海駅：広島県竹原市にある、JR呉線の駅。「ただのうみ」と読む。

## CX & GR Days

駅まで行く必要があり、少々面倒になります。そしてもう一つの理由は、**ただ単に乗りたかっただけ**です。だって乗りたいじゃん。

個人的には貧乏学生の長旅の友と言えば青春 18 きっぷなのですが、今回はウサギ島でウサギや廃墟と戯れる事が目的なので、体力温存のために鈍行での移動はやめました。

### ◆関鉄バスがいい仕事をしてくれたハナシ

上の行程を見ると、つくば→東京の乗換え時間が軒並み 10 分以下になっているのがお分かりいただけます。最初はもう 1 本早い TX の快速に乗る予定でしたが、出発直前になって葡萄酒君が「サンライズ出雲の乗車券を家に忘れた」と言って、家に帰ったため、このようなギリギリな移動になってしまいました。あらかじめ購入した乗車券は忘れないようにしましょう。

20 時台ともなると学内を走るバスの本数や TX の本数が少なくなるので、少しの遅れが大きな遅れになります。事実、この時も第三エリア前 20:21 発のバスが 5 分以上遅れると、東京駅での乗換えがかなりシビアになるという、最初からクライマックス状態。皆さんご存じの通り、学内を走るバスは 5 分程度の遅延は当たり前なので、もはや絶望的。しかし関鉄バスはやってくれました。まさかの定刻通りの運行です。ありがとう関鉄バス。愛してる。

この後、秋葉原駅で帰りに使う青春 18 きっぷを購入したのですが、もし関鉄バスが 5 分以上遅延したら青春 18 きっぷを購入する時間が無くなるのでどうしようとか騒いでいたのですが、杞憂に終わりました。というか青春 18 きっぷくらいあらかじめ購入しておきましょう。ギリギリの行動はダメ、ゼッタイ。

### ◆サンライズ出雲号・瀬戸号の紹介



サンライズ出雲号・瀬戸号に使われる 285 系電車

サンライズ出雲号・瀬戸号は、東海道線を走る唯一の寝台特急です。サンライズ出雲号とサンライズ瀬戸号は途中の岡山まで連結して運転されています。東京寄り 7 両がサンライズ出雲号で、高松寄り 7 両がサンライズ瀬戸号です。

下りの電車は東京駅を 22:00 に出発し、岡山に 6:30 頃に到着して、サンライズ出雲号とサンライズ瀬戸号に切り離します。サンライズ瀬戸号は宇野線・本四備讃線・予讃線を通って高松に 7:30 頃、サンライズ出雲号は倉敷から伯備線・山陰本線を通って出雲市駅に 10:00 頃に到着します。

東京から倉敷までの料金は学割を使って 18650 円(B 寝台シングル)。寝台料金がかかる分、新幹線利用時より 5000 円ほど高くなりますが、朝から行動できる点や、和気あいあいとした雰囲気など、値段に見合うだけの魅力はあると思います。

車内は木目調を基調とした高級感あふれるデザインでほとんどの部分が 2 階建てです。展望スペースやシャワールームがついており、まさに走るホテルと言えます。**そこらへんの平砂宿舎よりも住みやすいです。**

それでは簡単に、魅力ある車内設備の説明をしましょう。

### ◆客室

客室は、A 寝台 1 人用個室「シングルデラックス」、B 寝台 1 人用個室「シングル」、B 寝台 1 人～2 人用個室「シングルツイン」、B 寝台 2 人用個室「サンライズツイン」、B 寝台 1 人用個室「ソロ」、~~普通車~~ 普通車指定席「ノビノビ座席」の 5 種類があります。客室などの設備はサンライズ出雲号とサンライズ瀬戸号で、それぞれ同じ構造・配置になっています。

#### ・A 寝台 1 人用個室「シングルデラックス」

4 号車および 11 号車の 2 階にある、ブルジョワジー用の個室です。各部屋のサイズは B 寝台の個室と比べて倍くらいの広さがあり、B 寝台の個室にはない洗面台が設置されています。車端部にはシングルデラックス利用者専用のシャワールームもあり、B 寝台との格差を見せつけられます。くやしいのう。

#### ・B 寝台 1 人用個室「シングル」

今回我々が泊まった個室です。2 階席と 1 階席と平屋席(車端部)の 3 種類があります。写真に写っている部屋は大きな窓が特徴の 2 階の個室です。

室内のほとんどのスペースは、よくあるシングルサイズのベッドで埋まってしまいます。小さなデスクや若干の荷物置きスペースもあり、大荷物でなければ不自由しないと思います。(大荷物でも、靴を脱ぐスペースに置けば問題ないとおもいます。)

情報科学類生には嬉しい電源も 1 口用意されています。

枕元にはラジオ、アラーム付きの時計、緊急用の SOS ボタンがあります。ラジオは NHK-FM ラジオのみ視聴可能です。



B 寝台シングル。私の部屋。平砂宿舎より快適でした o(^-^)o

## CX & GR Days

SOS ボタンには、間違って押さないようにカバーがされています。押すと電車が急停車する<sup>\*</sup>ので、間違って押さないようにしましょう。

各部屋の扉には電子ロックがついているため、荷物を置いて展望室に行ったり、シャワーを浴びたりできます。電子ロックの設定時には 4 桁の暗証番号を入力するのですが、恐ろしいことに入力した暗証番号はロックボタンを押すまで平文で表示されます。見られないように十分気をつけましょう。

### ・B 寝台 1 人～2 人用個室「シングルツイン」

シングルツインという不思議な名前がついていますが、補助ベッド付きの 1 人用個室と言った感じです。平屋部分の中途半端にスペースが余った部分に存在し、広さはシングルと変わりませんが、かなり天井が高めの部屋になっています。ベッドの上に補助ベッドが設置されており、2 人で使う場合は補助ベッドを使用します。1 人で使っても 2 人で使っても、1 人あたりの値段は B 寝台シングルと同じです。

### ・B 寝台 2 人用個室「サンライズツイン」

4 号車および 11 号車の 1 階にある、シングルサイズのベッドが横に 2 つ並んでいる部屋です。部屋数はかなり少ないです。一人あたりの面積はシングルとほぼ変わりませんが、部屋が広い分、窮屈さを感じさせません。一人あたりの値段はシングルと同じです。我々はこの部屋を狙って乗車日の 1 ヶ月前(乗車券の販売開始日)に乗車券を購入しようとしたのですが、その時点で売り切れになっていました。シングル以外はかなり競争率が高いようです。

### ・B 寝台 1 人用個室「ソロ」

床下にモーターがついているため、2 階建て構造に出来ない 3 号車・10 号車にある部屋です。広さは B 寝台シングルとほぼ同じですが、天井が低く、やや窮屈な部屋です。また、モーターの上にあるため、走行音が気になるかもしれません。

しかし、値段が B 寝台シングルよりも 1000 円安いため、人気のある部屋です。

### ・普通車指定席「ノビノビ座席」

5 号車・12 号車にある空間です。部屋ではなく空間です。ノビノビ座席といいますが、座席はありません。簡単な仕切りで仕切られただけの空間が 14 箇所×上下 2 段の 28 箇所あり、絨毯がひいてあるだけの修行者向けスペースです。

とはいって、空調はしっかりと効いており、肌かけ毛布が 1 枚提供されるので風邪をひくことは無いでしょう。ただし枕は無いので、乗車する際はあらかじめ空気マクラを持ってきた方が良さそうです。

また、この席は**寝台料金が不要**で、指定席特急券と乗車券だけで利用できるため、個室とくらべてかなり安い値段で利用できます。そのため、かなり人気がある席になっています。

---

\*2 間違って SOS ボタンを押した人が居て、本当に電車が止まりました。

### ◆展望スペース

3号車と10号車にある4人掛け×左右2箇所のスペースです。大きな窓から外をながめつつ、友達と会話できるスペースです。すぐ隣にB寝台ソロ個室があるので、騒ぎすぎには注意しましょう。

電源はありませんが、ノートPCをバッテリ稼働させれば夢のサンライズプログラミングができます。

MAXコーヒー(持参)を片手に、夜の車窓をながめながらプログラミングが出来るのもサンライズ出雲号・瀬戸号の魅力ですね。



サンライズ出雲号でプログラミング@横浜駅

### ◆シャワールーム

合法的に電車内で全裸になれる貴重なスペースです。山形行きの新幹線と間違えて乗車しても、ここで全裸になれば罪には問われません。

シャワー利用の際には、車掌さんからシャワーカード(1枚300円也)を購入する必要があります。1枚のシャワーカードで6分間(実稼働時間)お湯が出ます。シャンプー・ボディーソープは備え付けてありますが、タオルはありません。あらかじめ持参するか、車掌さんから「サンライズ出雲号・瀬戸号オリジナルタオルセット」(200円也)を購入しましょう。シャワールーム内には残り時間が表示されるデジタル式のタイマーが設置されており、スリリングな「制限時間内に身体と頭を洗わないことになるとぞ」感を味わえます。また、揺れる車内でシャワーを浴びて転びそうになるのも寝台特急でしか味わえない体験だと思います。

私が利用した時は、最初の8秒は水がでて、残り時間1秒でお湯がストップしたので、5分51秒しか温かいシャワーを浴びる事が出来ませんでした。**差額の7.5円返せ!**

### ◆広島県周辺の鉄道事情

JR西日本は京阪神の大都市を結ぶ黒字路線を持っている一方で、山陽～山陰を結ぶ路線など、たくさんの赤字路線も持っています。そのため、一部の路線を除いて国鉄時代の車両がたくさん残っています。とりわけ広島周辺は京阪神から左遷してきた国鉄時代の車両が元気に走っており、ネット上の一部の人からは「國鐵廣島支社」などと言われています。

最近は塗装のコストを削減するために、1色のみの塗装が行われているようです。帯はもちろん、JRのロゴすら入れないという気合いの入れようです。広島周辺はあざやかな~~朱~~<sup>朱</sup>真っ黄色の塗装に統一する事が決まっています。

また、利用客の少ない路線では、線路をいたわって保守作業のコストを削減するためにわざと徐行する区間があります。ネット上では「必殺15km/h制限」と呼ばれています。今回の旅行では、呉線の一部区間で必殺15km/h制限が見られました。

いずれにせよ、JR西日本の苦しい財政状況が垣間見えます。みなさん、もっと電車に乗りましょう。これから毎日電車に乗ろうぜ?

## CX & GR Days

今回の旅行では、倉敷→三原ではサンライナー<sup>\*3</sup> 塗装の 117 系、三原→忠海では真っ黄色の 105 系に乗りました。両方とも国鉄時代の車輌ですが今日も元気に走っています。



サンライナー塗装の 117 系



真っ黄色の 105 系

### ◆忠海港のフェリーと呉線の関係

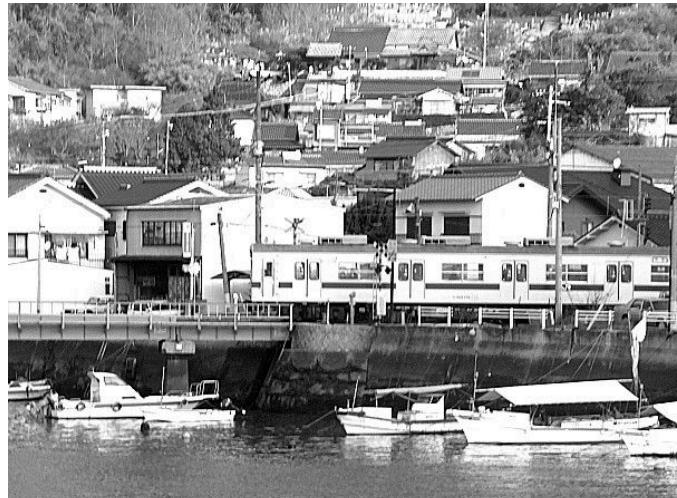
右の写真を見てください。これは、大久野島行きのフェリーから撮った呉線の写真です。

この写真から分かることは主に 2 つ。まず、呉線が非常に海に近い所を走っていること。そして、フェリーから電車が撮れるということは、この電車はギリギリでフェリーに接続しないという事です。

我々が乗ってきた電車(このページの右上の電車)が、忠海駅到着する 5 分前にも、フェリーが出ています。

どうやら、ここでのフェリーはギリギリで電車に接続しないという謎のダイヤを組んでいるようです。

待ち時間が 30 分以上あったので、フェリー乗り場近くに小さな売店にて、軽めの朝食をとりました。



フェリーから撮った呉線の 103 系電車

\*3 サンライナー：山陽本線の岡山～福山を走る快速「サンライナー」のこと。

## ウサギと廃墟の島

広島県に所属するウサギ島は正式名称を大久野島といい、周りを瀬戸内海に囲まれた周囲 4km 弱の小さな島である。戦前はこの島に数人の住民が暮らしていたのだが、戦争に際して政府に接収され要塞が建造される運びとなった。それ以降、この島は様々な形で戦争と関わっていく事となる。実際に島へと向かう前に、その痛ましい軌跡を辿ってみることにしよう。

事の始まりは明治 23 年 ————— つまり日清戦争が始まる 4 年前、広島県呉市に呉鎮守府海兵团が設置されたところにある。これに伴い周辺の都市は戦争とともに大きく発展していく事になった。そのうちの一つが大久野島と密接に関わっている竹原市忠海町である。

山と海に挟まれるようにして広がる忠海の町は、かつては商船の停泊所として発展した平和な港町であった。明治 33 年、日清戦争に勝利した政府は、軍港として大きな役割を果たしていた呉港をはじめ軍事上重要な拠点となった広島を防衛するため、この忠海に芸予要塞の建設を始めた。その一環として作られたのが大久野島の 12 門の砲台である。この砲台が実戦で使用される機会は無かったが、大正 13 年に同要塞の廃止が決定した後も大久野島は軍用地として戦力を蓄えていく。

昭和 4 年、火工廠忠海兵器製造所、いわゆる毒ガス工場が大久野島に設置される。毒ガス兵器は第一次世界大戦でドイツがその効力を実証して以来、各国が精力的に開発を続けており、日本もまた例外ではなかった。毒ガス兵器の使用はハーグ条約やジュネーブ条約などで禁止されているため、その開発や製造には機密性が求められる。また、事故や災害に際して付近の住民への影響を最小限に抑える必要がある。この二つの要件を満たし、また輸送の容易さを勘案した結果、軍部が工場の建設地として選択したのは、呉に程よく近い瀬戸内に浮かぶ大久野島であった。以後終戦に至るまで、この島は公式の地図から姿を消す。

竣工後、本土から大量の人員を招聘して毒ガスの製造が始まったが、実際に製造に携わる作業員達はある程度教育を受けていたものの、実際に自分たちが何を作っているのか、どの程度の危険性を持った作業であるのかは知らされていなかった。そのため作業中に数多くの作業員が事故で被曝し、負傷した。中でもイペリットの生産工程は特に危険が大きく、幾人もの作業員が皮膚や気管支を侵され倒れたという。イペリットは吸い込むだけでなく、ガスが肌に触れただけでも激しい炎症を起こす。まだ防毒技術が発達していなかった当時の防護服では全てを防ぎきる事は出来ず、マスクや靴の隙間から入り込んだガスによって全身が爛れる者が後を絶たなかった。それでも軍部の意向により製造は続けられ、ガスが充満して噎せ返る作業場の中、作業員は涙とくしゃみに耐えながらの作業を強いられたのである。この工場での正確な死傷者数は今でも分かっていない。この凄惨極まる環境については「地図から消された島 - 大久野島 毒ガス工場」(ドメス出版/武田英子著)<sup>\*4</sup> が詳しいので、興味のある読者は手に取ってみると良いだろう。実際に作業に携わった方の体験談が数多く収録されている貴重な資料である。

戦後、軍部の解散に伴い GHQ の監視下で海洋投棄や火炎放射器による焼却処分が行われ、この島に貯蔵されていた 7000 トン近い毒ガスは全て廃棄された。この処分にも多くの危険が伴い、負傷者が多発した。実際に製造に携わった者は毒ガスの恐ろしさを知っているため作業に加わろうとはしなかったので、経験の無い処分担当者は苦戦を強いられる事となり、日米問わず多くの被曝者を出す事となってしまう。島は消毒のために大量のサラシ粉<sup>\*5</sup> が撒かれ、昭和 22 年に日本政

\*4 一般的の書店には流通していないようだが、Amazon で購入できる。

\*5 散布されたサラシ粉は厚さ 3cm の層ができるほどの量であったという。

## CX & GR Days

府に返還された後も一般人の立ち入りは禁止されていた。

そして迎えた昭和 31 年。朝鮮戦争のために一時的にアメリカに弾薬庫として接収されていた大久野島は再び日本政府へと返還され、ついに一般人の立ち入りが可能になった。竹原市の要望により国民休暇村に指定された同島は、長く悲しい戦争の歴史に幕を閉じ、ようやく現在のウサギ島へと至るのである。

### 大久野島へ上陸

さて、堅い話はこの辺にして島へと向かう事にしよう。JR 忠海駅から徒歩数分の忠海港から直通のフェリーに乗って、まずは 10 分程度の船旅である。力強いスクリューの唸りを挙げて疾走するフェリーの窓から、ぐんぐんと近づいてくる大久野島。海上を吹き抜ける風は強く肌寒いものの、透き通るような青空が気持ちのよい快晴であった。船から降りると目に入るのは青い空にそびえ立つ赤い門、朝日を受けて鮮やかに映える姿が印象的である。



桟橋に立つ門

この桟橋からホテルまでは数キロということで、折角なので朝の空気を満喫しながら歩く事にした。まず出迎えてくれるのは、ウサギ島という愛称の由来であるウサギ達である。このウサギ達は戦後に放たれたものが野生化し、天敵も少ないのでこの島で爆発的に増えたものと聞く。しかし、



第一村人（？）

その数はせいぜい動物園程度だろうと高を括っていた私の予想を遥かに超えていた。桟橋から遊歩道へ出てすぐの広場に、ぱっと見るだけで 10 羽以上。こちらを見るや否や一斉に集まってくるウサギ達、しかも遠くからも続々と集まっているのである。観光客がエサをくれる事を憶えているのだろう、全く人間を怖がる気配はない。しかし島に降り立ったばかりの我々がエサを持っているはずもなく、群れる彼らを宥めずかしつつ宿へと向かうことにした。



子供に群がるウサギの大群



毒ガス貯蔵庫跡

道中多くのウサギと戯れながら歩く事、20分。ヤシのような南国風の木に囲まれたホテルに辿り着いた。やはり宿の近くは人が集まる所為だろうか、エサを求めて群がるウサギの数は道中とは比較にならない。

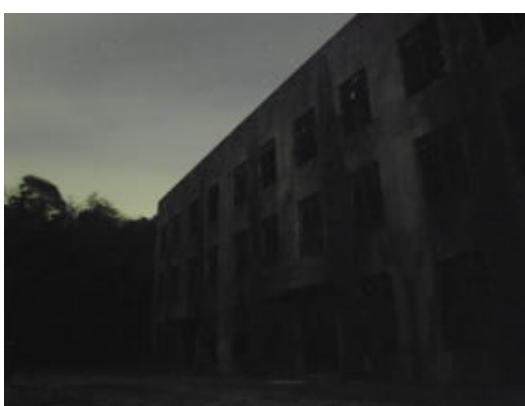
### 廃墟散策

フロントに荷物を預けた後はエサを貰い、餌付けをしつつ島を一周する事にした。目指すは島中に点在する旧陸軍の施設跡である。

宿から出発して右回りにぐるりと歩を進めると、最初に姿を現すのは毒ガスの貯蔵庫跡。半壊したコンクリート造りの無骨な建物に、焼却処分された際の黒い焦げ跡が痛々しく残っている。戦時中にはここに 600 トンもの毒ガスが貯蔵されていたという。



レンガ造りの原料貯蔵庫



真夜中の発電場跡

少し山道を上り、次に見えてくるのは中部砲台跡である。この砲台は日露戦争の前に撤去されたのだが、土台はそのまま残っており陣地の上に立つ事が出来る。また、砲台が撤去された跡は毒ガスの原料保管庫としても使用されていたようで、斜面を削るように建造されたレンガ造りの建物が各所に見受けられる。

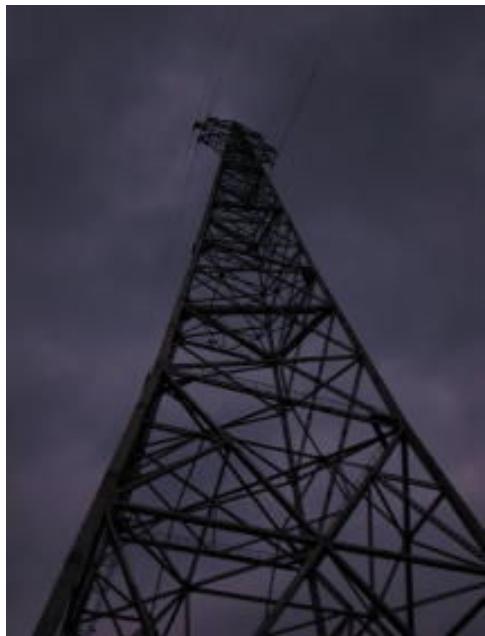
更に進む事数分、ついに辿り着いたのが本日のメインディッシュ、発電場跡である。ツタに覆われ、鬱蒼と茂る木々の中にひっそりと溶け込むその姿は何とも表現し難い不気味さを放っている。この「廃墟とウサギ」という何とも奇

妙な取り合わせは、世界広しといえどもここでしかお目にかかるないだろう。ゆったりとウサギ達が暮らす長閑な風景の中で痛々しく戦渦の爪痕を残すこの廃墟——その寂しげな佇まいの前に立つ私は、何處か遠くの世界へと誘われるような錯覚をおぼえたのだった。

## CX & GR Days

### そびえ立つ大鉄塔

廃墟を堪能したあとは、山頂にある鉄塔へ向かおう。この鉄塔は高さ 226m、本土との 2.4km の間に送電線を架ける重要な役割を果たしている。流石に鉄柵に囲まれていて足下まで行く事は出来ないが、ある程度の距離からでも十分に迫力のある眺めが楽しめるだろう。何とこの送電線、一番低い地点では海拔 45m まで下り、これは山の上に立っている鉄塔の足場よりも低く、この凄まじい建造技術<sup>\*6</sup> には驚愕を禁じ得ない。本土から引いた電気は大久野島を経由して愛媛県の大三島へと送られるとのこと。



高さ 226m を誇る大鉄塔



廃墟とウサギ

日程の都合で施設を全て回る事は出来なかつたが、この島の魅力を十分に感じる事が出来たと思う。現在我々が過ごしている「平和」が、どのような犠牲の上に成り立っているのかという事実を強烈に印象づけられた。自分達を取り巻く環境と、その過去について考え直す良い機会になるだろう。大久野島の宿の情報などは休暇村の Web サイト<sup>\*7</sup> から参照できる。

翌日、愛らしいウサギ達と寂しげな廃墟に想いを馳せつつ、ウサギ島に別れを告げた。



山道で見かけた仔ウサギ

\*6 しかも建造されたのは昭和 37 年、今から約 50 年も前であるというから尚の事驚きである。

\*7 <http://www.qkamura.or.jp/ohkuno/>

## ■冬だから北陸経由で帰ろう

### ◆経路

(12月28日)	忠海	10:19 発
	↓ 岩線	普通   三原行き
	三原	10:43 着 10:46 発
	↓ 山陽本線	普通   岡山行き
	岡山	12:18 着 13:14 発
	↓ 山陽本線	普通   相生行き
	相生	14:20 着 14:22 発
	↓ 山陽本線・東海道本線	新快速   野洲行き
	野洲	16:42 着 16:54 発
	↓ 東海道本線・北陸本線	新快速   長浜行き
	長浜	17:34 着 17:41 発
	↓ 北陸本線	普通   敦賀行き
	敦賀	18:19 着 18:35 発
	↓ 北陸本線	特急「雷鳥」33号   金沢行き
	福井	19:11 着 19:34 発
	↓ 北陸本線	普通   金沢行き
	金沢	21:00 着 21:03 発
	↓ 北陸本線	普通   富山行き
	高岡	21:40 着
	↓ タクシー	
	葡萄酒君の家	22:00 くらい着
(12月29日)	↓ 車	
	高岡	9:27 発
	↓ 北陸本線	普通   富山行き
	富山	9:45 着 10:24 発
	↓ 北陸本線	普通   直江津行き
	糸魚川	11:36 着 13:33 発 ←ここで昼飯のために途中下車
	↓ 北陸本線	普通   直江津行き
	直江津	14:12 着 14:29 発
	↓ 信越本線	普通   長岡行き
	長岡	15:57 着 16:32 発
	↓ 上越線	普通   水上行き
	水上	18:33 着 18:37 発
	↓ 上越線	普通   高崎行き
	高崎	19:39 着 19:58 発
	↓ 高崎線	普通   上野行き
	上野	21:46 着 21:55 発
	↓ 常磐線	普通   水戸行き
	北千住	22:07 着 22:11 発
	↓ つくばエクスプレス	快速   つくば行き
	つくば	22:45 着

## CX & GR Days

さて、復路についてですが、ウサギ島探検を終えた後なので、帰りはいくら体力を消費しようと、時間を消費しようと関係ありません。そこで青春 18 きっぷの出番です。今回あえて北陸を回って帰ってきました。山陽地方から東京に行くのであれば、東海道本線の利用が圧倒的に便利かつ早いのですが、その利便性を無視して北陸本線を回って帰ってきた理由は 3 つあります。まず、富山県高岡市に、旅行で同行していた葡萄酒君の実家があり、宿泊の許可が出た事。次に、冬の豪雪地帯を通ってみたかった事。そして、そもそも東海道本線は既に何往復もしていて**飽きた**事です。

つまり、どうみても一般的なルートでは無いのですが、そもそも「ひたすら東海道本線を通って広島から東京まで帰ってきました(^-^)v イエイ！」などというごくごく普通の記事が、この情報科学類誌 WORD で許されるはずがありません。あえて北陸を回るなど、しょうもない事に努力を惜しまないのが WORD 編集部員です。

そのようなわけで、ここからはこの頭の悪い経路について、淡々と鉄オタ視点で語ります。

### ◆大都会岡山といえば

経路を見ると、岡山駅で約 1 時間の待ち時間があるのが分かること思います。このような長い乗り換え時間はおみやげ購入タイムやお食事タイムなどに利用しましょう。岡山駅ではきびだんごが沢山売られています。控えめの味でインフィニティー的に食べ続けられます。味も色々ありますが、マスカットきびだんごがオススメです。

### ◆野洲駅



この駅名に反応を示さない情報科学類生はおそらくモグリですね。

### ◆北陸本線の鉄道事情 ～特急編～

ここでは JR 西日本の北陸本線についての紹介をしたいと思います。JR 北陸本線は、滋賀県の米原駅から琵琶湖の東岸沿いに日本海側へ北上し、福井県の敦賀駅から日本海沿いに新潟県の直江津駅までを結ぶ路線です。北陸の主要な都市を結ぶ路線ですが、主要な都市以外の利用客は芳しくありません。

もちろん、そんなことくらい JR 西日本は分かっているので、少しでも収益を増やすべく、たくさんの特急電車が走っています。特に敦賀～富山間は“特急街道”と呼ぶにふさわしく、例えば敦賀駅の富山方面の普通電車 24 本に対して特急・急行は合わせて 41 本も走っています。

さらに、“特急誘導”ダイヤと呼ばれるダイヤが見られます。今回の旅行の経路を例に紹介しましょう。

敦賀	18:19 着	18:35 発
↓ 北陸本線		特急「雷鳥」33 号   金沢行き
福井	19:11 着	19:34 発
↓ 北陸本線		普通   金沢行き
金沢	21:00 着	21:03 発
↓ 北陸本線		普通   富山行き
高岡	21:40 着	

今回の経路のうち、“特急街道”である敦賀～高岡間を抜き出したものです。敦賀～福井まで特急雷鳥に乗っています。仮にこの区間も普通電車を使った場合の経路を載せてみます。

敦賀	18:19 着	18:35 発
↓ 北陸本線		普通   福井行き
福井	19:41 着	20:38 発 ←！！！
↓ 北陸本線		普通   金沢行き
金沢	22:04 着	22:43 発 ←！！！！！！
↓ 北陸本線		普通   富山行き
高岡	23:21 着	

なんと、高岡駅到着が 2 時間も遅れてしまいました！ この理由は、敦賀駅 18:35 発の普通で福井駅まで行くと、福井駅 19:34 発の普通電車にわずか 7 分差で乗れず、福井駅と金沢駅で合計 1 時間半以上もの待ち時間が発生してしまうことがあります。

「たしかに高岡駅には行けるけど、福井駅 19:34 発の普通電車にわずか 7 分差で乗れないせいで 1 時間半ものロスタイム……！ ぐぎぎ…… くやしいっ！ ピクンピクン」

↓

「そうか！ 特急「雷鳥」で福井まで行けば良いんだ！ 少し金はかかるけど、まあいいだろう！」  
これこそが、“特急誘導ダイヤ”です。我々は敦賀～福井でホイホイと特急に誘われてしましました。

ちなみに今回乗った特急「雷鳥」は 1 日 1 往復のみの貴重な特急で、平成 24 年 3 月で廃止になる事が決まっています。廃止前に乗れて良かったと思っています。

## CX & GR Days

### ◆北陸本線の鉄道事情 ～普通車輌編～

特急編で述べた通り、北陸本線の特急の止まらない駅の利用者はかなり少ないので、あまり設備にお金をかけることが出来ません。当然車輌にもあまりお金をかけることが出来ません。そこで国鉄時代に、寝台特急や急行が廃止になった関係で大量に余っていた特急車輌や急行車輌を北陸本線の普通電車用に転用することで、車輌を補いました。新しい車輌が作れる日が来るまでの一時的な運用のつもりだったようですが、導入から四半世紀が経った今も元気に走っています。

そんなユニークな車輌達を紹介しましょう。

#### ・457系

元急行車輌です。

車内の温度計に国鉄のマークがついていました。つまり、その時代から特に手を加えられていないという事です。

白い車体に青い帯というデザインですが、広島周辺の鉄道事情でも述べた塗装の1色化により、順次**青一色**\*8になる事が決まっています。

側面から見るとブルートレインっぽくなるかも。



#### ・419系

寝台特急用の583系を普通電車用に改造した車輌です。一部の窓が開けられるようになったり、吊革を設置した程度の改造にとどめられています。

格納式の寝台は面倒くさいので未だにつけっぱなしです。もちろん乗客が格納されている寝台を展開することは出来ませんが。

元々長い編成だった車輌を3両編成単位にぶつ切りにして使っているので、運転室が無かった所にも運転室をあとづけています。

右の写真に写っている部分は、元から先頭車として作られている部分なのですが、先頭車ではなかった所に運転室をあとづけしたところは、本当に運転室をつけただけの簡単な改造にとどまっています。

その姿は次のページに！



\*8 JR西日本の塗装一色化計画は以下の通り。 山陽：黄色 北陸：青 能登半島：赤 京都：緑  
和歌山：青緑 非電化路線：朱色



あまりに個性的なその姿に、「食パン」と呼ばれています。こちらは近々廃車になる予定らしいので、青一色にはならないそうです。近々廃車になる予定と言っても、ここ数年間「来年こそは廃車にするから！」を繰り返しているので、あと2~3年くらいは走るかもしれません。

ここで挙げた2つの車輌は元々急行または特急車輌だったこともあります。イスの座り心地などは良いです。ただし、扉が2つしか無い事や、詰め込みが効かない事もあり、ラッシュ時には嫌がられている様です。あと、雪国を走るにも関わらず窓の気密性が低いため、すきま風がやたら寒かったです。

#### ◆ほくほく線美佐島駅に行こう！

信越本線の犀潟駅から上越線の六日町駅を結ぶ北越急行ほくほく線という路線があります。『電車でGO! 2』で有名な【要出典】路線です。最高時速160km/hを誇る特急「はくたか」と、それから逃げるために110km/h運転を行う普通電車が走っています。ほぼ全区間が高架または地下という高規格な路線です。

その中でも面白い駅が六日町駅から2駅目の美佐島駅。ほくほく線内で唯一トンネルの中にある駅です。TXのつくば駅で、電車が来る前に耳が痛くなる経験をした方が多いと思います。あれは電車がトンネルに入った事により、トンネル内の気圧が変化して生じる現象なのですが、この駅はその現象のもっとすごい版が起きます。

この駅は特急「はくたか」が容赦なく160km/hで通過するため、特急「はくたか」がトンネルに入った瞬間から、トンネル内に風が生じます。美佐島駅のホームにも強風が生じ、開通前の実験ではダミー人形が吹き飛ばされるという結果が得られています。そのため、美佐島駅では普通電車が到着する時以外はホームと待合室の間の自動ドアを閉めて対処しています。自動ドアの密閉性はそこまで高くないのですが、すきま風が吹き、駅には掃除機をかけたような音が鳴り響く始末です。

また、この駅は外にでても特にこれといった施設がないため、ほとんど利用客がいません。誰も居ない駅に響くすきま風の轟音。私はこれ以上のホラー駅を知りません。あと、夜に行くと星が綺麗です。

ちなみにほくほく線はJR線ではないので青春18きっぷでは利用できません。そのため、今回の旅行ではほくほく線は使いませんでした。

## CX & GR Days

### ◆上越線土合駅に行こう！

群馬県の高崎駅から越後山脈を越えて新潟県の長岡駅をむすぶのが、JR 上越線です。関東から新潟に抜けるルートは上越新幹線が主流なので、まわりに人家の少ない水上～越後中里間は極端に本数が少なく、臨時電車を除くと1日に5本しかありません。

その極端に本数の少ない区間に存在する土合駅は「日本一のモグラ駅」として有名です。この水上～越後中里間は最初に開通した地上を走る上り線と、後から複線化した際に開通した地下を走る下り線にわかっています。そのため、土合駅の上りホームは地上にあるのですが、下りホームに行くためには**462段**の階段を下りて地下ホームまで行く必要があります。

今回は土合駅で下車しませんでしたが、以前訪れたときの写真があるので紹介します。



三角屋根が特徴的な駅舎。



下り線ホームへ続く階段。ここから 440 段ほど続く。



駅前の風景。



地下にある下り線ホーム。

### ■そんなわけで

今回の旅行の目的地であるウサギ島はもちろんの事、道中の色々な所にたくさんのおもしろスポットやおもしろポイントがあると思います。こういったおもしろスポットやおもしろポイントを知っていれば、ウサギ島に着くまでの長い移動時間も面白く過ごす事が出来ると思います。実際に私は全く退屈しませんでした。

ぜひ今度の春休みにウサギ島に出かけてみてはいかがでしょうか？

一方 mitty は、新幹線<sup>9</sup>を使った。



---

\*9 電源車 N700 系

# MOZC カ レー

文 編集部 はろぺり

## ■ Mozc とは

Mozc とは Google 社が開発した日本語入力のオープンソース版の名称です。一般的には Google 日本語入力として公開されています。Google 社の工藤拓氏、小松弘幸氏の 20%プロジェクト<sup>\*1</sup> から始まって正式プロジェクトに昇格し、年末にはとうとうベータ版から正式リリースになりました。オープンソースにできない部分があるためか、Mozc と Google 日本語入力では機能が少し異なる部分もあるようです<sup>\*2</sup>。Google 日本語入力の開発版という位置付けでもあるので、ただ使うだけなら Google 日本語入力のほうがいいかもしれません。

## ■ 技術

Google 日本語入力・Mozc ともにウェブ上の膨大なテキストから変換辞書を作っています。ウェブ上のテキストにはたくさんの誤字が含まれているため「危機一発」<sup>\*3</sup>などのよくある誤変換が新しい名詞として辞書登録されていました。未知語の読みも機械任せなのか「雪歩」<sup>\*4</sup>の読みが「ゆきぽ」だったりといった残念な部分もあります。しかし Google 検索の「もしかして」機能を応用して補正をかけているので、そこまで多くの誤字は登録されていません。

技術的な面の概要が知りたい場合は、ウェブで公開されている妙な漫画を見ると良いと思います。この漫画では Mozc の技術的な概要が一通り説明されています。

<http://www.google.co.jp/intl/ja/ime/comic/>

## ■ 既知の問題

Mac の日本語配列キーボードでは、かな/英数を切り替えるボタンが便利らしいのですが、それが使えないようです。筆者自身は英字配列を使用しているのであまり関係ない話なのですが、このままでは ことえり<sup>\*5</sup>に勝てないので近いうちに改善されるでしょう。

## ■ インストール

ここでは一番メジャーな[要出典]MacOS Xへのインストールのみご紹介します。必要なものは Xcode、Qt、Python の 3つです。Qt は『Mac binary package for Mac OS X 10.5 - 10.6 (32-bit and 64-bit)』というものをインストールしてください<sup>\*6</sup>。Python に関しては、Mac にデフォルトで入っているもので十分です。

---

\*1 有名ですよね。就業時間の 20%は好きな研究に費やせるんです。

\*2 とはいっても、社内ではどっちも Mozc と呼ばれているらしく、実質的な差は大してないと思います。

\*3 正しくは「危機一髪」

\*4 アイマスのキャラ

\*5 MacOS X に搭載されている残念な日本語入力です。

\*6 QtCore, QtGui という Framework が必要です。

以下のコマンドでは gclient<sup>\*7</sup> をインストールして、それを使って Mozc のソースをチェックアウトしてきます。

```
$ svn co http://src.chromium.org/svn/trunk/tools/depot_tools
$ export PATH=`pwd`/depot_tools:"$PATH"
$ mkdir -p ~/src/mozc
$ cd ~/src/mozc
$ gclient config http://mozc.googlecode.com/svn/trunk/src
$ gclient sync
```

そして以下のコマンドで設定、ビルドを行います。

```
$ cd ~/src/mozc/src
$ python build_mozc.py gyp*8
$ python build_mozc.py build_tools -c Release
$ python build_mozc.py build
  -c Release mac/mac.gyp:GoogleJapaneseInputac/mac.gyp:gen_launchd_confs
```

その後 Mozc.app を"/Library/Input Methods"に移動させ、さらに以下の 2 つのファイルを、/Library/LaunchAgents へ移動させてください。

- src/out\_mac/DerivedSources/Release/mac/org.mozc.inputmethod.Japanese.Converter.plist
- src/out\_mac/DerivedSources/Release/mac/org.mozc.inputmethod.Japanese.Renderer.plist

再度ログインすると Mozc が使えるようになります。システム環境設定から IME の切り替えを行ってください。

## ■便利な機能

Mozc には細かいカスタマイズや便利な機能が備わっています。ここではその中からいくつかをご紹介します。

### ◆ローマ字テーブル設定

辞書登録に似た機能を使って「who」と打つと「うお」、「lu」と打つと「う」が入力できるなど、日本語入力システムごとに微妙に異なるローマ字変換のカスタマイズができます。辞書登録と違う点は、置き換えした後も変換は確定していないというところです。この便利な応用として zh, zj, zk, zl<sup>\*9</sup> などの入力を、変換を確定させることなく←↓↑→に置き換えることができます。

---

\*7 gclient は様々なバージョン管理システムをラップするツールのようです。

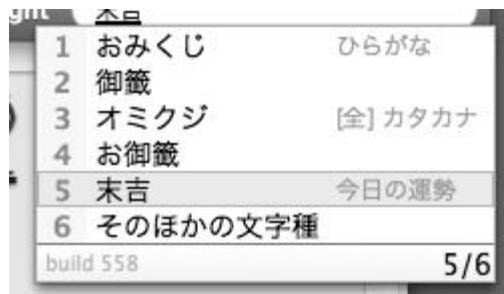
\*8 GYP は Chrome のために開発されたもので、環境に応じて Makefile・Xcode・Visual Studio のプロジェクトファイルを生成する歪みねえツールです。依存関係をハッシュや配列で列挙した Python スクリプトが設定ファイルなのですが、見た目が汚くて、中の人ですらキモいとか言っていました。

\*9 hjkl は、Vi/Vim でのカーソル移動に使うキーです。まあ常識ですね。

## Mozc カレー

### ◆おみくじ機能

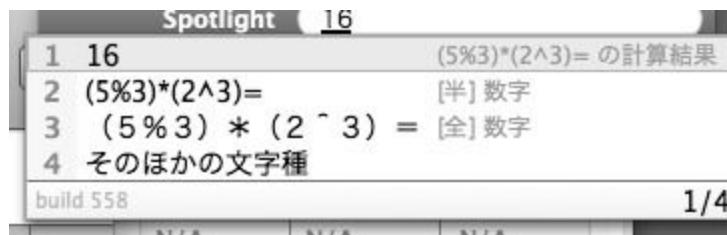
「おみくじ」と打って変換してみてください。今日の運勢が出ます。私の今日の運勢は末吉でした。別に便利でも何でもないですけど。



本日の筆者の運勢

### ◆ Calculator

すでに Google 日本語入力でも利用可能ですが、計算式を入力すると変換候補に、式を評価した結果が出てきます。



「(5%3)\*(2^3)=」の計算結果

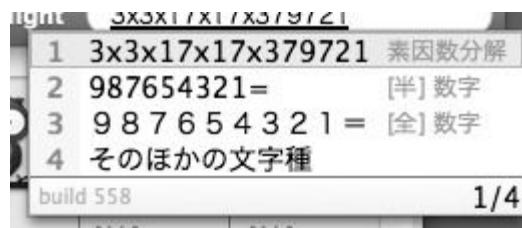
### ■改造

せっかくオープンソースなのでちょっと改造してみましょう、ということで、Calculator を参考に、Rewriter と呼ばれるレイヤに機能を追加してみようと思います。Rewriter は動的に変換候補を作ったり、前後の単語の情報から変換候補の順序を動的に入れ替えたりと、変換候補に対して手を加えることのできるレイヤです。Calculator やおみくじ、日付の変換の機能<sup>\*10</sup>、基数変換<sup>\*11</sup>もここで実現されています。どうやら学習機能もこのレイヤにあるようです。

今回は入力された整数を素因数分解して変換候補に表示する改造を施しました。誰しも、ふと「この数って素数かな?」と気になることが週に一回くらいはあるので、かなり実用的な機能です。紙面の都合によりコードは掲載しませんが実質 100 行も書いていません。~~GYP キモすぎだし、ドキュメント整ってないから不慣れなので私は丸一日費やしましたが、ひとまず動くようになりました。そのうちどこかで公開しようと思います。~~

\*10 例えば「きのう」から「2011/01/22」、「しあさって」から「平成 23 年 1 月 26 日」に変換できます。たいていの日本語入力には備わっている機能です。ことえりには無いけど。

\*11 10 進数の数値を入力して変換すると、16 進法・8 進法・2 進法に変換したものが変換候補に出ます。



987654321 の素因数分解の結果

### ■おわりに

今回の記事では Mozc ばっかりでカレー成分がありませんでしたが、次号ではカレーに関する記事を書こうと思います。

## 型制約の謎に迫る！ in Perl6

文 編集部 葡萄酒

皆さんこんにちは、葡萄酒です。前回は主に Perl6 の導入について触れましたが、今回は Perl6 の世界を軽く紹介したいと思います。時折 Perl5 との比較も交えて説明しますので、しばしお付き合いください。

### コンクリートからオブジェクトへ

手続き指向にオブジェクト指向を強引に後付けしたような 5 とは違い、6 では完全にクラスベースのオブジェクト指向言語に進化しました。これにより、いくらか黒魔術成分は失われてしましましたが、「スカラー変数に"何か"へのリファレンスが格納されているように見える……Data::Dumper 先生呼んでこい！」などという事態は発生しません。Perl6 では動的型付けの柔軟さを保ちながらも、全てのリテラルや変数は必ず何かのクラスに所属しています。一件矛盾したようなこの二つを、Perl6 はどうやって実現しているのでしょうか？これを説明するためには、まず Perl6 の型システムがどういうものであるかを理解する必要があります。

注）これからサンプルコードを幾つか示しますが、説明のために独自の表記を用いますのでご了承ください。「#>」から始まるコメントは直前の式を評価した際の端末への出力結果、「#!>」から始まるコメントは同様に評価した時のエラーメッセージを示します。かなり説明不足のコードが続きますが、意味は何となく把握できると思いますので頑張って読んでください。

### 型制約

Perl6 では変数に型制約を付けることが出来るようになりました。変数を宣言する際に変数名の前に型名を置くことで、その型以外のオブジェクトの代入を基本的に禁止できます。なお、この制約は省略が可能です。

```
# 型制約の例

my Str $hoge_str; # 文字列型に制約
$hoge_str = "hoge"; # ok
$hoge_str = 1; #!> Type check failed for assignment

my $hoge_any; # 型制約無し
$hoge_any = "hoge"; # ok
$hoge_any = 1; # ok
```

変数への代入だけでなく、関数呼び出しにも型は有用です。申し訳程度にしか仮引数の定義が出来なかった Perl5 とは違い、Perl6 では健康で文化的に定義が書けます。更に、関数のオーバー

コード<sup>\*1</sup>が可能になりました。multi修飾子を指定した関数は、違う型の引数をとるパターンを別に記述できます。

```
multi sub double (Int $var) {
    return $var * 2;
}

say double(2); #> 4
say double("hoge");
#!> Nominal type check failed for parameter '$var';
#!> expected Int but got Str instead

# 以下のようにディスパッチを追加すると
multi sub double (Str $var) {
    return $var x 2;
}

say double("hoge"); #> hogehoge
```

このように、実行時ではありますが変数への代入や関数呼び出しの際に型チェックが行われ、望まない型のオブジェクトを弾くことができます。しかし一方で、型制約を書かずPerl5のように変数を扱うことが可能になっています。この不可解とも思われる柔軟性は、実はPerl6のクラス構造と密接に関わっているのです。

## クラス

Perl6のクラス構造はどうなっているのでしょうか。以下の図に簡単に纏めてみました。

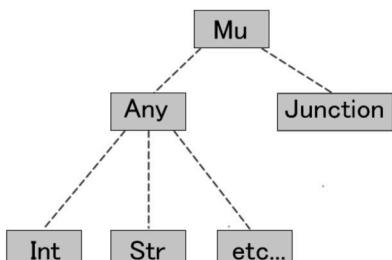


図1 - クラスの継承関係図

まずルートクラスに **Mu** クラス<sup>\*2</sup> があり、全てのクラスはこの **Mu** クラスを継承します。そして、更にそこから **Any** クラスと **Junction** クラスが派生しています。**Junction**についてここでは触れませんが、もう一つの **Any** クラスこそが型付けの鍵を握っているのです。

先程「変数を宣言する際に変数名の前に型名を置くことで、その型以外のオブジェクトの代入を基本的に禁止できます」と書きましたが、「基本的でない」例外があります。それを以下のコード

\*1 コンパイル時ではなく実行時に実際に呼ぶ関数が決定されるため、厳密にはマルチディスパッチといいます。

\*2 日本語の"無"から来ているらしいです。Most Undefinedの略とも。

## Learning Perl6 II

に示します。

```
class MyInt is Int {  
    method who_am_i {  
        return "I am MyInt!";  
    }  
} # Int クラスを継承した MyInt クラスを定義  
  
my Int $integer = MyInt.new; # ok  
say $integer.who_am_i; #> I am MyInt!  
  
# WHAT メソッドはインスタンスの所属しているクラスを返す  
say $integer.WHAT; #> MyInt()
```

最初に Int クラスを継承した MyInt クラスを定義しています。who\_am\_i メソッドを追加した以外は Int と何も変わりません。次に、Int 型に制約したスカラー変数\$intger を宣言しています。右辺では MyInt クラスのコンストラクタを呼んで\$integer に代入しようとしているのですが、何かおかしいと思いませんか？当然、MyInt.new が返すのは MyInt クラスのインスタンスです。しかし、先程も書いたように\$integer は Int 型に制約されているはずです。それにも関わらず代入は成功している、これこそが動的型付けを可能にするトリックなのです！

実は、制約された型を継承した派生クラスは例外的にその変数に代入できるという仕様があります。そして図 1 を見ると分かる通り、Junction 以外の組み込み型は全て Any 型を継承していることを考えると、何となく全貌が見えてきたのではないでしょうか？この推測を裏付けるために、型制約を書かなかった変数のクラスを見てみましょう。

```
my $var;  
say $var.WHAT; #> Any()
```

予想通り Any 型のインスタンスでした。つまり型を指定しなかった変数は暗黙的に Any 型として処理され、Any クラスを継承したクラスのインスタンスは何であろうと自由に代入できるという仕組みだったのです。

ちなみに「派生クラスとは言え、別のクラスを代入できて良いのか？」という疑問が出るかもしれません、基本的には問題ありません。なぜなら、派生クラスのインスタンスは基底クラスのメンバを全て持っていることが期待できるため、同じように扱うことができるからです。もちろん、派生クラスの制約に対して基底クラスのインスタンスを代入することはできません。

### まとめ

Perl5 や Ruby と同じように型に縛られることなく自由に変数を扱うことができ、さらにプログラマが望めば静的型付けのような型制約も可能な Perl6—— なかなか斬新で面白いと思いませんか？実際にコードを書いてみると分かりますが、明示的に型を書いたコードは Perl5 と比べて圧倒的に読みやすさが違います。動的型付けが主流であった LL の中に、今新しい風が吹き込みました。

# PCをゲームコントローラーで操作しよう！

文 編集部 ミレトス

## はじめに

お久しぶりです。最近 Twitter でぺろぺろ言うのが癖になってるミレトスです。皆さん冬休みはどうでしたか？筆者は年末にインフルエンザにかかり、新年をベッドの上で迎えました。今年は体調管理を目標にしていきたいと思います。(キツ)

さて、今回はコントローラーで PC を操作する方法を書きます。情報に来る人間なら誰しも一度くらいは「コントローラーで PC が操作できたらいいのに」「現実で検索が使えたらしいのに」

「どこでもドアがあればいいのに」という事を考へると思います。今回はその内の 1 つを私が叶えて差し上げましょう！

## メリット・デメリット

何をするにもメリット・デメリットは大事ですよね。ということでパッと思いつくものを挙げてみました。

### メリット

- ・ゲームみたいに操作できる
- ・PC の前に居なくても操作出来る
- ・マウス要らない子
- ・ベッドから出ない生活が出来るようになる（上級者のみ）
- ・コントローラーたん可愛いよコントローラーたんはあはあペロペロってなれる

### デメリット

- ・ベッドから出ないため運動不足に
- ・コントローラーペロペロしすぎてベトベトになる

こう比べてみると明らかにメリットの方が多いですね。つまりこれは導入するしか無い！

## 環境

筆者が今回解説する環境は以下の通りです。

OS: Windows XP SP2
ソフト : JoyToKey <sup>*</sup> Ver4.6
ゲームパッドコンバーター: ELECOM JC-PS101UWH
コントローラー: Sony 純製 Playstation2 コントローラー (以下 PS2 コントローラー)

何が言いたいかっていうと Windows での設定の仕方ってことですね。Mac や Linux の場合は手元に環境が無いので検証できません。アーサンセンドナー

\*1 JoyToKey : コントローラーの操作をマウスやキーボードの操作に変えてくれる素晴らしいソフト。サイトはこちら <http://www.ac.auone-net.jp/~jtk/>

## PC コントローラー

### 準備

まずコントローラーを PC に接続するコンバーターが必要ですね。USB 接続できるゲームパッド等をお持ちの方はそのまま大丈夫です。持っていない方は予め買っておきましょう。

次にソフトを入れます。今回使うのは「JoyToKey」というソフトです。別にインストール等は必要ないので、サイトからダウンロードしたら中のファイルをどこか適当な所に置いておけば大丈夫です。

### 設定

それでは早速ソフトの設定に移りましょう。JoyToKey.exe を起動すると以下のようないい画面が出てくると思います。



この右側が主に弄り回す場所となります。まず最初に、「Preferences」をクリックし、「十字キー×1」となっている所を「十字キー×2 + POV × 1」にします。その下の斜め入力の項やジョイスティックの数、しきい値などは特に設定する必要はありません。そして先程の「Joysticks」の方に戻ると次のページの画像のようになっていると思います。



ここで右側のボタンの欄と実際のコントローラーとの対応を示しておきます。今回は PS2 コントローラー用なので、ファミコンやスーパーファミ、64 コントローラーなどは対応していない可能性がありますのであしからず。

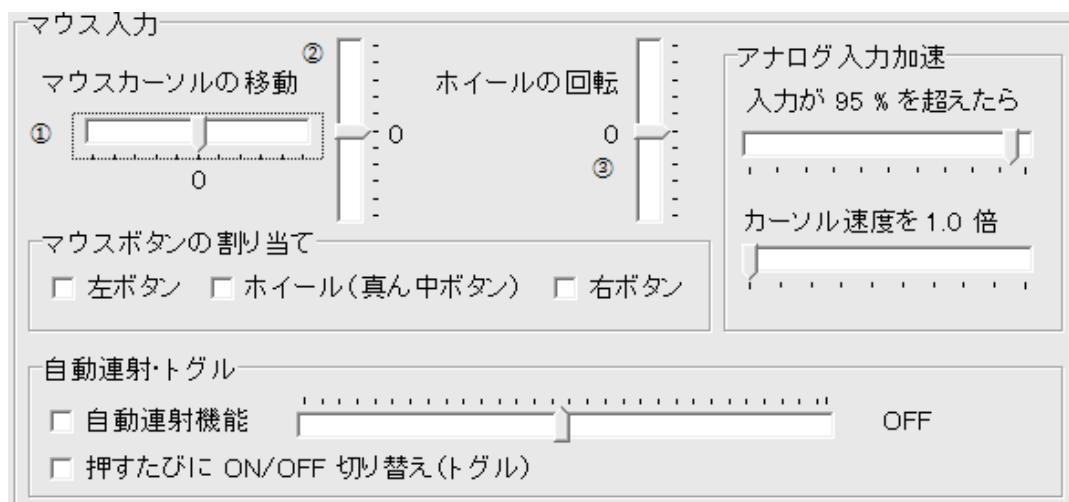
Stick1:←	左アナログスティック左	Button1	△
Stick1:→	左アナログスティック右	Button2	○
Stick1:↑	左アナログスティック上	Button3	×
Stick1:↓	左アナログスティック下	Button4	□
Stick2:←	右アナログスティック左	Button5	L2
Stick2:→	右アナログスティック右	Button6	R2
Stick2:↑	右アナログスティック上	Button7	L1
Stick2:↓	右アナログスティック下	Button8	R1
POV1:←	十字キー左	Button9	スタート
POV1:→	十字キー右	Button10	セレクト
POV1:↑	十字キー上	Button11	R3
POV1:↓	十字キー下	Button12	L3

これ以上のボタンは PS2 コントローラーには対応していないので意味ありません。残念。  
さて、それでは設定していきます。まず最初に新規作成を押して設定ファイルを作成します。  
これを切り替えることによって設定を変えることが出来るので便利です。早い人はこの辺りからコントローラーが愛おしくなってきます。

## PC コントローラー



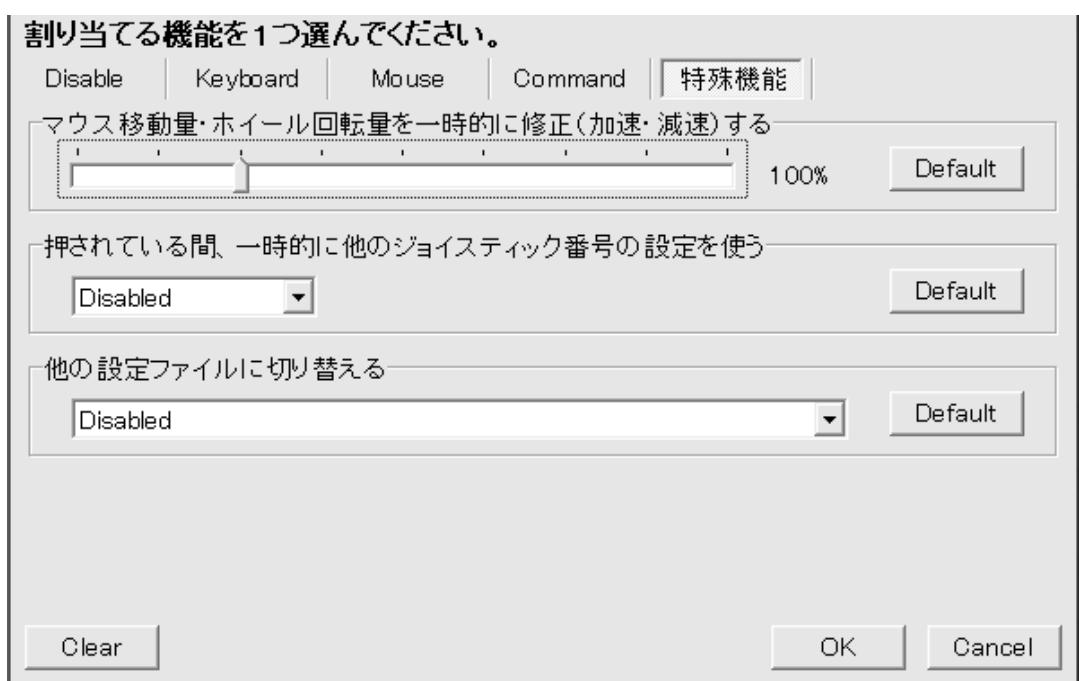
右側の「Button2」ダブルクリックしてみてください。上の画像のような画面が表示されたる思います。このソフトで設定できるのは、キーボード上のキー、マウス、ホイールの動きです。開いた直後は左上の Disabled の所が黄色になっている欄があると思います。そこでキーボード上のキーを何か押してみましょう。例えば A のキーを押すと A と表示されます。その状態で対応したボタンを押すと、A が打ち込まれます。「Button 2」に A を設定すると、「○ボタンを押す」という行為が「キーボードの A を押す」という行為に置き換わって PC に伝わっています。では次に上方にある「Mouse」を押してみましょう。下のような画面が出てくるはずです。



こちらではマウス機能を設定する事が出来ます。前ページの①でマウスの左右の動き、②でマウスの上下の動き、③でホイールの動きを設定することが出来ます。数値が大きい程マウスカーソルが早く動きます。このへんの設定は直感的に出来るようになっています。左に 50、上に 30 として斜めに動くように設定することも出来ます。マウスボタンの割り当ての項目にチェックを入れると、その機能が使えるようになります。左ボタンをチェックして、コントローラーの該当するボタンを押すと左クリックするということですね。

アナログ入力加速というのは、アナログスティックをどれだけ傾けたらどれくらいの倍速にするか、という設定です。例えば「50 %を超えたたら 2 倍」と設定すると、アナログスティックを 50 %以上傾けたらカーソルの移動速度が 2 倍になります。ね？だんだんペロペロしたくなってきたでしょう？才能のある人はこら辺で既に「やべえ！何これ！ペロペロすげえコントローラーたんすごいおペロペロペロペロ」となっています。ん？俺じゃないよ？

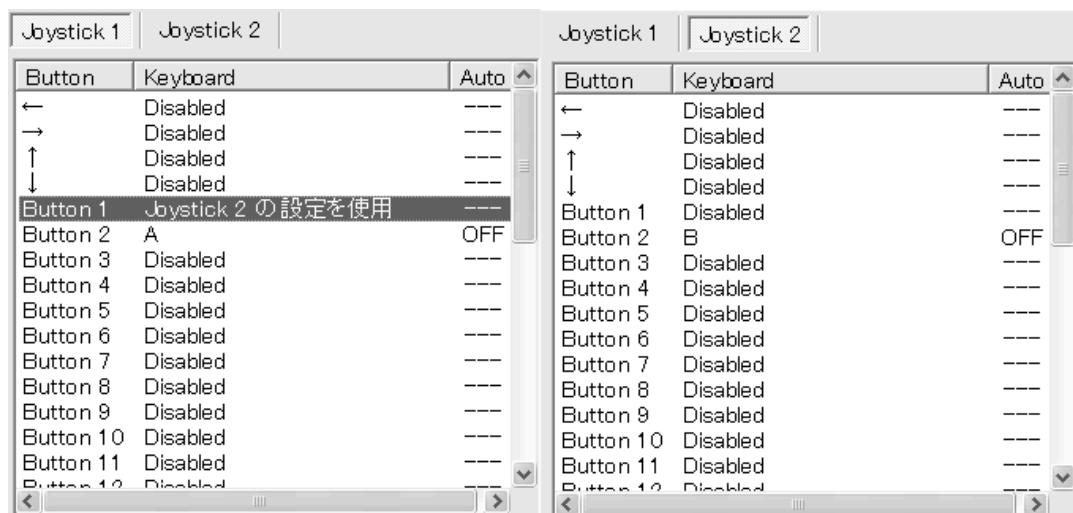
さて、先程から気になっている人もいるであろう特殊機能を見てみましょう。「特殊機能」を押すと下のような画面が表示されると思います。



一番上のマウスについての設定は、「これを設定したボタンを押しながらマウスカーソルを動かすと、指定した倍率分速く動く」というものです。例えば「Button2」(つまり○ボタン)を 200 %に設定します。左アナログスティックでマウス操作出来るようにしてある場合、○ボタンを押しながら左アナログスティックを動かすと、設定の 2 倍の速さでカーソルが動きます。

## PC コントローラー

他のジョイスティック番号の設定について説明しましょう。色々なボタンにあれもこれもと設定すると、もう設定する場所がなくなって足りないよー！もっとペロペロさせてくれよおおおおおおおおおおおお！！1となる場合があると思います（※当社調べ）。このような場合は、仮想的なコントローラーを作り、その設定を使うようにすることができます。ボタンの所の上の部分に「Joystick1」「Joystick2」とありますね。普段の操作は Joystick1 の方に設定されています。具体的な使い方としては、Joystick1 の方の Button2（○ボタン）に A キー、Joystick2 の Button2（○ボタン）に B キーが設定されているとします。そこで Joystick1 の Button1（△ボタン）の設定で先程の特殊機能の 1 段目で Joystick1 を選びます。すると以下の画像の様な状態になるとと思います。



この状態で何もせずに○ボタンを押すと A がタイプされます。しかし、△ボタンを押しながら○ボタンを押すと、B がタイプされます。そう、コントローラーの機能をまるまる切り替えられるのです！これを使えばもっといっぱいペロペロ出来るんだよ！ちなみに Joystick の数は Preferences で変えることができます。最大で 16 本まで増やせます。そこまで増やして何をするのかはわかりませんが……。

最後は設定ファイルの切り替えです。最初にみなさんは思い思いの名前をつけて設定ファイルを作ったと思います。用途別に合わせて何個か作ったときに、「この時だけはこっちの設定ファイルにしたいなー」となる場面があればこの機能の出番です。作った設定ファイルの中から切り替えたいファイルを選べば、そのボタンを押すと切り替えられるようになります。Joystick の切り替えに似ていますが、こちらはボタンから手を離しても切り替わった状態のままであることに注意してください。

さて、一通りの機能を説明したところで、どのように設定するのかを考えていきましょう。今回は筆者が普段使用している設定を紹介します。

応用編

Joystick 1			Joystick 2		
Button	Keyboard	Auto	Button	Keyboard	Auto
Stick1: ←	Mouse: ←(50)	---	Stick1: ←	Mouse: ←(75)	---
Stick1: →	Mouse: →(50)	---	Stick1: →	Mouse: →(75)	---
Stick1: ↑	Mouse: ↑(50)	---	Stick1: ↑	Mouse: ↑(75)	---
Stick1: ↓	Mouse: ↓(50)	---	Stick1: ↓	Mouse: ↓(75)	---
Stick2: ←	Disabled	---	Stick2: ←	Disabled	---
Stick2: →	Disabled	---	Stick2: →	Disabled	---
Stick2: ↑	Disabled	---	Stick2: ↑	Disabled	---
Stick2: ↓	Disabled	---	Stick2: ↓	Disabled	---
POV1: ↑	Ctrl, V	OFF	POV1: ↑	Disabled	---
POV1: →	Ctrl, T	OFF	POV1: →	Disabled	---
POV1: ↓	Disabled	---	POV1: ↓	Disabled	---
POV1: ←	Ctrl, C	OFF	POV1: ←	Disabled	---
Button 1	Joystick 2 の設定を使用	---	Button 1	Disabled	---
Button 2	Mouse: Left	OFF	Button 2	L-Win, D	OFF
Button 3	Mouse: Right	OFF	Button 3	Disabled	---
Button 4	Enter	OFF	Button 4	Disabled	---
Button 5	Mouse: Wheel(-15)	---	Button 5	Alt	OFF
Button 6	Mouse: Wheel(15)	---	Button 6	Mouse: Wheel(15)	---
Button 7	Disabled	---	Button 7	Tab	OFF
Button 8	F11	OFF	Button 8	Ctrl	OFF
Button 9	Disabled	---	Button 9	Disabled	---
Button 10	Disabled	---	Button 10	Disabled	---
Button 11	Disabled	---	Button 11	Disabled	---
Button 12	Mouse: Middle	OFF	Button 12	Disabled	---

こちらが筆者が設定している内容です。ネットサーフィンや Twitter の TL を眺めたり、ボクたち御用達の**女の子がちょっと多めの読むゲーム**をプレイする分にはこれで十分です。POV1 の設定を見てもらうと分かるのですが、1つのボタンに複数のキーを設定することも出来ます。先ほど設定した時も3つ登録できる場所がありましたね。登録したキーは上から順番に処理していくので、順番を間違えたりするとおりに動作しないことがあるので注意しましょう。他にも様々なファイルがあります。筆者はたまにトリックスター<sup>2</sup>というネットゲをやるのですが、スキルをファンクションキーに設定するゲームは他にもたくさんあると思います。マウスはまだしもファンクションキーはキーボードの上の方にあるし、押すの面倒だし……そう思った筆者はコントローラーの左側にファンクションキーを全て対応させることにしました。これで楽にゲーム出来るね！コントローラーたんまじ万能ペろべろ！

それともう一つ、相性の良いブラウザを紹介しておきます。それはマウスジェスチャー機能のあるブラウザです。Operaなどに代表されるこの機能は、右クリックしながらマウスを移動させる動作等によって、様々な操作をすることが出来ます。例えば右クリックしながらマウスを上に動かすと新規タブが開くように設定出来ます。大抵のブラウザでは追加で使えるようになっています。この機能があると、大抵の操作が右クリックとマウス移動で実現できるので、それ以外の

\*2 トリックスター：最近はイカ娘や人気声優などとタイアップしたりしている。サイトはこちら <http://www.trickster.jp/>

## PC コントローラー

場所にもっと色々な機能を入れることができます。更に ChromePlus というブラウザにスーパードラッグ<sup>\*3</sup> と呼ばれる機能があるのですが、これに似た機能があるブラウザとも相性がいいと思います。

ここまで来ると大抵の人は「コントローラーは俺の嫁」と言って嫁争奪戦に参加していると思うのですが、もしかしたら「タイピングも出来ないのかよだめじやん」という方もいるかもしれません……。ですがご安心ください。コントローラーでもタイピングは出来ます。確かにタイピング速度は早くないですが、出来ないことはないんです！だってコントローラーたん最強だから！！！！11漢字変換も可能です。筆者は今のところ Joystick を 7 本で実現しています。まだまだ荒削りなのでもっと減らせると思います。実用段階まで至った時にはまたご紹介いたしましょう。

### 終わりに

上級者になると布団から出ずにネットサーフィンし、1 日が過ぎてしまうことも少なく有りません。皆さんどうか用法・用量を正しく守ってお使いください。これによって 1 限に出られなくなったり、外に出るのが面倒になったり、学校に行かなくなったりしたとしてもそれはコントローラーたんが魅力的すぎるからで、私のせいではありませんのでその辺り勘違いしないように。



筆者の相棒のコントローラーたん

---

\*3 スーパードラッグ：リンク先をドラッグして適当な場所に引っ張ると新しいタブで開いたり、単語をドラッグすると検索してくれたりする機能。これがあると検索がとても楽です。

# WiFi 暗号がやばい件について

文 編集部 zer0day

## はじめに

※用語の説明はめんどいのでしません。分からぬ言葉は各自検索などしてください。

みなさんおはようございます。zer0day です。

今回のあやしいお話は、みなさんもお馴染みのことかと思います、無線 LAN の暗号についてです。旧型 Nintendo DS などを使いならばまだ「WEP」にお世話になってるのではないかと思います。

え？ WEP って何？酔っぱらいがコアダンプしそうなときの変な声？

の一の一。無線 LAN の電波通信を暗号化する規格の 1 つです。ほかにもより安全な暗号として WPA, WPA2 があります。

## 「より安全な」

そうなんです、WEP は実に馬鹿でしゃーない奴なんです。みなさんも WEP が 5 分で破られるだの、もうちょっと新しい話題では 8 秒で破られるだの、いろんなところで WEP の糞っぷりを噂に聞いていると思います。今回はそんな WEP がいかにダメか、実演をもって証明したい<sup>\*</sup>と思います。

## 用意する物

- ・無線 LAN 対応 PC、あるいは PC と外付け無線 LAN アダプタ
- ・某無線クラッキング用ソフトおよびそれがまともに動く OS (Ubuntu 推奨)

はい、これだけ。

WEP を破るのに特別な装備なんて要らないんです。大丈夫だ、問題ない。まあ、一番いい装備を頼むのなら「インターネットが一生タダ！」のあの GSKY<sup>\*\*</sup> とかいいですね。今となっては少々乗り遅れ感がありますが。

ちなみに、私は旧型 MacBook Pro に Ubuntu 10.04.1 を積んでいます。無線チップセットは Atheros AR5008。文句ナシ！

\*1 後述するお約束が守れない場合はマネしないように。法律にバッチリ抵触します。

\*2 これは出力がやばいので日本の電波法には違反します。やるなら電波暗室で。

## ALL YOUR WAVE ARE BELONG TO US!

### たのしいところ

まず、チップをモニタリングモードに落として、パケットの収集を始めます。

次に、近隣の無線アクセスポイントの情報を集めます。

```
File Edit View Terminal Tabs Help
Terminal Terminal Terminal
CH 11 ][ BAT: 1 hour 24 mins ][ Elapsed: 20 s ][ 2010-11-02 17:46
BSSID PWR Beacons #Data, #/s CH MB ENC CIPHER AUTH ESSID
00:0D:0B:4F:21:E3 -57 26 1 0 13 54 WEP WEP SECURITY_TEST
-58 29 23 0 1 54 WPA TKIP PSK
-66 26 0 0 6 54 WPA2 CCMP PSK
-70 39 0 0 6 54e WEP WEP
-76 33 0 0 7 54 WPA TKIP PSK
-82 6 0 0 7 2 . OPN
```

SSID、BSSIDなどを頼りにお目当てのアクセスポイントを定めます。チャンネルと BSSID を控えたら、ターゲットの通信だけを拾うことができます。

```
File Edit View Terminal Tabs Help
Terminal Terminal Terminal
CH 13 ][ BAT: 1 hour 15 mins ][ Elapsed: 3 mins ][ 2010-11-02 17:52
BSSID PWR RXQ Beacons #Data, #/s CH MB ENC CIPHER AUTH ESSID
00:0D:0B:4F:21:E3 -57 100 17423 92 13 54 WEP WEP SECURITY_TEST
```

通信が多い場合（例えば既に接続している人が動画をストリーミングしているなど）はこの状態で待つだけで、解析に必要なパケットは集まってくるんですよ。上の画面に「#Data」の数が表れますか、これが多いほど解析がスムーズに進みます。5万個ほど集まつたらよいよクラックです。

```
[00:06:57] Tested 291634 keys (got 41226 IVs)
KB depth byte(vote)
0 0/ 1 67(57484) 22(48096) EC(47660) F9(47408) C8(47248) EB(46812) 69(46424)
1 0/ 1 68(55884) 78(51732) 4E(51688) 26(49132) 9F(46540) 4C(48266) CA(47084)
2 0/ 1 6F(56340) FC(48436) DD(47856) 05(47712) F4(47704) 78(47656) 86(47372)
3 0/ 1 73(54260) 7D(50128) 5F(49120) E2(48996) D1(48544) 9F(48332) 7F(47948)
4 0/ 1 74(56564) AE(50184) AA(49016) 0E(48656) 52(48152) A9(47084) 9F(47040)
5 0/ 1 6C(58208) 2B(47252) 63(47072) D4(46900) FF(46852) D5(46840) AC(46812)
6 0/ 1 79(53208) B7(48696) B3(48600) 04(47552) D1(47404) 29(47340) CD(47264)
7 0/ 1 65(51600) 35(49172) 1C(48248) F9(47516) D2(47276) A2(47224) 14(47216)
8 0/ 3 D9(59468) 65(48176) E8(48148) D8(47772) EB(47400) 56(47236) 89(47268)
9 0/ 2 65(51820) 50(49752) 1A(47956) 66(47894) 10(47552) BF(47116) B9(46684)
10 1/ 1 D1(48444) 1B(48108) 7A(48028) 54(47776) 37(47732) 6B(47504) 3D(47396)
11 0/ 1 15(49456) C7(48540) F9(48140) 88(48128) 2F(47840) AB(47596) 14(46996)
12 0/ 1 30(52644) 35(50012) DE(49112) B2(48696) 10(47988) 1F(47852) 9E(47852)

KEY FOUND! [ 67:68:6F:73:74:6C:79:65:79:65:73:30:38 ] (ASCII: ghostlyeyes00 )
Decrypted correctly: 100%
```

あとはこいつがやってくれます。このコマンドを実行してから暗号キーを得るまで、運が良ければ1秒以下、長くて1分です。ここは少し確率的なところがあるので、IV<sup>\*3</sup>(#Data)を7万個集めても解読できない場合があるのでそれは運が悪かったということです。

\*3 初期化ベクター。WEP の弱点の根本的な原因であり、データパケットに含まれる。

どうでしょう、ここまで流れはだいたい 5 分です。WEP を実装している無線 LAN で大量のデータのやり取りをすると、お隣さんに暗号キーを簡単に解くチャンスを与えててしまうわけです。

WEP がここまで簡単に破られてしまう理由には、大きく次の特徴が挙げられます。

- ・鍵長が不十分 (40/104bit)
- ・IV が存在するため、鍵の復元が容易
- ・同一の鍵をいつまでも使い続ける。

WEP は 1997 年から「Wired Equivalent Privacy」の名を冠し使われ始めましたが、2001 年ごろから次々と弱点が見つかり、2008 年には、わずか 8 秒で WEP を破る方法である KOBECRACK が発明され、WEP はいよいよ気休めにもならない代物と化してしまいました。今も WEP のみ対応である製品は出回っており、なかなか WEP を脱することができないどころか WEP を使っているアクセスポイントが増えているのが現状です。こわいこわい。

### **WPAは安全？**

WEP がくたばってしまったので、後継の暗号規格としてより安全な WPA や WPA2 が発明されました。しかし、そんな WPA も弱いパスワードでは台無しなのです。それはどういうことかとすると、以下の通りです。

WPA-PSK の弱い部分はズバリ、認証・接続時の 4 パケット。この 4 パケットさえあればあとはパワーのあるマシンで総当たり攻撃をするだけなんです。同じソフトを使ってこれをやってみましょう。

### **WPA-PSKをつぶす**

WEP のときと同じ方法でお目当ての AP に目をつけます。

誰かが接続するのを待つか、既に接続されているマシンがあればそれをこちらから強制的に切断します (Deauthentication 攻撃)。このことについては次回号で詳しく取り上げますので、お楽しみに！

ハンドシェイクをゲットできたら、あとは辞書攻撃でもやって待つだけ。

### **どうしてこうなった**

WPA-PSK の最初の 4 ウェイハンドシェイクの部分を計算で出すことによって、拾ったハンドシェイクと一致するかどうか確かめることができます。しかし、暗号化処理を伴うため非常に長い時間がかかり、パスワードが見つかるとしたらたいてい数時間～数日後です。ただ私の実験環境下では明らかに辞書に載ってそうな文字列を選んだので、ものの数秒で破れました。

### **じゃあ何がいいの**

WPA2-AES に限ります！現行最強で、現在のところ静的解析法は見つかっていません。それでも先に述べたとおり、暗号鍵（パスワード）が単純だとアレなので、ASCII にして 30 文字以上を

## ALL YOUR WAVE ARE BELONG TO US!

おすすめします。ちなみに上限は 63 文字です。

### 次回予告

次回も無線 LAN をいじいじする記事の予定です。

主な内容は、

- ・「パケットインジェクションで WEP のクラックを加速させよう」
- ・「Deauthentication 攻撃で手っ取り早く WPA-PSK のハンドシェイクを手に入れよう」
- ・「MAC アドレス制限？なにそれおいしいの？」
- ・「野良 AP の恐ろしさ、お見せします」

### あとがき・注意事項

私がこんなダークな記事をやり方まで書いているのは、セキュリティの甘い設定の無線 LAN の怖さを自分の手で体験し、「WEP とか糞パスワードとかやべえな、早いとこ WPA2-AES にして 63 文字のパスワードを使おう」という動機にしてほしいからです。よい子は他人の AP クラックしちゃダメだぜ！

ちなみにまじめな話、自分が管理していない無線 LAN の暗号キーを復元する行為は、電波法第 109 条の 2 で明確に禁止されています。未遂もアウトなので許可を受けていない対象には絶対にやらないように。

完



## -Produce 1-

文 編集部 Santarh

### 第一幕 -私はアイドル-

現実は、クソゲーだ。そう思うことが普通である世界に引きこまれた僕は二次元の偶像にすがる。そんな世界の落とし神もこう言っている。

「フン、アイドルとか TV とか、もはや前時代の遺物だよ。今世紀は、ゲームアイドルの時代だ！！」（桂木桂馬/神のみぞ知るセカイ）

というわけで、素晴らしいゲームアイドルを、この手で創造しようではないか。その第一歩として、Kinect を利用したモーションキャプチャによるキャラクタ操作を行おう！……偶像を自ら操るという行為はある意味背徳的である。

### Kinect を PC に導入する

さて、この記事では Kinect と OpenNI を利用したモーションキャプチャがいかに楽かということを紹介していきます。

まずははじめに PC で Kinect を利用するために物理的に必要なものを以下に挙げます。

1. PC
2. Kinect
3. 十分な広さの空間

1についてですが、現在 OS は Windows, Ubuntu が stable で対応しています。また、unstable な最新版では Mac OSX に対応済みです。なお本記事においては Windows 環境を前提に説明します。

次に 2について。現在 Kinect を入手するには Kinect 同梱の Xbox360 を買う、Kinect センサー単体で買う、の 2つの方法があります。ここで注意すべきなのが、同梱版の Kinect センサーには独自形式の端子から USB 端子へ変換するケーブルが付属していない、ということです。この変換ケーブルを別途入手するには米国の Microsoft ストアでオンライン購入しなければなりません。よって、Kinect 単体で買うことをお勧めします。

最後に 3についてですが……こればかりは、どうこうなるものではありません。最低、直線距離で 2 m の余裕があれば全身は映ります。自らの踊りを実用的に写像したい人は、公式で推奨されているように両手を振り回せる程度のスペースを用意しましょう！

## 降臨！偶像大師

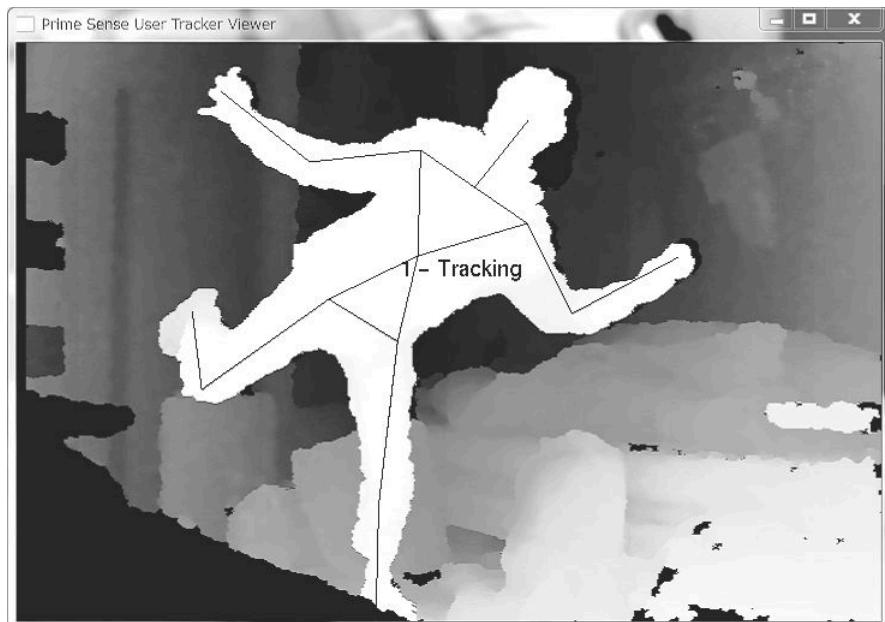
次に、必要なソフトウェアを以下に挙げます。

1. Kinect Sensor Driver (有志による Kinect ドライバ)
2. OpenNI (Natural Interaction のためのフレームワーク)
3. PrimeSense NITE (OpenNI に準拠したミドルウェア)

これらを、以下に示す順序でインストールしてきましょう。

- ① "Kinect Sensor Driver" のドライバをダウンロード<sup>1</sup> します。次に Kinect を繋ぎ、デバイスマネージャからドライバファイルを参照してドライバをインストールします。
- ② OpenNI のインストーラを OpenNI の Web サイト<sup>2</sup> から入手し、インストールします。
- ③ "Kinect Sensor Driver" に含まれている "PrimeSensor (Mod Kinect)" をインストールします。
- ④ "PrimeSense NITE" のインストーラを OpenNI の Web サイトから入手し、インストールします。この "PrimeSense NITE" のインストールにはライセンスキーが必要ですが、OpenNI の Web サイトに書いてあるのでそれをメモしておきましょう。

以上で導入は終了ですが、OpenNI は日々更新されているので分からなければググることを推奨します。きちんと導入できたかどうかを確かめるために、インストールした OpenNI に含まれているサンプルプログラムを実行し、正常に実行できるかどうか試してみましょう。画像は、サンプルプログラム "NiUserTracker.exe" を実行した様子です。ゴミゴミした部屋の中でも、正常に人物をトラッキング出来ていることが分かります。



\*1 <https://github.com/avin2/SensorKinect>

\*2 <http://www.openni.org/>

## Kinect を活用する

導入が終われば、出まわっている Kinect 対応ソフトはほとんど動かせることになります。てつとり早くモーションキャプチャの技術を体感したければ、MikuMikuDance<sup>\*3</sup> がお勧めです。思い思いのキャラクターを操作しましょう。

## OpenNI で開発する

さて、OpenNI を実際に自分で利用してみましょう。OpenNI は非常に簡単に、利用することができます。例えばエラー処理のコードを書かなければ、20 行ほどで深度画像を取得することができます。その最小限の、C++で書かれたサンプルコードを以下に示します。

```
#include <XnOpenNI.h>
#include <XnCodecIDs.h>
#include <XnCppWrapper.h>

...
XnStatus nRetVal = XN_STATUS_OK;
xn::Context context;
xn::DepthGenerator depth;
xn::DepthMetaData depthMD;

nRetVal = context.InitFromXmlFile( "SampleConfig.xml" );           // 注釈*4 参照
nRetVal = context.FindExistingNode(XN_NODE_TYPE_DEPTH, depth);
while (!XnOSWasKeyboardHit())
{
    nRetVal = context.WaitOneUpdateAll(depth);
    nRetVal = depth.GetMetaData(depthMD);
    const XnDepthPixel* pDepthMap = depthMD.Data();
}
...
```

かなり省略されたコードではありますが、これだけで pDepthMap に深度画像へのポインタが入ることになります。このデータは 11bit で 2048 段階のデータになっているので、RGB 表示したければ 256 段階に圧縮するなりするなどの工夫が必要になります。より詳しく知りたければ、OpenNI 同梱の Samples を参照することをお勧めします。残念ながら OpenNI の公式リファレンスは分かりにくいので、OpenNI に付属するサンプルコードを読んだ方がよく分かります。

さて、深度画像があればカメラ視点から見える物体までの距離がミリメートル単位で分かります。従来は画像から人間を検出しようと思っても大変難しい技術が必要でしたが、OpenNI+NITE は人間のボーンを簡単に取得することができます。

\*3 <http://www.geocities.jp/higuchuu4/>

\*4 "SampleConfig.xml" は "OpenNIInstallDirectory\\$\Data\\$SampleConfig.xml" と同一のファイルです。

## 降臨！偶像大師



画像は OpenNI を利用して、自作の DirectX11 アプリケーションにボーン情報と深度画像を読み込ませたものです。OpenNI を利用するのはもちろん初めてでしたが、OpenNI 利用部分の実装は 3 時間、300 行程度で済みました。

### さいごに

特に具体的な実装もなく、順当な紹介記事となりましたがいかがでしょうか。Kinect の魅力について十分分かって頂けたでしょうか。これだけ素晴らしいことができるにもかかわらず、たったの定価 14800 円です。僕は Xbox360 を 2 台持っていますが、持っていないあなたも Kinect を購入して新たなユーザインターフェースを体感してみてください。

今回はこれにて終わりですが、次回は具体的にスキンメッシュ等を組み込んでキャラクタを動かしてみる予定あります。乞うご期待！



# お詫びと訂正

文 編集部 ふあい

## ■おしらせ

昨年の 10 月に**ばらまかれた** 配布された社会学類誌「そおしあ～る」Vol.127において、多数の**だらしねえ誤字・誤表記**がたくさん見受けられました。思わず「多数」と「たくさん」で重複表現をしてしまうくらい、多数見受けられました。

我々 WORD 編集部は、このような**だらしねえ誤字・誤表記**の数々に遺憾の意を表明すると共に、社会学類誌委員会に代わって訂正を致します。お詫びはしません。だってこっちは何も悪くないし。



これが問題の冊子「そおしあ～る」Vol.127です。WORD 編集部員が学祭中に 3 冊も貰ってきたらしいですが、そんな事より中央上側に見える小麦ちゃんが一番かわいいのがお分かりいただけるでしょうか。

では、「そおしあ～る」Vol.127 の**だらしねえ表記**の数々をご覧下さい。

## お詫びと訂正

### ■ページ数とそしあの法則

さて、早速「〇〇ページの××という部分は誤りで、正しくは△△でした」というふうに指摘したいところですが、ページ数 자체が怪しい事になっているのでそこから見てみましょう。



問題の箇所です。28 ページの次が 27 ページになっていて、そんな事より小麦ちゃんがかわいいのがお分かりいただけるでしょうか。

これについて考察してみましょう。通常 N ページの次は N+1 ページなのですが、ここでは

$$N = 28 \quad \text{---} \quad ①$$

$$N+1 = 27 \quad \text{---} \quad ②$$

という連立方程式が成り立っています。

ここで、①を②に代入すると、

$$29 = 27$$

両辺から 27 を引いて、

$$2 = 0$$

という式が成り立ちました。これを「そしあの法則 (Law of Socia)」と呼ぶ事にしましょう。「そしあの法則」は現実世界においてどのように応用できるのでしょうか。

そしあの法則を用いると、 $21000 = 20998+2 = 20998+0 = 20998$  = (中略) = 10500 という式が成り立つので、21000 円のおせちを 10500 円で販売する事ができるし、おせちの品数が 33 品でも 25 品でも等価である事が言えるし、テストの点数が 0 点でも 100 点と等価なので単位が貰えるはずだし、そもそも単位取得数 0 であっても、それは 100000000000 単位取得済みという状態と同じなので、偉大なる山口学類長先生——っ！！俺だ———っ！！！  
学位をくれ———っ！！！<sup>1\*</sup>

というわけで、「そしあの法則」は想像を絶するほど夢と希望に満ちあふれた素晴らしい法則である事が分かりました。こんな素晴らしい法則に出会えた私は、きっと特別な存在なのだと感じました。この法則に対してはどうお礼を申し上げていいか分かりません。だって別に感謝してないし。

### ■本文をつつこう！

つまるところ、信じがたい事にページ数が間違っているという事と、小麦ちゃんはかわいいという事が分かったところで、最初の記事から順に本文を見てみましょう。<sup>2</sup>

#### ◆ p.4-p.9 残暑納涼企画筑波大学七不思議を追え！

- ・ p.6 不振に思い→不審に思い  
よくある誤変換。
- ・ p.8 車に跳ねられてしまった→車に撥ねられてしまった  
私は車の知識には長けていませんが、車は跳ねないと思います。
- ・ p.9 お願いいいたしたい。→お願いたしたい。  
「先生がいらっしゃいやがった」レベルのちぐはぐな表現。

#### ◆ p.11-p.13 ソシア新歓○×△！？2010

- ・三點リーダ(…の代りに、中黒 3 つ(・・・)が使われていますが、三點リーダを使ったほうが日本語として適切。
- ・ p.12 の女の子がかわいいと思います。  
でも小麦ちゃんのほうがかわいいと思います。

#### ◆ p.14-p.22 完全 AV マニュアルⅢ

- ・「ははあ、えっちなビデオと思わせておいて、オーディオの話題なんですね、わかります。」と思ってみたら、えっちなビデオの記事だったでゴザル。  
内容は頑張ってるけど、学類誌として出すにはどうなんでしょう。どうなんでしょう。
- ・ p.17 AM 9 ; 0 0 → AM 9 : 0 0  
時刻の表示にはセミコロン(;)ではなくコロン(:)を使う。
- ・ p.20 車所持者と保有者で表記揺れが発生している。

---

\*1 ください。

\*2 以後、ページ数の表記に関しては「そしあの法則」適用後のページ数で表記します。例えば、本来は 33 ページだけど 31 ページと書いてある箇所は、p.31 と表記します。

## お詫びと訂正

- ・ p.20 歯移転→配点  
とても分かりやすい誤変換。

### ◆ p23-p.26 ゆる突！おみやげ日本一周

- ・ p.24 茨城県の文の冒頭のインデントが半角 1 文字分余計にある。
- ・ p.25 山梨県の文の先頭 5 行文だけ半角 1 文字分左に寄っている。

### ◆ p.27-p.28 絆～ライブイベントを見るオタクたちの友情～

- 全体的に改行が少ないので、もうすこし読みやすく改行していただきたい。
- ・ p.27 8／1 放送予定  
これが発行された 10 月 10 日時点で 8 月 1 日は終わっているので、「予定」とつくのはおかしい。
  - ・ p.27 客席数 3 6 0 1  
下の脚注には客席数 3600 とある。何故 1 席増えたのだろうか？ 補助椅子だろうか？ そしあの法則だろうか？
  - ・ p.27 の脚注 収容人数 3800 人。客席数 3600。  
表記揺れ。収容人数には単位がついているのに、客席数には単位がついていない。  
そもそも、全角の数字と半角の数字が同一記事内で存在する事自体が表記揺れ。<sup>\*3</sup>

### ◆ p.35-p.38 若くない大学生による「青春」考察

- ・ p.39 世間における誰か別の人の送った→世間における誰か別の人が送った  
「の」が連続して読みにくい。  
また、直後の「あるいは誰かが」にあわせて「が」にするべき。

### ◆ p.39-p.42 ひとりでラーメンできるもん！

- ・ 目次だと「ラーメン一人でできるもんっ！」となっている。どちらが正しいか分からぬので、ここでは記事のタイトルにあわせた。
- ・ 三点リーダ、中黒が混在している。  
中黒の数が 2 つだったり 3 つだったり、全角だったり半角だったりと、表記揺れの嵐である。

### ◆ p.43-p.46 食べ比べ企画 なっとう

- ・ p.46 インデントがなぜか 3 段になっている。

### ◆ p.47-p.50 100 点取るまで諦めま '10 19 歳夏、カラオケ夢物語

- ・ 目次だと「カラオケ 100 点取るまで帰れま 1 0」となっている。
- ・ 全文を通して、インデントが無い。

### ◆ p.51-p.54 ほーむめいど おぶ レイショクべんとう

- ・ 全文を通して、インデントがあつたり無かつたりする。
- ・ p.52 所有時間→所要時間
- ・ p.53 整形→成形

---

\*3 情報科学類としては、全角の数字自体が気持ち悪い存在ですが、ここでは触れないでおきます。

もしもアンパンマンの顔の型を使ったのならば「成型」。

- ・p.53 アンパンマンぽてと→アンパンマンポテト

**◆ p.55-p.58 ペーパードライバーへの道**

- ・p.58 「帰郷して、のち、免許」の本文1行目が半角1文字分ほど右にずれている。  
さらに、インデントが無い。

**◆ p.59-p.65 社会の窓**

- ・p.60 730人を超える参加をいただきました→730人を超える方々にご参加いただきました  
不自然な日本語。

**◆ pp.66-67 緊急増刊 年間ギャロッパ**

- ・P.66 参加するを→参加する者を

**◆ pp.68-69 アンケート集計**

- ・P.69 WORDさんとの絡みはできているので

**えっ、何それ怖い。**

**■ WORD編集部の場合**

WORD編集部では、このようなだらしねえ記事を生み出さないために、編集部員全員で記事の推敲を行っています。誤字・脱字はもちろんの事、漢字表記とひらがな表記が混ざってないか、重複表現が無いか、見づらいレイアウトをどう改善するか、ソースコードにバグは無いかなど、様々な歪みねえ修正を行っています。

さらに発行の許可をいただく際に、情報科学類長である山口喜教先生<sup>\*4</sup>にチェックをしていた  
だき、☆ KENZEN ☆な記事だけを掲載しています。

**■というわけで**

**これからも、清く健全な情報科学類誌 WORD をよろしくお願いします。**

---

\*4 単位ください！

# この一歩から、未来が始まる。～LTE レビュー～

文 編集部 Genyakun

### はじめに

今までの記事を見てくれている人も、そうでない人もここにちは。

今回は今までの記事<sup>\*1</sup>とはちょっと違った方面である、モバイル通信について取り上げたいと思います。

### モバイル通信とは

モバイル通信とは、外出先などの移動環境で行う通信のことです。このような環境において情報のやりとりを行うためには、通信をするためのデバイスが不可欠です。この記事では通信を行うためのデバイス、特に LTE 対応端末について、詳しく掘り下げながらレビューをしていきたいと思います。

### モバイル通信の歴史

先ほど「LTE」という言葉が出ましたが、LTEという物を説明する前にまず今までのモバイル通信の歴史を軽くさかのぼってみましょう<sup>\*2</sup>。

#### PDC（第2世代）

アナログ式の携帯電話（第1世代）からデジタル式に変更された際に採用された規格のこととで、NTT docomo（以下 docomo）で言うと mova 等に採用されている日本ではかなりメジャーな規格となっています。この規格では最大 28.7kbps の通信が可能です。

#### W-CDMA（第3世代）

docomo の FOMA や、SoftBank の 3G サービスが使用している規格のこととで、別名 UMTSとも呼ばれています。また、この規格は現在のデファクトスタンダード的な存在であり、最大で 384kbps の通信が可能です。

#### HSPA（第3.5世代）

HSPA とは、W-CDMA を拡張してより高速な通信を行うために策定された規格で、ダウンロード側を高速化する HSDPA とアップロード側を高速化する HSUPA で構成されています。HSDPA では下りだけが 7.2Mbps になり、HSDPA に HSUPA を組み合わせた場合には下り 7.2Mbps、上り 5.7Mbps の通信が可能です。

ちなみに、HSPA にはこれ以外にも HSPA+などの複数の内部規格が存在しますが、ここでは複雑化を避けるため、省略します。

### LTE とは

さて、このようなモバイル通信の歴史に軽く触れた上で、新しい LTE という規格についてご紹介しましょう。

LTE とは Long Term Evolution の略称で、先ほどの世代でいうと第3.9世代目となります。なぜここまで細かく第3世代が分かれているかというと、先述の3.5世代までは「同じ規格」をベースとして拡張を行ってきたため、世代の刻みが細かくなっているのです。ところが、LTE はこれまで

---

\*1 ミクさんかわいいよミクさん。

\*2 cdmaOne(CDMA2000)については、紙面の都合上説明を省かせていただきます。

での第3世代の規格との互換性は全くなく、新しい規格と捉えても良いものです。

これだけ聞くと、じやあLTEは第4世代でいいじゃないかと思われる方もいらっしゃると思うますが、そもそもLTEは次世代規格である第4世代への移行をスムーズにするために開発された規格であるため、和訳すると「長期的進化」となるような名前を与えられているのです<sup>3</sup>。

では実際にLTEでは何が改善されるのか、大きく分けると三つあります。

### ・通信速度

LTEでは、規格上最大300Mbpsにも達するスループットを得ることが可能です。しかし、現時点では端末の制約などがあり、日本国内での利用可能な最大速度は75Mbps(一部屋内エリア)と、37.5Mbps(屋外エリア)になっています。つまり、現時点ではW-CDMAの100倍、屋内エリアに限って言えばHSDPAの10倍で通信することが可能になります。

### ・周波数利用効率の向上

LTEは従来の第3.5世代と比較すると、周波数の利用効率が約3倍まで向上しています。周波数利用効率が向上することは、同じ周波数帯で、より多くの人が通信が可能になるため、キャリア側からするとユーザ単位の通信コストが減るというメリットがあります。

### ・低遅延

遅延とは、相手のマシンに対して要求を送信し、その結果が返ってくるのに掛かる時間のことです。従来の第3.5世代では実測<sup>4</sup>で300-500msecとかなりの遅延が発生していました。そこで、LTEは遅延を限りなく抑えるという方針で設計され、実測でも遅延が大体100msec前後に抑えられています。

このわずかな遅延がインターネットにどう影響するか理解しにくいと思いますが、ブラウザでリンクをクリックしてから、レスポンスが返ってくるまでの間が短くなるという風に考えていただいて問題ありません。また、ゲーマーな方々は「ラグる」などの単語に見覚え（聞き覚え）があると思われますが、まさにその状況です。

LTEでは、従来規格に対して上記のような改良が施されており、より快適にネットワークを使用することが可能になりました。

そして国内ではdocomoが、このLTEを採用したサービス「Xi(クロッシャイ)」を2010年12月24日に開始しました。筆者はサービスイン当日に購入してきたので、Xi(LTE)の詳細なレポートを次のページからどうぞ。

\*3 ただし、アメリカの一部通信キャリアにおいては、LTEのことを第4世代と表記している場合がありますが、これも間違いではありません。

\*4 www.google.co.jpまでpingを送った際の応答時間を計測しました。

### 料金プラン

docomo の Xi で採用されている料金プランは、定額プランではありません。二年契約が前提のプランでスタートは 1,000 円、通信量が 5GB までは 6,510 円の料金プランです。しかし、5GB を超えると以後 **2GBごとに 2,625 円の追加課金**が発生します。ただし、現時点では docomo 側も利用者の動向を見たいのか、2012 年 4 月 30 日までは 4,935 円で使い放題となっていますが、今後の不安が残る料金プラン設定であることは間違いないでしょう。

### 契約、開封

というわけで料金プランの設定に若干悩みましたが、筆者はヨドバシ Akiba にてデータ通信端末の「L-02C」を契約してきました。端末価格は 0 円だった上に 100 円で Netbook が一台ついてきました。これなら PC を持っていない方でも安心して購入できます。

さて、開封してみましょう。開封すると本体と、本体を入れるケース、USB 延長ケーブルにワンセグチューナーのようなクリップが付属していました。ちなみに、SIM カードは赤色でした。



### セットアップ

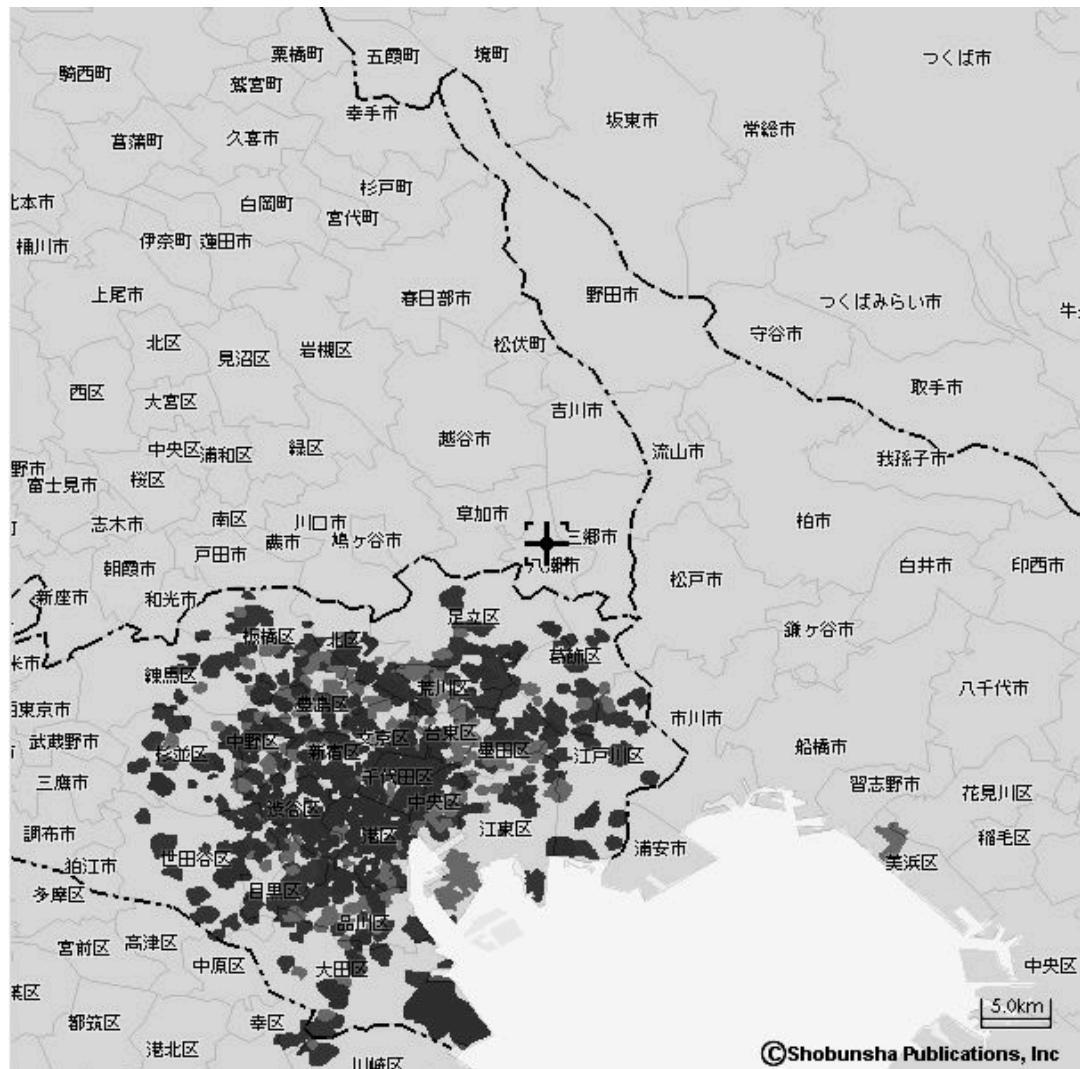
早速コンピュータに接続してみましょう。意外と幅が大きく、両隣にある USB や HDMI コネクタなどが使用できなくなる覚悟が必要です。



コンピュータに接続すると、どうやら内部にインストーラが格納されているようで、自動でドライバと接続ソフトがインストールされました。接続ソフトの詳細については後述しますが、Flash で出来ているようです。どうやって OS の API にアクセスしているんでしょう……。

## エリア

接続ツールのインストールも終わったので、早速 LTE で接続したいと思いますが、まずはその前に docomo の公式サイトで接続できるエリアを調べましょう。下の図は Xi のエリア図で、黒いところが今後のエリア拡張の予定も含むエリアです<sup>5</sup>。



——現時点では、つくば市ではLTEが使えず、実質使えるのは山手線沿線ということで、筆者としては想定の範囲内です。しかし、現時点ではLTEだけを目当てに買うのは控えた方が良いかと思われます。

ちなみに今、だけと書いたのは、L-02C は HSPA にも対応していて、LTE 圏外のエリアでは通常の FOMA エリアの HSDPA や HSUPA を使うことが出来るからです。そのため、とりあえず契約さえすれば（LTE の恩恵は別として）全国どこでも使えるということです。

\*5 エリア図については <http://www.nttdocomo.co.jp/support/area/kanto/xi/index.html> から閲覧することができます。

## LTE レビュー

### 接続

さて、エリアも調べたところで早速接続してみましょう。接続は専用のソフトで行いますが、専用のソフトは端末を PC に接続するだけでインストーラが起動するのすぐにセットアップ出来るような仕組みになっています。また、ドライバだけをインストールして、Windows 標準のダイアルアップ機能を使用することも出来るようになります<sup>6</sup>。



LTE エリアで接続すると、左上のインジケーターが左図のように LTE になり、LTE エリア外で接続すると左上のインジケーターが右図のように HSDPA (HSUPA) 等になります。また、接続先が標準では mopera U という ISP のアクセスポイントである mopera.net になっていますが、Xi では ISP も別契約のため、他の ISP が対応すればそちらに接続することが可能です。

### 速度測定

実際に LTE エリアで接続した後にスピードテストをしてみました。測定条件は次の通りです。

- ・マシン : Let's note CF-N9 (Core i5 540-M, RAM 4GB)
- ・OS/ブラウザ : Windows 7 Professional / Google Chrome 8.0.552.224
- ・通信モジュール : L-02C
- ・通信環境 : すべて LTE 環境

上記の環境を用いて、Radish Network Speed Testing<sup>7</sup> にて 3 回測定した結果<sup>8</sup>と、その平均速度と測定した時間帯<sup>9</sup>を次に示します。

\*6 この問題は 2 月に予定されているアップデートで改善されるとのこと。

\*7 <http://netspeed.studio-radish.com/index.html>

\*8 速度については小数点第二位まで記載し、残りは切り捨てています。

\*9 測定結果には基地局の混雑具合などの都合も含まれるため。

秋葉原 芳林公園 (15 時頃)

	Down (Mbps)	Up (kbps)
1st	4.45	116.30
2nd	4.82	53.06
3rd	6.31	80.82
平均	<b>5.19</b>	<b>83.39</b>

UDX 2F AKIBA SQUARE 付近の屋外 (15 時頃)

	Down (Mbps)	Up (kbps)
1st	3.20	283.70
2nd	3.02	254.30
3rd	3.62	205.80
平均	<b>3.28</b>	<b>247.93</b>

37.5Mbps 対応エリアの屋外では大体このような結果になりました。

この他にも、いくつかの地点で 1 回だけ測定をしたので参考までに掲載しておきます。

	Down (Mbps)	Up (kbps)
高田馬場周辺 (15 時頃)	5.91	98.91
TX 北千住駅ホーム (20 時頃)	5.99	312.5
秋葉原 Sofmap 買い取りセンター (屋内 / 15 時頃)	4.12	143.6
東武伊勢崎線北千住駅付近 (移動中 / 16 時頃) <sup>*10</sup>	8.35	260

70Mbps 対応エリアにて測定した結果を示します。現時点では、70Mbps 対応エリアは公式サイト<sup>\*11</sup>によると羽田空港周辺と羽田空港へ接続される駅の「周辺」<sup>\*12</sup>だけとなっています。

羽田空港の P4 駐車場 4F (12 時頃)

	Down (Mbps)	Up (kbps)
1st	8.96	608.5
2nd	7.33	731.8
3rd	5.82	742.3
平均	<b>7.37</b>	<b>694.2</b>

羽田空港第 2 ビル 3F (12 時頃)

	Down (Mbps)	Up (kbps)
1st	7.78	1095
2nd	8.76	1082
3rd	10.41	1049
平均	<b>8.98</b>	<b>1075.3</b>

羽田空港 国際線ターミナル 3F (13 時頃)

	Down (Mbps)	Up (kbps)
1st	2.57	182.2
2nd	5.71	397.9
3rd	3.34	449.5
平均	<b>3.87</b>	<b>343.2</b>

羽田空港第 1 ビル B1F (14 時頃)

	Down (Mbps)	Up (kbps)
1st	21.15	1239
2nd	14.18	1066
3rd	14.89	1070
平均	<b>16.74</b>	<b>1125</b>

70Mbps 対応エリアでは、多くの場所において 37.5Mbps 対応エリアを上回る成績であることが確認できます。

\*10 この場所だけ、www.speedtest.net で計測しました。

\*11 <http://www.nttdocomo.co.jp/support/area/kanto/xi/spot/index.html>

\*12 駅構内は HSDPA 接続になりました。なんという罫。

### まとめ

ここまで駆け足で LTE サービスの紹介をしてきましたが、実際に使ってみると LTE エリアでのレスポンスの良さに感動できます。数値的に見ると ping<sup>\*13</sup> が最小 44ms、平均 95ms とそこそこ高速なのがわかり、Web サイトの閲覧やリモートデスクトップなどのネットワークの応答速度がソフトの使い心地に直結するようなケースにおいて大きな恩恵を得られます。

また、先ほどの速度測定の結果にもありました、通信速度が 3Mbps というケースがあったものの、測定した限りでは 37.5Mbps エリアにおいては 3Mbps ~ 8Mbps、70Mbps 対応エリアでは 3 ~ 16Mbps、ピークでは 21Mbps と、理論値に近い値とは限らないものの、実使用においてはそれほど不満のない速度が出ます。ただし、上り方向の速度が著しく出ていないのには疑問を感じざるを得ないところで、これは端末上の問題なのか、Skype や Ustream などのストリーミング系の帯域を制限しているのか、あるいは ISP の設備の問題なのかは定かではありませんが、上り方向に関してはあまり期待をしない方が良いと思われます。

しかし、これらのメリットがある一方で、現時点ではこれらの恩恵を受けることの出来る LTE のエリアが関東では山手線沿線のみで、そのエリア内であるにもかかわらず屋内では通信が不安定になってしまふ場所があつたりと、はっきり言ってエリアは充実していません。しかし、まだサービスインをしてそれほど日も経ってないので、エリアについてはあまり批判しにくいところではあります。ただ、このような LTE エリア外においても HSDPA 等の 3G 回線で接続できるというところは評価が出来ます。この時、LTE から 3G 回線へのローミングはセッションが切断されずに自動で行われます<sup>\*14</sup> が、逆の 3G 回線から LTE へのローミングは手動で再接続する必要があり、ここに改善の余地があります。

ちなみに、これは LTE 自体とはそれほど関係がありませんが今回購入した L-02C のソフトウェアの完成度が低く、デバイスを接続した後に専用のランチャを実行しないと端末が認識されなかつたり、セッションが確立されているにもよるに見えて通信が出来ない状態に陥ったり、LTE が不安定なエリアにおいて 3G 回線にならなかつたり、そもそもソフトウェアが Flash で作られていてちょっともっさり気味なのが残念なところです。しかも、現時点ではこの端末が LTE を使う上での唯一の選択肢であるため、早急な別メーカーによる端末や WiFi ルーター型の端末の発売が望まれます。

なお、個人的な感想としては、現状では速度も、LTE のエリアも WiMAX に負けている状態なので、まだ一般ユーザにお勧め出来るまでにはなっていない、発展途上のサービスという印象を強く受けました。とはいって、このようなサービスが増え、UQ WiMAX やイー・モバイル等の他社との競争を促すには良いサービスではあると思っています。

---

\*13 ping の宛先は先ほどと同じく [www.google.co.jp](http://www.google.co.jp) です。

\*14 つまり Ustream やニコ生を見ながら LTE → HSPA エリアの移動が出来ます。

おまけ

さて、ここからはおまけとして Wi-Fi ルーターである「Portable WiFi」、そのフレッツ版である「光ポータブル」に、Xi の SIM を差し込むとどうなるか検証してみたいと思います。

そもそも、Xi の SIM カードは既存の FOMA 端末では使えないということがまことしやかに噂されていたので検証してみました。

**とても重要な注意：メーカが想定していない方法での接続のため、定額通信の対象にならずパケ死しても筆者は責任を負いかねます。あらかじめご了承ください。**

使用した機材は、PWR-100D(光ポータブル SIM ロック版)です。手順としては、まず、L-02C の SIM カードを取り外して、PWR-100D に差し込みます。そして電源を入れて、光ポータブルの AP に接続します。その後管理画面のステータスを見てみると……！

製品名	PWR-100D Ver.1.80 (R2.03/B101102_001WWWE_4735)	
機器名		
外部接続	クレードル接続中(LANポートとして動作)	
電源	55%(AC/USB充電中)	
Internet接続モード	未接続	
Internet(無線LAN)	接続状態 プロファイル名	待機中
	MACアドレス	00:24:██████████
Internet(3G/HSPA)	接続状態 ネットワーク 電話番号 電波状態	無効 未接続 090-████████ 圏外
	IMEI	████████████████████████████████
LAN	IPアドレス サブネットマスク DHCPサーバー MACアドレス	172.20.0.1 255.255.255.0 有効 00:24:██████████
無線(802.11g)	無線状態 SSID 認証方式 暗号化 ANY接続 プライバシーセッセーテー 無線チャンネル 接続端末数	制限なし ████████████████████████████████ WPA/WPA2 mixedmode - PSK TKIP/AES mixedmode 許可する 使用しない 1 チャンネル(自動設定(1-11ch)) 1 台

この通り、無事に認識されていることが確認できました。筆者はチキンなので接続はしませんでしたが、この調子だと普通に接続が出来そうな感じです。

## use Perl::Object 0;

```
use Perl::Object 0;
```

文 編集部 mitty

### お詫びと訂正<sup>\*1</sup>

前回の記事(16号)の「目次」では、『次は第1章の「useによる既存モジュールの活用とpackage宣言」から始める』としていましたが、多角的に検討した結果、まずは初步的な導入部分を用意して、第1章は年度を改めてから始めた方が良かろう……という結論に至りました。従いまして今回は第0章と称して、「Perlのデータ構造およびCPANからのモジュール導入」を扱います。

なお、掲載されているソースコードは、以下の処理系で確認しております。

- ActivePerl 5.12.2 build 1203 x86 および x64 on Windows 7 x64
- perl 5.10.1-8ubuntu2 x64 on Ubuntu 10.04.1 LTS (Lucid Lynx) x64

### 予定

#### **第0章 Perlのデータ構造およびCPANからのモジュール導入(今回)**

第1章 useによる既存モジュールの活用とpackage宣言

第2章 Perlでのクラス・インスタンス・メソッド

第3章 既存モジュールの拡張・継承

第4章 POE: Perl Object Environmentの紹介

第5章 POEを使ったマルチタスキング

第6章 POEによるイベントドリブン生活

第7章 並列クローラの作成

### Perlにおける変数の型

Perlで書かれたコードに頻繁に出てくる「\$」「@」「%」などですが、これは *Sigil<sup>\*2</sup>* と呼ばれていて、変数の「型」を大まかに表しています。

「\$」は「スカラー変数」、「@」は「配列変数」、「%」は「連想配列変数(ハッシュ)」と呼ばれ、右の「ソースコード1」を見て貰うと分かるかと思いますが、スカラー変数は値を一つ、配列変数は値を複数持ります。これ以外にもファイルやディレクトリを扱う際に用いる「ハンドル」や「サブルーチン」であることを表す「&」、Sigilの親玉である「型グローブ」を示す「\*」などがありますが、追々触れていくことにしましょう。

### **ソースコード1**

#### use0-01.pl

```
1:  #! /usr/bin/perl -w
2:
3:  use strict;
4:  use warnings;
5:  use feature qw(say);
6:
7:  my $hoge = 0;
8:  my @hoge = (0, 1, 2, 3, 4, 5, 6, 7);
9:  my %hoge = (
10:    zero => 0,
11:    one  => 1,
12:    two   => 2,
13:  );
14: say $hoge;
15: say join(", ", @hoge);
16: say "0 != 1" if ($hoge != $hoge[1]);
17: foreach my $key (keys %hoge) {
18:   say "$key => $hoge{$key}";
19: }
```

\*1 お詫びと訂正：某ふあい氏の記事とはたぶんきっとおそらく関係がありません。

\*2 Sigil：元々は魔術などの分野で使われる「紋章」といった意味を持つようです。カタカナだと「シジル」になるらしい。なお、「プログラミング Perl vol.1」では"funny character"と書かれているが、そう呼ばれているのを今知ったくらいに使われているところを見ない……。

「ソースコード 1 の実行結果」で示されるように、「\$hoge」と「\$hoge[1]」は違う変数です。そして「\$hoge{one}」もまた違う変数(の要素)を指しています。

なお、ここまでで疑問に思われた方も居るかと思いますが、Perl5までのPerlにはいわゆる「int」「string」のような、データそのものの種類を表す型はありません。

『例1) 「==」と「eq」の違い』を見て下さい。[Ctrl+D]までの太字の部分が入力したコード、[Ctrl+D]より下が出力結果です。

出力結果から分かるように、「==」演算子はオペランドを数値として比較し、「eq」演算子は文字列として比較します。文字列を数値として評価すると「0」として扱われるため、「例1」のような結果となります。

変数に型が無い、ということはその分記述に曖昧さが生じますが、むしろ慣れるとこの自由度の高さからストレス無く書けるようになるでしょう。

なお、以降のソースコード中で出てくる「\$\_」や「@\_」などの、Perl特有の特殊変数については、前回の記事<sup>\*3</sup>で一部触れているので、説明を省略している箇所があります。

### ソースコード1の実行結果

```
$ ./use0-01.pl
0
0, 1, 2, 3, 4, 5, 6, 7
0 != 1
one => 1
zero => 0
two => 2
```

### 例1) 「==」と「eq」の違い

```
$ perl -w
use feature qw(say);
say ((0 == 0) ? "true" : "false");
say ((0 == 0.0) ? "true" : "false");
say ((0 == "0.0") ? "true" : "false");
say ((0 eq 0) ? "true" : "false");
say ((0 eq "0.0") ? "true" : "false");
[Ctrl+D]
true
true
true
true
false
```

## スカラーコンテキストとリストコンテキスト

Perlにおける変数の型として、「スカラー変数」と「配列変数<sup>\*4</sup>」があることは既に述べましたが、Perlにおいて「スカラー」および「リスト」の2単語は変数の型にとどまらず、「コンテキスト」と言われる文字通りコード中で式がどのように評価されるか、という意味でも使われます。

\*3 前回の記事：[http://lab.mitty.jp/word/use\\_Perl\\_Object/use\\_Perl\\_Object.pdf](http://lab.mitty.jp/word/use_Perl_Object/use_Perl_Object.pdf) あるいは、WORD16号「侵略されたいでゲソ号」をご覧下さい。

\*4 配列変数：より正確には、値が複数代入できる変数を、単純な配列変数として扱うか連想配列変数として扱うかによって、取り出される値が違うだけ過ぎません。

```
%hoge = (zero => 0, one => 1, two => 2); for (%hoge) { print $_, ", " };
# => one, 1, zero, 0, two, 2,
```

「%hoge」も上記で示されるように単純な配列として扱えます。ただし、「@hoge」とは異なり(定義時や要素追加などの)順番は保存されません。

## use Perl::Object 0;

「ソースコード 2」の 9 行目を見て下さい。配列変数である「@hoge」を、スカラー変数である「\$fuga」へ代入する場合、「@hoge」は「スカラーコンテキスト」で評価されます。配列変数をスカラーコンテキストで評価した場合、得られる値はリストの「要素数」となり、この場合「\$fuga」には「8」が代入されます。

一方、10 行目では「@hoge」は「リストコンテキスト」で評価され、リストが返されます。

では、次の「例 2) スカラー変数とリストコンテキスト」はどうでしょうか。

「\$hoge」自身はスカラー変数ですが、「()」で囲むことによってリストコンテキストとして扱われ、「@hoge」の要素の一番目「0」が代入されます。

「コンテキスト」には他に「ブール値コンテキスト」などがあり、これは真偽値として評価される場合が該当します。「ブール値コンテキスト」は「スカラーコンテキスト」の一種なので、変数に以下で述べる特定の値以外が含まれている限り、真となります。

「ソースコード 3」において「@hoge」から「pop」関数によって一つずつ要素を取り出されますが、要素がある限り while ループからは抜け出さないことになります。

Perl5において「真」とならない、つまり「偽」として扱われるデータは以下の通りです。

- 0
- 空文字列
- 空リスト
- undef (未定義値)

空ではない任意の文字列は、数値として評価した場合には「0」と評価されるため、「偽」となります。

注意して欲しいのですが、「例 3) スカラー変数とリスト変数の真偽値」で示されるとおり、ブール値コンテキストで評価すると「\$hoge = 0」は「偽」となりますが、「@hoge = (0)」は要素が一つあるため「真」となります。

ソースコード 2

### use0-02.pl

```
1:  #! /usr/bin/perl -w
2:
3:  use strict;
4:  use warnings;
5:  use feature qw(say);
6:
7:  my @hoge = (0, 1, 2, 3, 4, 5, 6, 7);
8:
9:  my $fuga = @hoge;
10: my @fuga = @hoge;
11: say join(", ", $fuga);
12: say join(", ", @fuga);
```

ソースコード 2 の実行結果

```
$ ./use0-02.pl
8
0, 1, 2, 3, 4, 5, 6, 7
```

### 例 2) スカラー変数とリストコンテキスト

#### \$ perl -w

```
(@hoge) = @hoge = (0, 1, 2, 3, 4, 5, 6, 7); print $hoge;
[Ctrl+D]
0
```

ソースコード 3

実行結果

### use0-03.pl

```
1:  #! /usr/bin/perl -w
2:
3:  use strict;
4:  use warnings;
5:  use feature qw(say);
6:
7:  my @hoge = (0, 1, 2, 3, 4, 5, 6, 7);
8:
9:  while (@hoge) {
10:    say pop @hoge;
11: }
```

### \$ ./use0-03.pl

7
6
5
4
3
2
1
0

## 例3) スカラー変数とリスト変数の真偽値

```
$ perl -w
@hoge = ($hoge = 0); print "true" if @hoge;
[Ctrl+D]
true
```

リファレンス

変数を受け取り値を 2 倍にして返す関数を考えます。受け取る変数がスカラー変数か配列変数か分からないとすると、以下の「ソースコード 4」および「ソースコード 5」のように分けて書くことがまず考えられます。

## ソースコード4

```
use0-04.pl
1:  #! /usr/bin/perl -w
2:
3:  use strict;
4:  use warnings;
5:  use feature qw(say);
6:
7:  my $hoge = 1;
8:
9:  say scalardouble($hoge);
10:
11: sub scalardouble {
12:     my $scalar = shift;
13:
14:     $scalar *= 2;
15:     return $scalar;
16: }
```

## ソースコード4の実行結果

```
$ ./use0-04.pl
2
```

## ソースコード5

```
use0-05.pl
1:  #! /usr/bin/perl -w
2:
3:  use strict;
4:  use warnings;
5:  use feature qw(say);
6:
7:  my @hoge = (0, 1, 2, 3);
8:
9:  say join(
10:    ", ",
11:    listdouble(@hoge)
12: );
13:
14: sub listdouble {
15:     my @list = @_;
16:
17:     return map {
18:         $_ * 2
19:     } @_;
20: }
```

## ソースコード5の実行結果

しかし、扱いたいデータが常に、スカラー変数あるいは配列変数のどちらかであるとは限りません。

ここで、「ソースコード 6」で示されるように「¥(円記号あるいはバックスラッシュ)」演算子を用いて、引数となる変数の「リファレンス」を渡すようになると、スカラー変数か配列変数かで関数を分ける必要が無くなります。

## use Perl::Object 0;

### 例4) 2次元配列(失敗)

```
$ perl -w
use feature qw(say);
@hoge = ( (1, 2), (3, 4) );
say "no exist" unless exists $hoge[1][1];
say $hoge[3];
[CtrI+D]
no exist
4
```

### 例5) 2次元配列

```
$ perl -w
@hoge = ( [1, 2], [3, 4] );
print $hoge[1][1];
[CtrI+D]
4
```

また、いわゆる「n 次元配列」のような変数を扱いたい場合を考えてみましょう。単純に考えると「例4) 2次元配列(失敗)」のようになりますが、これは単純な一次元配列と同じになってしまいます。

実際に「\$hoge[1][1]」に値を代入するには「例5) 2次元配列」のように書く必要があります。

「()」だった部分が「[]」に変わっていますが、これにより「『配列のリファレンス』を要素を持つ配列」が「@hoge」に代入されているのです。

さて、ここで「『ソースコード 6』では『\$ [...]』とか『@ [...]』、あるいは『\$\$ret』といった変な Sigil があるのに、リファレンスが含まれるはずの『例5)』ではそれが見あたらない」と気づいた方は鋭いです。

実は、「\$hoge[1][1]」は「\$hoge[1]->[1]」の省略形で、リファレンスの中身を取り出す「デリファレンス」演算子である「->」が隠れているのです。

### ソースコード6

```
use0-06.pl
1:  #! /usr/bin/perl -w
2:
3:  use strict;
4:  use warnings;
5:  use feature qw(say);
6:
7:  my $hoge = 1;
8:  my @hoge = (0, 1, 2, 3);
9:
10: my $ret = double($hoge);
11: say $$ret;
12: my @ret = double(@hoge);
13: say join(
14:     ", ",
15:     @{$ret[0]},
16: );
17:
18: sub double {
19:     my $reference = shift;
20:
21:     my $type = ref $reference;
22:     if ($type eq "SCALAR") {
23:         ${$reference} *= 2;
24:     }
25:     elsif ($type eq "ARRAY") {
26:         @{$$reference} =
27:             map { $_ * 2 }
28:                 @{$$reference};
29:     }
30:     else {
31:         warn "something wrong";
32:     }
33:     return $reference;
34: }
```

### ソースコード6の実行結果

```
$ ./use0-06.pl
2
0, 2, 4, 6
```

「例 6) 2 次元配列の構造」を見て頂くと、「@hoge」の構造が理解頂けるかと思います。

つまり、「\$hoge[1][1]」というコードは、ARRAY のリファレンスである「\$hoge[1]」を「->」演算子でデリファレンスし、得られた配列の 2 番目の要素「\$hoge[1]->[1]」にアクセスしているのです。

「ソースコード 6」では「->」演算子ではなく、「\${スカラーリファレンス}」や「@{配列リファレンス}」といった形でアクセスしているわけです。

なお、文法的に曖昧さが無い場合は「\${...}」や「@{...}」の「{」と「}」は省略することが可能で、「\$\$ret」のように書けます。

#### 例6) 2次元配列の構造

```
$ perl -de 0
@hoge = ( [1, 2], [3, 4] );
print join(", ", @hoge);
[Ctr+D]
ARRAY(0x11efdf0), ARRAY(0x129dce0)
```

#### リファレンスによる複雑なデータ構造

「[]」による配列のリファレンス以外に、「{}」によるハッシュのリファレンス、「sub { ... }」によるコードリファレンスなどがあります。これによって階層化された複雑なデータ構造が表現できます。

#### 例7) Twitter用botの設定ファイルをロードした後、Data::Dumperを通して出力したもの

```
{
  'hashtag' => [
    '#perlbot',
    '#bot'
  ],
  'allow' => {
    'screen_name' => [
      'twitter'
    ],
  },
  'mail' => {
    'pickup' => [
      'search',
      'mention'
    ],
    'to' => [
      'info@example.com',
    ],
    'subject' => 'new tweets for retweetbot are found',
    'from' => 'retweetbot@twit.example.jp (Cron Daemon)',
    'server' => 'localhost',
    'contenttype' => 'text/plain; charset="ISO-2022-JP"'
  },
}
```

「例 7)」で示されるデータ構造が「\$conf」に代入されている場合、「@{ \$conf->[mail] {to} }」にアクセスすると、「(info@example.com)」という配列が得られます。

## use Perl::Object 0;

### CPAN

さて、ここまで Perl のデータ構造に関してはある程度理解して頂けたかと思います。文法なども掘り下げると色々と楽しいのですが、きりがないのでこの辺にして、Perl でコーディングするときに大変便利な CPAN について解説しようと思います。

CPAN とは「Comprehensive Perl Archive Network」の略語で、Ruby における RubyGems や PHP における PEAR に近いでしょうか。とはいってもの、あまりにも規模が大きい<sup>5</sup>ため、大抵のライブラリやちょっととしたツールであれば車輪の再発明をするまでもなく既に CPAN に登録されていることが多いです。

CPAN に登録されているモジュールは、<http://search.cpan.org/> から検索することが出来ます。たとえば Twitter に関する(と思われる)モジュールは 118 件(2011/01/21)あるようです。

### CPAN モジュールの使い方

詳しくは今後の記事で再度触れますぐ、簡単に説明すると、「`use Module::Name;`」でモジュールをロードし、「`$instance = Module::Name->new( arguments );`」で「インスタンス変数」を新しく用意して、そのインスタンス変数を通じてメソッド呼び出し、というのが基本的な使い方になります。モジュールによっては「クラスメソッド」にあたる関数が用意されていることもあります。

独断と偏見で、よく使われるモジュールを紹介してみます。

#### • LWP::UserAgent

LibWWWPerl つまり WWW 関係の Perl ライブラリのうち、Perl をウェブクライアントとして使う際に必要となる機能をまとめたモジュールです。wget コマンドをイメージするとどう使うか把握しやすいかも知れません。

<http://search.cpan.org/> に接続を試行し、成功すれば得られたコンテンツを、失敗した場合はステータスを表示するコード「lwp-ua.pl<sup>6</sup>」を次に示します。

The screenshot shows the CPAN search interface with the word 'twitter' entered in the search bar. The results page displays 118 found modules, with the first few listed below:

- Net::Twitter**  
A perl interface to the Twitter API  
Net-Twitter-3.14002 ★★★★☆ (6 Reviews) - 02 Nov 2010 - Marc Mims
- Net::Twitter::Lite**  
A perl interface to the Twitter API  
Net-Twitter-Lite-0.10003 ★★★★★ (3 Reviews) - 27 May 2010 - Marc Mims
- Twitter::Badge**  
Perl module that displays the current Twitter information of a user  
Twitter-Badge-0.02 - 09 May 2008 - Arul John
- Dancer::Plugin::Auth::Twitter**  
Authenticate with Twitter  
Dancer-Plugin-Auth-Twitter-0.02 - 17 Jan 2011 - Alexis Sukrieh
- Net::Twitter::Role::API::REST**  
A definition of the Twitter REST API as a Moose role  
Net-Twitter-3.14002 ★★★★☆ (6 Reviews) - 02 Nov 2010 - Marc Mims
- Twitter::Shell**  
Twitter From Your Shell!  
Twitter-Shell-0.03 - 08 May 2007 - Daisuke Maki
- Catalyst::Authentication::Credential::Twitter**  
Twitter authentication for Catalyst  
Catalyst-Authentication-Credential-Twitter-0.01001 - 06 Dec 2009 - Jesse Stay
- Net::Twitter::Antispam**  
Making Twitter usable  
Net-Twitter-Antispam-0.02 - 27 Jun 2009 - James Laver

\*5 規模が大きい：「89356 Modules, 8717 Uploaders」だそうです。(2011/01/21 08:55 JST)

\*6 lwp-ua.pl : <http://search.cpan.org/perldoc?LWP::UserAgent> の「SYNOPSIS」ほぼそのままです。

**lwp-ua.pl**

```
1:  #! /usr/bin/perl -w
2:
3:  use LWP::UserAgent;
4:
5:  my $ua = LWP::UserAgent->new;
6:  $ua->timeout(10);
7:  $ua->env_proxy;
8:
9:  my $response = $ua->get('http://search.cpan.org/');
10:
11: if ($response->is_success) {
12:     print $response->decoded_content; # or whatever
13: }
14: else {
15:     die $response->status_line;
16: }
```

• Data::Dumper

「Data::Dumper 先生」と敬称で呼びたくなるほど便利なモジュールです。これを `use` しておくと、既に出てきたように「Dumper」というクラスメソッドに引数を渡すことで、データ構造を再現した形で中身を出力してくれます。`print` 文などに渡せば OK。

ウェブページをスクレイピングしたりする際に、メソッドから返ってくるオブジェクトの構造がよく分からなかったりする場合は先生の出番です。Twitter 関連のモジュールである「Net::Twitter::Lite」を使って RT ポットを作る過程で、返値を実際に出力したもののが <http://lab.mitty.jp/trac/lab/wiki/Dev/Twitter/Perl/Dumper> にありますので参考まで。

• Web::Scraper

最近その存在を知ったのでまだあまり使いこなせていませんが、非常に強力なスクレイピングモジュールです。`scraper` というコマンドラインツールも付いてきます。同じような目的では、「HTML::TreeBuilder」や「WWW::Mechanize」などといったモジュールもありますが、「Web::Scraper」はそれらに比べ少ないコード量で済む設計になっているようです。

この他にも、本当に多種多様なモジュールがあります。是非自分で探してみて下さい。

**CPAN モジュールの導入**

「LWP::UserAgent」「Data::Dumper」は CPAN からダウンロードしなくとも、perl 本体の配布パッケージと一緒にになって導入されているはずです。それ以外の「Web::Scraper」や「Net::Twitter」などは自分で導入する必要があります。

このとき、導入方法には大まかに次の三つがあります。

## use Perl::Object 0;

- CPAN 上のモジュールのページから tarball をダウンロードし、make する
- cpan コマンドにモジュール名を渡し、インストールする
- その他、OS 付属のパッケージ管理ツールを用いる

後に挙げた方法ほど容易になります。cpan コマンドに関してはウェブ上に文献がいくらでもありますし、筆者自身普段はパッケージ管理ツールを用いているので、ここではパッケージ管理ツールを用いた導入方法を簡単に説明しておきます。

### ・ Windows

Windows 用のメジャーな Perl のバイナリは「ActivePerl」という名前で配布されています。これには「ppm」という、GUI ツールが付属しています。デフォルトのインストール先は「C:\Perl\bin」です。

cpan コマンドなどもそうですが、「ppm」もモジュール間の依存関係を自動で解決して必要なモジュールを追加で導入してくれます。また、既に build 済みの形でダウンロードされるため、cpan とは違い make を行わず、(Perl プロジェクトではない) 外部のライブラリを必要とすることはありません。

その代わり、CPAN に置かれてるパッケージの全てが網羅されているわけではなく、特に GNU のライブラリなどに依存しているモジュールは無いことが多いです。

### ・ Ubuntu

aptitude または apt-get で導入することができます。パッケージ名にちょっと癖があり、たとえば「WWW::Google::Calculator」であれば「aptitude install libwww-google-calculator-perl」で導入できます。こちらは ppm よりも網羅されておらず、2000 パッケージ強しかないようです。

Perl のモジュールは依存関係が入れ子になっていることが多く、あるモジュールを導入する際に、それが依存しているモジュールを導入しようとするとさらに依存が発生して……、と予想以上にモジュールの導入に時間が掛かることがあります。このため、導入対象のモジュールそのものは ppm や aptitude などのパッケージ管理ツールには載っていない場合でも、あらかじめ依存するモジュールを導入してから対象モジュールを cpan コマンドで導入すると、比較的楽に入れることができます。

## 最後に

如何でしたでしょうか。かなり端折った部分も多く、まだまだ Perler として力不足を感じていますが、この記事が楽しい Perl 生活のきっかけにでもなれば、幸いです。

### ハイパー添削タイム(あるいは蛇足)

前回の記事に掲載しました mycat に対して、ありがとうございます。

- <http://twitter.com/uasi/status/4133434205143040>

今回の WORD に *print for <>;* って Perl のワンライナーが載ってたけど *print <>;* でもいいんじゃないかな。どっちもリストコンテキストだから *print* する前にファイル全体を読み込むはず。それとも *for <>* は最適化されんのかな。

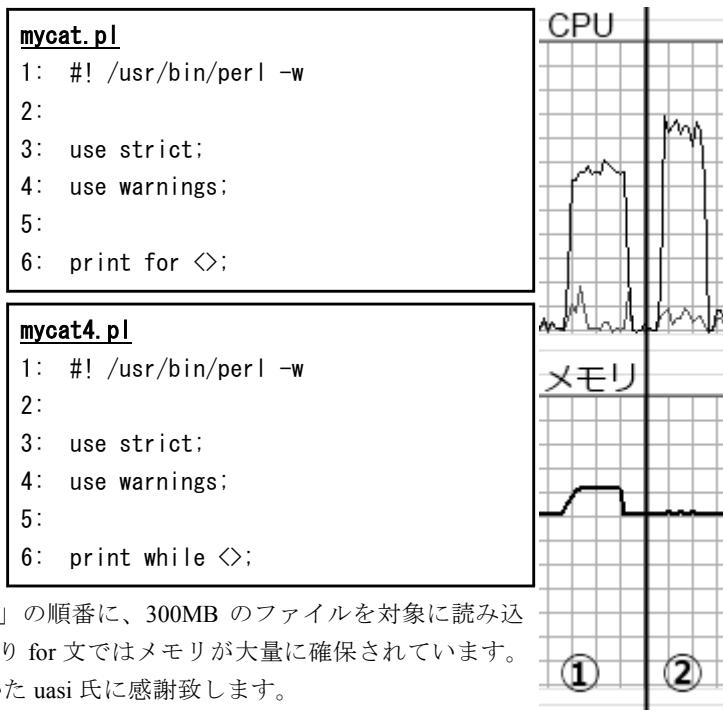
- <http://twitter.com/uasi/status/4133886183350272>

1 行ずつ読むなら *print while <>;* すべき。

while の場合山括弧演算子によって、ファイルから一行ずつ読み込まれ画面に出力されますが、*for/foreach* ではリストコンテキストで評価されるため、ファイルから一度に全て読み込んでしまいます。前回の例では問題にはなりませんでしたが、読み込み対象が巨大なファイルの場合は問題が生じます。

右端のグラフは上段が CPU 使用率、下段がメモリ使用量です。というか Windows のタスクマネージャーです。

①「mycat.pl」、②「mycat4.pl」の順番に、300MB のファイルを対象に読み込みを実行しましたが、見ての通り *for* 文ではメモリが大量に確保されています。  
ありがとうございました uasi 氏に感謝致します。



次に添削についてですが、「perl」のコマンドラインオプションに興味深い記述があります。

```
$ perl --help
-n           assume "while (<>) { ... }" loop around program
-p           assume loop like -n but print line also, like sed
```

どうやら、ループを書かなくてもループを追加してくれるオプションがあるらしい……。

結果、mycat はさらに短くなりました。

ここまで来ると、何がしたいのかコードからは完全に不明<sup>7</sup> ですが、何事もなかったかのように動作します。さすが Perl だ、なんともないぜ。

```
mycat5.pl
1: #! /usr/bin/perl -wp
```

\*7 完全に不明：コード量ゼロですし。

## use Perl::Object 0;

このコマンドラインオプションを用いることで、今回ソースコードに行番号を付けて出力するために用いた「add-line-number.pl」は、以下のように書くことが出来ます。

### add-line-number.pl

```
1: #! /usr/bin/perl -w
2:
3: print "$_.:$t";
```

Perl で書かれたコードは大変シンプルになることがお分かり頂けたかと思います。

## 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/>
- <http://www.cpan.org/>
- perldoc コマンド
- Google 先生

## 免責

本記事の内容は、引用部分や素材など、著作権その他の権利が筆者に帰属しない物、ある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<sup>\*8</sup>」か、あるいは「2-clause BSD license<sup>\*9</sup>」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。

本記事の内容は、以下の URL でも公開しております。

<http://lab.mitty.jp/word/>

内容に間違い・質問などありましたら、twitter:@mittyorz まで連絡して頂ければ幸いです。

\*8 Artistic License : <http://dev.perl.org/licenses/artistic.html>

\*9 2-clause BSD license : <http://www.freebsd.org/copyright/freebsd-license.html>

# 情報科学雑誌 WORD 読者アンケート

文・題字 編集部 ふあい

## ■あいさつ

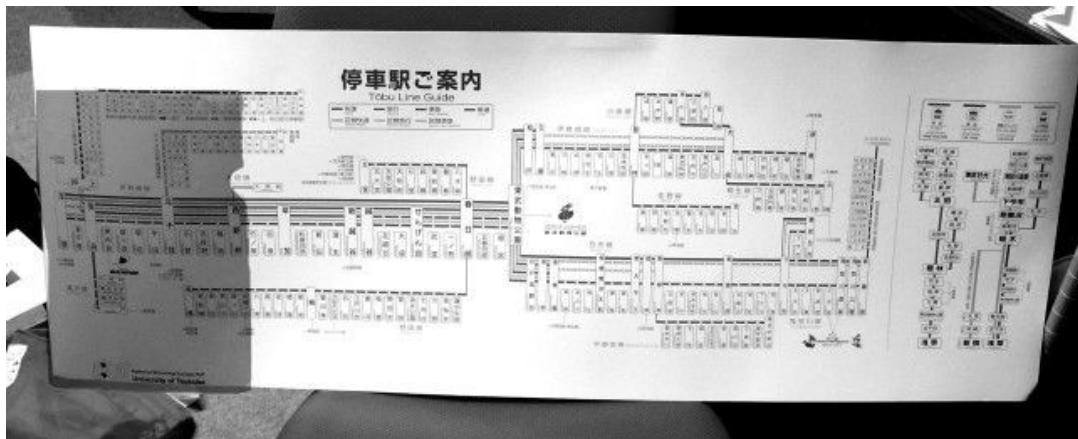
皆さん、おはこんばんちは<sup>1</sup>。前号実施した WORD 読者アンケートですが、なんと**4人**の方から回答を頂きました。アンケート回収 BOX にコンビニのレシートでも入れられるんじゃないかと期待心配したのですが、杞憂でしたね。回答をして下さった 4 人々に感謝いたします。

ちなみに粗品の COJT グッズ<sup>2</sup>は誰一人として貰いに来ませんでした。  
前号の記事ではちゃんと粗品の紹介をしなかったので、魅力が伝わっていないのかもしれません。  
そこで、アンケートの結果に移るまえに、粗品の COJT グッズの紹介をします。

## ◆こんなにすごい！ COJT 特製クリアファイル

クリアファイルは 1 人につきなんと**5枚**貰えます。色は黄色・赤・クリア・緑・青の 5 色。なんと初回のゴレンジャイよりも色数が豊富です。

気になるサイズは以下の通り。



どこの家庭にもある東武線の路線図が、少しほみだす程度の驚きの収容力！ これでプリントの管理もバッチリです。

\*1 おはこんばんちは：おはよう+こんにちは+こんばんは

\*2 COJT グッズ：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」の教室デザインアンケート用に、偉大なる徳永先生が道楽で作ったグッズ。

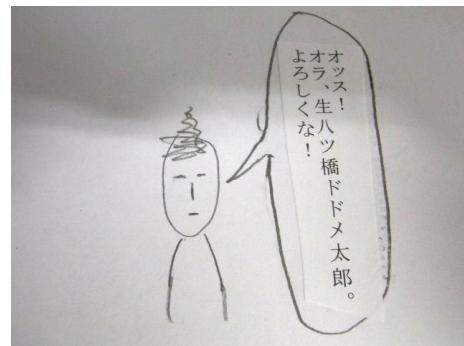
## WORD 読者アンケート

### ◆これは便利！ COJT 特製テープのり

修正テープ感覚で使える地味に便利なのりです。使い方は簡単！



①貼りたい所にのりを転写する



②貼りたい物を貼り付けて完成！

ステイックのりよりも手軽に使えるところが魅力です。紙がシワシワになることもありません。

魅力がたっぷり伝わったところで、本題のアンケート結果発表に参りましょう。

### ■アンケート結果

#### ◆ Q1:所属を教えてください。

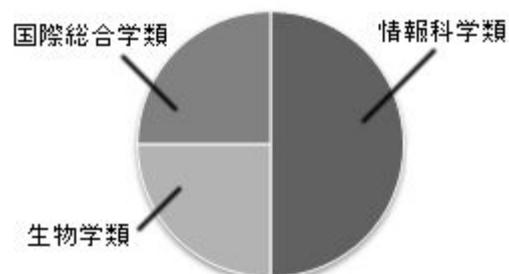
生物学類：1人

国際総合学類：1人

情報科学類：2人

非常にコアなネタが多い WORD ですが、他の学類の方々にも読んでいただいているという事が分かりました。

ありがとうございます。



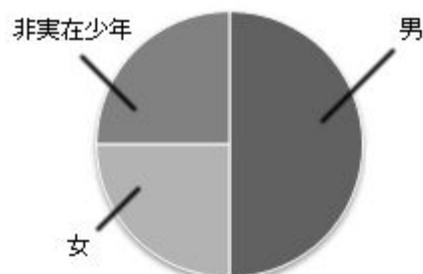
#### ◆ Q2:性別を教えてください。

男：2人

女：1人

非実在少年：1人

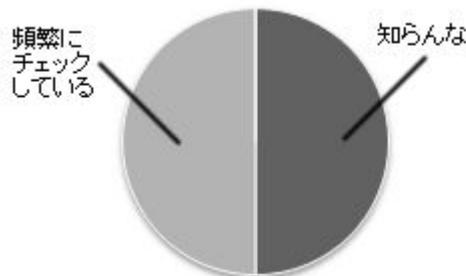
編集部は男ばかりですが、読者は性別によらず、色々な方々に楽しんでいただいているようです。



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

頻繁にチェックしている：2人

知らんな：2人



半分の方には知られていないようなので、ここで宣伝しておきます。

WORD Press では、WORD のバックナンバー や、その他どうでもいいエントリを全世界に発信しています。

WORD 新刊の配布告知などもしているので、是非チェックしてください。IPv6 にもバッチリ 対応しています！！！↓の URL にいますぐアクセス！！

# <http://www.word-ac.net/>

### ◆ Q4:16号の記事で、良かったと思う記事があれば教えてください。

#### ・表紙：1票

› 表紙みて取るの余裕でしたゲソ。

ありがとうゲソ！

#### ・電子の歌姫はアイドルの夢を見るか：1票

› DIVA はやったことなかったが、見て興味を持った。しかし、アイマスとコラボする必要が

› あったかは、アイマス民(笑)として疑問(^q^)

そんなことよりアイドルマスターゼノグラシア見ようぜ！！！！！！！！

#### ・最近のポケモン事情～育成編～：1票

› BW もってないのでググりました(ドレディアとエルフーン見るために)

› もっと萌えそうなポケモンを教えてください。

編集部随一のポケモンマスターの IX 氏に伝えておきます。

#### ・GRな日々：1票

#### ・RVMを使っていくつかの Ruby 実装を使ってみよう：1票

#### ・use Perl::Object; : 1票

#### ・革命的で魔法のようなラクダ。: 1票

#### ・ICT 中間報告会 2010 : 1票

色々な記事に対して、均等に票が入っているようでした。これからもこんな感じの誌面作りに努めて参ります。

### ◆ Q5:16号の記事で、良くなかったと思う記事があれば教えてください。

#### ・WORD 読者アンケート：1票

› とりはずしづらい

› (一番最後にするといいのでは)

わああああああああああああああああ、ごめんなさい！！1

## WORD 読者アンケート

### ◆ Q6:過去の記事に関して感想がありましたら、ご記入下さい

› バックナンバーを見ました。面白かったです。自分の学類もこんなのがあれば……。

(国際総合学類、名も無き龍 さん)

お褒めの言葉ありがとうございます。一時期は WORD とそおしゃ～るだけになった学類誌ですが、最近になって情報メディア創成学類誌 MAST が設立されたり、知識情報・図書館学類でも学類誌設立の噂があつたりと、学類誌が増える傾向にあるようです。やる気のある人が 5 ~ 6 人くらい集まれば国際総合学類誌も設立できるかもしれませんね。

› 青春 18 きっぷで青春を取り戻したい。

(情報科学類、多加枝 鉄見 さん)

青春 18 きっぷ関連は、15 号の青春 18 きっぷ記事の他、今号のウサギ島の記事も参考にしていただければ幸いです。とりあえず東海道本線を 1 往復してみることをオススメします。片道 10 時間程度で済み、電車の接続が良く、うまく予定を組めば特急車輌に乗れるので初心者にもオススメです。

### ◆ Q7:自由記述欄です。何でもいいから書いてくださいね。

くぎゅうううううううううう  
うううううううううううう  
うううううううううううう  
うううううううううううう  
うううううううううううう

でもあざちはが好きです。

(国際総合学類、名も無き龍 さん)

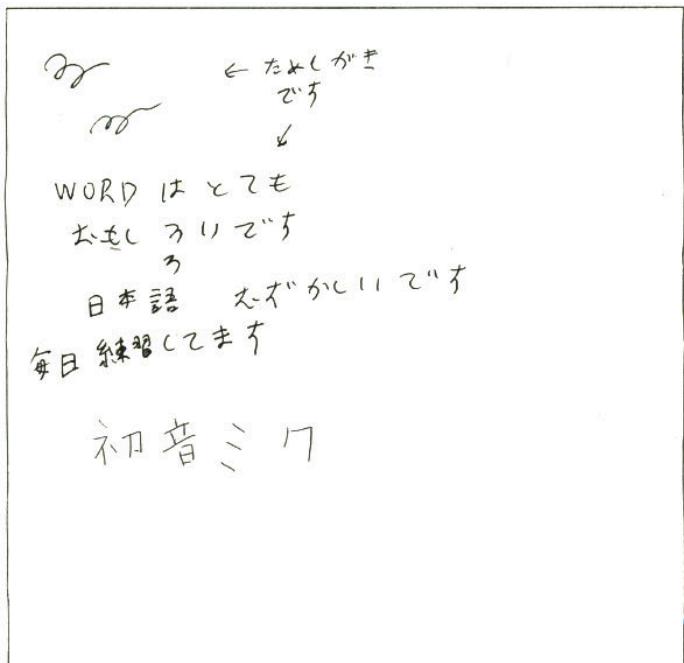
今回のアンケートの、記念すべき第 1 号回答です。ありがとうございます。

なんだよ！前半のくぎゅう関係ないじゃんかよ！！！！！！

とりあえずゼノグラシア 見ようぜ！！！！

留学生の方でしょうか？  
WORD を読んでいただき、また  
アンケートに回答していただき  
ありがとうございます。WORD  
では発行の前に念入りに推敲を  
行い、清く正しく美し  
い日本語を使うよう心がけてい  
ます。初音ミク。

ところで、初 の字が間違っ  
てますよ。



(生物学類、ジンバブエ さん)

キタコレ！！！ さす  
が情報科学類だ！！ この調子で  
ラティアスも頼む！！  
というわけで、今回のアンケート  
で唯一のイラスト付き回答  
でした。ああ、ありがたや。

#### ◆というわけで

今号もアンケートやります！  
アンケート粗品のクリアファイル  
とテープのりを希望する方は、  
WORD 編集部(3C212)まで直接お  
申し出下さい。先着 10 名分だけ  
あります。

WORD 編集部前その他、計算機  
室前などにもアンケート回収  
BOX を設置します。よろしくお  
願いします。



(情報科学類、多加枝 鉄見 さん)

# WORD編集部へのお誘い

文 編集部 ららぼ

## 新入生大募集

我々、WORD 編集部は情報科学類の公式団体であり、情報科学類誌「WORD」の発行をしています。これは筑波大学内では3つしか存在しない「学類誌」のうちの1つです。

「WORD」はコンピュータに関することのみならず、様々なことを取り上げる総合エンターテイメント雑誌です。年に数度発行され、主に第3エリア3A、3B、3C棟にて配布されます。

WORD 編集部にはこんな編集部員がいます。例えば……

- ・段ボール遣い Keiyac
- ・バイリンガルなスーパーハカー zer0day
- ・朝鮮半島に詳しい ふあい氏
- ・漆黒の堕天使 はろぺり総書記
- ・ガチでヤヴァイ子マッドサイエンティスト PJ

ほかにも多くの個性的な編集部員がいます。また、編集部員の大半は情報科学類生ですが、他学類の学生も存在しています。

「WORD」は我々が真心をこめて執筆、編集、製本しています。といっても、編集部における拘束時間は殆ど無く、週1回の編集会議、年に数度の製本作業といった程度です。なので、他のサークルとの掛け持ちは何も問題ありません。実際、他のサークルと掛け持ちしている編集部員もたくさんいます。みんな、のびのびと過ごしています。

以下の項目に当てはまる人はぜひ、学生ラウンジ隣の編集部室（3C212）へ。常に（夜も）開いているのでいつでも見学に来てください。

- ・雑誌のようなものを編集したい人
- ・議論好きな人
- ・AC入試で入学した人
- ・それ以前にACな人
- ・ネットワーク管理経験者
- ・常に近くにコンピュータがないと発狂する人
- ・コンピュータの中に恋人がいる人
- ・ゲーム（ジャンル問わず）が好きな人
- ・Web系に強い人（ただしPHP, Flashは除く）

4月、または5月に新歓を行う予定なので、こちらの参加も大いに歓迎いたします。詳細は追って連絡させていただきます。

その他質問のある方は [word@coins.tsukuba.ac.jp](mailto:word@coins.tsukuba.ac.jp)までメールして下さい。

# 編集後記

情報科学類誌



From College of Information Science

## WORD おせち特盛号

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)

2011年2月14日 初版第一刷発行

(5 1 2 部)