

# WORD

2008.5 From College of Information Science

5

特集

学類新歓後記

神々の言霊二千八

マイコンで作る  
wiiコントローラもどき

新連載

THE BINARYMASTER  
バイナリマスター

LINQ in C# 3.0

1時間で始めるFlex

オーディオベーシック?  
青年男子ダイエット

WORDでは  
古い記事の使い回しは  
しておりません。号

# ～ 目次 ～

学類新歓後記	3
StarWORDS	10
MAX プリンの逆襲 /The Pudding Strikes Back	16
THE BiNARYM@STER	21
LINQ in C# 3.0	26
マイコンで作る Wii コントローラもどき	33
Flex 入門 1	47
オーディオベーシック？	52
神々の言霊	60
青年男子ダイエット	62
ICPC のお誘い	65

## 学類新歓後記

文 編集部 mitty

一年生の諸君は、あの日のことを覚えているだろうか。

あの日、情報科学類新入生歓迎委員会は燃えていた。<sup>\*1</sup> おりしも雨の予報が出ていた 4/13、新歓委最後の活動となる、新一年生のためのお花見会である。オードブル、鍋、bingoゲーム、景品。せっかくいろいろこれまで準備してきた最後の集大成が、雨なんかで中止されではたまらない。

これは、新入生歓迎のために立ち上がった、上級生たちの物語である。

### 閑話休題

と、某~~国~~公共放送の Project ×みたいなノリで始めてみましたが、そんな大それなものでもなくだらだらと新歓委員会の舞台裏とかを紹介していきたいと思います。 mitty 視点ですので独断と偏見に満ちあふれていますが、~~まあいつものこと~~ご了承ください。

4/3

明日からの新入生入居手続き手伝いのために、「情報科学」という看板を急遽制作する。というか、直前になるまで誰も思い出さないのはお約束なのだろうか。

WORD に A3 ノビの印刷ができるプリンタがあるので、インクと紙を用意して借りることにした。

MS WORD で Fontsize を 200 くらいに指定して印刷。



完成。やっつけっぽいのは仕様なのであきらめよう。

\*1 草冠じやないですよ、念のため。

## 学類新歓後記

夕方、平砂の方で大量に捨ててあるのを覚えておいた木のブロックを拾いに行く。三往復くらいするけれどまだある。



これやどかりセール<sup>\*1</sup>で買うと4個で1200円くらいしたよなあ…。もったいない。ちなみに、この日をあわせてそれまでの一ヶ月くらいで**100個近く回収**。何に使ったのかはそのうちまた後日談で記事にしたいと思うので、こうご期待<sup>\*2</sup>。なお、余った分は入居手続きの際に学類のテントに来た新入生にプレゼント。日取りの都合で一の矢の方にのみとなってしまったので、あしからずご了承のほどを。

### 4/6

新入生第一回顔合わせ会@第三エリア食堂。

~~人がゴミのよう~~に集ま予想を遙かに超える大変な参加人数で、危うく席が足りなくなるところだったのは内緒。昨年はなんというか、皆で出されたものを黙々と食べるというお通夜のようなイベントだったからなあ…。

今年の一年生は今後イベントの折にはいろいろと活躍してくれそう、という期待が持てた一日。

### 4/7

WORD入学祝い号の素材写真を撮りに、夜の大学を徘徊する。大気君こわいよ大気君。



\*1 宿舎入居にあわせて、電器店のアイデンとココストア(旧ホットスパー)が行っているセール。その実あまり安くない。

\*2 本当は今号で載せたかったんですが、時間的都合が…寝かせてくださいorz

4/10

学類のオリエンテーションで、産業技術総合研究所に行く。昨年は JAXA だったので、いけなかつた方もこれでコンプリート。今年から選択肢が増えた食品総合研究所はどうなったのだろう。行ったはずの新歓委員長からコメントが全くなかったのが気になるところだ。

4/11

13 日の本番に向け、**禁則事項です**の GO or NOT GO を決めるべく、試作に入る。試作でまずいようであればキャンセル予定だったが、材料が変わった割にはなかなか美味しく作れた。私自身は前のパッケージの時の方が後味が良かった気がするのだが…。

まあ、**禁則事項です**を材料にしている時点で何をどうしようと無駄な抵抗な訳だが。

4/12

bingoゲームの買い出し。昨年は新歓委員が使っていないものなどを持ち寄って集めた感が強かったのだけれど、今年は物品募集の連絡をするのを忘れていた&新歓予算が余りそうなので予算で買うことにした。

というわけで我等が趣都アキハバラへ。  
買った物は以下の通り。

#### ドンキホーテ

bingoカード 100 (正確には景品ではないが)

自転車ワイヤーロック

LED ライト

通常チープチビング<sup>\*1</sup>

萌え無限チープチ×三種類<sup>\*2</sup>

計 ¥6526

\*1 あまりチープチしている感じがしない、と私の周りでは不評。やはり実物でないと皆満たされないらしい。

\*2 通常版無限チープチは 100 回に一度変な効果音がなるとのことだが、こちらは効果音ではなく変な声がするらしい。声優にくぎみーを使っている辺りがかなりキてる。正式名称は「ぷち萌え」。バンダイなにやってんだもっとやれという声が聞こえてきそうな商品。なお、この記事を書く際に公式サイトを見てみたら、~~中の~~キャラクターのブログなるものが開設されていた。やつら本気だ(ry

なお、残念ながらツンデレバージョンはドンキ以外も含めて三店舗で探したけれど見つからず。やはり人気なのだろうか。

## 学類新歓後記

ヨドバシカメラ

CAT5e 5m LAN ケーブル× 3 本

24 枚入り CD/DVD ケース

7 口 OA タップ

マウスパッド

CAT7 0.2m LAN ケーブル<sup>\*1</sup>

計 ¥5590

あきばお～こく

orz マグカップ<sup>\*2</sup>

初音神社 電脳厄除<sup>\*3</sup>

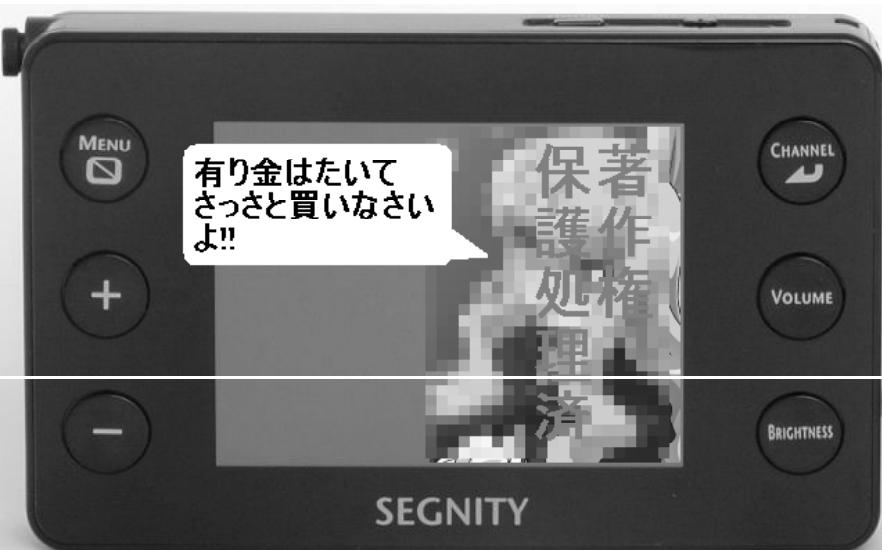
さいたましてもいいですかタオル<sup>\*4</sup>

おでん缶 牛筋・大根入り<sup>\*5</sup>

初音ミク inside シール<sup>\*6</sup>

SEGNITY<sup>\*7</sup>

計 ¥9594



※画像は mitty の勝手なイメージです。

\*1 10Gbit 対応の、(転送量が)多い日も安心設計。プラグの金属部分が金メッキされているという凝り様。もちろん下位互換もばっちりで 10BASE-T のケーブルとしても使える。ちなみに、10Gbit 対応の NIC (Network Interface Card、いわゆる LAN カード) は 10 万円程度する。

\*2 今日もまた自主休校してしまった… orz と黄昏れながら飲むのですね、分かります。

\*3 みく～、みくみくみく～。

\*4 4 年前に買って以来私も愛用していますが、この手の商品にしては驚くほど丈夫です(色落ちもほとんどしない)。サイズがバスタオルと普通のタオルの間くらいで丁度良いので重宝しています。

\*5 実はおでん缶食べたことありません。

\*6 「no one inside(中に誰もいませんよ)」を探したのだけれど無かったのでこれに。

\*7 ツンデレワンセグ。これも声優さんはくぎみーだなあ…。

ココストア

メタルラック

計 ¥4980

全景。



ぶっちゃけ、あきばお～こくだけで良かった気も。変なモノならいくらでもあったし。もっとマニアックなモノも買ってこようかとも思ったのだけれど、あまりマニアックすぎてもウケが悪そうだったのでこのようなラインナップに。使えそうなものと、使えそうで使えないもの、最初からネタ指向の三本立て。なお、予算的には2万円を予定していたので超過というか、メタルラック買わなければ良かった気がする。<sup>\*1</sup>

#### 4/13

新歓集大成となる最終日。当日朝8時、集まってきた新歓委員たちの顔は外の天気と同様に曇っていた。前日の雨予報をそのままなぞるかのような、今にも降り出しそうな空模様に、委員会内にも中止意見や、規模を縮小して実施などの意見が出る。

3C棟ラウンジで実行しようという意見もあったが、参加人数が人数なのでおそらく警備の人見とがめられるだろうということから却下。結局、第三エリアよりはるかかなたの1H棟の一階ピロティ一部分で行うことになった。

---

<sup>\*1</sup> 新歓委全体としては予算以下になっているのでご安心を。

## 学類新歓後記

1H 棟一階はそもそもパーティをやることを想定したような作りになつてない<sup>\*1</sup>。そのため屋外で実行したときのために用意しておいたブルーシートを4枚ほど広げ、テーブルを急遽用意して委員自家製の豚汁と予約してあったオードブルを配膳。そうこうしているうちに予定期刻の11時半になる。

時折雨は降るもののか配していたような本降りにはならなかつた。皿に盛られた料理や鍋があらかた片付けられてくると、皆の関心はいよいよビンゴゲームに。

想像以上にネタ指向の景品が人気で、萌えブチブチは一瞬でなくなつてしまつた。獲得した人に公式サイトの商品説明を読み上げてもらったのは言うまでもない。<sup>\*2</sup>

そして、ネタ指向景品が少ないことを詫びる委員に対して「ここをどこだと思っているんですか、情報科学類ですよ!?」などと色々な意味で頼もしい答えを返してくれる一年生の興奮も最高潮になつた時、~~マジック~~ワンセグTVのSEGNITYの仕様説明<sup>\*3</sup>をしていると、それは起きたのである。

---

\*1 もつとも、そんなことを想定した作りになつてゐる教室なんてものがあつたら見てみたいところではあるが。

\*2 たとえば…

幼なじみ編:料理と100円ショップ巡りが大好きな女の子です。隣に住む幼なじみを甲斐甲斐しく毎日起こしに行くなど、世話好きな性格をしています。人混みや騒音が苦手で、パニックなることがしばしば……。将来の夢は「お嫁さん」と小学生の頃から密かにあこがれています。

from [http://gigazine.net/index.php?news/comments/20080207\\_mugen\\_moe/](http://gigazine.net/index.php?news/comments/20080207_mugen_moe/)

\*3 「声をあててているのはあの釘宮さんです」という説明に対して、会場の興奮は最高潮に。今年の一年生は本当に将来が~~七~~楽しみです。

*A short time ago in Tsukuba very,  
very near...*

# ST\*R WORDS

## Episode MMVIII A NEW PUDDING



*It is a period of new commers.  
Sophomoric students, welcoming  
freshmen to their new college,  
have won their first victory  
against the very big pudding.*

*In the battle, Welcome committee  
spies managed to steal secret  
recipe to the WORDian's  
ultimate weapon, the MAX  
PUDDING, the largest bucket  
pudding with enough power to  
destroy everyone's tongue and  
stomach.*

エピソード 2008 新たなるプリン

時は新歓時期。右も左も分からぬ新入生を秘密結社へと取り込む新しい学校へと歓迎すべく活動していた二年生たちは、その誕生以来人々を脅かし続けてきた巨大プリンに対し初勝利を上げたばかりであった。のちに伝説となつた激戦、その噂を聞きつけた二年生たちは新歓委員会を組織し、WORD 帝国の最終兵器「MAX プリン」、すなわちいかなる人間であっても舌と胃を破壊されてしまう恐るべきパワーを備えた究極のバケツ DE プリン、その隠された設計図を入手すべくスパイを送り込み、多大なる犠牲を強いて奪取することに成功していた。

しかし、そのスパイとは…

文 編集部 いのひろ

デス・スター（第3エリア某所）

新たなる 8 リットル MAX プリンを手に、暗黒面の使者たち（以降、我々）は奇襲の準備に取りかかっていた。

この作戦は、我々 WORD 編集部員 2 年生が新入生に、もはや情報科学類名物と言っても過言ではない 8 リットル MAX プリンの洗礼をお見舞いすべく企画されたものである。我々は暗黒面の人間であるため、全身を黒くした（スーツを着た）。そして本番でプリンを持って突入するメイドさん（！）であるが、これを精銳揃いの情報科学類でも希に見る変態度を誇る「ひなたちゃん（牛久市、19 歳、男性）」がやることになった。満場一致、と言うよりかは本人の強い希望を叶えてあげたと言ったほうが正しい。

我々は綿密な打ち合わせをした後、新入生が第 2 回顔合わせ会で盛り上がっていると思われる 1H 棟へ向かったのである。



第 3 エリア 食堂前

今回の MAX プリンは、昨年度の記事「Project MAX」で開発した MAX プリンとはひと味違った。これはプリンの素を「プリンミクス」から「プリンエル」に変えたからである。カワチ薬局による情報では、ハウス食品が「プリンミクス」の生産を中止したようである、ということがわかった。仕方が無く、我々は「プリンエル」を用いたのである。

しかし「プリンミクス」がいわゆる「普通のプリン」を作るプリンの素であり、「プリンエル」は「カスタード風味プリン」を作るプリンの素であった。このため、「プリンエル」は前回の MAX プリンよりも甘みが増している事が容易に想像できた。文字通りひと味違ったのである。

さて作戦の詳細を説明しよう。まず私、いのひろとオーディオ野郎かづきおが MAX プリンが展開される折りたたみ式のテーブルを持って、顔合わせ会の場所に参上する。もちろん新入生・

## ST\*R WORDS

新歓委員はいきなり怪しい男二人組がバケツを持ってくることなんか知らない。そしてその正体がプリンだということも知らない。大切なことなのでもう一度言う。新入生・新歓委員は何も知らないのである。

テーブルを設置した私とかづきおは一度退散し、その時に備える。そのとき、それはプリンによる奇襲、そのものである。

我々は今回の突入の為にプリンを運ぶ台車を独自に開発していた。この台車には UPS (Uninterruptible Power Supply; 無停電電源装置)、iPod Shuffle、DELL コンピュータのデスクトップ PC 用ステレオスピーカが搭載されている。読者の中には、我々が考えていた事を容易に想像できる方もいるのではないだろうか。



第2、3エリア付近にて（稻妻マークの箱の中にUPSが搭載されている）

作戦の説明に戻ろう。

我々はタイミングを見計らい、奇襲をかけるのである。すなわち「帝国のマーチ（ダースベイダーの登場シーンでおなじみのアレ）」を爆音で再生しながら、ひなたちゃんが台車でプリンを運ぶのである。黒ずくめの我々は彼の後ろについて行き、プリンの展開を手伝う。私のひろとかづきお、バイナリーマスター suma がプリンの展開を、そして yasuharu が記念すべき彼の初めてのメイドのお仕事を DV カメラで撮影するという作戦であった。

## ベースキャンプ（共同研究棟A1階）

まず我々は 1H 棟にほど近い「共同研究棟 A」にベースキャンプを張った。その時まで待機である。当日、我々がほどよいタイミングを知る事ができたのには理由がある。我々には強力なスパイ mitty がいたのである。彼は新歓委員として会計の役職をこなしていた。しかし同時に、我々と通信していたのである。ちなみに我々の通信は SSL で暗号化されていなかった為、傍受されているのではないかと少し不安であった。こうして我々は mitty からの情報によって、最適なタ

イミングで突入することができたのである。



ベースキャンプにて、お辞儀の練習をするひなたちゃん



1H 棟付近（左から、かづきお、ひなたちゃん、yasuharu、suma、そして私）

#### 情報科学類第2回顔合わせ会 会場（第1エリアH棟1階）

「その時」である（歴史が動いた）。我々は共同研究棟 A を出発し、1H 棟へ向かった。作戦通り、私とかづきおがテーブルを運び、bingoゲームで大盛り上がりの 1 年生のところへ突入したのである。戸惑う 1 年生・新歓委員を尻目に我々は無言でテーブルを組み立てた。しかし彼らはすぐにbingoゲームを再開したのである。この後とんでもないことが起こるとも知らずに。

一度 1H 棟から撤退した我々は、本突入の準備に取りかかった。スピーカのつまみを最大出力にし、iPod Shuffle の再生ボタンを押した。「てーてーてーてーてーてーてー♪」、帝国のマーチが響き渡った。

## ST\*R WORDS



1H 棟に突入した我々。プリンを運ぶひなたちゃん

ひなたちゃんを先頭に、1H 棟に突入。唖然とする 1 年生。ひなたちゃんはテーブルの脇に台車を止めた。私は台車からバケツを取り、机の上に勢いよく置いた。フタを外し、かづきおがタッパーを被せた。いよいよ展開である。我々はタイミングを合わせ、バケツをひっくり返した。かづきおがバケツを引っ張り上げると、新たなる 8 リットル MAX プリンがそこに誕生したのである！！！1（もちろんこの間、爆音「帝国のマーチ」はリピート再生である）。



バケツを引っ張り上げるかづきお。新たなる MAX プリン爆誕の瞬間である

1年生から歓声があがった。と同時に、「あれ、WORD の人じやない?」「あの人見たことある」という声が上がった。言い忘れていたが我々は事前に 100 円ショップでサングラスを購入、当日は装着していた。しかしながら、既に WORD に出入りしていた新入生によって、我々の正体は半ばバレたも同然であった。

プリンを展開した我々は、ひなたちゃんの一礼の後、退散した。



切り分けられる新たなる MAX プリン

その後、我々が送り込んだスパイ mitty の情報によれば、8 リットル MAX プリンは 50 人の新入生でも完食することはできなかつたようである。我々のこの壮大な作戦は成功したのである。



新入生 50 人でも完食できなかつた、恐るべき 8 リットル MAX プリン

## MAXプリンの逆襲/The Pudding Strikes Back

あるいは蛇足

文 編集部 mitty

### MAXの異常な糖度 また私は如何にして心配するのを止めてMAXを愛するようになったか

MAXは偉大である。三学の暗黒卿 Inohiroによって私が~~オース~~の暗黒面<sup>1</sup>プリンのカラメル面<sup>1</sup>に染まるようになったのか、その過程に関して、私は真に驚くべき真実を見つけたが、この余白はそれを書くには狭すぎるため次の機会を待つとしよう。

### メイド服

あのメイド服は如何にして用意されたのか。実は12日、世を忍ぶ仮の姿として筑波大学情報科学類新入生歓迎委員会会計を担っていた私は、bingoゲームの景品を買うために訪れた趣都アキハバラにおいて秘密の任務を遂行していた。

#### すなわち、メイド服の購入である。

一口にメイド服といってもその種類は千差万別である。個人的にはスカートが短いものは正統派とは言い難い。手袋は長い方がよい。カチューシャはありふれているので、あえて違った形のものを選ぶのも一興。

そういう要望に答えてくれるのは、そう、専門店をおいて他にないのである。

アキハバラには、調べた限りでもメイド服を扱っている店舗が片手以上ある。ただ、そのうち大半はコスプレの一環として扱っているので、「専門店」ではない。モノを選ばないのであればドンキホーテでもそれっぽいものは売っているが、やはりスカートが短い。これは許せないものがあるので、素直に専門店に向かう…と言いたいところであったが、新歓用の景品を揃えておかないと怪しまれてしまうため、先に景品を購入した後に向かうこととなった。

bingoゲームの景品を揃えたころにはかなり時間がたってしまっていた。仕方ないため、一番最後に向かおうとしていた一番専門性が高そうな店に向かう。メイド服以外扱っていないという文字通りの専門店である。

その建物に近づくと、何故か眼鏡と書いてある。はて、もしかして移転したのだろうか…。

眼鏡は眼鏡でも、働くメイドさんのための眼鏡屋さんであった。別店舗という訳ではなく併設しているらしい。さすがひと味違う。

オーダーメードも取り扱っているらしい。というか二階が工房で、大きいミシンが置いてあったので店内でも縫製しているのだろう。

既に店内には3名ほど先客(男性)が居て、メイドさん(店員(女性))と歓談していた。話しぶりからすると常連さんらしい。そうこうしているうちにもう一名登場「遅れてごめーん。」メイド服専門店で待ち合わせとは、さすが客のレベルもひと味違うようだ。

実際に着るのが私ではないため、前日に測っておいたひなちゃん<sup>2</sup>のスリーサイズをお伝えし、

\*1 MAXにカラメルの素を溶かすと本当に~~ダ~~黒いです。

\*2 正式名「ひなた」。牛久市が誇る、汎用変態決戦兵器である。建造から19年を経ているがその変態性にはますます磨きがかかっている。なお、どこと戦っているのかはWORD七不思議に数えられるらしい。

予算とのかねあいおよび店頭在庫から見繕ってもらう。メイド服本体とは別に帽子、靴下、手袋を購入。しめて 38,115 円<sup>\*1</sup>。ちなみに、パーティ用ではなくまともに縫製されているコスプレ衣装はたいてい同じ値段帯かそれ以上なので、ちょっと買うか迷ってる人は注意しよう。<sup>\*2</sup>

とりあえず着てみた。



~~苦勞~~と試行玄人志向の中の人とは関係ありません。きっとたぶん。



\*1もちろん、全て自腹である。

\*2女装…ではないにしろ、誰しも逸般的な服を買ってみたいと思ったことの一度や二度はあるに違いないと私は思うのだが、皆さんはどうだろう。

## 学類新歓後記 the dark side

メイド服と機関銃<sup>\*1</sup>。



悪の手先、ローアングラーと戦うの図

着せる対象のひなちゃんよりも購入者が楽しんでいたのは秘密である。



---

\*1 WORD 読まない奴は死刑！

### 打ち合わせ

成功の秘訣、それは綿密に組まれた計画と、打ち合わせである。

しつこいまでの予行演習、実際に衣装を着ての動作。そして、何よりも諦めない<sup>\*1</sup> 姿勢が重要である。

実はスパイだった mitty を始め、カラメル面に染まった WORD 有志の意志は固かった。その意気込みは打ち合わせにも現れようものである。



スパイのくせに一番偉そうな mitty



---

\*1 何を？

## 学類新歓後記 the dark side

### 一年生諸君の反応

ひなちゃんを目にした一年生の反応は劇的であった。いや、一年生どころか、もう筑波の異常時空<sup>\*1</sup>に一年間さらされて免疫が出来てきているはずの新歓委員の二年生たちでさえ、混乱を免れ得なかった。新歓委員 O に至っては狂ったように笑い続けていたほどである(実話)。



※やばすぎてお見せできません。

### まとめ

いかがだっただろうか。きっとこの記事を読んで、「MAX 作ってみたい！！！ 1」と思った諸君もいることだろう。そんな君は今からでも遅くない、ぜひ情報科学類の暗黒面 WORD に参加しようではないか。

---

\*1 aberrant cosmosphere、通称 AC。直訳すると「A たまおか C」である。(大嘘)



**LESSON 1**

文 編集部 suma

はじめに

どーも、ご無沙汰しております。suma でございます。コンシューマーゲーム機の某ゲームソフトの解析記事を数回にわたって載せていいたいと思います。

内容としては、ネタ+ほんのわずかの技術ネタ+解析をしていく過程のレポートを予定しています。加えて、ターゲットに依存しない初步的な解析（リバースエンジニアリング）テクニック、解析ツール作成といった技術的な内容の解説も考えています。

本記事はコピーや改造行為は目的としたものではありません。そのため、解析対象、一部の解析方法や解析結果につきましては多くの箇所を伏せさせていただきますので、その点をご了承ください。また、当然のごとくこの記事を参考するしないに関わらず解析した結果は個人だけでお楽しみください。もう大人ですから、そのあたりは自重できますよね。

それでは、一部誇張表現が見苦しいかもしれません、記事をお楽しみください。

ソフトがやってきた

解析を始める前のこと、それは今年のゴールデンウィークに入る前のことです。WORD 編集部員の ranha 君が某ソフトを買ったと聞きました。そう、それはニコニコ動画で MAD 動画作成が盛んな、自分がプロデューサーとなって以下略なゲームのことです。

ranha 容疑者の話によりますと、Amazon の割引率が異常ということでした。それを聞いて私は先日 Amazon の商品紹介メールにてそのゲームがお勧めされていたことを思い出しました。私がそのメールを見た時に「Amazon 自重！！」と心の中でつぶやいたことを覚えています。

さて、諸処の事情は飛ばしまして、実際にプレイした時の会話に移ります。

最初は、ranha 君もとい、らんは P (P はプロデューサーの意) からプレイを始めます。

## らんは P 爆誕！！！！！ 1 1

少佐「どうやって描画してるんでしょーねー」

ろんず「60fps くらいかなー？」

suma「選択肢が地味にうける・・・」

らんは P「なんかただの作業になってきましたよ」

〇〇〇「UI が腐ってるなー」

・

・

## THE BiNARYM@STER

こんな感じでプレイが進みました。もちろん私も一回だけ sumaP としてプレイさせてもらいました。肝心のゲーム内容の方ですが、ゲームをあまりプレイしない私にとってはゲームバランスが整っていると感じました。ゲームエンジンは一部を除いてシンプルそうな感触です。

### ゲームを解析して××なことをしてみたい><

特に解析を始めた理由といえるほどのことはないのですが、答えは「そこにバイナリがあったから」です。プラスアルファで、今月の WORD は紙面がいつになく余ってるとか（r y

解析をしていく目的は人それぞれです。解析して得られた結果からツールを作つて人に自慢するということも考えられます。しかし、私はそういったことにあまり興味がなく、PC (Windows) でプレイできる互換エンジンを作成できれば Cool だと考えています。しかし、モデルビューアネタでひどくバッシングを受けた人がいるとか、エンジン作成はスペック的に無理だとか…始まる前にネタ切れの危機です><

とりあえず、~~十~~年くらい時間をかけてモチベーションが続く限りバイナリと戯れ、それに合わせる形で個人的、つまり自分専用に楽しめるツールを少佐と一緒に作れたらいいなと思っています。

ちなみに解析するメリットは特にありません。自分の知らないアーキテクチャについて学んだり、2 進数・16 進数が好きになったり、バイナリデータの理解度が深まったり、ゲームデータのファイルフォーマットを脳内デコードできるようになります。といったスキル修得が考えられます。が、ちょっとびりマイナ一分野です。いちいち考えていてもらちが明きません。解析を始めることにしましょう！

### データを抜き出すお仕事

まず最初は、ゲームディスクからデータを抽出する簡単なお仕事をします。PC 用のソフトであればデータの取得は簡単ですが、コンシューマゲーム機のディスクからデータを取得しなければなりません。ファイルシステムによっては PC からコピーすることができますが、今回のブツはプロテクトがかかっていました。面倒だったので、プロテクトに関してはググって解決しましたが、場合によっては解析ツールの自作が必要になりそうです。

プロテクトの解除は注意が必要です。日本国内における法律では、著作権保護を目的としたプロテクトの解除、暗号を解いてコピーできるようにする装置（機器・ソフト）の製造・頒布（ようするに配布）は違法（刑事罰の対象）となっています（正しい内容については出版物を調べるのがお勧め）。とりあえず、ツールを使う側は処罰されないことになっていますが、わからないことや不明瞭な点があれば自重しておきましょう。いわゆる DVD の暗号解除ツールがこれにあたりますので、うかつに作ったり配布しちゃだめなのです>< 完全に正しく説明できる自信も私にはないので、詳しくはググレカス><、ということで。

### バイナリを見てニヤニヤする前に

データ抜き出しが終わったら、次は解析作業に移ります。と言いたいところですが、その前に情報収集をします。ソフトによっては既に解析されて、ウェブ上に資料やツールが掲載されていることもあります。また、作業が進んでから、解析ツールや資料が既出だったことを知るとモチベーションが下がってしまいます。快適な解析ライフを送るためにも、解析の前になるべく多く

の情報を収集しておきましょう。

今回のターゲットの場合は、音声の抽出、画像の展開、一部のファイルフォーマットに関する資料を発見することができました。

もちろん、ゲームの解析はファイルフォーマットだけでなく、実行プログラムの解析も行います。いわゆるリバースコードエンジニアリングという行為です。

このゲーム機の実行ファイルについて調べみると、プログラムが暗号化・デジタル署名されてパッケージ化されているとわかりました。ディスクからデータを抽出したときと同様に、詳細をウェブから探します。ツールは発見することができましたが、プログラムはメーカーの希望で公開停止になっていました。再配布しているサイトを探しますが、中々見つけることができません。心が折れときそうです><

ここでコンピュータサイエンスを学んでいる者として、ツール自作にとりかかりたいとも思いました。しかし、開発と解析するコストが非常に大きいです。また、せっかくツールを作成しても人目に晒すことができません。自作ツールは散りゆく運命にあります。そういうわけで仕方なく、サイトやフォーラム（掲示板）でファイルを再配布していないか探し続けました。Google 先生にお世話になること数時間、最終的にツールをダウンロードでき、パッケージを展開することができました。とにかく、解析には諦めずに行動し続ける根気が必要です。

### 実行ファイルを眺めてみる

これで実行ファイル解析の準備がひとつ完了しました。最初はいきなり逆アセンブル（disassemble）せず、バイナリエディタで実行ファイルを開きます。バイナリエディタなどのツールで、プログラムが利用しているライブラリ、文字列などを調べ、プログラムの構造や流れを推測・想像します。初めから逆アセンブルしても構いませんが、大量に出力されるアセンブリコードの海からプログラム全体を把握するのは簡単ではありません。

バイナリエディタで 16 進数のダンプ画面をよく見てくると、何かが見えてきます。ここで私を差し置いて「オペコードが読める！ 読めるぞー！」とか言っちゃう人はこのまま別の記事でも読んでください。そんな人を楽しませるネタは提供できません>< 私が聞いた話ですが、x86 の命令（おそらく全命令）をバイナリからスラスラ読める人が世界に数人いたりするそうです（CPU アーキテクチャによって大きく増減しそうですね）。これがハッカーの中でも上級の、いわゆる Wizard って人たちのことだと思います><

次は、16 進ダンプで実行ファイルのヘッダー（先頭部分）を眺めます。慣れればファイルフォーマットらしき部分が見えてきます。何も見てこない人は、ビットマップファイルや、Jpeg 画像のファイルフォーマットで訓練するとよいでしょう。たいてい、32 ビット長の整数（or 符号なし整数）や、00 と FF のパターンがあり、それぞれ何かを意味しています。何回か繰り返していくと、いつのまにか 16 進数が好きになるかもしれません。がんばって 16 進数を 10 進数に暗算できるよう訓練しておきましょう！ 一瞬で暗算できたらかっこいいですよ！

今度は、文字列ダンプからでも実行ファイルフォーマットを探します。多くのファイルには、先頭部分にファイル固有の情報を示すとヘッダーと、ファイルを示すシグネチャ（特定の文字列）があります。そのため、実行ファイル先頭数バイトを見れば、どのプラットフォームを対象とし

## THE BiNARYM@STER

ているのかおまかに知ることができます。既存の有名な実行ファイルフォーマットについては、 Wikipedia を見ると幸せになれるかもしれません ( Wikipedia 「オブジェクトファイル」を参照)。私の場合は、ファイルを開いたときは、最初に文字列ダンプを流すように見て、それから 16 進ダンプを見ます。文字列でフォーマット等を確認し、次にバイナリデータからパラメータ (32 ビット整数とか) 等を読み取るのです。

今回ターゲットの実行ファイルは有名な実行ファイルフォーマットでした。最近のゲーム機は環境がリッチになり、ディスク容量が豊富になったのも実行ファイルフォーマットに関係ありそうです。しかも、Assert カログ出力らしきメッセージ、一部のライブラリ名まで含まれていました。私は一般に販売されているソフトウェアにデバッグ文字列が残っていることはないと思っていましたが、最近はそういう時代なのかもしれません。ゲーム、しかもコンシューマーゲーム機のソフトからですよ、わらわらと g k b r みたいに色々な文字列が出てくると、なんだか逆に自分が釣られてしまったのかと錯覚してしまいます (´・ω・｀)。

さて、せっかくなので含まれている文字列をなめるように見ていきます。ゲーム開発者の方に遠慮なんてする必要ありません>< 個人的なお遊び…なのですから…いやまあ、こうやって記事にして晒す段階で個人で済まないですが、よい子の読者の方々は、この記事を読んでも悪いことしませんよね？ お願いですから個人的な範囲でとどめるようお願いします。お約束はこのあたりにして、内容は自重して伏せつつ解析を続けていきたいと思います。ちなみに発見した文字列には…色々ありました。

### ぎやくあっせんぶらあー

さて、続いてついにプログラムを逆アセンブルします！ 逆アセンブラーをどこかから探してきましょう。国内にはまず存在しないので、海外旅行 (海外サイトの検索) をします。Windows・Linux 向けの逆アセンブラー・デバッガは簡単に手に入りますが、コンシューマーゲーム機で利用可能な逆アセンブラーは中々存在しません。現在の主要コンシューマーゲーム機 (PS3, Xbox360, Wii) はいずれも PowerPC をベースとした CPU アーキテクチャを採用しています。実行ファイルのパッケージ展開の件と同じように、逆アセンブラーまでも Google 先生にお世話になりました…と思いきや、実行ファイル展開ツールを作成した作者さんが逆アセンブラーも作成していました。

しかし、その逆アセンブラーは IDA Pro という逆アセンブラー向けのプラグインでした。IDA Pro はセキュリティ監査目的や、その他あらゆる CPU (x86 問わず、組み込み、ゲーム機など) に対応したプロフェッショナル向けの逆アセンブラーです。それだけ超高機能な製品だけに、PowerPC に対応したバージョンは 985 米ドルも値段がしました。お仕事で使うならこの価格に納得できますが (仕事用なら安いもんです)、さすがに「ゲーム解析」のためだけに購入する気になりません。うーむ、困りました。

逆アセンブラーが許されるのは小学生までだよね！

とかいうことなんでしょうか！！ 切ないです>< つまり人が作ったツールに頼らず逆アセンブラー自作するか脳内逆アセンブルしろ、と。正直、i386 の逆アセンブラーは作りたくありませんが、PowerPC なら作っても良いような気がしてきます。幸いにも、情報学類・情報科学類の計算機には Power Mac があります。CPU が PowerPC 970 なので、生の Power アーキテクチャと戯れる

ことができます。まぁ、実際に逆アセンブルを作るには時間や労力かかるので、喜々として作業を始める気にもならないので、また次に後回しにすることにします。

### **おわりに**

少しはゲームを解析していく雰囲気や、解析のアプローチを理解していただけたでしょうか？  
今回は解析の取りかかりを主にまとめましたが、次回は少佐がファイルフォーマットを解析や、ツール作成について解説できればいいと思っています。

# LINQ in C# 3.0

文 編集部 いのひろ

WORD 編集部のいのひろです。2 年生になりました。昨年度は技術系の記事を書こうと思いつつ、MAX プリン 8 リットル作る企画（もちろん今年度も作りますが）や大学でよりよく寝る的な記事を書いていました。今年は短めの記事を連載してみたいと思っていました。そして今回の記事がその試みです。何はともあれ、今年度もよろしくお願いします。

## C# 3.0 と .NET Framework 3.5

Windows 上でしか動かないことから情報学類・情報科学類からあまり人気がない（と思われる）.NET Framework ですが、最近どんどん.NET で様々な事ができるようになっています。その中でも LINQ と呼ばれる.NET 上での新しいデータアクセス手段について C# の新しい機能を含めて簡単に解説してみたいと思います\*1。

.NET Framework の現在の最新バージョンは「3.5」になっています。この 3.5 では今回解説する LINQ と呼ばれる機能や ASP.NET AJAX、それらのために改良された C#/VB.net の新しいバージョンがリリースされました (C# 3.0/Visual Basic.net 9.0)。

Microsoft の IDE である Visual Studio も 2008 がリリースされ、Windows Vista が正式にサポートされました。

3.5 は前バージョンの 3.0、その前のバージョンの 2.0 を土台に構成されています。2.0 については既に多くの方がご存じだと思いますが、3.0 についてはけっこう知らない人が多いのではないかと思います。ということで 3.0 の中身についても簡単に振り返ってみることにします。

## .NET Framework 3.0

3.0 には Windows Presentation Foundation (WPF) , Windows Communication Foundation (WCF) , Windows Workflow Foundation (WF) , Windows CardSpace などの機能が追加されました。

Windows Presentation Foundation (WPF) は、主に Windows Vista 向けのこれまでにない UI を備えたアプリケーションの開発に用いる機能です。XAML (ザムル) と呼ばれる XML ベースのマークアップ言語で UI を、C#/VB.net でロジックを書きます。Microsoft は WPF アプリケーション用の UI を作るデザイナ向けスイート (笑) スイート 「Microsoft Expression」 をリリースしています。

---

\*1 この記事および取り扱うネタは少なからず C# の文法を理解している必要があります。読む側にとっても、書く側にとってもあまり重たい記事にしたくないという考え方から、C# の詳しい書き方、メインに取り扱うネタ以外の技術については、本当に簡単にしか触れません。あらかじめご了承下さい。しかしながら C# は本当に簡単にプログラミングすることができるので、身構えずにどんどん始めてみる事をおすすめします。Visual Studio 2008 も技術職員室アプリケーション班 (3E110) で無償でレンタルすることができます。

個人的には「C#によるプログラミング入門 (<http://ufcpp.net/study/csharp/>)」がおすすめです。



WPF アプリケーションの例「Family.Show」（家系図アプリケーション）

Windows Communication Foundation (WCF) は、Web サービスなどを構築・実行する新しい基盤です。新しいといつても、これまでの ASP.NET Web サービスや.NET Remoting などの既存のテクノロジを生かしつつ、新たなプログラミングモデルなどを導入したものです。WCF を用いて P2P アプリケーションなどを構築することもできます。

Windows Workflow Foundation (WF) は業務アプリケーションなどの動作を視覚的に（状態遷移図・ワークフロー図などで）表現することによって、アプリケーションソフトウェアの保守性や管理性の向上を目指すための技術です。学生にはちょっと無縁かもしれません。

Windows CardSpace は、Windows Vista から導入されたデジタル個人情報の管理などに用いる技術です。たとえば、オンラインショッピング時に Web サイトを経由せず Windows から直接お店とクレジットカードの情報をやり取りするような仕組みがなどが含まれています。が、実際ほとんど普及していない気がします。

### .NET Framework 3.5

3.5 ではこれらのコンポーネントの拡張なども行われました。

3.5 で追加されたその他の機能としては、ASP.NET AJAX や Windows Form の拡張、Base Class Library (BCL) の拡張などです。

ASP.NET AJAX とは、2.0 ベースの ASP.NET に Ajax の要素を加えたもので、いくつかのコンポーネントを配置することによって Ajax 処理を埋め込んだ Web サイトを開発することができます。またオープンソースで多数の Ajax コンポーネントが開発されているので、それを自由に利用す

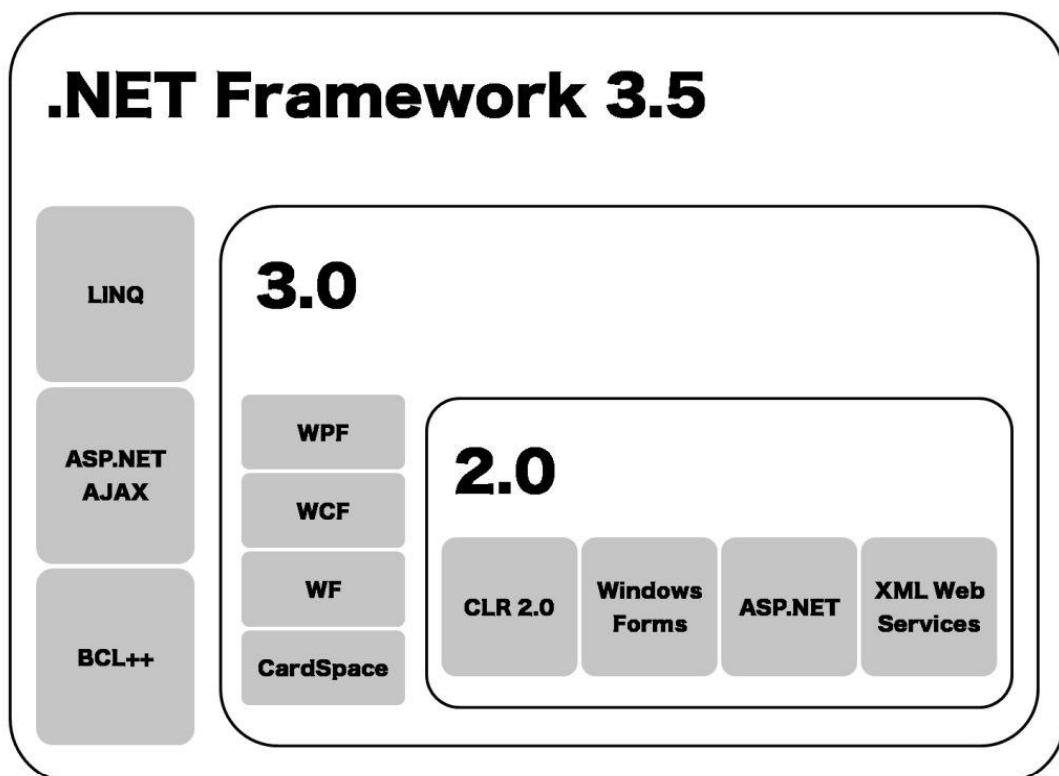
## LINQ in C# 3.0

ることも可能です。

Windows Form の拡張では、Windows Vista の新しいファイルオープンダイアログに対応した Windows Form コントロール、同じく Windows Vista の UAC (User Account Control; ユーザーアカウント制御) に対応したアプリケーションの開発をサポートする機能などが追加されました。

Base Class Library (BCL) の拡張では、近い将来リリースされる予定である SQL Server 2008 での新しいデータ型 (DateTime2 型、DateTimeOffset 型など) に対応した新しいクラスが追加されています。

またこの記事を書いている 5 月半ばに、.NET Framework 3.5 SP1 Beta のリリースがありました。SP1 では SQL Server 2008 で新たに追加される予定の 7 つの型<sup>\*1</sup> が完全にサポートされる予定です。



.NET Framework 2.0-3.5 概略図

### LINQ

さて、話題を LINQ に変えます。

LINQ (Language INtegrated Query; 統合言語クエリ) とは、前に書いたように.NET Framework での新しいデータアクセス手段です。これまで.NET でデータにアクセスするときは ADO.NET というテクノロジを使いました。ADO.NET では DB やその他のデータにアクセスするために Reader/Writer オブジェクトのインスタンスを作り、それを媒介としてデータにアクセスする方法

\*1 7 つの新しい型（詳しい説明は割愛します）：Hierarchyid 型（階層データを簡単に扱える）、日付/時刻型の強化（Date 型、Time 型、DateTime2 型、DateTimeOffset 型）、空間データ型（Geography（地理）型、Geometry（ジオメトリ）型）

でした。

これに比べて、LINQ とは ORM (Object-Relational Mapping; オブジェクト関係マッピング) の.NET 向け実装であり、C#などのオブジェクト指向プログラミング言語から SQL のようなクエリ（問い合わせ）式と呼ばれる式を用いてデータを簡単に扱うことが可能になりました。Ruby on Rails を使ったことある人は、ActiveRecord と同じようなものだと言えばおわかりになるのではないでしょうか。

しかし LINQ は DB に対するデータアクセスだけを提供するものではありません。XML 文章やその他の様々なデータソース (MS 製品以外のものも含めて) に、どれも同じようにアクセスすることができます。

早速 LINQ のコード、C# 3.0 の新しい機能である「クエリ式」を用いたコードを見てみましょう。このコードは、簡単な学生の情報が入っているテーブル<sup>\*1</sup> から学籍番号と一致する学生の氏名を引っ張ってくるものです。

```

1      static void Main( string[] args )
2      {
3          UniversityDataContext university = new
UniversityDataContext();
4
5          var names = from student in university.Students
6                      where student.StudentId == 345
7                      select student.Name;
8
9          foreach( var name in names )
10         {
11             Console.WriteLine( name );
12         }
13     }

```

5 行目から 7 行目にかけてが「クエリ式」です。SQL ライクな式をプログラミング言語で記述することができるのが大きな特徴です。

このクエリ式は、「大学 (university) の生徒 (Students) の中から、学籍番号 (StudentId) が「345」である生徒 (student) を選択し、その生徒の名前 (Name) を要求するクエリ」と解釈することができます。これはとても自然な語順だと思いませんか？

このコードが実行時にどのような SQL を実行しているかというのは、Visual Studio から簡単に確認することができます。適当な場所 (foreach 付近) にブレークポイントを配置し、デバッグ実行 (F5) します。ブレークポイントで実行が中断されると「ローカル」と書かれたウインドウに現在の「names」の中身が表示されます。

---

<sup>\*1</sup> テーブルの構造 : Id (int, null を許容しない、Identity 属性、主キー)、Name (text、null の許容)、StudentId (int、null の許容)

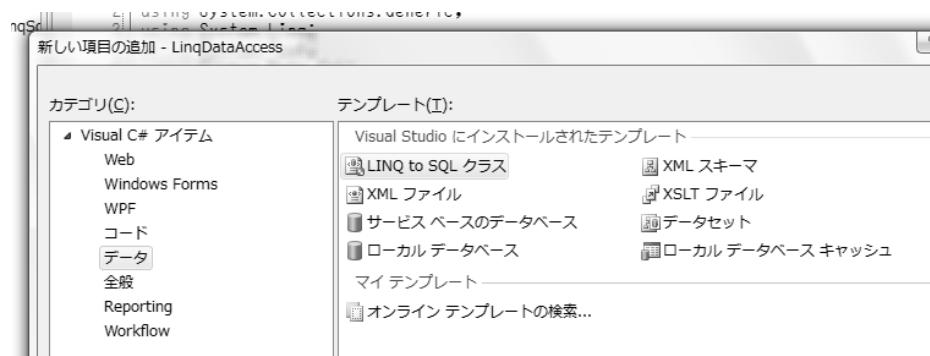
## LINQ in C# 3.0

名前	値
args	{string[0x00000000]}
university	{LinqDataAccess.UniversityDataContext}
newStudent	{LinqDataAccess.Students}
names	{SELECT [t0].[Name]FROM [dbo].[Students] AS [t0]WHERE [t0].[StudentId] = @p0}
deleteStudent	null

### 実行される SQL コマンド

これが実際に発行されている SQL コマンドです。この SQL コマンドは `foreach` でデータを列挙するときに発行・実行されます。(実際には「クエリ式」→「拡張メソッド\*1」→「SQL コマンド」の順番で変換されます。)

LINQ を用いて SQL Server にアクセスするときは「`DataContext` クラス」(ここでは `UniversityDataContext`) のオブジェクトインスタンスを生成して、それを用いてデータにアクセスします。Visual Studio 上で「`LINQ to SQL クラス`」を追加すると、その追加したクラスについての `DataContext` クラスが利用できるようになります。

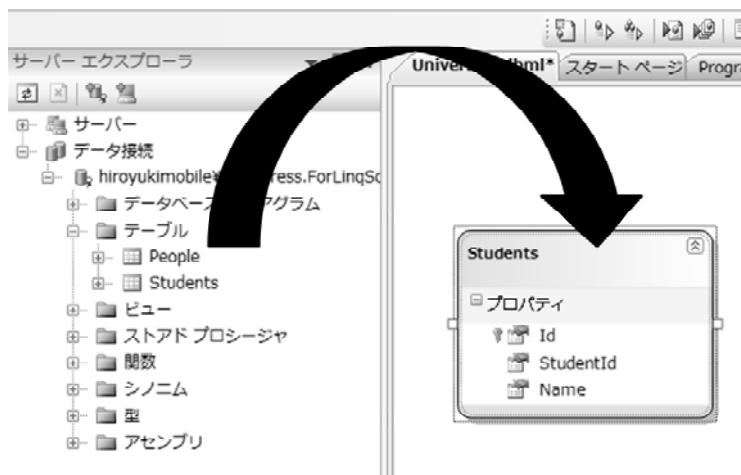


新しい項目の追加から「`LINQ to SQL クラス`」を選択する

(今回は「`University.dbml`」と名前をつけた → `UniversityDataContext` が利用できるようになる)

\*1 拡張メソッド : C# 3.0 からの新機能。クラスの継承などをしなくても、既存のクラスにメソッドの追加をすることができる。しかしコードのどこにそのメソッドが書いてあるかわからなくなることがあるので、多用するべきではありません。今回の例では、C#コンパイラは以下のようにコードを解釈し、SQL を発行しています。

```
var names
    = Students.Where( student => student.StudentId == 345 ).Select( student
=> student.Name );
```



データベースに接続して適当なテーブルをドラッグしてくる

これまでに C#でのプログラミング経験がある方は、「var」という新しい型に驚くと思います。JavaScript でプログラミングを経験したことがある方は、あの「var」かと思われるかもしれません。

この新しいキーワード「var<sup>\*</sup>」はいわゆる「暗黙的型付け（もしくは型推論）」と呼ばれるもので、コンパイラがコンパイル時に自動的に型を判断してくれるものです。これは C# 3.0 からの新機能であり、LINQ の為に開発された機能のひとつです。「var」は「variable」（変数）から来ています。先ほどの「クエリ式」のコードを見て、SQL を書き慣れた人だと違和感を覚えたかもしれません。なぜなら select キーワードがコードの一番後ろに来ていたからです（SQL はコードの一番始め）。これは select を最後に持つてこないと型推論ができないくなってしまうためです。

次に新しい生徒（Student）をテーブルに追加するコードを見てみましょう。

```

1  Students newStudent = new Students
2  {
3      Name = "Inohiro",
4      StudentId = 890
5  };
6  university.Students.InsertOnSubmit( newStudent );
7  university.SubmitChanges();

```

Students クラスの新しいインスタンスを作り、それを DataContext クラスの InsertOnSubmit メソッドにわたすだけです。しかし挿入の SQL コマンドが実行されるのは、SubmitChange メソッドが実行される瞬間であることを覚えておいて下さい。

余談ですが、Visual Studio 2008 Beta2 (.NET Framework 3.5 Beta) の時から、このデータを追加するメソッドや、後述するデータを削除する delete メソッドの名前が変りました。それまでは

---

\*1 「var」キーワードは万能ではありません！ 「var n;」といった宣言では変数 n に何が入るコンパイラがかわからぬため、コンパイルエラーが発生します。

## LINQ in C# 3.0

「university.Students.add( newStudent );」と書けば良かったのですが、SubmitChange メソッドが実行されたときに DB に書き換え（挿入、削除、編集）のクエリが実行されるため、それを意識させる名前（InsertOnSubmit / DeleteOnSubmit）に変更されました。

次に学籍番号が 890 である生徒を探し、そのデータを削除するコードです。

```
1  Students deleteStudent
    = university.Students.First( student => student.StudentId ==
890 );
2  university.Students.DeleteOnSubmit( deleteStudent );
3  university.SubmitChanges(); // 除籍！！(^_^);
```

削除する時も SubmitChange メソッドが呼ばれたときにレコードが削除される SQL コマンドが実行されます。

1 行目の「university.Students.First メソッド」は引数の条件に当てはまるレコードの 1 番最初のものを選ぶというものです。またその First メソッドの引数が見慣れない記法になっている事に気づきましたか？これもまた C# 3.0 からの機能である「ラムダ式」と呼ばれる書き方です。この書き方は少し省略しています。本来の書き方は下のようになります。

```
university.Students.First( ( Students student ) => student.StudentId ==
890 );
```

これは First メソッドの中で、別の条件式を評価していることになります。「First メソッドの引数は Students 型であるが、その中でも StudentId が 890 である student を引数に渡します」という解釈をすることができます。C# 2.0 での「匿名メソッド」でも同じような事ができましたが、こちらのほうが書くのが簡単です（ちなみに下のようになります）。

```
university.Students.First(
    delegate( Students student ) { return student.StudentId == 890;
} );
```

匿名メソッドでできることは、すべてラムダ式で書くことができます。コードが冗長になるので、ラムダ式を使うことをおすすめします。C# 3.0 の開発者さんは「ラムダ式が匿名メソッドよりも早く C# に取り入れられていれば、匿名メソッドは必要なかつただろう」とまで言っています。

これまでの方法である「SqlConnection」オブジェクトのインスタンスを生成し、SQL 文を書いて、実行して結果を得るという方法と比べたら、LINQ は非常にシンプルであり、プログラマにとってもフレンドリーであることがわかります。是非試してみて下さい。

今回は SQL Server に対してデータの検索をしてみましたが、次回は XML ドキュメントに LINQ でアクセスする「LINQ to XML」について取り上げてみたいと思います。

## マイコンで作るWiiコントローラもどき

文 編集部 yasuharu

### はじめに

最近では、マイコンを使ったプログラミングを取り扱っている本がいくつかあり、非常に高機能なものが安価に買えるようになっています。そこで、今回は、DesignWave という雑誌の5月号の付録を使って Wii コントローラもどきを作成していきます。

### 用意するもの

用意するものは以下の通りです。

DesignWave 2008年5月号×1  
USB ケーブル（マイコン側はミニB端子）×1

（図1は、左が雑誌、右上がUSBのケーブル、右下が付録基板）

これだけで、Wii コントローラもどきができてしまします。DesignWave のこの号の付録は、すばらしいことに最初から加速度センサが接続されており、また、USB の端子が実装されています。なので、雑誌を買ってきてケーブルを接続するだけで使うことができます。コントローラを作るにあたっては、この二つだけあればできてしまいます。また、付録基板には以下のような特徴的な部品が載っています。



図1：用意するもの

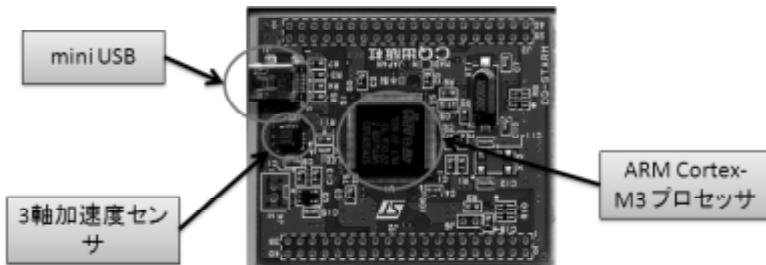


図2：付録基板についているもの

Q : DesignWave 2008年5月号の雑誌が売っていないのですが…。

A : 確かに、どこを探しても品切れが多いです。DesignWave の発行元である CQ 出版のページなどを見てみても、在庫は無い状態です（まあ、だいたいこういうのは、みんなしてまとめ買いしてすぐに売り切れてしまうものなので、在庫がないのは仕様です。かくいう私も2冊買ったことですし…。こういうものは、早めに販売情報を入手してまとめ買いするほかないでしょう）。ネットで少し探してみたところ、2008年5月22日現在、ツクモのロボット王国という

## マイコンで作る Wii コントローラもどき

ところでのみ販売されているのを確認しました。

Design Wave Magazine 5月号

<http://robot.tsukumo.co.jp/goods/4910165550585/>

あと、確認していませんが、秋葉原の書泉にも置いてあるそうです。

ただ、この記事が発行される頃には、もう既に売り切れてしまっているかもしれません。そんな場合は、「どうにかしろやゴルア！」とか、メールを書いて私の方まで送っていただければ、市販の部品を使って作る方法について記事にするかもしれません。

今回の記事では、そこら辺のことはある程度、考慮しながら書いていますので、持っていないという方でも有用な情報であるように努めています。今回使用するマイコンの使い方だけでなく、他のマイコンでも同じように使えることも一部書いてありますので、是非読んでみてください。

### コントローラの仕様

コントローラだけ作っても、何を操作するのか決めないとどうしようもないで、操作する対象を決めます。というわけで、毎度おなじみで、もうとつくの昔に飽きられていると思う「シューティングゲーム開発講座」をベースとして話を進めていきます。「シューティングゲーム開発講座って何？」という方は、何とかして WORD のバックナンバーをあさってきてもらうか、「<http://yasuharu.net/word/>」からダウンロードできるので見てみてください。

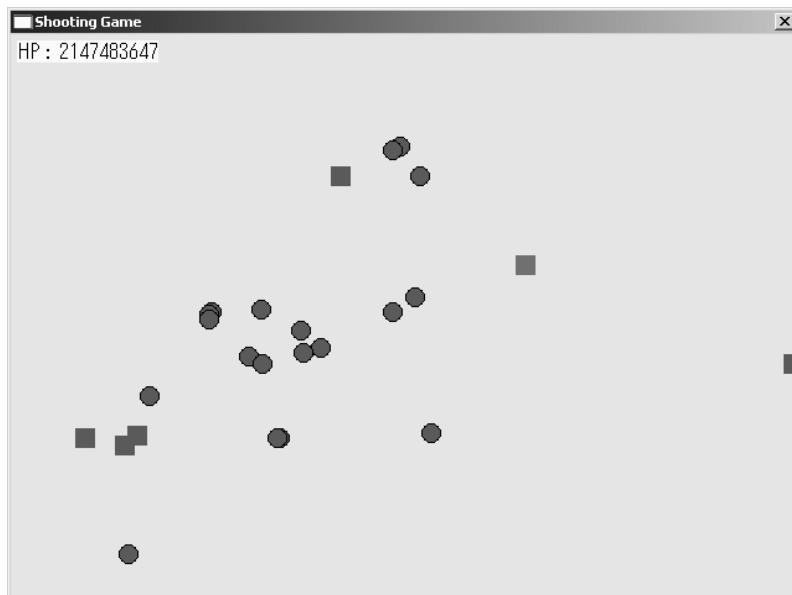


図3：シューティングゲーム開発講座で作成したもの（改造済み）

このシューティングゲームにおいて、ユーザの機体を今回作るコントローラでコントロールできるようにします。付録基板の加速度センサを使って、傾けたら傾けた方向に機体が移動するようにします。弾の発射については、基板側で制御するわけではなくて、パソコンのスペースキーで発射するようにします。まとめてみると、次の表の通りになります。

画面上の動作	コントローラでの操作
右へ移動	右へ傾ける
左へ移動	左へ傾ける
上へ移動	(操作する人から見て) 奥に傾ける
下へ移動	(操作する人から見て) 手前に傾ける
弾を発射	スペース

シューティングゲーム開発講座で開発したシューティングゲームでできる動作は、この程度なのでこれで問題ないです。あとは、これに加えて傾き具合で速さも変えられるようにしましょう。

### 作成の方針

まずは、雑誌に従ってドライバなどをインストールして、雑誌についているサンプルプログラムを動かしてみましょう。サンプルプログラムでは、付録基板を上下させることで、かかるがぴょんぴょん跳びはねるゲームが収録されています。また、雑誌上ではハイパーテーミナルで接続して、「\*」キーを押すことで加速度センサの値をマイコンが出力するようになる、と書いてあります。このことから、マイコン側のプログラミングをすることなく、付録基板から加速度の値を取ってくることができる、ということがわかります。このマイコンには、既にサンプル用のプログラムが書き込まれており、そのプログラムのおかげでこのようなことがで



図 4：ハイパーテーミナルで加速度センサの値を表示

もちろん、マイコンも Micro Computer という名前が表すとおり、小さいコンピュータです。多くの方がご存じの通り、コンピュータは、プログラムがなければただの鉄のかたまりです。本来なら、マイコンを買ってきてマイコンで動作するプログラムを書いてやる必要があります。しかし、今回は前述の通り既にサンプル用のプログラムが書き込まれており、それを利用することでコントローラを作成していきます。非常に楽ですね。

マイコン側のプログラミングをしないのでいいのであれば、「Windows のプログラムは書けるけど、マイコンはやったことがない…。」といった方でも、とっつきやすいかと思います。もちろん、マイコンでのプログラミングができるにこしたことはないので、興味がある方は是非挑戦してみてください。新しい世界が見えてくると思います。

マイコンからデータを取ってくることはできることはわかりました。そうなると、どうやってプログラム中からそのデータを読み込んでやるのか？という問題になります。マイコンとパソコンを物理的にどうやって接続しているのかを確認すると、USB を使って接続してい

## マイコンで作る Wii コントローラもどき

ます。しかし、デバイスマネージャから確認をすると仮想 COM ポートとして認識されています。



図 5：仮想 COM ポートで接続している

加速度センサの値が取得できたらしめたものです。煮るなり焼くなり好き放題に調理してやれば良いわけです。今回は、傾きを検知してユーザの機体を動かすようにするので、そのようにプログラムを書いてやります。

### COM ポートとは？

いきなり COM ポートという言葉が出てきましたが、これはいったい何なのでしょう？

COM ポート (Communication Port) とは、シリアルインターフェースの総称です。USB や IEEE1394 といったものも、シリアルインターフェースですので、厳密には COM ポートと言った場合、対象に入ります。しかし、マイコンを取り扱う場合、COM ポートと言うと RS-232C という規格のインターフェースのことを指すことが多いです。同じようにして、この COM ポートのことをシリアルポートと呼ぶこともあります。ここでは、RS-232C のことを COM ポートと呼んで説明していきます。

RS-232C 自体は、マイコンとコンピュータをつないだり、コンピュータとコンピュータをつなぐために用いられます。



図 6：RS-232C(オス)



図 7：各ポートの関係

仮想 COM ポートとして Windows と接続している場合、Windows 上では COM ポートとして見えています。なので、COM ポートを制御してマイコンからのデータの送受信をしてやればいいわけです。

それでは、Windows で COM ポートの制御を行う場合、どのように行うのでしょうか？ 方法としては、以下のような方法があります。

- Win32 API
- Microsoft Communication Control

- ・.NET Framework
- ・直接操作する

最後の方法を除いて、COM ポートの制御を行うのは簡単です。今回は、Win32 API を用いて説明をします。

### 実装しよう

仕様も決まり、マイコンとデータを送受信する方法もわかりました。それでは、実装に取りかかっていきましょう。まずは、元となるシューティングゲームのソースコードを持ってきましょう。ソースコードは以下の URL にあります。

[http://yasuharu.net/word/shooting/4/ShootingGame\\_src\\_10.zip](http://yasuharu.net/word/shooting/4/ShootingGame_src_10.zip)

ダウンロードが完了したら、ダウンロードしたプロジェクトを開いてください。

今回、コントローラで操作できるようにするにあたって、既存のコードをできるだけ修正せずに、かつ、できるだけ手間をかけずに実装していきます。これは、既存のコードに対してぐちゃぐちゃと修正を行っていくと、その修正の影響範囲が予想できない部分まで影響する可能性があるためです。また、手間をかけないというのは、こういうのは基本的にスピード勝負で、いかにして興味がなくなる前に実装し終えるかが重要だと思っています。なので、できるだけ素早く的確に実装します。

COM ポートの制御は、以下の図の手順で行います。実際に使用する関数と一緒に説明をしていきます。

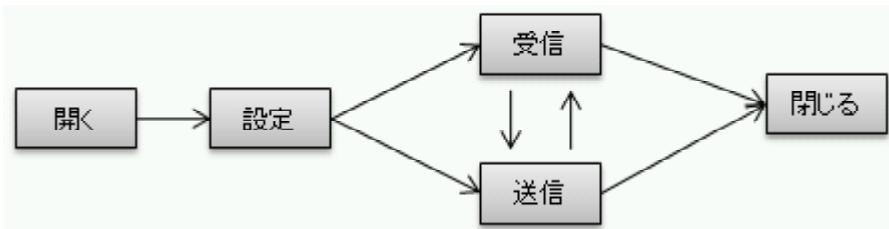


図 8 : COM ポートの制御

### COM ポートを開く

最初に、COM ポートのデバイスを開きます。このデバイスを開くためには、CreateFile 関数を使用します。CreateFile 関数自体は、ファイルを開くことに使うことが多いのですが、COM ポートやプリンタポートなどのデバイスを開く場合にもこの関数を使用します。シューティングゲームに組み込むために、COM ポートを開く関数を OpenCom 関数として定義すると以下のようになります。

```

HANDLE OpenCom (int ComNum)
{
    char ComName[12] = "";
    HANDLE hCom;
    sprintf(ComName,"%s.COM%d",ComName,ComNum);
}
  
```

## マイコンで作る Wii コントローラもどき

```
hCom = CreateFile(
    ComName, GENERIC_READ|GENERIC_WRITE,0,
    NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL
);

return hCom;
}
```

OpenCom 関数は、引数として COM ポートの番号を取ります。COM のポートは、それぞれ接続する場所によって番号が決まっており、例えば、図 5 では「COM6」となっていることがわかります。CreateFile 関数で COM ポートを開くときには、ファイル名の代わりに「¥¥¥COMn」(n は数字) という形式で、どの COM ポートを開くか、という情報を与えてやります。他の引数については、ファイルオープンの時とほとんど変わらないので、MSDN (<http://msdn.microsoft.com>) なりで調べてみてください。

戻り値として、開いた COM ポートのハンドルを返してやります。

Q：エラー処理がされていないのはなぜ？

A：すいません。本来ならハンドルが確保されているかどうかしっかりとチェックして、ハンドルが確保されていなければエラーを出すなどの処理が必要になります。例えば、CreateFile 関数であれば、MSDN を見ると以下のように書かれています。

「関数が失敗すると、INVALID\_HANDLE\_VALUE が返ります。拡張エラー情報を取得するには、GetLastError 関数を使います。」

引用元：<http://msdn.microsoft.com/ja-jp/library/cc429198.aspx>

つまり、CreateFile 関数の場合、ハンドルを取得した段階でハンドルの中身が「INVALID\_HANDLE\_VALUE」でないかどうかを調べる必要があります。今回のコードでは、ここの部分が書かれていません。実際に業務用のプログラムを書くときなど、エラーチェックをしていなければ話になりません。エラーを見過ごして処理を行ってしまうと何が起こるかわからないためです。

今回のような、遊びのプログラミングであればエラー処理をしていなくても、自分がこまるだけでいいのですが、後々の事も考えてエラー処理を書くような癖をつけておくことをおすすめします。

これは、ショーティングゲーム開発講座の方でもそうなのですが、紙面の都合上、ほとんどエラー処理を行っていませんでした。申し訳ありません。本来はこのようなプログラムは良くありませんので、各自でエラー処理を書いてもらうようお願いします。

### COM ポートの設定をする

次に、開いた COM ポートの設定を行います。COM ポートの通信速度の設定やデータの送受信時のタイムアウト時間の設定を行います。今回は、どちらも行わなくとも COM ポートの制御はできたので、COM ポートの設定について、詳細な説明はせず、関数の紹介と簡単な説明のみを行います。

まず、COM ポートでデータの送受信を行うときには、送受信を行うための通信速度やパリティの設定などをします。詳しくは説明しませんが、マイコンをいじっていると通信速度の設定をちゃんとした値に設定していなかったため、データが化けて受信されていました。

とがあります。あらかじめどの通信速度で送受信を行う必要があるのか、ということはわかりますので、ちゃんとした値に設定してやりましょう。この設定をするには、GetCommState 関数を使って、現在の設定が入った構造体の変数を取得し、必要な項目の値を変更して、SetCommState 関数で設定を変更します。

もう一つ、データ送受信時のタイムアウトの設定を行います。COM ポートのデータを送受信する場合、同期的にデータの送受信を行うと、指定されたサイズ分読み込むまで処理が送受信を行う関数の中で止まってしまいます。そのため、タイムアウトの値を設定して、一定時間経ったら強制的に送受信を停止させて、元の処理に戻るようにします。これを行うためには、GetCommTimeouts 関数を使って現在の設定が入った構造体の変数を取得し、必要な項目の値を変更して、SetCommTimeouts 関数で設定を変更してやります。

### COM ポートへデータ送信

一番最初に CreateFile 関数を使ってファイル操作と同じようにして COM ポートのハンドルを取得しました。送信や受信もファイル操作と同じようにして、COM ポートを開いたときのハンドルを使用して ReadFile 関数や WriteFile 関数を使います。

ここでは、COM ポートへのデータ送信方法について説明します。そんなに行数もないのと、特に説明なしにコードを見ていきます。COM ポートからデータを送信する関数を PutCom 関数として定義します。

```
int PutData (HANDLE hCom,char* Data)
{
    DWORD WriteByte = 0;
    WriteFile (hCom,Data,strlen (Data) ,&WriteByte,NULL);
    return WriteByte;
}
```

この関数については、特に説明をしなくとも問題ないかと思います。引数として、書き込み先の COM ポートのハンドルと書き込むデータを渡してやります。戻り値は、書き込みが完了したバイト数です。ファイルを書き込むときの操作と全く変わりません。WriteFile 関数自体の引数については、MSDN で調べてみてください。

### COM ポートからデータ受信

送信の時に説明をしたように、受信の時には ReadFile 関数を使ってデータの取得を行うと説明をしました。ただ、これだけでも動作はするのですが、あらかじめ COM ポートのバッファに溜まっているデータのサイズを調べて、その上で必要な分のデータだけ受信した方が安定するようです。

バッファに溜まっているデータのサイズを調べるために、ClearCommError 関数を使用して調べます。ClearCommError 関数に COMSTAT 構造体の変数を渡して、その変数の cbInQue 要素にバッファに溜まっているデータのサイズが格納されています。今回も同じようにして、COM ポートからデータを受信する関数を GetData 関数として定義します。

```
int GetData (HANDLE hCom,char *Data)
{
    DWORD Error;
    COMSTAT Status;
```

## マイコンで作る Wii コントローラもどき

```
    DWORD Size;
    DWORD ReadByte;

    ClearCommError (hCom,&Error,&Status);
    Size = Status.cbInQue;
    ReadFile (hCom,Data,Size,&ReadByte,NULL);

    return ReadByte;
}
```

ClearCommError 関数については、先に説明をしたとおり、COMSTAT 構造体の変数を渡してやります。ReadFile 関数も特にファイル読み込みと変わったところはありませんが、ClearCommError 関数から取得したバッファに存在するデータのサイズを ReadFile 関数の読み込むバイト数として指定しています。GetData 関数の引数には、COM ポートのハンドルと受信したデータの格納先の変数を与えてやります。また、戻り値は読み込んだデータのサイズです。

### COM ポートを閉じる

最後に COM ポートを使い終わったら COM ポートを閉じてやる必要があります。これを行うには、CloseHandle 関数を使用します。そんなに行数もないでの、特に説明なしにコードを見ていきます。COM ポートを閉じる関数を CloseCom 関数として定義します。

```
void CloseCom (HANDLE hCom)
{
    CloseHandle (hCom);
}
```

関数に関する説明は、特にすることはないので、わからなかつたら調べてみてください。開いたポートは閉じないと Windows が開いたままだと認識して、他のアプリケーションから COM ポートが開けなくなってしまいます。そのような場合は、一度、再起動するしかないようです。

### シューティングゲームの処理に手を加える

COM ポートの制御に関する長々しい話はここで終わりです。これまで説明してきた関数を使って、COM ポートから取ってきた加速度センサの値から、ユーザの機体を操作する部分について説明をします。

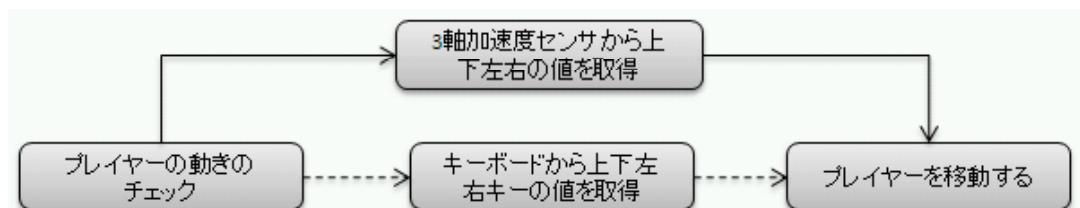


図 9：キーボードから 3 軸加速度センサへ

## マイコンで作る Wii コントローラもどき

先ほどの図を見てください。破線部が今までのシューティングゲームの構造です。プレイヤーの動きのチェックという動作の中で、キーボードから値を取得して、その値からプレイヤーの移動を行っています。これを 3 軸加速度センサの値から上下左右のキーボードが押されたときと同じような動作としてやれば、周りにできるだけ影響を与えずに実装してやることができます。つまり、キーボードをそのままそっくり 3 軸加速度センサに置き換えてやればいいわけです。ただし、キーボードからの入力は ON / OFF のどちらかで情報が戻ってきますが、3 軸加速度センサの場合にはそのようにはいかないので注意が必要です。

まずは、先ほど示した関数をソースコード中に記述します。今回は、ソースコードがつらつらと並んでいるだけでは、疲れるだけなので、最終的に完成したものを Web 上で公開するようになります。どこに記述するか、ということはそちらを参照してください。

次に、COM ポートを取り扱うためのハンドル用の変数をウインドウプロシージャの先頭の方で static として宣言します。

```
static HANDLE hCom;
```

COM のハンドルは、ウィンドウが初期化された時に一緒に開きます。これは、WM\_CREATE の部分の最後に書いてください。

```
hCom = OpenCom(6);
PutData(hCom,"*");
```

COM ポートの番号は、各自の環境に合わせるようにしてください。その次に送っている "\*" は、加速度センサの値を送信するようにマイコン側に命令をします。これは、最初の方に説明をした通り、加速度の値をマイコンから出力させるようにするための命令です。

そして、一番大きいプレイヤーの移動を制御している関数の修正です。これは、いくつかに分けて説明をしていきます。それでは、見ていきましょう。

```
void PlayerMove(Player *mPlayer,HANDLE hCom)
{
    int Key_Left = 0,Key_Right = 0,Key_Up = 0,Key_Down = 0;
    char Data[4096];
    int Read;
    int Ret;
    int x,y,z;
    int d = 50;
    int c = 4;

    Read = GetData(hCom,Data);
```

まずは、関数の定義と変数の宣言です。Data のサイズが大きいのは、GetData 関数の中でどれだけのサイズが入るかわからないので、あふれることがないように多めにとってあります。實際には、GetData 関数内で上限を設定するなどして、バッファが溢れないようにするための仕組みが必要でしょう。最後の方の d や c は、それぞれ閾値の範囲と移動時の係数を保持しています。これについては、後から詳しく説明をします。

最後の部分では、Data 変数の中にマイコンから受信したデータを入力しています。

```
Ret = sscanf(Data,"%d,%d,%d",&x,&y,&z);
```

## マイコンで作る Wii コントローラもどき

```
if(Ret != 3)
{
    return;
}
```

マイコンから受信したデータの形式は、図 4 でデータを受信している様子を見るとわかるのですが、「X 軸の加速度,Y 軸の加速度,Z 軸の加速度」という形式で入力されます。この入力されたデータを `sscanf` 関数を使って、それぞれの要素に分解してやります。`sscanf` 関数は `scanf` 関数と同じようなもので、書式に従って分解してほしい文字列を引数に渡すことで、指定した書式の通りに分解してくれます。

ただ、もし、それぞれの軸の値が正常に取れていなくて、各加速度を保持するための変数に値が入力できないことを考えて、`sscanf` の戻り値をとっています。`sscanf` の戻り値では、正常に入力ができた変数の数が戻ってきます。これを使うことで、仮に、3 個の変数にそれぞれ値が入っていなかった場合、問題が発生してしまうのを防いでいます。

```
if(x <= 2048 - d)
{
    Key_Left = -1;
}

if(2048 + d <= x)
{
    Key_Right = -1;
}

if(y <= 2048 - d)
{
    Key_Down = -1;
}

if(2048 + d <= y)
{
    Key_Up = -1;
}
```

それぞれの加速度センサから加速度が取れたら、それぞれの加速度の値を見て、どのキーが押されているか、ということを判断します。これについては、詳しく説明をします。

まず、3 軸加速度センサが水平にして机に置かれた場合、どう反応するか、というのを見てみます。そのときの結果は、 $(gx,gy,gz) = (2047,2047,2866)$ 付近を示すはずです。どうしてそうなるのか、というのは雑誌の説明を見てもうとして、ここでは、コントローラの実装に必要な部分について話していきます。この加速度の値ですが、値の範囲は、0 ~ 4095 の範囲で加速度の値としてマイコンから出力されています。また、加速度の範囲は、サンプルのプログラムでは、 $\pm 2G$  の範囲となっています。ここで重要なこととして、重力がかかっていない状態は、2047 付近であり、1G の重力がかかっているとき 2866 付近を示す、ということが重要です。のことから、中央値が 2047 で、-1G だったら 2047 - (2866 - 2047) など、だいたいの予測がつくかと思います。

ところで、それぞれの軸はどのように設定されているのでしょうか？ これは、実際に動かしてみて調べた方が早いので、実際に調べてみました。それぞれの軸は、以下のように設定

されています。左の方がマイコンを真上から見た図、右の方が、横から見た図です。

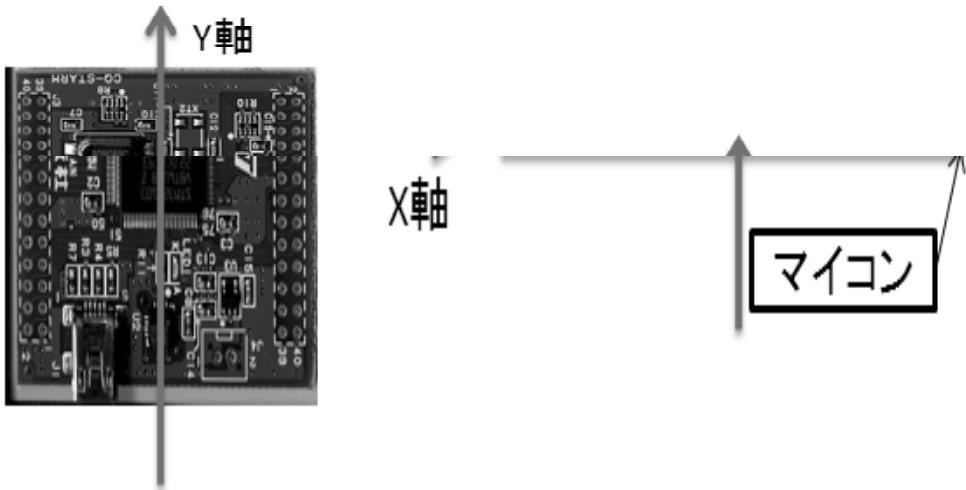


図 10：設定されている軸

このような軸に対して、机の上に水平に置いた状態を考えると、次の図のようになります。この時、Z 軸方向には重力がかかり、常に 1G 分の加速度がかかっていることになります。また、X 軸、Y 軸の方向には、加速度はかかっていません。なので、水平に置いたときは、 $(gx,gy,gz) = (2047,2047,2866)$  の付近となるわけです。

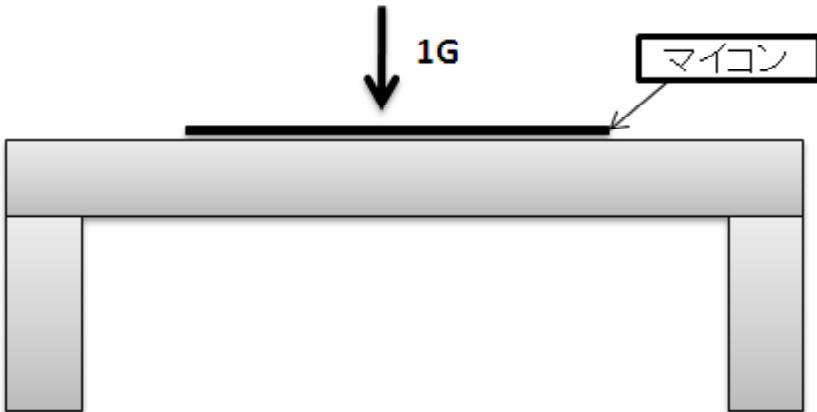


図 11：机に置かれたマイコン

それでは、コントローラを作成するにあたって、この加速度をどのように使用するか、ということを考えてみましょう。コントローラの動作としては、傾けるということを操作の基本としています。ですので、次のように考えることができます。

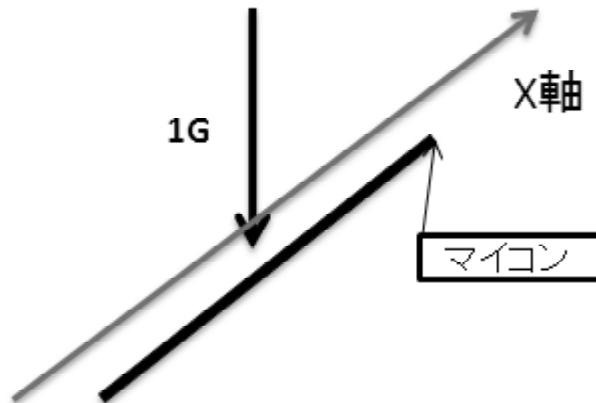


図 12：左に移動するために傾いたマイコン

例えば、上の図のように左方向に傾けて、左に進む場合を考えてみます。この時、X 軸の値だけを考えれば、重力が及ぼす加速度により、X 軸の加速度は加速度がかかっていないときより減少しています。逆に、右に傾けるとこの加速度の値は、増加します。従って、次のように考えることができます。

今の X 軸の加速度を見て、(無重量状態の加速度 - 一定の閾値) > X 軸の加速度だったら、左にキーが押されていると判断する。

この時、閾値というのは、誤差を丸めるためのものです。どういうことかというと、実際にマイコンを触ってみてもらって、動かしてみて、加速度の値を見てみます。すると、2047付近の値より少し正か負にずれていたり、水平においても値が時々刻々と変化する、というようになります。そこで、このようなズレや変化を誤差と見て、閾値を設定してやります。この閾値が先ほど変数宣言の時に設定した d の値です。

今の条件で考えてみると(2047 - 50) < z となれば、キーが押されていることとするわけです。同じようにして、右、上、下もそれぞれ設定します。これが、先ほど示したソースに関する説明です。

それでは、実際にユーザの機体を動かすところを見ていきます。

```

if(Key_Down < 0 && Key_Right < 0)
{
    SetXToPlayer(mPlayer,mPlayer->Position.x + MOVE_POINT_NANAME * c * (x - 2048) / 2048);
    SetYToPlayer(mPlayer,mPlayer->Position.y + MOVE_POINT_NANAME * c * (2048 - y) / 2048);
    return;
}
if(Key_Down < 0 && Key_Left < 0)
{
    SetXToPlayer(mPlayer,mPlayer->Position.x - MOVE_POINT_NANAME * c * (2048 - x) / 2048);
    SetYToPlayer(mPlayer,mPlayer->Position.y + MOVE_POINT_NANAME * c * (2048 - y) / 2048);
    return;
}
if(Key_Up < 0 && Key_Right < 0)
{
    SetXToPlayer(mPlayer,mPlayer->Position.x + MOVE_POINT_NANAME * c * (x - 2048) / 2048);
    SetYToPlayer(mPlayer,mPlayer->Position.y - MOVE_POINT_NANAME * c * (2048 - y) / 2048);
    return;
}

```

```

{
    SetXToPlayer(mPlayer,mPlayer->Position.x + MOVE_POINT_NANAME * c * (x - 2048) / 2048);
    SetYToPlayer(mPlayer,mPlayer->Position.y - MOVE_POINT_NANAME * c * (y - 2048) / 2048);
    return;
}
if(Key_Up < 0 && Key_Left < 0)
{
    SetXToPlayer(mPlayer,mPlayer->Position.x - MOVE_POINT_NANAME * c * (2048 - x) / 2048);
    SetYToPlayer(mPlayer,mPlayer->Position.y - MOVE_POINT_NANAME * c * (y - 2048) / 2048);
    return;
}
if(Key_Left < 0)
{
    SetXToPlayer(mPlayer,mPlayer->Position.x - MOVE_POINT * c * (2048 - x) / 2048);
    return;
}
if(Key_Right < 0)
{
    SetXToPlayer(mPlayer,mPlayer->Position.x + MOVE_POINT * c * (x - 2048) / 2048);
    return;
}
if(Key_Up < 0)
{
    SetYToPlayer(mPlayer,mPlayer->Position.y - MOVE_POINT * c * (y - 2048) / 2048);
    return;
}
if(Key_Down < 0)
{
    SetYToPlayer(mPlayer,mPlayer->Position.y + MOVE_POINT * c * (2048 - y) / 2048);
    return;
}
}

```

各方向に動かすときに動かし方の度合いによって、移動する量を変更しています。例えば、「 $(2048 - x) / 2048$ 」その方向に動きうる量の割合を取って、加速度が一番かかっているときに一番速くなるようにしています。また、c という定数がありますが、これは、割合に対する係数です。割合だと最大値が 1 となってしまうので、一番加速度がかかっているときでも、今まで通りの速度しか出ないことになってしまいます。それを補正するために係数 c をかけてあります。

最後に、PlayerMove 関数の呼び出しを変更します。これは、関数の引数の定義が変わっているためです。

PlayerMove (&mPlayer,hCom);

### 完成

以上の説明で Wii コントローラもどきの完成です。実際に動かしてみましょう。どうでしょうか？ いろんな方向に傾けてみてその方向に意図した通りに動いているでしょうか？

紙面上だと、最終的にどのような結果になるのか、ということをお見せすることができませんが、完成して、実際に動かしてみると、キーボードで操作するのとは違って、傾けるだけで操作することができるので、非常に面白いです。また、傾けて敵をよけようとするには、結構テクニックが必要です。

## マイコンで作る Wii コントローラもどき

### まとめ

今回は、DesignWave の付録基板を用いて、実際に手にとって動かせるようなプログラムを作成してみました。ハードウェアとソフトウェアを組み合わせることで、ソフトウェアだけではコンピュータ上の画面で動いているだけのものが、手にとって動く様を見ていると、非常に面白いです。

もし、この記事を読んで興味を持たれましたら、マイコンによるプログラミングについて挑戦して見てみましょう。最近では、雑誌に付録基板がついていたり、そうではなくても、インターネットなどを通じて、部品の一つ一つを購入して組み上げることも可能かと思います。

雑誌の付録基板に関しては、CQ 出版社が発行している「トランジスタ技術」という雑誌の 8 月号に付録基板がついてきて、その基板は、USB のポートが基板自体に実装されていて、そのまま挿すだけで使えるようになっています。比較的手頃な値段で購入することができるのでも、試しに買ってみるのもありでしょう。

いずれにしても、興味を持った技術についてどんどんいじってみることをおすすめします。

今回の記事については、こちらでも公開します。また、記事中に出てきたソースコードについてダウンロードできるようになっています。

Web ページ: <http://yasuharu.net/word/controller/>

間違いの指摘などありましたら、以下の連絡先まで遠慮無く連絡してください。

メールアドレス: [word@yasuharu.net](mailto:word@yasuharu.net)

# 1時間ではじめるFlex

文 編集部 ろんず

## Flash

WORD 読者の情報科が専門学類生なら、様々な WEB ページで Flash メディアコンテンツを見かけることがあると思います。



Flash メディアコンテンツの例<sup>\*1</sup>

図にあるようなコンテンツは AdobeFlash などのソフトを利用して開発されています。Flash を用いると、インタラクティブな WEB ページを作ることができます。どちらかというとデザインとかそっちの方面の島のように感じるところが強いです。

## Flash から Flex へ

その根本となる、Flash 自体はすばらしい技術で、プラットフォームにあまり依存せずに作り手の意図する挙動をさせることができます。

この点は Java アプレットなど似ています。Flash には、Java アプレットに対する JavaVM のように、AVM<sup>\*2</sup> が存在します。AVM は FlashPlayer として配布されています。Flash ファイル (SWF ファイル) の実体である ActionscriptByteCode (ABC) は、この VM 上で実行されることになり、OS は抽象化されます。つまり、クライアントの環境を意識せずにコーディングすることができます。

また Flash は、クライアントアプリケーションのような操作性の高いユーザインターフェースを持つ WEB アプリケーションの開発によく利用されます。もともとマルチメディア用途を意識して開発されているだけあって、容易に見栄えするインターフェースを作ることができます。

最近では、WEB アプリケーションの付加価値として使いやすさが求められるようになりました。このような Flash の持つ性格は、そのような需要とあいまって WEB アプリケーションとして採用される例が、見られるようになりました。

しかし、ご存知の方も居られるとおもいますが、Flash コンテンツの作成は「AdobeFlash」などのツールを用いて行います。余分な機能もついているので、エディタとコンパイラがあれば十分なブ

\*1 <http://www.iosysos.com/>より引用

\*2 ActionScript Virtual Machine 最新は version2

## Flex 入門一時間目

プログラマにとっては使いにくいのではないかと思います。それ以前に**価格が高い**等の問題もあります。

そこで、登場したのは Flex です。Flex は、プログラマ向けのアプリケーションフレームワークで、コンパイラが無料で公開されています。Adobe の WEB ページ<sup>\*1</sup>では、Flex は以下のように説明されています。

ブラウザ、PC、OS の違いを超えて、同一のユーザエクスペリエンスを提供できる、表現力豊かな Web アプリケーションを開発・管理するための、開発効率の高いオープンソースフレームワークです。

### Flex で何ができるのか

では Flex で何ができるのかというと、

<http://examples.adobe.com/flex3/componentexplorer/explorer.html>

<http://examples.adobe.com/flex2/consulting/styleexplorer/Flex2StyleExplorer.html>

にサンプルがあるので、見ればなんとなくわかると思います。

競合する技術として Ajax がありますが、Ajax のできることは結構 Flex でできたりしますし、機能によっては Flex のほうが高性能です。ただ、WEB ブラウザと連携する機能自体は Flex にはありませんので、Javascript との連携で補わなければならない部分もあります。そこで、Javascript と連携するための機能も実装されています。

もちろん、マルチメディアコンテンツも扱えるので、ストリーミングができたりと機能はいろいろ。

また、つい先日公開された FlashPlayer10 β では 3D や GPU がサポートされ、今後いろいろと遊べそうです。

以下、この記事は Windows で Flex を開発することを前提に書いています。利用する Flex の Version は 3 ですので、対応する言語として ActionScript3 を元にソースコードを記載しています。

### コンパイラのインストール

では早速、コンパイラを入手しインストールします。今回は FlexSDK3 を用います。コンパイラは以下の URL からダウンロードできます。

<http://www.adobe.com/cfusion/entitlement/index.cfm?e=flex3email>

ページの下のほうにある「I have read the Adobe Flex 3.0 SDK License, and by downloading the software listed below I agree to the terms of the agreement.」にチェックをして、「Download the Flex 3.0 SDK for all Platforms」をクリックすれば OK です。

次にダウンロードしてきたファイルを展開します。ファイルは「flex\_sdk\_3.zip」となっていると思います。

展開するとその中身に「bin」フォルダがあり、その中にある「mxmlc.exe」がコンパイラです。これを直接起動することでコンパイラが起動します。

さすがにこのままだと不便なので、環境変数の PATH を通しておくといいでしょう。

---

\*1 <http://www.adobe.com/jp/products/flex/>

### Flex のコーディング

Flex アプリケーションを作成するときには、二種類の言語を用います。「MXML<sup>\*1</sup>」と「ActionScript」です。

MXML は XML で、Flex アプリケーションのユーザインターフェースを記述できます。

MXML タグは ActionScript のクラスまたはそのプロパティに相当し、コンパイル時に解析され、対応する ActionScript クラスが生成されます。その後、これらの ActionScript クラスがコンパイルされて ActionscriptByteCode が生成され、SWF ファイルに保存されます。

つまり、MXML は ActionScript では複雑になるコードを簡単に記述できるものです。

基本的に、MXML でインターフェースなどの見た目を、ActionScript3 でボタンが押されたときなどの動作を記述します。

### Hello, Word

では早速こんなコードを書きます。このコードは画面上に Hello, Word というラベルのボタンを表示するものです。

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
  <mx:Button id="wordBtn" label="Hello, Word!" />
</mx:Application>
```

これをファイル名「HelloWord.mxml」として保存します。その後、このファイルをコンパイルします。

これで ActionscriptByteCode が生成されました。生成されたファイルは「HelloWord.swf」です。

```
$mxmlc HellowWord.mxml
Loading configuration file C:\FlexSDK\frameworks\flex-config.xml
HelloWord.swf (150407 bytes)
```

後はこれを FlashPlayer などで開くと、以下のようになります。



\*1 Macromedia Flex Markup Language

## Flex 入門一時間目

### イベント処理

先ほどの HelloWord は、ボタンを押したときのアクションなどはまったく記述されていません。そこで、次にボタンを押したときにアラートが出るようにしてみます。

Flex はイベント駆動型アプリケーションです。そこでボタンがクリックされたイベントを検知するようにイベントハンドラを定義します。Flex のイベントハンドラには二つの記述方法があります。

### **MXML を用いたイベントハンドラの記述**

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
    <mx:Script>
        <![CDATA[
            import mx.controls.Alert;
            public function clickHandler(event:Event):void{
                Alert.show("おされた～");
            }
        ]]>
    </mx:Script>
    <mx:Button id="wordBtn" label="Hello, Word!" click="clickHandler(event)"/>
</mx:Application>
```

### **ActionScript を用いたイベントハンドラの記述**

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" creationComplete="init() ">
    <mx:Script>
        <![CDATA[
            import mx.controls.Alert;
            import flash.events.MouseEvent;
            public function init():void{
                wordBtn.addEventListener(MouseEvent.CLICK, clickHandler);
            }
            public function clickHandler(event:Event):void{
                Alert.show("おされた～");
            }
        ]]>
    </mx:Script>
    <mx:Button id="wordBtn" label="Hello, Word!" />
</mx:Application>
```

どちらのコードも、ボタンがクリックされたとき clickHandler メソッドを呼び出します。この時、Event オブジェクトを引数に渡します。イベントによって Event オブジェクトから取り出せるイン

スタンスは異なります。クリックイベントの場合はクリックされたとき、Alt キーや Ctrl キーが押されていたかなど情報を取り出すことができます。

`creationComplete` というイベントハンドラは、MXML コンポーネントがすべて初期化された後に発生する「`mx.events.FlexEvent.APPLICATION_COMPLETE`」を検知することができます。Script タグで MXML で記述されたクラスに対して操作する場合は、クラス（この場合 `wordBtn`）が初期化されていなければならぬため、`creationComplete` から `init` メソッドを呼び出し、そこでイベントハンドラを定義します。

MXML に直接イベントハンドラを定義すると、インターフェースを記述しながらイベント動作を記述することができますが、コードの分離をすることができません。一度にたくさんのイベントハンドラを定義する場合は ActionScript でイベントハンドラを定義するほうがよいです。

コードの分離ですが、Script タグは以下のようにするとその中身を分離することができます。

```
<mx:Script source="HelloWordAction.as" />
```

Script タグの中身を `HelloWordAction.as` というファイルに分離しました。

### 終わりに

今回はここまでですが、次回からはいろいろと遊んでいこうと思います。

Flex の詳細については、AdobeFlex3 リファレンスガイドを見ると良いと思います。非常に詳しく記述されているので巷の本を買うより有益です。

AdobeFlex3 リファレンスガイド  
[http://livedocs.adobe.com/flex/3\\_jp/langref/](http://livedocs.adobe.com/flex/3_jp/langref/)

## オーディオべーしっく？

### オーディオべーしっく？

文責 かづきお

## そもそも、なぜオーディオか

### オーディオなんて今時流行らねえよ！

そんなことを言われても、好きな物は好きなのですから、どうしようもないじゃないですか。  
どこぞの少佐風に言えば、

「諸君 私はオーディオが好きだ 諸君 私はオーディオが好きだ 諸君 私はオーディオが  
大好きだ

スピーカーが好きだ セパレートアンプが好きだ レコードプレーヤーが好きだ バックロードホーンが好きだ ドームトゥイーターが好きだ インシュレーターが好きだ

大音量で 小音量で 町中で オーディオルームで ヘッドフォンで スピーカーで  
この地上に存在する ありとあらゆるオーディオが好きだ

バックロードホーン特有のスピード感あふれる低音が好きだ

雄大なオーケストラの中でティンパニーの音が鳴った瞬間など心がおどる

ESコーンならではの生々しいヴォーカルが好きだ

宇多田ヒカルの FirstLove を聴いたときなど思わず拍手をしてしまうほどだ  
限定ユニットが好きだ

年月がたちすぎて修理ができないと言われたときはとても悲しい

スピーカーで大音量で聴くのが好きだ

出先でポータブルプレーヤーで圧縮音源を聴くのは屈辱の極みだ

諸君 私はオーディオを 地の底から響くような低音を望んでいる

(ry 」

といった具合でしょうか。この感情を少しでも共有できたら幸いです。



お気に入りのレコード盤

そもそも、オーディオと言うと変な誤解をされることが多いのですが、私は「好きな音楽を、好きなように聴くこと」と定義しています。その意味で考えればでかいスピーカーや高性能アンプなどが必ずしも必要なわけではありません。iPodのような携帯型プレーヤーにイヤフォン、後は好きな曲のソースがあれば立派にオーディオできます。

しかし、私はあえて大きなスピーカーを並べ、大音量で聴きます。それはなぜか。そうやって聴くことが私は好きだからです。

しかも、私の楽しみはそれだけではなく、スピーカーやアンプを作ったりもしています。むしろ、そういった工作と、自作コンポーネントで聴く好きな音楽が私のオーディオの大部分を占めているのかもしれません。

しかし、先に言ったことと矛盾するようですが、やはり良い音で聴くことはオーディオを楽しむことには必要不可欠ではないでしょうか。

良い音で聴くと言っても、音を良くする方法は沢山あります。そもそも、良い音の定義が人それぞれなので良い音などと一概に言えないことは確かです。そこで私は、手を入れたときに一番変化がわかりやすい場所、音の出口であるスピーカーやイヤフォン、ヘッドフォンを変えてみることをおすすめします。ポータブルプレーヤーのイヤフォンを交換して楽しんでいる人で、交換したときの音の変化に驚いたという人は少なくないと思います。

しかし、私はスピーカー派の人間です。そこで、スピーカーについてこれから暑く語りたいと思います。

### スピーカーの良さ

スピーカーの良さとは、空気感。これに他なりません。音は耳で知覚しますが、空気の震えは体全体で体感することができます。映画館で映画を見たときの爆発シーンの衝撃を思い浮かべてもらえるとわかりやすいと思います。これはイヤフォンには絶対にありません。



B&W のスピーカー「805S」

次に、音の像。私がイヤフォンよりスピーカーを好む一番の理由はこれです。私は主に女性ヴォーカルを好んで聞くのですが、これをイヤフォンやヘッドフォンで聴いた場合、頭の中で歌っているような感覚に襲われた経験があるのは私だけではないと思います。スピーカーだと、うまく配置をすれば目の前に歌手が実在するかのように音の像を造ることができます。また、オーケストラなど音源の多い曲を聴いた場合などでも、スピーカーだとどこでどの楽器が鳴っているか判るほどの像を造ることもできます。

確かにスピーカーは場所をとり、鳴らせる場所、時間を選びますが、その欠点を補ってもあまりある良さがあると思います（イヤフォンにはイヤフォンの良さがあり、それを否定しているわけではありません。イヤフォン好きの人たち、ごめんなさい）。

### じゃあ、スピーカーをアップグレードしよう

イヤフォンやヘッドフォンをアップグレードしたことがある人は多いと思います。しかし、ス

## オーディオベーしっく？

ピーカーをアップグレードしたとこのある人は少ないのではないでしょうか。しかし、単品コンポーネント<sup>\*1</sup>でなくともスピーカーの交換できるプレーヤーは多いのです。ミニコンポなどはほぼ確実に交換ができます。交換ができるということは交換をしろということです（乱暴ですが）。

では、どんなスピーカーと交換すればよいか。乱暴な言い方をしてしまえば、すべてのパッシブスピーカー<sup>\*2</sup>と交換することができます。数千円のものから数百万円のものまでスピーカーの種類は多種多様ですが、すべてが候補になります。

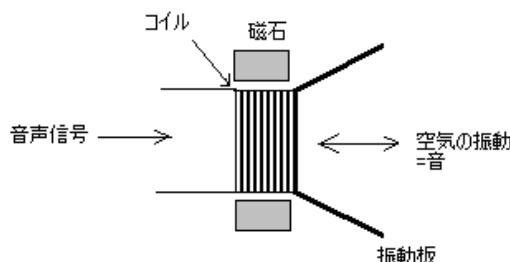
そう言われても困ってしまうので、私がお勧めするのは、自作スピーカーです。なぜそんなに種類があるのにわざわざ作るかというと、そのほうが面白いからです。皆さんだってPCを自作しませんか？それと同じです。さらに、今自作PCを作ろうとしても、一昔前のように自作の方が安いというわけにはいかなくなりつつあります。DEOLやh○が売っているPCは自作では太刀打ちできないほど安いものもあります。しかし、自作スピーカーは違います。メーカー製よりも安くて良いものができると私は断言します。

まあ、自作といってもユニットと呼ばれる振動する部分は作るわけにはいかないので（学研からユニットを作るセットが売られていますが）ユニットは市販のものを使います。自作するのはそれを入れる箱（エンクロージャーという）です。

たかだか箱を作ったくらいで、自作とはおこがましい、と思われるかもしれません、スピーカーで一番重要なのはこの箱、エンクロージャーなのです。同じユニットを使っていても、箱の設計で全然違う音が出ます。なぜかというと、スピーカーの周波数特性<sup>\*3</sup>はエンクロージャーによって決定されるからです。周波数特性が変われば、簡単に言ってしまえば、違う音が出ます。

## スピーカーを作ろう

さて、いよいよスピーカーを作るわけですが、その前に、スピーカーとは何ぞやという話をしたいと思います。簡単に言ってしまえば電気信号を空気の振動に変換する装置です。まあ、このことは言わずもがな、誰でも知っていることだと思いますが、その仕組みを知っている人は少ないのではないかと思います。その仕組みといつても、そんなに大げさなものではなく、簡略化して説明すると、その構成要素は、磁石と、コイル、振動板と呼ばれる一種の紙から成ります。



円形の磁石の中にコイルを配置し、そこにアンプで増幅された電流を流します。すると、電磁力の作用で、コイルは力を受けます。そのコイルに、振動板を取り付けておけば、振動板が振動し、空気の振動が発生するというわけです。この、磁石とコイルと振動板をひとくくりにしたシステムが、スピーカーユニットと呼ばれるものです。

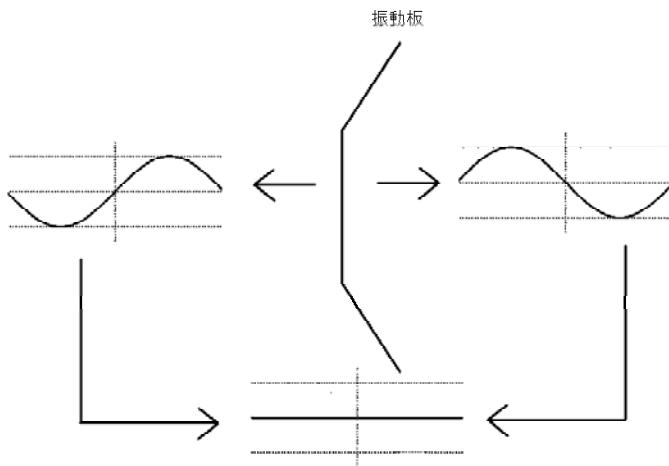
では、エンクロージャーは何をするかというと、一言で言うと、音を遮断します。どんな音を遮断するかというと、振動板の裏から出る音を遮断

\*1 単品コンポーネント アンプ、プレーヤー、スピーカー、チュナーなど役割ごとに筐体が分かれているもの。

\*2 パッシブスピーカー スピーカー側に増幅器（アンプ）を持たないもの。スピーカー側に増幅器を持つものはアクティブライドスピーカーといいます。

\*3 周波数特性 どの周波数の音がどのレベルで出ているかを調べたもの。f特とも。

します。振動板はとても薄くて軽い紙からできている場合が多いのですが、これが振動すると表と裏で逆位相の音波が発生します。この逆位相の音波は、互いに干渉し合い、打ち消し合います。



特に、周波数の小さい低音域は回り込みしやすいので、表から出る音も裏へ届き、裏もまた然りです。この結果、ほぼ中音域から下の音は消えてしまします。それを防ぐために、エンクロージャーと呼ばれる箱を作り、振動板後ろ側の音を箱の中で消すことにより表から発生する音のみを聞こえるようにします。

エンクロージャーも、種類はいくつか存在し、ただ単に後ろ側に箱を取り付けるだけの密閉型、今回作る予定のバスレフ型、箱ではなく、大きな一枚板で前と後ろの音を分離する平面バッフルなんというものもあります。

その中で、今回作る予定のバスレフについてですが、これは、市販のスピーカーの中で一番よく見かけるタイプのエンクロージャーで、密閉された箱に、ダクト、またはポートと呼ばれるパイプやスリットが付いているタイプです。このダクトの役割は、振動板の裏側から発生する音を、共鳴させることで増幅することです。これによって、密閉型では殺されるだけだった振動板の裏側で発生した音を、有効利用できるというわけです。

さて、長々と話をしたところで、今回作るスピーカーの話をします。

今回のコンセプトは、「ミニコンポのスピーカーに置き換わる、小音量でも美しく響くスピーカーです」。

ここだけの話ですが、このコンセプトは今決めました。fostex というメーカーの FE83E というユニットを使った小型バスレフを作ろうと思い立って、作ったものがそういったコンセプトのものになったという、行き当たりばったりなスピーカーです。ごめんなさい。m(..)m

fostex は、業務用の録音機器を手がけるメーカーとしてご存じの方もいるかと思いますが、スピーカーユニットのメーカーとしてもその分野ではかなり有名で、このメーカーの OEM をつかったスピーカーも数多く存在します。

ほかにもユニットメーカーはあるのですが、今回は fostex 製を使います。理由は、私がこのメーカーの音が好きだからです。

まずはユニットを決めます。ユニットと一口に言っても、その種類は多々ありますが、今回はフルレンジと呼ばれる種類のものを使います。フルレンジは、その名の通り、低域から高域まですべての音を再生することができます。ほかにユニットの種類としては、ウーファー、スコーカー、トゥイーターなどがあります。しかし、これらは相互に組み合わせて使うことが前提であり、ネットワークと呼ばれるフィルターをかませた配線をしなければいけないので今回は使用しません。

フルレンジにもいくつか種類がありますが、あまり大きなものは置く場所がないので今回は直

## オーディオベーしっく？

径 8 センチのものを使います。fostex の 8 センチユニットは 3 種類あって、型番でいうと FE83E、FE87E、FF85K です。値段はそれぞれ一個 3000 円前後です。この中で FE87E は AV 用の防磁型なので除外するとして、今回は FE83E を使いました。FF85K は次回（次回があればですが）使いたいと思います。



FE83E

エンクロージャーも、今回は出来合いのものを使います。自作じゃないじゃないか！ という方、ごめんなさい。本来は設計図を引いて、板を切って、組み立てるべきなのですが、誰にでもできるということを考え、出来合いのものにしました。買って来たのはコイズミ無線オリジナルモデル。BB-16 です。このエンクロージャーには、直径 2 センチ、長さ 5 センチのダクトが付いています。

かかった費用は、

FE83E 2,580 円 × 2

BB-16 2,800 円 × 2

計 10,760 円

さて、ユニットもエンクロージャーもそろいました。後は組み立てるだけです。今回のものはエンクロージャーにターミナルと呼ばれる接点があらかじめ付いていて、さらに内部配線用のケーブルにファストン端子まで付いているので半田付けも不要。ドライバーが一本あれば完成です。

あらかじめユニットに付いていたウレタン製の輪をケーブルに通しておき、ケーブルをユニットの端子に差し込むだけです。赤い色が付いている端子に、赤い色の付いているケーブルを接続します。後はねじでユニットを固定するだけ。

ユニットを固定するときに注意することは、なんといってもユニットに傷をつけないこと。手が滑ってユニットにザクッとやってしまったときは 3 日間はブルーになります（私は経験ありませんが）。あらかじめ下穴を開けてからやるといいでしよう。

ユニットを取り付け終わったら完成。どうですか？ 簡単でしょ。



ユニットに内部配線を結線

## いよいよ音出し

さあ、いよいよ音を出してみましょう。ケーズデンキや石丸電気にいけばオーディオケーブルが手に入ります。専用の高いものでなくてもかまいません。というか、高いものは1メートル十萬円を超えるケーブルもあるそうです。しかし、そんなケーブルは必要ないです。

接続方法は、今までのスピーカーがつながっていた様に接続するだけです。具体的には、赤い端子同士、黒い端子同士を接続します。プラス、マイナスと書いてあつたらプラスをスピーカーの赤い方に接続しましょう。

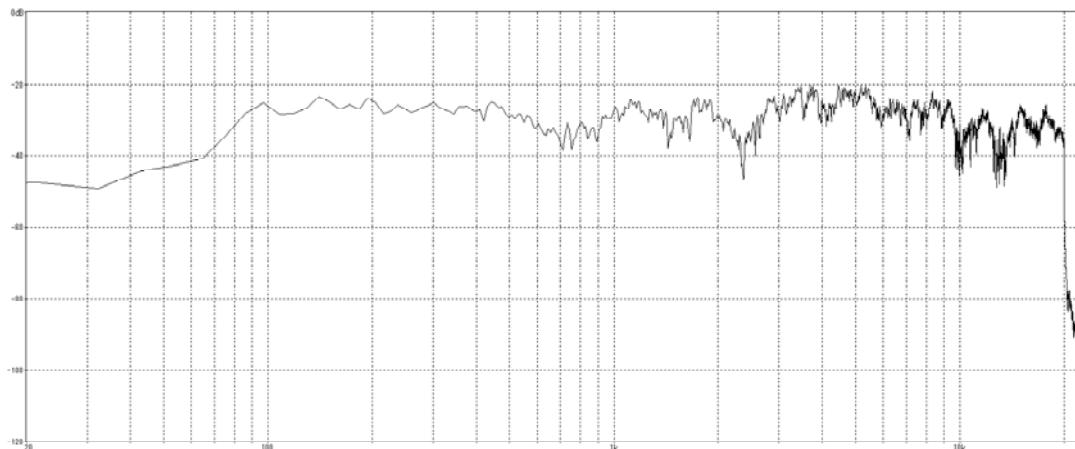
さあ、音を出します。好きな曲を好きな音量で流してみてください。ユニットが8センチなのであまり大きな音量は出せませんが、常識的な音量であれば十分再生できます(私のような爆音好きは例外とします)。



完成したスピーカー

今回作ったスピーカーの周波数特性を載せておきます。これは、私の部屋の騒音源を止めた状態で、ノートPCとSkype用のマイクで測ったものです。音源はオーディオテスト用のCDに収録されている、20Hzから20KHzまでのスイープ音です。本当はもう少しきちんとした機材で測りたいところですが、予算の都合上いま手元にあるものを使いました。

音源は20Hzから20KHzまで同じ強さで音が入っているので、特性グラフがフラットに近いことが正確な音が再生できていることの一目安になります。



ちょっと見にくいですが、横軸が周波数。右に行くに従って音が高くなります。ただ、縦に入

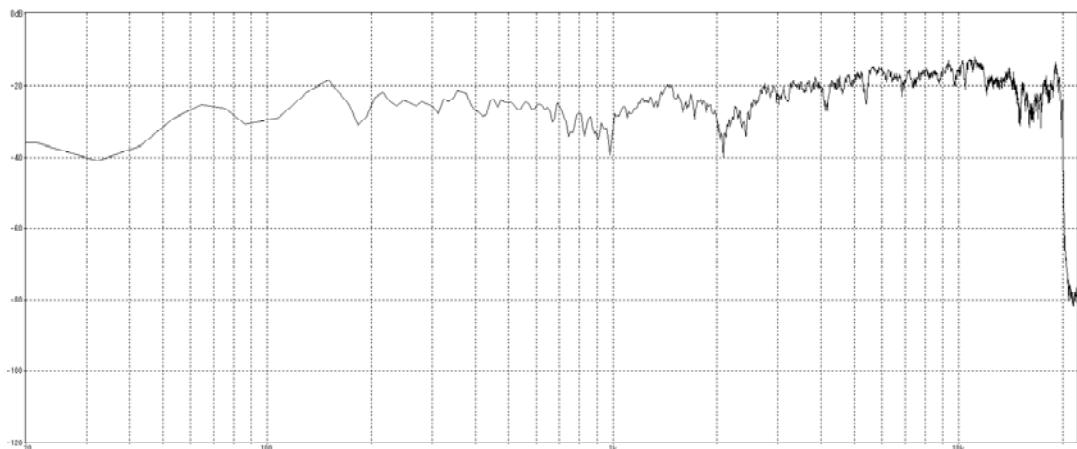
## オーディオベーしっく？

っている線をみると判るとおり間隔が一定でないので注意が必要です。横軸の目盛りは左から 20Hz、100hz、1KHz、10KHz です。

評価ですが、35Hz 以下と 20KHz 以上はマイクの性能なので除外するとして、80Hz 以下がちょっと寂しいですね。このあたりはたとえばドラムやティンパニーのような打楽器の音に影響してきます。これはユニットの性能限界です。

いくらフルレンジといえど、再生できる限界はあります。一般的に、振動板のサイズが小さいほど低音が苦手で、逆に大きいほど高音が苦手になります。ユニット単体の特性をみると 140Hz ~ 30KHz となっているので、これでも十分再生できている方だと思います。2.5Khz、10KHz、14KHz 付近がちょっと下がっているのも気になります。しかし、聞いている限り、気になるほどではないようです。むしろ、こうやってグラフにしないと気づきませんでした。

参考までに、私のメインスピーカーの特性を載せておきます。



前のグラフと比較してどうでしょうか。このスピーカーは fostex の FE88ES-R というユニットを使った CW 型バックロードホーンタイプのスピーカーです。CW 型バックロードホーンというのは、エンクロージャーの種類の一つで、メーカーからは発売されていません。所謂自作の醍醐味というところでしょうか。これについては次回の記事で作りながら説明していく予定です。

ちなみにこのスピーカー、低音は 50Hz くらいから再生できているのですが、70Hz ~ 120Hz にかけての山と谷が非常に気になるところです。聞いているとボワンボワンとします。高音は今回のスピーカーと比べて分かると思いますが非常に強烈。一癖も二癖もあります。しかし、それがカワイイのです。これをうまくならすのがオーディオの醍醐味なのですヨ。

さて、今回作ったスピーカーに戻っていろいろ聴いてみました。周波数特性だけ見ても、何も判りません。特性はあくまで目安であって、このスピーカーの全てではありません。

私がいつも試聴に使うのは、宇多田ヒカルの「FirstLove」とアクアプラスの「Pure -AQUAPLUS LEGEND OF ACOUSTICS-」というアルバムです。前者は、1000 万枚以上売れた邦楽歴代一位のアルバムなのでご存じの方も多いと思います。後者は、アクアプラスという、所謂「読むゲーム」のメーカーのゲームの BGM のアレンジアルバムなのですが、曲も、歌も、収録もすばらしいというオーディオ好きのためにあるかのような CD なのです。「ToHeart2」や「うたわれるもの」といった言葉に身に覚えがある人は、是非買って聴いてみることおすすめします。



今回作ったバスレフと私のメインスピーカー

て、音の雑さが消え、低音もより出るようになります。

逆に、ジャズは苦手かもしれません。金管楽器の音がチープになってしまい、シンバルの音ももう少し金属的な響きがほしいところです。何よりも低音が足りないのが痛い。

しかし、FirstLove を小音量で聴いた場合、私のメインスピーカーよりも良いんじゃないかと感じる瞬間がありました。バランスのとれた周波数特性が効いているのかもしれません。

ほかにもカットギターの弦の張った感じなどは素晴らしい。FAKiE というグループの CD によく合う。

え？ プラシー♪じゃないかって？ 断じてそんなことはありませんよ。ええ。

皆さんはこれを聴いてどう感じるのでしょうか

## 終わりに

さて、どうだったでしょうか。少しでもオーディオの良さが伝わってくれたら幸いです。次回は FF85K を使ったバックロードホーンに挑戦してみたいと思います。

この記事に対する意見、感想は s0711447@coins.tsukuba.ac.jp まで。

このスピーカーを聴かせてほしい、オーディオについて詳しく知りたい、もっと面白い記事を書け！ JBL 以外はスピーカーじゃねえ！ 共鳴管はどうした！ 等々、ご意見お待ちしております。

今回はほかの曲もいろいろ聴いてみましたが、今回のスピーカーの特徴はずばり、「中高音域の美しさ」にあります。ヴォーカルだけでなく、クラシックギターやバイオリンなどの弦楽器にもよく合うようです。音像の定位も良好で、歌手が浮かび上がってくるようでした。

ちょっと意外だったのが、男性ヴォーカルも良い感じに鳴らしてくれるということでした。私は以前、このユニットを使ったバックロードホーンタイプと呼ばれるスピーカーを聴いたことがあるのですが、そのときの感想は、とにかく上品なユニットで、美しく鳴るのだけど、押しが弱いなあとといったものでした。しかし、今回のスピーカーは張りがあり、音の線が太い感じがしました。多少音が雑なのですが、それが良い味を出していました。これは、買ったすぐの新しいユニットであることが関係しているかもしれません。スピーカーは鳴らし込んでいるとだんだん可動部分がこなれてきて、

# 神々の言霊二千八

文 編集部 dorio

## 心構え

大学には神が住んでいる。

Unix の神、インターフェースの神、アナログフィルタの神…

数えきれない程の神々の講義で、私たちは知識という名の果実を齧る。

その神々の、ありがたいお言葉を集めたのが今回の記事。

心して読んでほしい。

「もう帰りたいよ…」

通信の神 K

コンピュータリテラシの時、Coins のシステムがダウンして講義が進められない状態になった時のお言葉。

不安定なシステムは神をも不安定にしてしまうという真理を、入学 したばかりの学類生に教えてくれた。

「S NYなんて潰れてしまえばいいんですよ。」

インターフェースの神 T

S NY は技術力なんてない、あるのは「技術力があると思わせる技術だ」という言を力説。

その後、

「ただ、PlayStation シリーズだけは本気みたいですね。どんどん技術を開発していくって、潰れちゃって下さい。」

「親戚が重役やってるけど、路頭に迷う所見てみたいよね。」と言い放ち、会場を凍り付させた<sup>\*1\*2</sup>。

「パソコンに向かったって、何も生まれませんよ。」

体育専門学群の神 H

新入生の体育オリエンテーションで放たれた言葉。

そのときは、情報の新入生のオリエンテーションという事を、神はご存知だったはずだ。

敢えて発言したのか、何も考えずに発言したのか。

情報と体専は仲が悪いらしい、という噂を信じた瞬間であった。

\*1 後日、学務からお叱りが来たらしい。

\*2 お叱りが来た理由は、S NY の件についてなのか、はたまたパチン と某国との繋がりについて言及したからなのか、それとも某氏をパチコ先生呼ばわりしたからか。カオスな所との戦争が感じられた

「日本の大学はここがダメでね、AT ベル研とか MIT ではみんなやつてる。」

信号解析の神 T

テレビでしか聞いたことないような聖地と筑波を比べられても、と思うかも知れない。  
しかし、これは「筑波大学も、気合で MIT やベル研と張り合える！」という事を暗に示している。

これは、私たちに対する叱咤激励なのだ。

…きっと。

「C 言語なんてね、大学入ってからやつたって遅いんですよ。」

並列計算の神 T

情報 1 年が一学期で受ける講義「解析 1」で放たれた言霊。

大きな夢を持って入学した新入生を、現実と向き合わせる為の優しさが籠っている。

ある情報筋によると「トップクラスの人たちになろうと思ったら、やっぱり物心付く前からコンピュータをイジってないとダメだよ、というつもりで言った。」との事。

神様、しかし言葉が多少不足している気がします。

「課題は本物の Office でやって下さい」

オートマトンの神 K

「Mac の Office と Windows のそれとは、完全に互換が無いから」というお言葉と共に発せられた。

Mac の Office は Microsoft が出していても偽物なのだろうか、という議論を巻き起こした。

「計算機室に Mac は置いてあるけれども、Mac なんて社会に出たら使わないから」という事も仰せられ、一部の人たちを敵に回した(かもしれない)<sup>3</sup>。

「設計変更してもらつたんですよ。」

電子回路の神 I

T Y TA の某高級車、CR WN を買って、色々問題があつたらしい電子回路の神 I。

不具合について色々ディーラーに文句をついているうちに、車に搭載されている制御コンピュータまで仕様変更をさせてしまった。

技術者クレーマーは、怒鳴りクレーマーよりもよっぽど恐ろしい、という事を私たちに教えてくれた。

大学に降臨される、様々な神。

君たちも講義で、神々が発する「言霊」を心に刻んでいってほしい。

\*3WORD 編集部では、Windows のシェアが 30% 程度です。特にこれを書く意味はありません。

# 青年男子ダイエット

文 dorio

それが、始まりだった。

花金<sup>\*1</sup>RanRan。WORD 部屋では「今週終わった！ RanRan ! RanRan !」という歓喜の雄叫びが飛び交い、異様なテンションで RanRan へと繰り出す。

しかし、彼らは共通の悩みを抱えていた。

**そう、体重と言う名の悩みと。**

**Case1: らふにんの場合**

WORD 編集部一のメタボ候補。

家では意外と食べないようだが、大学に入ってからは

朝:家でご飯

昼:家からお弁当

おやつ:粉クリでパンやパフェ

間食:学食でカレーやラーメン

夕食:RANRAN で BIG丼

といった、一日 5 食という非常にフリーダムな食生活。

しかし、趣味の献血で「肝臓ヤバいっすよ wwwwww」<sup>\*2</sup>と言われ、献血拒否されてしまった。最近病院で定期的に血液検査をしてもらうほど気にしている…が、**食べる事は生きる事とほぼ同義**というのが彼のジャスティス。なかなか食事制限が出来ない。

**Case2: どりおの場合**

WORD のメタボ候補二番手。

成人式で誰にも声を掛けてもらはず、「俺マジぼっち？」と思いつきり回んでいたら「太りすぎて誰か分からんかった。」という言葉が飛んできた。そして「教師席はあちらです。」と言われた時、何かが心を貫いた。

2 年前の写真を見せると「本当に同一人物?」と言われるのは当たり前、時には三つ下の弟と買い物へ行って親子と間違われる事も。

大学生活二年間で得たものは、16kg ほどの脂肪と血圧の上昇が 30 ほど。

\*1 花の金曜日。次の日から休日という事で、サラリーマンたちが居酒屋へ繰り出す日。街で最も Core Dumper が溢れる日としても有名。

\*2 よく中年が言われる言葉。これを聞くと彼らは酒量のダイエットを始めるが、やはりなかなか長続きしない。

**Case3:ろんずの場合**

ぱっと見、太っているようには見えない。しかし、それは着痩せという名のマジック。  
マックへ行けば「ハンバーガー四個分下さい」と注文し、コーディングをしていれば片手にはスナック菓子を摘む。素晴らしい程のテンプレート的なメタボ予備軍の生活。  
まだ大丈夫まだ大丈夫と思っていたら、70kg Over してしまっていた。

**出会い、そして決意。**

それぞれに「早くどうにかしないと…」と悩み、しかしながら行動に起こせなかった三人。  
だが、手を取り合い、力を合わせる事を誓った彼らの前には、不可能という言葉は意味を持たないだろう。

**必死な彼らが紡ぎ出すダイエットストーリー。**

**WORD 編集部サリカルダイエット、始まります。**

**まあそんな感じで、長期企画が始まった訳ですが。**

今回の企画にあたり、ダイエット期間としてまず2月から春休みまでの一ヶ月の体重を記録しました。

各自それぞれ工夫をして生活を送りました。以下、三人が行ったダイエットの工夫です。

**らふにんの工夫**

- ・RanRan / 夢屋の回数を減らす。
- ・ご飯お代わり自由のファミレスにおいて回数を減らす。
- ・毎日計ることで自分の意識を改善する。

**どりおの工夫**

- ・RanRanでの注文時、一つサイズを落とす(Big 丼から Big 丼小盛り)。
- ・フライングガーデンではご飯お代わりを5杯から3杯へ。トンQでも同様。
- ・ストレスが溜まると太るので、食べ放題では我慢しない。メリハリ大切。

**ろんずの工夫**

- ・一日二食に変更。
- ・決して無理はしない。
- ・ガムを食べる。

以上、やる気の無さが溢れ出ているダイエットの方法ですが、これ以上ガチガチに固めると「そんなにメンドクサイんだったらダイエットなんかしたくない！」<sup>1</sup>と言いくだすような人しか居ないので、この程度で始める事にしました。

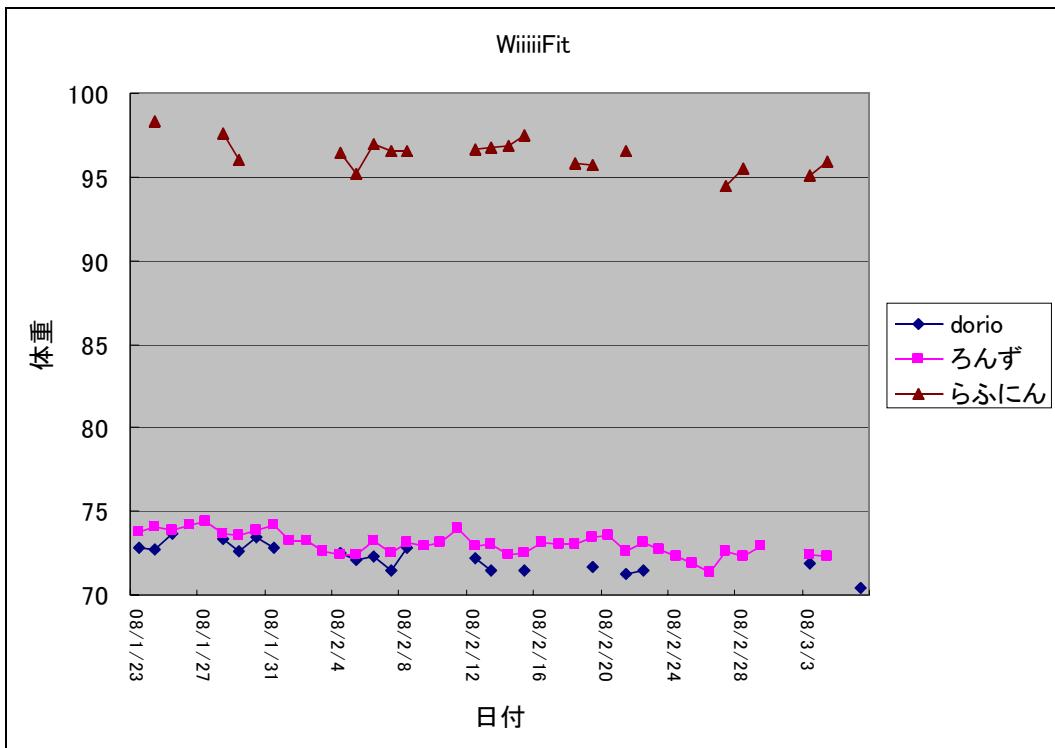
---

\*1 例文) 「もういいや、疲れたからパフェでも食いに行こうぜ！」

## 青年男子ダイエット

んで、結果は

以下、2-3月の体重のデータ推移を示します。



えっと、白黒で非常に見づらい気がしますが、ほとんど体重変化無しという事だけは読みとって貰えると思います。

まあ、やった事と言えば出来るだけ体重計に乗る程度ですから、当たり前と言えば当たり前なんですがね。

しかし、継続は力なり。〈del〉飽きなければ不定期連載記事として、これからも続けて…いけたらいいという事を前向きに検討している旨をここに表明したい所存でございます\*1。

そこの体重で悩みを持っている君も、この記事と一緒に

**ダイエット、やらないか？**

\*1 つまりは、春休みを明けてから誰も体重を計っていない、というか、そもそも企画の存在を忘れていたという事です。

# ICPC やらないか？

文 yuzuhara 編集 ろんず

## チャレンジャー募集！！

プログラミングにかける熱い情熱！もっとオーダーを小さくッ！さらに少ないコード量でッ！！より早くッ！プログラミングの腕を競うコンテスト、みんなトウゲザーしようぜ！

・・・厳密に言うと、こういうノリではありませんが（汗）  
みなさんが大学や趣味で磨いてきたプログラミングの腕を、もっと大きい場所で発揮し、競い合ってみませんか？

## ACM ICPC について

プログラミング競技の場として、大学対抗の ACM ICPC という国際プログラミングコンテストがあります。これは、3人チームを組み、1台のPCを共有し、時間内に与えられた課題を解いていくという競技内容です。以下問題例です。

## ICPC 得点集計ソフトウェア

### 概要

世界道化コンテスト (International Clown and Pierrot Competition; ICPC) は興行界で権威も人気も最高の行事である。

このコンテストの特色のひとつは、採点に当たる審判員の数が多いことでときには百人にも上る。審判の数は競技者ごとに異なる。というのは、採点対象の競技者と少しでも関係のある審判は、その競技者の演技の採点から一時的に外れるからである。

基本としては、ひとりの競技者の演技についての審判の点数の平均がその競技者の点数になる。常軌を逸した視点から採点する審判が点数に大きな影響を与えないよう、最高点と最低点は除外する。

もし最高点をつけた審判が複数いたら、そのうちのひとつだけを無視する。最低点についても同様である。平均点には端数があるかもしれないが、それは切り捨てて最終的な点数は整数値とする。この行事をテレビ中継向けにスピードアップするため、ある演技の点数を審判全員の採点から計算するプログラムを書くことをあなたは頼まれた。

### Input

入力はそれぞれが競技者の演技ひとつに対応するいくつかのデータセットからなる。入力のデータセット数は 20 以下である。

データセットの最初の行はある演技の採点に当たった審判の数  $n$  ( $3 \leq n \leq 100$ ) である。引き続く  $n$  行には各審判のつけた点数  $s$  ( $0 \leq s \leq 1000$ ) がひとつずつ入っている。 $n$  も各  $s$  も整数である。入力中にはこれらの数を表すための数字以外の文字はない。審判名は隠されている。入力の終わりはゼロひとつの行で示される。

## ICPC やらないか？

### Output

出力は、入力データセットごとにそのデータセットに対応する演技の点数を十進整数で記した一行である。出力行には他の文字があつてはならない。

私たち筑波大チームは、去年 2 チームで参加しましたが、残念ながらインターネット予選を突破することができませんでした。

今年ももちろん挑戦するつもりですが、2 チーム構成もできるかできないか、という人数しかおらず、また、M1,B4 の人に固まつていて、ノウハウの継承もままならないというのが現状です。

一人でも、友人と誘い合わせてでもよいので、興味を持った方、是非連絡を下さい。

### コンテストまでの流れ

コンテストに向けた予定は以下のとおりです。

#### 5月～6月末

- ・TopCoder に参加

TopCoder とは、週に 1 回ほどのペースで開催されているオンラインのプログラミングコンテストです。

- ・集まって集中的にアルゴリズム、過去問の練習  
予定は未定です。

#### 6月末

- ・国内インターネット予選

インターネット上で、予選が開かれます。6 問を 3 時間で解きます。

### 連絡先など

忠鉢洋輔 : bachi@osss.cs.tsukuba.ac.jp

Wiki : <http://www.osss.cs.tsukuba.ac.jp/~bachi/icpc>

確認が取れ次第、ML で連絡を取りあうカタチになります。連絡待ってます！

# 情報科学類誌

# WORD

WORDでは古い記事の  
使い回しはしておりません。号

発行者

情報科学類長

編集長

武井 裕也

制作・編集

筑波大学情報学群

情報科学類誌WORD編集部

(第三エリアC棟212室)

印刷

第三エリアF棟印刷室

2008年5月

初版第一刷発行