

WORD

27

2013.6

From College of Information Science

あっ…(冊子)号



目次

豚大学	3
パワーフード鶏頭水煮編	9
LXC による軽量な仮想環境	16
高知 × 鳥取の鼓動	35
自己をプリキュアとして高めるための素敵なグッズ紹介	38
LatestC++	47
GR な日々。XVI	53
新入生の新入生による新入生のための PC 購入のススメ	59
WORD 読者アンケート 2ndSeason	66

豚 大 学

文 編集部 IX

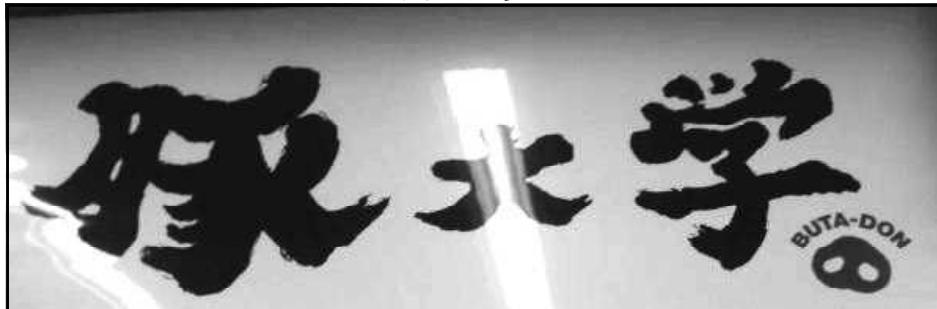
1. はじめに

サウナ大学^{*1}くんが解体か、悲しいなあ。

名前だけ知つて行こう行こうと思っているうちに行くことが出来なくなる、それはとても悲しいこと。

そんなことより、お腹空いたし食べ歩きのブログ巡回すつか！ お、シグナル・〇ッソ^{*2}の胃袋強化作戦更新されてんじやん。最近は麺多めだったけど、そろそろご飯物かねえ……お、今回は豚丼か。ほうほう、新橋に「豚大学」って店があるのか……

ぶただいがく
ん？豚大学！？



ひやあああああああ新しい大学だああああああ！！！！！

サウナ大学じゃ食指動かなかつたけど、豚肉だったら体脂肪率40%の俺でも満足できるぜ！

そういうやこの前の健康診断で BMI^{※3} が 34 とか出てたなあ……

ひやっほおおおおおおおおおおおおおおおおおおおおお
健康的いいいいいいいいいいいいいいいいい！ ！ ！ ！ ！

行かなくっちゃ……

*1 サウナ大学：東新井にあったスパ&サウナ。名前を知ってるけど行ったことない、そんな人は多いはず。

*2 シグナル・○ツソ：<http://signalrosso.blog.fc2.com/>

*3 BMI：人の肥満度を表す体格指数。日本肥満学会的には 22 が最適値らしい。

2. いざ行かん豚大学へ

2.1 登校

豚大学に行きたい、そんなとき、あなたは山手線を使って新橋に行くことが出来ます。



はい、新橋に着きました。SLがあることでおなじみの日比谷口です。

さあ、向かって左側のニュー新橋ビルへ向かいましょう。豚大学はもうすぐそこ。

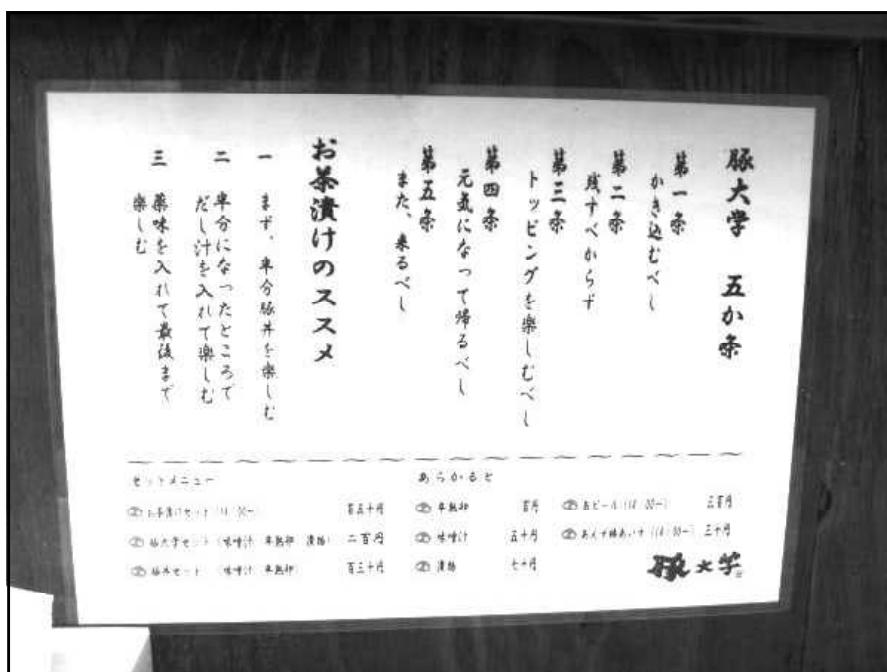
2.2 入学

1,000円札を券売機に食わせて、学科を選びましょう。小～特大、あなたの偏差値に合わせてボタンを押します。

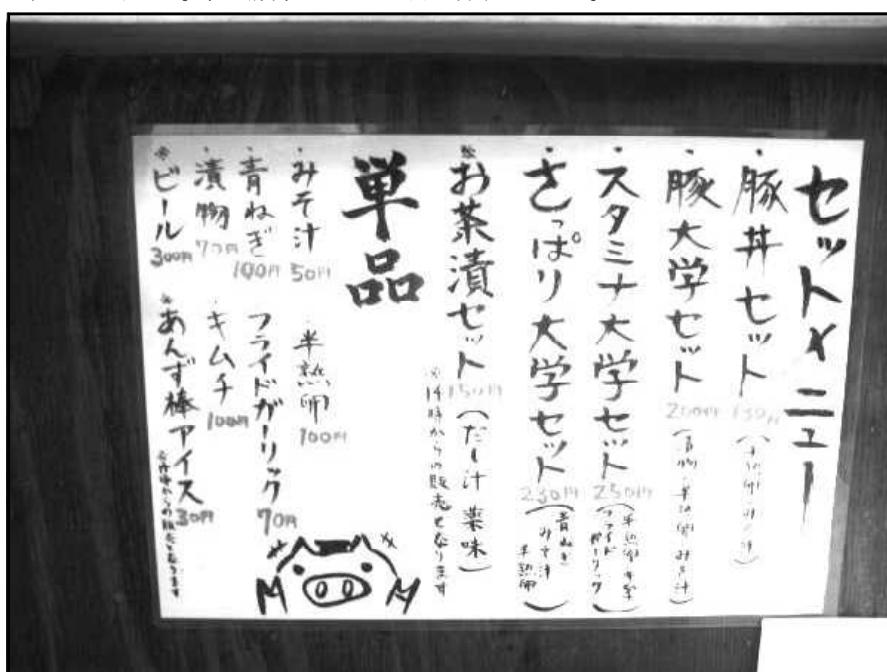
小：480円 ちょっと少なめです	中：630円 ちょっと多めの標準だと思います
大：780円 しっかり満腹感じます	特大：990円 総量1000g 挑戦しませんか

直前に調べた情報によると、一般的な人は中でちょうど良いらしいとのことですが、もちろん私は特大一直線。焼くタイプの豚丼はほぼ初めてだから今から楽しみです。

カウンターには、豚大学 五か条なる学則とオプション一覧が貼られています。



お茶漬けに逃げてはいけない。私は豚丼そのものと向き合うのだ……。



さっぱり大学、スタミナ大学なる姉妹校もあるのか……。

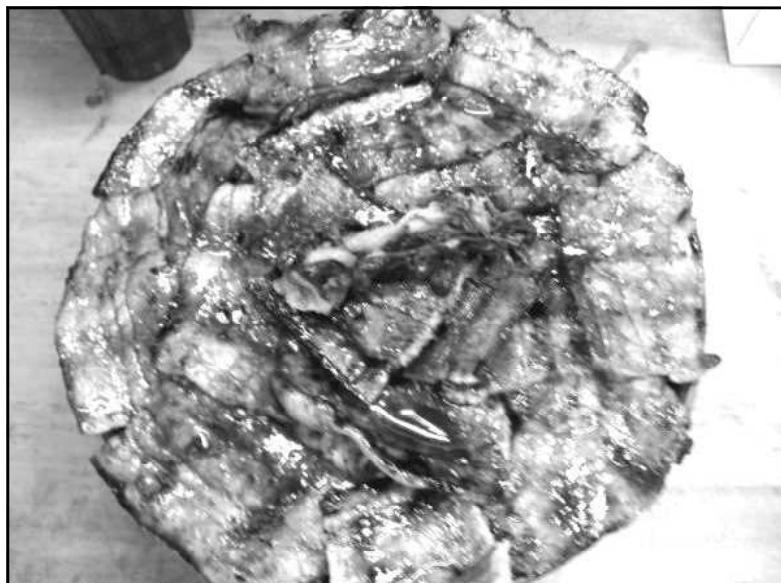
2.3 学生生活

10分ほど待って着丼。油断せずに行こう。



このサイズで990円！ 1野口切ってますよ！ 1野口以下！ お得感抜群ですぜ！

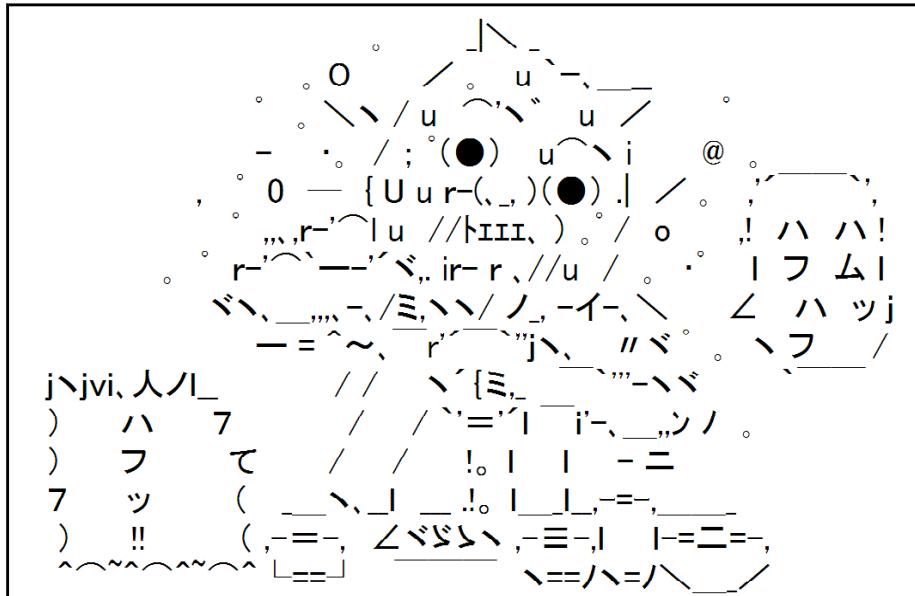
奥のコップと比較してみると大きいのが分かるでしょう？



いやあ壮観ですなあ……まるで切り株のようだ。さあ、

炭火が香ばしいですねえ。いい感じにこんがりしていておいしいです。丼中央に添えられた菜の花が箸休めとして意味をなさないくらいには肉々しい……脂はデブの大好物ですよ！

半分くらい食べてちょっと満腹感が襲ってきましたが、ここで止まつてはいけません。スパートかけてかつ込みます。



ハッフハッフムッシャムッシャハッフハッフムッシャムッシャハッフハッフムッシャムッシャ



ふう……激しい食事でした。

2.4 卒業

無事完食して、特大学科の課程をすべて終了しました。今回は単位の取り方が雑だったので、次回入学時はオブション自由単位を駆使して華麗に首席卒業したいですね。

そして心地よい満腹感と共に私は神田^{*4}へ向かうのであった……。

3. おわりに

焼くタイプの豚丼は初めてでしたが、なかなか美味しかったです。予定も詰まっていたので10分ほどで完食する形になりましたが、今度はゆっくりと味わってみたいものです。しかしまあ、コストパフォーマンスが実に良い。サラリーマンの街だからでしょうか。

皆さんも新橋にお越しの際は、是非入学してみてはいかがでしょうか。

■店舗情報■

店名 : 豚大学

住所 : 東京都港区新橋2-16-1 ニュー新橋ビル1階

営業時間 : 10:30~21:45(平日)

11:00~15:00 16:30~20:15(土日祝)^{*5}

席数 : 12席

*4 神田 : mu ○ en 関東オフの聖地。

*5 (土日祝) : 5月19日からしばらくの間、日祝が定休日になるようです。これから行こうと思っている方はご注意を。

パワーフード 鶏頭水煮編

文 編集部 ≠

外では気温も意識も高まる陽春を迎える中、危うく筆者は留年するところだった^{*1}。何をするにもなんとなくやる気が出ない。おかげで、トイレで手を洗った後乾燥機を使うことも、コーンの先までアイスを押し込みながら食べることも、ペットボトルの本体とキャップとラベルを正しく分別して捨てることも、土日に秋葉原に行くのに毎回回数券を使うことも、スペゲッティを茹でるときに塩をひとつまみ入れることも全て億劫だ。

とにかく新年度を無気力で迎えるのはまずい。俺に力があれば……生きる力が……

はじめに

人は、光合成のように体内でエネルギーを生産する事は出来ず、エネルギーの全てを食べ物から補給し生きている。エネルギーに生きたければ、よりエネルギーな食べ物を食べるのが通例だ。”You are what you eat”というフレーズもある。もし仮に、一日の活力が食べ物によって決まるならば、良い食べ物を日常的に食べているほど人生を優位に進めることができるだろう。しかしながら、うなぎ、すっぽん、クロイモリなど、精のつく食べ物は数あれど、どれも高価で学生には易しくない。そこで本記事では、学生に可能な、日々を強靱に過ごすための「パワーフード」を紹介する。調理は発起人である葡萄酒氏と共に行った。

パワーフードに関する記事はほぼ2年ぶりとなり、今回が2回目だ。前回は「アメリカンパワーフード」と題し、最も不健康的なデザートと報道されたヘビースイーツ「スニッカーズパイ」を紹介した。興味を持たれた方がいらっしゃれば公開されているWORDバックナンバー18号を参照して欲しい^{*2}。

前回の「スニッカーズパイ」は「ハイエンドカロリー」という観点からのパワーフードであった。あれから2年、「美味しいんば」や「孤独のグルメ」による学習を積んだ筆者は、素材そのもの的重要性、また料理全体の印象が与える効果を再認識した。葡萄酒氏がパワーフードの食材として「鶏頭水煮」が提案したのはそのさなかである。今回はビジュアルインパクト、つまり食べる人の精神を揺さぶるという観点からの「パワーフード」だ。

鶏頭水煮は文字通り鶏の頭の水煮である。今まで全く見たことがなかったが、ドッグフードとして販売されているらしい。最近ではペットショップでも中々取り扱っていないため、日本では犬、それも極少数の犬しか食べていないと見て良いだろう。また、理科の授業で解剖実験に使われることもあるらしい。今時やっているところはあるんだろうか。

*1 新TWINSでは、単位の充足状況が確認できるようなので、私の経験したような悲劇はもう起きないだろう。

*2 http://www.word-ac.net/?page_id=721

実際に購入してきたものはトマト缶と同様のサイズで1缶およそ150円。

開封してみると、獣の匂いと共に灰褐色の頭部が詰められており、ゼラチン質と滲み出た油分に浸かっている。表面は羽毛を抜いた後のぶつぶつがびっしりと覆い、くちばしの間から見える舌や濁った眼球が脱力感を感じさせる。鶏頭は繊細に扱わねば崩れてしまうほど脆く、持ち上げると一息遅れて粘着質な音が鳴った。取り出した頭を写真1に示す。1缶6首^{*3}。2缶購入したため12首だ。価格にして300円程度だが、途轍もない負債を抱えてしまったかのような気持ちになる。

鶏頭はグロテスクというよりは冒涜的だった。信心も動物愛護精神も旺盛でない筆者でさえ、埋めて供養したほうが良いかも知れないと考えたほどである。開封前までは筆者、葡萄酒氏ともに意気揚々とした雰囲気だったが、開封してからは不思議な時間が流れ始めた。



写真1：R.I.P

≠「とりあえず出したけどどうしよう。」

葡「実は頭蓋骨が固すぎて食う部分ないんじゃないかな。」

≠「箸で持ってみた感じ結構やわそうだった。」

≠「寄生虫館^{*4}で見たけど、脳を食べるのは寄生虫感染のリスクが高いらしいよ。」

葡「加熱すれば大丈夫だろう。臭みもありそうだし、下処理はしっかりしよう。」

≠「そういうえばパッケージには『活動的な愛犬に』って書いてたけど、犬以外が食べても大丈夫なのかな」

葡「そういうのは大抵但し書きが書いてあるよ。」

≠「本当だ。犬以外には与えないでくださいって書いてあるね。」

*3 単位あってる？

*4 目黒寄生虫館。様々な寄生虫の標本を見たり、グッズを購入することが出来る。象牙病感染者の陰嚢が見所。

葡萄「まあ今日は与えるわけじゃなくて、自発的に食べるから。それに肝臓の弱い犬が食べて大丈夫なんだから、人が食べてもそんなに問題ないと思うよ。」

≠「何と言つても普段食べてる動物だし大丈夫だよね。」

葡萄「うん。」

≠「……。」

下処理

手順

1. 塩と酒で茹でてしっかりと加熱する。

万が一にも健康被害にあわないよう、どの料理でも始めに入念な下処理を行うことをおすすめする。

これ以降、レシピの手順では、下処理の記載は省略している。

レシピ1：鶏頭の刺身

手順

1. わさび醤油を付けていただく。

通はやはりわさび醤油で、各部位をじっくりと味わうものだ（震え声）

得体の知れない臓器を含んだタンパク質を丸かぶりできるほど二人は豪胆ではなかった。脳天から箸で分解していくと大豆のような脳、ハリのない眼球やよく分からないぶよぶよ、それからたくさんの骨になった。二人ともタイミングをはかりつつ口に運ぶ。始めのうち筆者は、鶏頭をわさび醤油で頂くというよりは、わさび醤油を鶏頭で頂いていた。

以下は勇気を出して口に含んだ、それぞれの部位についてのレビューだ。

脳 : レバーと同様に口当たりはまろやかでコクがある。

皮 : 考えて見れば、通常の鶏皮と同じだった。

トサカ : 鶏皮とほとんど変わらないが、より油の香りが強くしっかりとした弾力がある。

眼球 : 臭みはなく、ほんのりと甘味がある。

くちばし : 他の骨と比べても固めだが食べられないこともない。無味。

よく分からぬぶよぶよ : ぶよぶよ。

……見た目は悪いが、思っていたほど味は悪くない。

「美味しんば」作中でも、料理について豊富な知識を持つ山岡さんが「鶏で一番美味しいのは皮と内臓」と断言している^{*4}。その両者を合わせ持つ鶏頭は実は鶏を味わう上で適した部位なのかもしれない。

レシピ2：鶏頭ごぼう

鶏頭ビギナーの皆さんには、やはり舌に馴染んだ家庭料理の体を装って攻略するのがよいだろう。ここでは鶏ごぼうを踏襲した鶏頭ごぼうを作った。

手順

- 1.ごぼうやこんにゃくを一口大に切り、水にさらしてアクを抜く
- 2.鶏頭をごま油で炒め、塩でかるく下味をつける
- 3.軽く焦げ目がついたら鍋から取り出し、ゴボウとこんにゃくを炒める
- 4.2で炒めた材料を鍋にいれ、酒と薄めためんつゆを入れて煮る
- 5.水分を飛ばして完成

ここまで調理してようやく鶏頭にかぶりつく決心がついた。控えめにかじると、よく煮た骨特有の歯ざわりがシャリシャリと頭に響く。脂身がやや臭みを感じさせるが、通常の鶏肉よりもジューシーだ。何よりこんにゃくとごぼうがうまい。煮るときに生姜を入れるとより美味しくいただけるだろう。



写真2：認めたくないお袋の味

レシピ3：鶏頭のエスニック焼き

レシピ2で生理的嫌悪感を抱く皆さんには、レシピ3あるいはレシピ4による攻略をおすすめする。

一般的に料理のおいしさは食べる人の心構えに依存することが知られている^{*5}。レシピ3ではこの心構えを利用して攻略を狙う。心構えが影響する例として、ある料理を高級食材を一流のシェフが調理したものだと信用した後で食べる料理は、そうしなかった料理よりも美味しく感じる傾向があることが知られている。その料理に対するポジティブイメージが重要ということだ。

*4 美しひんば コミックス60巻 「居酒屋、新メニュー!？」

*5 要出典

つまり現時点で筆者が抱いている、「鶏頭そのものがあまりにも食べて良いものではなさそうだし、気持ちが悪い」という偏見を払拭すれば、美味しく鶏頭を食べることが出来るだろう。

このレシピは自分との戦いだ。自己イメージに沿うように調理するのがふさわしい。

手順

- 1.想定する情景にふさわしい具材を揃える
- 2.思い思いに調味料をふりかけつつ炒める
- 3.頭の中で情景を思い浮かべながら食べる

筆者が想定するのは、「実はこういう見た目は普通だし、みんな重宝している美味しい食材なんだ」という状況だ。調理後、一息に食べるべく想像を開始した。

僕はアマゾン川流域の森深くの部族の一員。鶏頭は貴重なタンパク源だ。普段はシャーマンのおじさんしか食べられないのだが、今日はお祭りでみんなに鶏頭が振舞われる事となった。直火で焼いた鶏頭はとても美味しいのでみんな大はしゃぎだ。手元の鶏頭は焼きたてでとてもいい匂いがする。さあ食べよう。何しろこの鶏頭は、僕が今朝まで大切に育ててきた鶏のものだ。毎朝鳴き声がうるさかったが、今朝も元気に走りまわっていて可愛い奴だった。なのに今では……

むしや

まずくはなかったが全部食べる気にはならなかった。

具体的な調理として筆者は、ネギ油を敷き、ナムプラーとニヨクマムとトムヤムパウダーを加えて炒めた後、ネギと香菜と花椒を振りかけた。ちなみに花椒は山椒の強化バージョン、トムヤムパウダーはトムヤムクンに用いられるレモングラスなどのハーブを調合したいい匂いの調味料、ナムプラーはくさい発酵調味料、ニヨクマムはくさい発酵調味料、香菜はくさい植物だ。タイ、ベトナム、中国の多国籍料理となっており、筆者の「エスニック」に対するイメージの杜撰さが現れている。



写真3：鶏頭のエスニック焼き

レシピ4：鶏頭といちごのフレッシュタルト

ビジュアルで生理的嫌悪感を抱いているのならば、ビジュアルで対処するのが最も効果的だ。一口にビジュアルと言えど様々な方向性がある。今回は生活になくてはならない癒し要素である「可愛さ」に注目し、可愛いスイーツの代表格であるいちごタルトをベースにレシピを作成した。スイーツであるため別腹での攻略も可能にしている。

手順

- 1.バターをレンジで軽く温め、ポマード状に柔らかくする
 - 2.1と砂糖とすり混ぜ、卵黄、アーモンドプードル、小麦粉と良く混ぜあわせる
 - 3.30分程度冷蔵庫で寝かせる
 - 4.18cmタルト型にサラダ油を塗り、小麦粉をまぶしておく
 - 5.タルト台に流し込むアーモンドクリームを作るため、1とは別にバターをレンジで軽く温め、ポマード状に柔らかくする
 - 6.5に砂糖とすり混ぜ、アーモンドプードルとラム酒を加え混ぜあわせる
 - 7.寝かせた3の生地をのばしてタルト型に貼り付ける
 - 8.フォークで穴をあけたのち6を流しこむ
 - 9.170度で余熱しておき、レンジで25分焼く
 - 10.冷ました後、いちごと鶏頭を飾り付ける



写真4：何の儀式だろう

やはりメルヘンとホラーは親和性が高い。いちごのフレッシュさと鶏頭のフレッシュじやなさが見事なコントラストだ。口に含んだ時に一瞬油臭さで眉をひそめる事となるが、食感もタルトのカリカリで緩和されていて、後味はいちごタルトだ。いちごに埋もれる鶏頭に気がつかずにうっかり口に運んでしまうだろう。

まとめ

本記事では、ビジュアルインパクトを与えるパワーフードとして、「鶏頭水煮」を用いた料理を幾つか紹介した。筆者としては、「頭脳パン」を作れなかつたのが心残りだ。本記事を読んでこんなレシピもあるぞ、こんなパワーフードがあるぞという情報、またご意見ご感想がありましたら、巻中のアンケートからお願ひします。最後になりますが、鶏頭水煮を食べることによって生じた問題に対して責任を負うことは出来ません。



2口サイズでお弁当にもぴったり

LXCによる軽量な仮想環境

文 編集部 mitty

1 はじめに

Kernel Hacking をしたり、Linux Distribution の新しいリリースを試用してみたり^{*1}、あるいは素性の知れないソフトウェアを試してみたり^{*2}する際に仮想環境は欠かせないのですが、ハードウェア仮想化支援の付いていない x86 CPU が少数派になりつつある^{*3} 昨今では、皆さんも QEMU-KVM や VirtualBox、VMware^{*4} といった仮想マシンモニタ^{*5}を活用のことと思います^{*6}。

本記事では VMM とは異なる手法で仮想化を実現する、Linux Container^{*7}について紹介したいと思います。

2 LXC とは

2.1 VM/VMM と LXC の違い

QEMU-KVM などの VMM では、ホスト OS^{*8} とは完全に別の OS を VM 上で動かすことが可能ですが。多くの VMM では物理的なハードウェアと同等のデバイスを、エミュレーションによって仮想的に用意する^{*9} ことで、ゲスト OS^{*10}からはあたかも実体を持って存在している計算機の上で動作しているかのように見えます。

これに対して LXC ではハードウェアのエミュレーションは行わず、ホスト OS^{*11} とコンテナは全く同じカーネル、同じメモリ空間上で動作します。コンテナに属するプロセスやファイルツリー、ネットワークといったリソースのみを Linux カーネルの機能を使って隔離することで、ホスト OS とは別の環境が動作しているかのように見えるわけです。

^{*1} そういえば Debian 7.0 Wheezy が 5 月頭にリリースされました。

^{*2} 行儀の悪いソフトはしまっちゃおうねえ

^{*3} CPU の仮想化支援の対応状況

<http://ark.intel.com/Products/VirtualizationTechnology>

<http://sites.amd.com/us/business/it-solutions/virtualization/Pages/client-side-virtualization.aspx>

^{*4} 少し内容が古くなってしましましたが、情報科学類生は VMware Workstation/Fusion を条件付きながら無料で使用することができます。WORD22 号の「筑波大学の計算資源」で紹介したがあるので、よければ参考にしてください。

^{*5} Virtual Machine Monitor、略して VMM。以下仮想マシンとあわせて VM および VMM と省略します。

^{*6} 「おまえは今まで潰した VM のインスタンス数をおぼえているのか？」

^{*7} こちらも、以下 LXC とします。

^{*8} 物理的なハードウェア上で直接動作している OS のこと。VMware vSphere Hypervisor のように、ホスト OS と呼べる部分が無い VMM も存在します。

^{*9} 完全仮想化、準仮想化、物理デバイスのバススルーなど、厳密には色々ありますが、説明をはじめると本記事どころか本一冊でも取まらないので詳しくは省きます。

^{*10} VM 上で動作する OS のこと。

^{*11} LXC の場合はコンテナの中と外で OS が違うわけではないので、「ホスト OS」という書き方はあまり適切では無いですが、「コンテナの外」と書くのも少し間抜けな感じがるので、VM の「ゲスト OS」「ホスト OS」にあわせて「コンテナ」「ホスト OS」と記述することにします。

2.2 LXC の長所と短所

LXC の長所は

- エミュレーションを行わないため、VM を使った場合と比べてディスク I/O やネットワーク通信などにオーバーヘッドが無い
- CPU やメモリ、ディスク I/O 速度の制限が柔軟に行える
- ホスト OS のデバイスを必要に応じて直接扱える
- VM のディスクイメージと違い、ホスト OS からコンテナの中のファイルを直接扱える

などが挙げられます。

エミュレーションを行うと必ずつきまとうのがオーバーヘッドの問題です。存在しないハードウェアをエミュレーションによって存在しているかの様に見せかけたり、本来は一つのホスト OS で専有するデバイスを複数のゲスト OS から代わる代わる利用するために調停を行ったりと、元々必要でなかった余計な処理が発生してしまうからです。このオーバーヘッドをいくらかでも減らそうと、準仮想化やハードウェアによる仮想化支援など様々な試み^{*12} が行われています。LXC ではエミュレーションを行わないため、この種のオーバーヘッドとは縁がありません。

また LXC では、コンテナ内からホスト OS のデバイスへ自由にアクセス出来ないよう、cgroup という Linux カーネルの機能を使って制限を掛けているのですが、この cgroup を適切に設定することで、コンテナに属するプロセスは特定の CPU コアのみで実行されるようにしたり、利用出来るメモリの上限を設定したり、ディスクに必要以上に負荷を掛けられないよう制限したりすることができます。

さらに、cgroup の制限を意図的に緩めることで、コンテナ内からホスト OS のデバイスへアクセス可能にすることも出来ます。

一方、LXC の短所は

- ホスト OS と異なるアーキテクチャや異なる OS、異なるバージョンのカーネルを利用出来ない
- コンテナにおいてリソースを過大に消費すると、特にメモリやディスクの空き容量などでホスト OS に影響が及びやすい
- ホスト OS からコンテナの隔離が十分でない部分があるため、コンテナの中からホスト OS の見えては困る情報が見えてしまったり、ホスト OS を制御出来てしまう場合がある
- カーネルに脆弱性が見つかった際に、その脆弱性を利用してコンテナの中からホスト OS へ悪影響が及ぶ危険性がある

といったことがあるでしょう。

^{*12} この辺りも色々と面白い……というか筆者の研究ネタがまさに VMM なのですが、今回は LXC の話なので詳しくはまたの機会（あるのか？）にでも。

ホスト OS とカーネルを共有するという LXC の仕様上、異なる OS やカーネルが利用出来ないのは仕方ないと言えます。ただし、QEMU と組み合わせることで ARM などの x86 以外のアーキテクチャで動作するコンテナも作れるようです^{*13}。

ホスト OS へ影響が及びやすいという点についてですが、KVM-QEMU などの VMM の場合、ゲスト OS に割り当てるディスクやメモリ容量は VM を作成する際に決定され、その値を上限としてそれ以上のリソースをゲスト OS が消費することは出来ません。

しかし LXC の場合、ファイルシステムやメモリ空間はホスト OS と共有されるため、コンテナ内で巨大なファイルを作成したり、ひたすらメモリを消費するプログラムなどを動かしたりした場合、ホスト OS 全体でディスクの空き容量が無くなったり、メモリが足りなくなったりすることがあります。

コンテナの中からホスト OS を直接制御出来るという抜け道は、ある意味 VM では簡単には出来ない利点とも言えるかもしれません、基本的には望まれた仕様ではないため、AppArmor などの強制アクセス制御機構^{*14}を使って塞がれています。なお、Ubuntu 13.04 以降^{*15}、User Namespace というカーネルの新しい機能も使っているようです。

カーネルに脆弱性があった場合でも、上記 AppArmor などを使ってその影響がコンテナの中からホスト OS に及ばない様に設定することが出来るでしょう。

3 LXC を使うための準備

LXC を利用するにあたって必要な準備は Linux のディストリビューションによって異なりますが、筆者が普段使っている Ubuntu 12.04 Precise Pangolin をベースに説明を行っていきたいと思います。Ubuntu 13.04 Raring Ringtail や Debian 6 Squeeze、Debian 7 Wheezy を使っていて気付いた点などは随時注意書きしていきます^{*16}。

ちなみに、色々試行錯誤してみましたが Debian 7 ではうまくコンテナを起動することが出来ませんでした^{*17}。筆者の手順が悪いのか、何かバグがあるせいなのかは執筆時点では不明です。

3.1 必要なパッケージ

まずは *lxc* パッケージを導入しましょう。Ubuntu に関しては、

```
$ sudo aptitude*18 install lxc
```

^{*13} この場合は CPU のエミュレーションを伴うため、速度は犠牲になります。

^{*14} 少なくとも Ubuntu では AppArmor を使っています。RHEL (Red Hat Enterprise Linux のこと。業務用途で使われることが多い) では SELinux を使うのかしら。

^{*15} 正確には Linux カーネル 3.8 以降。以下のサイトなどが参考になるでしょう。

<http://d.hatena.ne.jp/defiant/20130213/1360760602>

Linux カーネル 3.8 の User Namespace 機能 (I) - TenForward の日記

^{*16} RHEL、Gentoo などの人はごめんなさい。でも、RHEL はともかく Gentoo 使っている人ってこんな記事読まなくても自分でどうにか出来る気がする…… (偏見)。

^{*17} なんという出落ち。どうも起動途中に何らかの理由でコケているようなのですが、詳しく調べていないのでどうすれば直るのか執筆時点では不明です。

^{*18} Ubuntu では apt-get、Debian では aptitude を使うことが好まれているようですが、筆者は何となく aptitude を使い続けています。推奨パッケージを同時にインストールしないことが多いので、aptitude-without-recommends の省略形として aptitude -R が存在するのが理由の一つかもしれません。(apt-get だと-no-install-recommends で、省略出来ない)

なお、Debian 7 から aptitude より apt-get が推奨されるようになりました。

<http://www.debian.org/releases/wheezy/amd64/release-notes/ch-upgrading#upgradingpackages>

として推奨パッケージ (*Recommends*) もインストールしていれば、通常の使い方ではパッケージの追加は特に必要ないはずです。

Debian では、上記に加えて以下のパッケージを追加すると良いでしょう。

debootstrap

Debian 6 では、なぜか推奨パッケージにも入っていません。コンテナを作成する際に *debian* テンプレートや *ubuntu* テンプレートを使う際は必要となるので導入しておくと良いでしょう^{*19}。

bridge-utils

ネットワークを使用したい場合、ホスト OS のネットワークカードにコンテナの仮想ネットワークカードをブリッジするために必要となります。

3.2 その他のパッケージ

必要に応じて導入しておくと便利なパッケージです。

yum

RHEL やそのクローンで使われているパッケージ管理ツールです。fedora テンプレートや oracle テンプレートを使う際に必要となります。

curl

fedora テンプレートを使う際に必要となります。

virt-manager

QEMU-KVM や Xen といった VMM を制御するために使われることが多い GUI ツールですが、LXC にも部分的に対応しています。

lvm2

物理ディスクを直接パーティションで区切ってファイルシステムを作る方法では、パーティション、ファイルシステムの作り方がディスク構成によって制限されます。物理的なディスク構成と論理的な「区切り」を分けて管理することで、より自由にシステムを設計出来るようにするためのツールが LVM です。

LVM の使い方について詳しい説明は省きますが、LXC ではコンテナのファイルを LVM で作成したボリュームに保存することが出来ます。

3.3 cgroup ファイルシステムの mount

Ubuntu では必要ありませんが、Debian では LXC から cgroup を使えるようにするため、fstab^{*20}に自分で追記する必要があります。

*19 どの Linux ディストリビューションを模して仮想環境を作成するのか、その方法をディストリビューションごとにまとめたのがテンプレートです。*debian* テンプレートは Debian の仮想環境、*ubuntu* テンプレートは Ubuntu の仮想環境を作成するために使います。詳しくは 4.4 にて後述します。

*20 システム起動時にファイルシステムをどのようにマウントするか、設定を記述するファイル。

/etc/fstab

```
cgroup /sys/fs/cgroup cgroup defaults 0 0
```

追記後、*mount -a* するか、OS を再起動してください。

```
mitty@wheezy-lxc:~$ mount | grep cgroup
cgroup on /sys/fs/cgroup type cgroup (rw,relatime,perf_event,blkio,net_cls,freezer,devices,cpuacct,
cpu,cpuset)
```

の様になつていれば大丈夫です。

4 LXC コンテナの作成と起動

4.1 コンテナの作成

コンテナの作成には *lxc-create* コマンドを用います。

```
mitty@precise-lxc:~$ sudo lxc-create -n hoge
No config file specified, using the default config
'hoge' created
```

この例では、*hoge* という名前のコンテナを作成しています。*-f* というオプションで、コンテナの初期設定ファイルを指定出来ますが、Ubuntu の場合指定しないと lxc パッケージによって用意されたデフォルトの設定ファイルが使われます^{*21}。Debian の場合、設定ファイルは空のファイルとして用意されます。

4.2 コンテナの起動方法

コンテナを起動するコマンドとしては、*lxc-execute* と *lxc-start* の二つのコマンドが用意されています。man 7 lxc によると、

lxc-execute

引数で与えられたコマンドを、*lxc-init* を経由して起動する。*lxc-init* は与えられたコマンドが終了するまで待ち続ける。*lxc-init* の PID は 1 になり、コマンドの PID は 2 になる。

lxc-start

与えられたコマンドをコンテナの環境下で直接起動する。引数が無い場合、*/sbin/init* *22 を起動する。

という違いがあるようです。*lxc-init* が何なのかというと、どうやら/*proc* や/*sys* などプロセスの起動に必要な特殊なファイルシステムのマウントなど、下準備を行ってくれるコマンドのようです。

筆者の感覚としては、

lxc-execute

特定のプログラム一つを、ちょっと起動して少し試したらすぐ終了する場合などに用いると良さそう。

*21 12.04 では/etc/lxc/lxc.conf、13.04 では/etc/lxc/default.conf がデフォルトの設定ファイルです。

*22 システム起動時に最初に起動され、全てのプロセスの親プロセスとなるプログラム。PID が 1 のプロセスがそれ。

lxc-start

LXC を使わない場合であれば新しく物理マシンを用意したり VM を作成したりしていたような、一つの独立したシステムとして用意したコンテナを起動する場合に使用する。

という感じです^{*23}。

4.3 lxc-execute によるコンテナの起動

4.1 で作成した、hoge コンテナの中で ps コマンドを実行してみましょう^{*24}。

```
mitty@precise-lxc:~$ sudo lxc-execute --name hoge /bin/ps ax
  PID TTY      STAT   TIME COMMAND
    1 pts/0    S+        0:00 /usr/lib/lxc/lxc-init -- /bin/ps ax
    2 pts/0    R+        0:00 /bin/ps ax
mitty@precise-lxc:~$
```

-name でコンテナ名を指定します。-n と省略が可能です。

ホスト OS 上で ps ax を普通に実行すると本来は以下の様になるはずですので、ホスト OS と別の仮想環境で動作していることが分かります。

```
mitty@precise-lxc:~$ ps ax
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss        0:00 /sbin/init
    2 ?        S          0:00 [kthreadd]
    3 ?        S          0:00 [ksoftirqd/0]
    4 ?        R          0:02 [kworker/0:0]
    6 ?        S          0:00 [migration/0]
    7 ?        S          0:00 [watchdog/0]

(snip)
```

ps の代わりに bash を指定すれば、ホスト OS で動作している他のプロセスは全く見えない状態でシェルを使うことも出来ます。

```
mitty@precise-lxc:~$ sudo lxc-execute -n hoge /bin/bash
root@precise-lxc:/home/mitty# pstree -aAp
lxc-init,1 -- /bin/bash
`-bash,2
  `-pstree,12 -aAp
root@precise-lxc:/home/mitty# exit
mitty@precise-lxc:~$
```

なお、コンテナ名として lxc-create で作成していないコンテナ名を指定すると、デフォルトの設定で一時的にコンテナを作成し、コマンド終了時に自動的に削除するようです。

*23 man 7 lxc でも「To summarize, lxc-execute is for running an application and lxc-start is for running a system.」と書かれています。

*24 筆者の環境では、Ubuntu 13.04においては AppArmor のせいかうまく動きませんでした。また、Debian 7 ではなぜか ps コマンドの PID が 3 になりました。

4.4 テンプレートを使用したコンテナの作成

4.3において、lxc-execute でコンテナを起動し、任意のプロセスを仮想環境に隔離する方法を紹介しました。しかし、実はこの方法では元々あったファイルは見えてしまい、ネットワークも完全には分離されません^{*25}。

```
mitty@precise-lxc:~$ ls -l
total 34820
-rw-r--r-- 1 mitty mitty 35651584 Oct 16 2012 mini.iso
drwxr-xr-x 2 mitty mitty 4096 May 17 07:16 works
mitty@precise-lxc:~$ brctl show
bridge name      bridge id          STP enabled     interfaces
lxcbr0           8000.de5f56d06c57    no            vethGMsUJh
virbr0           8000.000000000000    yes
mitty@precise-lxc:~$ sudo lxc-execute -n hoge /bin/bash
root@precise-lxc:/home/mitty# ls -l
total 34820
-rw-r--r-- 1 mitty mitty 35651584 Oct 16 2012 mini.iso
drwxr-xr-x 2 mitty mitty 4096 May 17 07:16 works
root@precise-lxc:/home/mitty# brctl show
bridge name      bridge id          STP enabled     interfaces
lxcbr0           8000.d6cd8f232797    no            vethGMsUJh
                                         vethaq0Hw9
virbr0           8000.000000000000    yes
```

従って LXC では通常、システムライブラリファイルや各種アプリケーションの実行ファイルをホスト OS とは別に、コンテナごとに配置してからコンテナを起動します。ライブラリファイルや実行ファイル、あるいはそれらをインストールするためにどのパッケージを導入するか、そしてパッケージの導入方法などはディストリビューションごとに違うため、3.1 で少し触れたように、特定のディストリビューションを模したコンテナを作成するにはテンプレートを使うのが一般的です。

テンプレートを使ったコンテナの作成では、大まかに以下のことが行われます。

1. コンテナの動作に必要なファイルを、*debootstrap* などを使って配置する
2. ホスト OS のリソースへのアクセス可否を定義した設定ファイルを用意する

テンプレートを使わなくとも、1 と 2 のいずれも自分でコマンドを叩いても良いのですが、面倒ですし何より間違える可能性があります。そのため、ある程度決まった手順でコンテナが作成出来るよう、lxc-create では-t でテンプレートを指定出来ます。

テンプレートの実体はシェルスクリプトで、以下に置かれています。

Debian 6, Ubuntu 12.04 /usr/lib/lxc/templates/

Debian 7, Ubuntu 13.04 /usr/share/lxc/templates/

*25 より厳密に隔離された状態でプロセスを起動するために、スクリプトを経由して lxc-execute するという方法があるようです。

<http://shyouhei.tumblr.com/post/467629329/lxc-execute>

ト部昌平のあまり reblog しない tumblr-lxc についてくる lxc-execute(1) がお手軽すぎてヤバい

例えば Ubuntu 13.04 では次の様になっています。

```
mitty@raring-lxc:~$ ls -1 /usr/share/lxc/templates/
lxc-alpine
lxc-altlinux
lxc-archlinux
lxc-busybox
lxc-debian
lxc-fedora
lxc-opensuse
lxc-oracle
lxc-sshd
lxc-ubuntu
lxc-ubuntu-cloud
```

様々なテンプレートがありますが、筆者は普段 *ubuntu* テンプレートを使っているので、まずはそれを使って *ubuntu01* という名前でコンテナを作成してみましょう。いずれのテンプレートもファイル名が *lxc-*で始まっていますが、テンプレートとして指定する場合はその部分を除き、例えば *ubuntu* とだけ指定します。

インターネット回線の速度にもよりますが、だいたい 10 分程度で完了すると思います。

```
mitty@precise-lxc:~$ sudo lxc-create -t ubuntu -n ubuntu01

No config file specified, using the default config
debootstrap is /usr/sbin/debootstrap
Checking cache download in /var/cache/lxc/precise/rootfs-amd64 ...
installing packages: vim,ssh
Downloading ubuntu precise minimal ...
I: Retrieving Release

(snip)

I: Checking component main on http://archive.ubuntu.com/ubuntu...
I: Retrieving adduser

(snip)

I: Base system installed successfully.
Installing updates

(snip)

Download complete
Copy /var/cache/lxc/precise/rootfs-amd64 to /var/lib/lxc/ubuntu01/rootfs ...
Copying rootfs to /var/lib/lxc/ubuntu01/rootfs ...

##  
† # The default user is 'ubuntu' with password 'ubuntu'!
# Use the 'sudo' command to run tasks as root in the container.
##

'ubuntu' template installed
'ubuntu01' created
```

下線で示したように、debootstrap でダウンロードしたパッケージをキャッシュしておき、二回目以降はそこからコピーするためコンテナの作成がとても速くなります。また†の行では、ubuntu というユーザが ubuntu というパスワー

ドで作成されたと表示されています*26。

4.5 コンテナの作成オプション

lxc-create -h でコンテナ作成時に指定可能な基本的なオプションが表示されます。テンプレート使用時に追加で指定可能なオプションは、*lxc-create -t template-name -h* で表示されます。

```
mitty@precise-lxc:~$ lxc-create -t ubuntu -h
usage: lxc-create -n <name> [-f configuration] [-t template] [-h] -- [template_options]
usage: lxc-create -n <name> [-f configuration] [-t template] [-h] [fsopts] -- [template_options]
      fsopts: -B none
      fsopts: -B lvm [--lvname lvname] [--vgname vgname] [--fstype fstype] [--fssize fssize]
      fsopts: -B btrfs
              flag is not necessary, if possible btrfs support will be used

creates a lxc system object.

Options:
name       : name of the container
configuration: lxc configuration
template   : lxc-template is an accessible template script

The container backing store can be altered using '-B'. By default it
is 'none', which is a simple directory tree under /var/lib/lxc/<name>/rootfs
Otherwise, the following option values may be relevant:
lvname     : [for -lvm] name of lv in which to create lv,
              container-name by default
vgname     : [for -lvm] name of vg in which to create lv, 'lxc' by default
fstype    : name of filesystem to create, ext4 by default
fssize     : size of filesystem to create, 1G by default

template-specific help follows: (these options follow '--')
/usr/lib/lxc/templates/lxc-ubuntu -h|--help [-a|--arch] [-b|--bindhome <user>] [--trim] [-d|--debug]
      [-F | --flush-cache] [-r|--release <release>] [ -S | --auth-key <keyfile>]
release: the ubuntu release (e.g. precise): defaults to host release on ubuntu, otherwise uses latest
      LTS
trim: make a minimal (faster, but not upgrade-safe) container
bindhome: bind <user>'s home into the container
          The ubuntu user will not be created, and <user> will have
          sudo access.
arch: the container architecture (e.g. amd64): defaults to host arch
auth-key: SSH Public key file to inject into container
```

書いてあるとおりですが、一部説明すると、

-B 4.4 で述べたライブラリファイルや実行ファイルを、ホスト OS 上でどのように格納するか指定出来ます。デフォルトでは/var/lib/lxc/**container-name**/rootfs以下*27に直接保存されますが、3.2 で少し触れたように、LVM を使ってホスト OS とは別のパーティション*28 に格納することも出来ます。

*26 アカウント名とパスワードはテンプレートに埋め込まれているため、*lxc-create* のコマンドラインオプションなどで変更することは不可能です。テンプレートを改造すればいいのですが、時間の関係で本記事ではそこまで触れることができませんでした。

*27 Debian 7 では *lxc* パッケージのインストール時に任意の場所へ変更可能です。

*28 LVM は logical volume manager の略なので、正確には「ポリューム」と呼びます。

-F | -flush-cache

4.4で述べたとおり、LXC ではコンテナにインストールされるパッケージをキャッシュし、次からはダウンロードしなくて良いようにします。一度キャッシュすると基本的にアップデートされないため、コンテナ作成後にコンテナ内で *aptitude update; aptitude upgrade* する必要があつて面倒だつたりします。手動で/var/cache/lxc/以下のファイルを削除しても良いのですが、それをコンテナ作成時に同時にやってくれるオプションです。

-r | -release

ubuntu テンプレートでは、コンテナにインストールされるパッケージはデフォルトでは最新の LTS リリースのもの、すなわち執筆時では 12.04 になりますが、これを任意のリリースへ変更することが可能です。

テンプレートオプションはテンプレートごとに違いますので、詳しくは実際に実行してみてください。

4.6 lxc-start によるコンテナの起動

作成したコンテナを *lxc-start* で起動してみましょう。*lxc-create* した時に表示されたとおり、アカウント、パスワードとも *ubuntu* でログイン出来ます。

```
mitty@precise-lxc:~$ sudo lxc-start -n ubuntu01

Ubuntu 12.04.2 LTS ubuntu01 console

ubuntu01 login: ubuntu
Password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-41-generic x86_64)

(snip)

ubuntu@ubuntu01:~$ ps ax
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:00 /sbin/init
   88 ?        S      0:00 upstart-socket-bridge --daemon
  125 ?        S      0:00 upstart-udev-bridge --daemon
  140 ?        Ss    0:00 /sbin/udevd --daemon
  147 ?        S1    0:00 rsyslogd -c5
  182 ?        Ss    0:00 dhclient3 -e IF_METRIC=100 -pf /var/run/dhclient.eth0
  204 ?        Ss    0:00 /usr/sbin/sshd -D
  230 ?        Ss+   0:00 /sbin/getty -8 38400 tty4
  234 ?        Ss+   0:00 /sbin/getty -8 38400 tty2
  236 ?        Ss+   0:00 /sbin/getty -8 38400 tty3
  239 ?        Ss    0:00 cron
  255 ?        Ss    0:00 /bin/login --
  259 ?        Ss+   0:00 /sbin/getty -8 38400 tty1
  280 lxc/console S      0:00 -bash
  290 lxc/console R+   0:00 ps ax
ubuntu@ubuntu01:~$ brctl show
bridge name      bridge id               STP enabled     interfaces
ubuntu@ubuntu01:~$ ifconfig -a | egrep 'Link|addr'
eth0      Link encap:Ethernet HWaddr 00:16:3e:54:31:50
          inet addr:10.0.3.32  Bcast:10.0.3.255  Mask:255.255.255.0
          inet6 addr: fe80::216:3eff:fe54:3150/64 Scope:Link
lo       Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
```

lxc-execute の時と同様、コンテナごとに独自のプロセステーブルになっていることが分かります。また、ネットワークも分離されています。Ubuntu の場合、デフォルトではホスト側に lxcbr0 という仮想ブリッジデバイス^{*29}が用意され、それを通じて NAPT^{*30} で外のネットワークへ抜けられるようになります。

なお、ubuntu テンプレートの場合 sshd がインストールされるので、ホスト OS 側から ssh を使ってリモートログインすることも可能です。

```
mitty@precise-lxc:~$ ssh ubuntu@10.0.3.32
The authenticity of host '10.0.3.32 (10.0.3.32)' can't be established.
ECDSA key fingerprint is 94:67:83:e2:c4:f9:d4:ff:ad:48:aa:e4:37:67:89:04.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.3.32' (ECDSA) to the list of known hosts.
ubuntu@10.0.3.32's password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.2.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
Last login: Fri May 24 07:54:02 2013
ubuntu@ubuntu01:~$
```

4.7 コンテナの終了

ホスト OS と同様、shutdown コマンド^{*31}によって終了、再起動が可能です。

```
ubuntu@ubuntu01:~$ sudo shutdown -h now
[sudo] password for ubuntu:
ubuntu@ubuntu01:~$
Broadcast message from ubuntu@ubuntu01
(/dev/lxc/console) at 8:04 ...

The system is going down for halt NOW!
* Asking all remaining processes to terminate... [ OK ]
* All processes ended within 1 seconds... [ OK ]
* Deconfiguring network interfaces... [ OK ]
* Deactivating swap... [ fail ]
umount: /run/lock: not mounted
umount: /run/shm: not mounted
mount: cannot mount block device /dev/disk/by-uuid/53e5aa6f-282a-4a0c-88ab-7cbd88ff355a read-only
* Will now halt
mitty@precise-lxc:~$
```

あるいは、ホスト OS から lxc-shutdown コマンドによって制御することも出来ます^{*32}。

```
mitty@precise-lxc:~$ sudo lxc-shutdown -n ubuntu01
```

また、ホスト OS から lxc-stop コマンドによって強制終了させることも出来ます。

Debian 6 の場合、lxc-stop コマンドしか用意されていませんでした。また、ホスト OS のネットワークデバイスが隔離されずにコンテナ内で直接使用されているせいか、コンテナの終了あるいは再起動を行うと、ホスト OS のネットワークがダウンするという現象が見られました。設定ファイルを編集するなどしないと使いにくいですね……。

^{*29} イーサネットレイヤ（L2）でデータを中継するための仮想デバイス。

^{*30} Network Address Port Translation のこと。内側（この場合コンテナ）と外側（ホスト OS が繋がっているネットワーク）で異なるネットワークを構築する場合に、その間でアドレス変換をして中継するための仕組み。Linux では「IP マスクレード」とも呼ばれる。

^{*31} 筆者は主に poweroff と reboot コマンドを使っています。

^{*32} 具体的には終了の場合 SIGPWR シグナル、再起動の場合 SIGINT シグナルがコンテナに送られているようです。

4.8 コンテナのデーモン化

`lxc-start` を使ってコンテナを起動するとコンテナのコンソールに入出力が奪われてしまい、他に端末を立ち上げない限りホスト OS では他のことが出来なくなってしまいます。そのままコンテナを動作させつつホスト OS で別の作業を行うために、`lxc-start` にはコンテナをデーモン化するための`-d` オプションがあります。

```
mitty@precise-lxc:~$ sudo lxc-start -n ubuntu01 -d
mitty@precise-lxc:~$
```

これで、バックグラウンドでコンテナが起動します。

起動しているコンテナも含め、現在ホスト OS 上で利用可能なコンテナの一覧を得るには、`lxc-list` を使います。

```
mitty@precise-lxc:~$ sudo lxc-list
RUNNING
    ubuntu01

FROZEN

STOPPED
    fedora01
    hoge
    lucid01
    ubuntu02
```

Ubuntu 13.04 で `lxc-list` を使うと、「`lxc-list` は廃止予定なので `lxc-ls --fancy` を使え」と表示されます。コンテナに割り当てられた IP アドレスも表示されるので `lxc-list` より便利です。

```
mitty@raring-lxc:~$ sudo lxc-ls --fancy
NAME      STATE     IPV4          IPV6   AUTOSTART
-----
hoge      STOPPED   -           -       NO
ubuntu01  RUNNING   10.0.3.14  -       NO
```

デーモン化したコンテナの場合も、デーモン化していない時と同様コンテナの中で `shutdown` するか、`lxc-shutdown` あるいは `lxc-stop` を使って終了させることができます。

4.9 lxc-console によるコンテナのコンソールへの接続

コンテナをデーモン化しない場合、`lxc-start` によってコンテナのコンソールへ自動的に接続されますが、コンテナをデーモン化した場合は `lxc-console` コマンドによってコンテナのコンソールへ接続することが出来ます。

```
mitty@precise-lxc:~$ sudo lxc-console -n ubuntu01
Type <Ctrl+a q> to exit the console

Ubuntu 12.04.2 LTS ubuntu01 tty1

ubuntu01 login:
```

コンソールから切断するには、下線 の通り `Ctrl+a q` とタイプします。

4.10 コンテナの自動起動と自動終了

ホスト OS の起動と終了にあわせてコンテナを自動的に起動し、終了させることも出来ます。

```
mitty@precise-lxc:~$ sudo ln -s /var/lib/lxc/ubuntu01/config /etc/lxc/auto/ubuntu01.conf
```

この例では、*ubuntu01* コンテナを自動起動、自動終了するようにしています。

```
mitty@precise-lxc:~$ sudo lxc-list
RUNNING
    ubuntu01 (auto)

FROZEN

STOPPED
    fedora01
    lucid01
    ubuntu02
```

ただし、筆者の経験では比較的頻繁（2割くらい）に自動起動に失敗するようです。一部のコンテナだけ起動してその他はなぜか自動起動しない、ということもあるのであまり信頼しない方が良いかもしれません。

5 LXC で出来ることと出来ないこと

さて、前章まででホスト OS とは別の仮想環境が用意出来たわけですが、この仮想環境を使ってどういうことが出来るか、あるいは出来ないか、紹介していきたいと思います。

なお、この章以降、筆者が普段使っている Ubuntu 12.04 以外ではほとんど検証していませんので、あらかじめご了承ください。

5.1 ホスト OS とは異なるリリースバージョンを入れてみる

ubuntu テンプレートを使うと、指定しない限り LTS の 12.04 のコンテナが作成されます。これを、例えば 10.04 にしてみましょう。リリースバージョンの指定には precise や lucid といったコードネームを使用します。

```
mitty@precise-lxc:~$ sudo lxc-create -t ubuntu -n lucid01 -- -r lucid
```

-r で指定するだけです。簡単ですね。テンプレートにオプションを渡す場合は、ハイフン二つの後に指定する点に注意してください。

バージョンダウンして何が嬉しいのかと言われそうですが、例えばビルドするのに古いコンパイラが必要なソフトウェアがあると考えてください。ここでは Linux カーネル 2.6.32.12 を例にしてみます^{*33}。

2.6.32.12 はかなり古い^{*34}ため、Ubuntu 12.04 や 13.04 のリポジトリからインストールされる gcc ではビルドすることが出来ません。

^{*33} 特定のバージョンのカーネルに依存するカーネルモジュールがあってですね……。

^{*34} <https://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.32.12.tar.xz> のタイムスタンプは「26-Apr-2010」となっています。古いつていうレベルじゃねえぞ！

```
mitty@kernel-precise:~/works/linux-2.6.32.12$ gcc -v
(snip)

gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1ubuntu5)
mitty@kernel-precise:~/works/linux-2.6.32.12$ make deb-pkg
(snip)

make KBUILD_SRC=
  CHK      include/linux/version.h
  CHK      include/linux/utsrelease.h
  SYMLINK include/asm -> include/asm-x86
  CALL    scripts/checksyscalls.sh
  CHK      include/linux/compile.h
  VDSO    arch/x86/vdso/vdso.so.debug
gcc: error: elf_x86_64: No such file or directory
make[4]: *** [arch/x86/vdso/vdso.so.debug] Error 1
make[3]: *** [arch/x86/vdso] Error 2
make[2]: *** [arch/x86] Error 2
make[1]: *** [deb-pkg] Error 2
make: *** [deb-pkg] Error 2
```

しかし、ここで lucid コンテナを用意して同じように make すると、問題なく deb パッケージファイルをビルドすることが出来ます。

```
mitty@kernel-lucid:~/works/linux-2.6.32.12$ gcc -v
(snip)

gcc version 4.4.3 (Ubuntu 4.4.3-4ubuntu5.1)
mitty@kernel-lucid:~/works/linux-2.6.32.12$ sudo*35 make deb-pkg
(snip)

dpkg-deb: building package 'linux-image-2.6.32.12' in '../linux-image-2.6.32.12_2.6.32.12-3_amd64.deb'
.'
```

また、ホスト OS より新しいリリースバージョンを導入することももちろん可能です。本記事の組版には TeXLive 2012 を使っている^{*36} のですが、TeXLive 2012/Debian の導入方法を紹介しているサイト^{*37} で指定されている *texlive-lang-cjk* パッケージが Ubuntu 12.10 以降にしか存在せず、仕方がないので 12.04 上に 12.10 のコンテナを用意してコンパイルしています。

```
mitty@word-tex:~$ uname -a
Linux word-tex 3.2.0-43-generic #68-Ubuntu SMP Wed May 15 03:33:33 UTC 2013 x86_64 x86_64 x86_64 GNU/
  Linux
mitty@word-tex:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 12.10
Release:        12.10
Codename:       quantal
```

^{*35} sudo しないと、ビルドの終盤で chown が出来ずに make が失敗するため、sudo しています。本当は sudo せずに fakeroot 等を使って回避すべきなのでしょうが、面倒くさいので……。

^{*36} TeX による組版については WORD25 号の「WORD 氏のための LATEX2ε 入門」が大変参考になります。

^{*37} <http://oku.edu.mie-u.ac.jp/~okumura/texwiki/?Linux/Ubuntu>

5.2 ホスト OS とは異なるディストリビューションを入れてみる

試しに *fedora* テンプレートを使って *fedora01* コンテナを作成してみましょう。3.2 で述べたとおりホスト OS 側に *yum* と *curl* が必要なので、あらかじめ導入してください。

```
mitty@precise-lxc:~$ sudo lxc-create -t fedora -n fedora01

No config file specified, using the default config
This is not a fedora host and release missing, defaulting to 14. use -R|--release to specify release
Checking cache download in /var/cache/lxc/fedora/x86_64/14/rootfs ...
Downloading fedora minimal ...
Fetching from http://ftp.tsukuba.wide.ad.jp/Linux/fedora-archive/fedora/linux/releases/14/Everything/
x86_64/os//Packages/fedora-release-14-1.noarch.rpm

(snip)

Complete!
Download complete.
Copy /var/cache/lxc/fedora/x86_64/14/rootfs to /var/lib/lxc/fedora01/rootfs ...
Copying rootfs to /var/lib/lxc/fedora01/rootfs ...setting root passwd to root
installing fedora-release package
Error: Cannot retrieve repository metadata (repomd.xml) for repository: fedora. Please verify its
      path and try again
container rootfs and config created
'fedora' template installed
'fedora01' created
mitty@precise-lxc:~$
```

ちょっとエラーが出ていますが、とりあえず作成出来たようなので起動してみましょう。上記の 下線 で示された通り、アカウント、パスワードともに *root* です。

```
mitty@precise-lxc:~$ sudo lxc-start -n fedora01
      Welcome to Fedora
Setting hostname fedora01:                                     [ OK ]
Checking filesystems                                         [ OK ]
mount: can't find / in /etc/fstab or /etc/mtab
Mounting local filesystems:                                    [ OK ]
Enabling /etc/fstab swaps:                                    [ OK ]
Entering non-interactive startup
Bringing up loopback interface:                               [ OK ]
Bringing up interface eth0:
Determining IP information for eth0... done.                  [ OK ]
Starting system logger:                                       [ OK ]
Mounting other filesystems:                                    [ OK ]

Fedora release 14 (Laughlin)
Kernel 3.2.0-41-generic on an x86_64 (console)

fedora01 login: root
Password:
[root@fedora01 ~]# ps axw
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:00 /sbin/init
  219 ?        Ss     0:00 /sbin/dhclient -1 -q -lf /var/lib/dhclient/dhclient-eth0.leases -pf /var/run/dhclient-eth0.pid eth0
  269 ?        S1     0:00 /sbin/rsyslogd -c 4
```

```

304 ?      Ss      0:00 login -- root
306 tty1   Ss+    0:00 /sbin/mingetty /dev/tty1
308 tty2   Ss+    0:00 /sbin/mingetty /dev/tty2
310 tty3   Ss+    0:00 /sbin/mingetty /dev/tty3
312 tty4   Ss+    0:00 /sbin/mingetty /dev/tty4
313 console Ss     0:00 -bash
325 console R+     0:00 ps axw
[root@fedora01 ~]# ifconfig -a| egrep 'Link|addr'
eth0      Link encap:Ethernet HWaddr 0A:A0:D8:2B:96:24
          inet addr:10.0.3.49 Bcast:10.0.3.255 Mask:255.255.255.0
          inet6 addr: fe80::8a0:d8ff:fe2b:9624/64 Scope:Link
lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
[root@fedora01 ~]# shutdown -h now
/proc/self/fd/9: line 2: /sbin/plumouthd: No such file or directory
initctl: Event failed
Shutting down interface eth0:                                [ OK ]
Shutting down loopback interface:                            [ OK ]
Sending all processes the TERM signal...                  [ OK ]
Sending all processes the KILL signal...                [ OK ]
Saving random seed:                                     [ OK ]
Syncing hardware clock to system time Cannot access the Hardware Clock via any known method.
Use the --debug option to see the details of our search for an access method.
                                         [FAILED]
Turning off swap:  swapoff: /dev/sda5: swapoff failed: No such file or directory
                                         [FAILED]
umount2: Device or resource busy
umount: /dev/console: device is busy.
        (In some cases useful info about processes that use
         the device is found by lsof(8) or fuser(1))
Halting system...
mitty@precise-lxc:~$
```

ubuntu テンプレートを使ったコンテナと同様、問題なく起動、終了出来ました。

5.3 LXC で出来ないこと

しつこいようですが LXC ではホスト OS とカーネルを共有しているため、VM ではごく普通に出来ることが出来なかつたりします。ただし、Ubuntuにおいては AppArmor によって制限されているだけの場合もありますので、AppArmor を適切に設定することで可能になることもあります。もちろん AppArmor の制限を緩めると、その分コンテナの中からホスト OS へ影響を与えることが容易になるため注意が必要です。

例としては以下が挙げられます。

デバイスの *mount*

NFS や block デバイス、loopback デバイス、あるいは CIFS などのリソースは使う際にローカルのディレクトリにマウントする必要がありますが、少なくとも設定を変更しない限りコンテナの中ではマウント出来ません。Ubuntu の場合、loopback デバイスは AppArmor の設定を変更^{*38} することでコンテナの中でもマウン

^{*38} <http://www.mail-archive.com/lxc-users@lists.sourceforge.net/msg03671.html>
Re: [Lxc-users] loop mount inside container

ト出来るようになります。NFS および CIFS については、色々試行錯誤してみましたが筆者の環境ではマウントすることは出来ませんでした。

カーネルモジュールのロード

VMware Workstation for Linux など独自のカーネルモジュールを使っている場合、そのカーネルモジュールをコンテナの中でロードすることは出来ません。

6 参考文献

本記事を執筆するにあたって参考にしたサイトなどを紹介したいと思います。

本記事では詳しくは触れませんでしたが、RHEL およびそのクローンの一つである CentOS に LXC を導入する場合、ソースからビルドする必要があったり、SELinux のポリシーを手動で設定したりと、少し難易度が高いようでした。

http://gihyo.jp/admin/column/01/vm/2011/lxc_container

「仮想化ソフトウェアと chroot の “いいとこ取り” — LXC で実現する VPS」 | gihyo.jp … 技術評論社」

VM/VMM と LXC、あるいはその他のコンテナ技術の違いに焦点をあてて、LXC の紹介をしています。

<http://gihyo.jp/admin/serial/01/ubuntu-recipe/0226>

「第 226 回 LXC で軽量仮想環境の活用：Ubuntu Weekly Recipe」 | gihyo.jp … 技術評論社」

Ubuntu 12.04 LTS 上での LXC を導入の仕方を、具体的な例を交えて紹介しています^{*39}。

<http://www.slideshare.net/enakai/lxc-8300191>

「Lxc で始めるケチケチ仮想化生活？！」

LXC がどのようにしてコンテナごとにプロセスやネットワークを分離しているのか、また CPU やディスク I/O、ネットワークといったリソースを制限する際にどのように行えばよいのか紹介されています。

http://www.slideshare.net/masahide_yamamoto/osc2011-nagoya

「LXC 入門 - Osc2011 nagoya」

コンテナの中からホスト OS を制御出来てしまう抜け道について触っています。なお、Ubuntu の各バージョンでどのようにして抜け道が塞がれているかは、<https://wiki.ubuntu.com/LxcSecurity> を参照すると良いでしょう。

<http://d.hatena.ne.jp/defiant/20111228/1325067535>

「lxc の仮想ネットワークのパフォーマンス測定 - TenForward の日記」

コンテナのネットワークを様々な条件で設定して性能を評価しています。1000BASE だと、もう劇的な差は出でないようにも見えます^{*40}。

<https://help.ubuntu.com/12.04/serverguide/lxc.html>

Ubuntu 12.04 向けの公式マニュアルです。

*39 ぶっちゃけこれがあれば本記事は要らない子のような……。

*40 10G が一般家庭まで普及してくるのっていつになるんだろう……。本記事とは離れますぐ、どちらかというと VM で SR-IOV する話とかの方が筆者としては興味深いですね。

<http://d.hatena.ne.jp/defiant/20111213/1323771409> SR-IOV を有効にする (3) - TenForward の日記

<https://help.ubuntu.com/13.04/serverguide/lxc.html>

Ubuntu 13.04 向けの公式マニュアルです。

12.04 のマニュアルと比べて、コンテナの起動時、終了時などに指定したプログラムを実行出来るようになったこと、*python3-lxc* パッケージによって Python からコンテナを扱えるようになったことなどが追記されています。

<http://wiki.debian.org/LXC>

Debian 向けの公式マニュアルです。

<http://wiki.centos.org/HowTos/LXC-on-CentOS6>

CentOS 6 向けの公式マニュアルです。

<http://hirolovesbeer.hatenablog.jp/entry/2012/01/06/073431>

「CentOS6.2 で LXC - hirolovesbeer's diary」

<http://d.hatena.ne.jp/enakai00/20110529/1306658627>

「RHEL6.0 で LXC (Linux コンテナ) - めもめも」

<http://inokara.hateblo.jp/entry/2013/03/31/172636>

「Ubuntu 12.10 で CentOS 6 の LXC テンプレートを試す一部始終 - ようへいの日々精進」

7 最後に

本当は今回の記事で、コンテナのより具体的な利用法や周辺ツール、テンプレートの改造、コンテナ内からホスト OS が制御する方法や、cgroup によるリソース制限など、よりディープな内容にも踏み込んだかったのですが、圧倒的に時間が足りませんでした。よって続き（があれば）それらについて触れたいと思います。

7.1 おまけ

LXC はオーバーヘッドが無いから VM より速いよ、と既に何度か触れましたが、ではどれくらい違うのか簡単にですがベンチマークを取ってみました。準備が面倒だったので Linux カーネルのビルド^{*41}のみです。

表 1: 物理マシンのスペック

Motherboard	MSI X58 Platinum (MS-7522)
CPU	Intel Core i7 940 2.93GHz
Memory	Hynix PC3-8500 2GB x6
Storage (RAID1)	WDC WD5000AAKS-00YGA0 Hitachi HDP725050GLA360
OS	Ubuntu 12.04.2 LTS x86_64

^{*41} CPU、メモリに関しては VM でもあまりオーバーヘッド無いだろうし、たぶんディスク I/O くらいしか違つてこないと思われる……。

表 2: QEMU-KVM による仮想環境

VMM	qemu-kvm 1.0+noroms-0ubuntu14.8
CPU 割り当て	8 cores
Memory 割り当て	8096MB
Storage 割り当て	64GB with VirtIO
OS	Ubuntu 12.04.2 LTS x86_64

VM の作成は virt-manager を使い、OS type : Linux、Version : Ubuntu 12.04 LTS として行いました。

カーネルのビルドは以下の手順で行っています。

```
$ cp /boot/config-3.2.0-44-generic .config
$ yes '' | make oldconfig
$ /usr/bin/time make deb-pkg -j9
```

表 3: make deb-pkg に要した時間 (分:秒 . ミリ秒)

バージョン	物理マシン	VM	LXC
3.2.45	29:12.630	34:17.870	29:10.320
3.9.3	28:13.270	33:35.550	28:02.950
3.10-rc2	26:24.800	31:54.250	26:32.290

LXC であれば、実機と同様の性能を発揮出来ることが分かるかと思います。

高知×鳥取の鼓動

文 編集部 ジオン

1 はじめに

2013年3月24日。私はあるイベントに参加するため、編集部の無季氏、UTM氏と共に秋葉原を訪れていた。そのイベントとは『まんがの聖地と呼ばれたい！高知×鳥取 in AKIBA』。自称まんが王国鳥取県が、同じくまんが王国を名乗っている高知県と修好通商条約を結ぶための式典が行われたのである。今回はこのイベントを軽くまとめようと思う。

2 イベントレポート

実はまんが王国と最初に名乗ったのは高知県である。高知県は『アンパンマン』のやなせたかし氏や『フクちゃん』の横山隆一氏を始めとし、数々の漫画家を輩出した県であり、人口あたりの漫画家数では全国1位である。そのため漫画に力を入れており、毎年高校生の中から有望な漫画家のたまごを発掘する『まんが甲子園』なども開催されている。このようにまんが王国を名乗ってきていた高知県だったが、最近になり鳥取県もまんが王国を名乗り始めたことを知る。そこで、両県知事の話し合いの結果、まんが王国同士互いに協力することが決定され、今回のイベントが開かれることになった。

イベント当日は曇りで、野外ステージで行われる会場は非常に冷え込んでいたが、私の予想に反して到着したときには会場には10人ほど^{*1}の人だかりができていた。ステージにはでかでかとイベントのタイトルと共に両県知事の握手をする面白いイラストが描かれていた。これは文章で表現するよりも見てもらったほうが早いため写真を掲載する。何故か両県民がいがみ合っている点、「まんがの聖地！」ではなく「まんがの聖地と呼ばれたい！」と謙虚な姿勢をとっている点が面白い。



まんがの聖地と呼ばれたい!! 高知×鳥取

あそぼ♪

いつしょに

©小村博明・くさな里相

高知×鳥取のイベントで何故秋葉原かはつこんではいけない

*1 すごい！人がいる！！

開演前から並んでいた人々にイベントの重役と思われる人間(のちに司会とわかる^{*2})がイベントに来た目的を尋ねて周っていたが、ほとんどの人間がこのイベントに呼ばれる下田麻美氏^{*3}、明坂聰美氏^{*3}、2名の声優のトークショーが本命だったらしく、純粋に鳥取がらみで来た我々は少数派だった。

11時30分、物品販売開始と同時に開場。我々はステージの前方の席に座ることができた。イベント開始後は立ち見の人が大勢いたことを考えると幸いだった。

12時00分、イベントが開始される。イベント自体は両県知事が出演するからか、人気声優が出演するからか、(ご当地アイドルがいるからか、)わからないが、撮影禁止となっていた。そのため記事に使えそうな写真が撮れなかつたため、ここからは簡略してイベントの状況を伝える。

イベントは前半の部、後半の部で分かれており、それぞれで地方アイドルによるライブショー、声優によるトークショーが行われた。アイドルのライブショーでは高知から『はちきんガールズ』、鳥取から『バードプリンセス』『Chelip』が参加。これまでライブショーといったものに縁のなく、会場の雰囲気に慣れない私たたが、色々な意味で興味深いものだった。~~アイドルってエターナルフォースブリザードが使えるんですね~~^{*4}。2名の声優によるトークショーはほとんどの来場者の本命だっただけあり、異様な盛り上がりを見せた。その時だけ会場の人口密度が異常にあがったことから流石秋葉原といったところである。下田氏は鳥取県出身、明坂氏は両親が高知県出身ということで呼ばれており、決して人集めのために呼ばれたわけではないということを補足しておく。内容、観客共に前半後半ほぼ同じだったため、後半の必要性がわからなかったが、盛り上がったのでよしとする。

前半後半が終わると高知県知事尾崎正直氏と鳥取県知事平井伸治氏がそれぞれ坂本龍馬、大国主命^{*5}のコスプレをしてステージに登場。そこに下田氏に加え^{*5}高知出身のベテラン声優島本須美氏^{*6}が続いて登壇し、条約締結式が始まった。まず両県知事がここに至るまでの経緯を説明する。尾崎正直氏は簡潔で要点を押さえた話をし、カリスマ性あふれる人物、平井伸治氏は話の中に自虐ネタ^{*7}や時事ネタを含ませるなど頭の回転が速い、コミカルな人物という印象を受けた。

経緯の説明が終わると条約の立会人として『あしたのジョー』の作者として知られるしばてつや氏^{*8}が登場。観

*2 司会はサブカル好きで一部に有名らしいNHK長野のアナウンサー松岡忠幸氏

*3 代表作『THE IDOLM@STER』双海亜美・真美など

*3 代表作『メルルのアトリエ ハーランドの錬金術士3』メルルなど

*4 相手は死ぬ

*5 神話『因幡の白うさぎ』の登場人物。因幡とは鳥取県東部の旧国名である

*5 高知県出身ではないからかこの時明坂氏は登壇しなかった

*6 代表作『風の谷のナウシカ』ナウシカなど

*7 鳥取県が唯一スタバの無い県になったことに対し、「スタバはなら日本一のものがある」とコメントした知事だが、実は日本一大きい砂丘は青森県の猿ヶ森砂丘。鳥取砂丘は観光できる砂丘として日本一なのだ

*8 しばてつや氏は東京出身で高知県、鳥取県ともに関係性はない。なぜしばてつや氏が呼ばれたかはわからない

衆と報道陣が見守る中、両県知事がサインを交わし、条約が締結された。

報道陣が県知事の握手する姿を撮影し終えると、高知出身のやなせたかし氏、鳥取出身の水木しげる氏のビデオメッセージが紹介された。長い時を生きているお二人のビデオメッセージはとてもインパクトのあるもので、私の拙い文章力では到底表現できるようなものではなかった。式典の様子はニコニコ生放送で中継されてた。気になる方はニコニコ動画で視聴できるかもしれない各自調べてみて欲しい。

ビデオメッセージが終り、ゲストとのジャンケン大会が行われた後、全てのイベントが終了した。

3 最後に

条約締結から約2ヶ月。まんが王国両国の動きを軽くインターネットで調べてみたが、未だ目立った動きはないようだ。しかし、式で両県知事はまんが王国サミットを共催したいと述べていたので近いうちに何かあるかもしれない。各県としては、鳥取県は今年の7月13日から8月25日まで『まんが博・乙』⁹を開催、高知県は今年の11月2日と3日にマンガフェスティバル『まんさい』を開催することを決定している。

言わずもがな鳥取県は人口がもっとも少ない県である。そして高知県は行ったことがない県1位らしい¹⁰。共に厳しい行政をまんがによって改善しようと励む両県はこれを機に協力して頑張っていってもらいたいものである。

*9 まんが博(無印)についてはシリーズ第1回で軽く触れているので是非読んでみて欲しい。WORDのバックナンバーは

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

*10 旅行誌「じやらん」調べ

自己をプリキュアとして高めるための素敵なグッズ紹介

文 編集部 みみずのひもの

1 導入

もう数えで二十になろうという私みみずのひものは、お恥ずかしながらプリキュアというコンテンツを趣味にしている。趣味と言っても特に大した活動をしているわけではない。ただほぼ毎週日曜日8時30分に起床してプリキュア本編を視聴したり、巷に並ぶプリキュアグッズやプリキュアCDアルバムを購入したり、人目を憚ってこっそりプリキュアの映画を拝見する程度の些細な嗜みだ。私も前作「スマイルプリキュア」から視聴を始めた所詮にわかのファンに過ぎず、「初代から欠かさず撮りためて全部DVDに焼いてますブヒブヒ」などと宣う大き



プリキュアへの変身を試みる Santarh 氏の図

なお友達には、逆立ちしたところで到底敵わない。とはいえるプリキュアなるコンテンツに、片足どころか腰までずつぶり埋まってしまったこともまた事実。小さいお子様のみならず成人男性にまで強い影響力を与えるプリキュアには、只々敬意を表するばかりだ。

いい年してプリキュアなんて……と嘆く、正常な思考の持ち主も多くいらっしゃるだろう。だが決して侮ることなれ。時にコメディな、時にシリアスな展開を含んだ巧みなプリキュアのストーリー展開は、30分間の起承転結で実にすっきりとまとめられている。さらに話の中で得られる教訓は、時として我々大学生にも深い思慮を促すはずだ。「嗚呼、今日も今日とて堕落してしまった……」などと嘆く情報科学類生も昨今多い。そんな時はプリキュアを見るだけでも、明日から生きる活力がふつふつと湧き上がってくるのではないだろうか。

そしてプリキュアを視聴することにより得られる絶大な効能が3つある。1つは日曜日の朝8時30分から健康的に活動ができること。1つは「好きなアニメは？」と聞かれた時に、堂々と「プリキュアです」と比較的好感度の高い回答を返せること。そして最後に若い女性との会話の糸口になることだ。私事で恐縮だが先日親戚の女児(4)と会話に詰まった時、スマイルプリキュアの話で花を咲かせることができた。まさにプリキュア様様仏様である。

さて、ここまでプリキュアへの愛が高まってくると、私の中でプリキュアへの変身欲求が沸き上がりてくるのも当然ではないだろうか。こういったことを書くと親愛なるWORD愛読者の皆様はきっと今頃、ああ、何言ってんだこいつ頭可哀想なんだなあ、と生温かい瞳で私の記事をぢっと見つめていることだろうかと容易に想像ができる。だがよくよく考えてみて欲しい。一度人間として生まれたからには、ヒーローに変身して人の役に立ちたいと望むものも自然な感情の1つだ。男児が仮面ライダーを見て仮面ライダーに変身したいと思うのは至極当然であり、女児がプリキュアを見てプリキュアになりたいと願うのも当たり前のことだ。こうして本誌編集者みみずのひものは自分で自分に正当な理由をつけて、プリキュアへの変身を説得することに成功した。そしてラブリーコミ

ューン^{*1}にキュアラビーズ^{*2}を装着することでプリキュアへの変身を試みたのだった……。

プリキュア！ラブリング！

聞き慣れた軽やかなメロディーに包まれ、みみずのひものは変身を始めた。さあ、これで私も歴代33人目となる伝説の戦士、キュアみみずのひものへとその姿を変えるのだ！そして自己の肉体をしがない大学生から、女児にとっての永遠の憧れプリキュアへと昇華させるのだ！さあ我が精神よ、解放されるがいい！

……しかし1分待てど2分待てど変身バンクは訪れなかつた。

おかしい、何故だ、不良品か！

早急にタカラ○ミーのお客様相談センターへ連絡する必要があると感じた私は、受話器に手をかけたところで思い留まつた。待てよ、私は大事なことを忘れているんじやないか。そういえば私はメルヘンランドの~~姫~~妖精さんのお姿を、現実においてまだ拝見したことがない^{*3}。ひょっとしてまだ私には、プリキュアとして耐えうる精神の修行、つまりは魂のステージレベルが低いのではないのではないだろうか。ふと脳裏に過ぎたのは、コンビニやスーパーに並ぶ数多のプリキュアグッズ達だ。プリキュアグッズはスナック、チョコボールから果てや入浴剤まで多岐に渡るが、あれはひょっとして自身をプリキュアへと高める仮具のような役割を担つているのではないかだろうか。プリキュアグッズを使いこなすことが、プリキュアという選民にアセンションするための手段の1つではないのだろうか。私の中での推理は確信へと徐々に変化していった。

そこで今回は自身をプリキュアへと高めるため、私みみずのひものが実際にプリキュアグッズを体験することで得られたノウハウを、今回皆様に無料でご提供させて頂く。また、もう自分は既にプリキュアだという方にも、どのような状況においてプリキュアグッズが役立つかを、本誌記者が勝手に想定したので参考にしていただければ幸いだ。これを読んであなたもプリキュアへの道を極めて頂きたい。

*1 WORD 編集部員 Santarh 氏の私物。現代社会いかなる需要に対しても、そのニーズを満たすため一応の解決法が試みられている。例えばプリキュアに変身したいというお客様のニーズに答えるため、タカラ○ミーからは毎年プリキュアの変身アイテムが比較的安価で提供されている。ラブリーコミューンというのは決して共産主義思想集団の名称などではなく、今年度担当のドキドキプリキュアが変身時に用いるスマホ形状のアイテムのことだ。

*2 WORD 編集部員 Santarh 氏の私物。キュアラビーズとは、ラブリーコミューンをラジカセとした時のカセットテープのようなものだ。これを入れ替えてラブリーコミューンにセットすることで、変身したり必殺技を撃ち込んだりすることができる。

*3 歴代のプリキュアにはそれぞれ付き添いのマネージャー的役割として、妖精学校出身の一部エリート妖精がサポートしている。いかなる社会においても学歴はものを言うということがよく分かる。

2 鍛錬

基本編

まず自分がプリキュアへ変身することを考えた時、変身アイテムはどうしても外せない必需品である。ドキドキ！プリキュアの場合はラブリーコミューン、スマイルプリキュアの場合はスマイルパクトと、年々プリキュアによってアイテムが異なるので注意して頂きたい。

・ラブリーコミューン

アニメ本編でも使用されている変身アイテム。ラブリーコミューンに好きな妖精を選ぶことで、ドキドキ！プリキュアの4人のうちから好きなプリキュアに変身できる。その他必殺技の射出や、プリキュアとの電話、相性占い、ミニゲームなどが楽しめる。お値段は3,000円前後。それでも高くてお小遣いでは買えない！という赤貧プリキュアの皆様の為に、ミニサイズのプラスチック製ラブリーコミューンというものがある。こちらは1個280円という駄菓子並の価格で販売されているので、ラブリーコミューンの事前練習や耐久実験をしたい時などに有効だ。



真のラブリーコミューン（左）と紛い物（右）

・ラブハートアロー

ドキドキ！プリキュアの敵ジコチューや、その幹部ジコチュートリオは話数が進むに連れ、戦闘能力が強化されていく。そこで我々プリキュアもより強力な課金武器⁴を投入することで、ジコチューとの闘争において何としても勝利を収めなければならない。そこで今回オススメさせて頂く商品が、こちらのラブハートアローだ。一見只のおもちゃの弓矢のように見えるがその威力は絶大。強化されたジコチュートリオも容易に撃退することができる。お値段は約5,000円。ちなみにアローというたいそうな名前がついていながら、現時点では実際弓としてちゃんと使っているのはキュアハート1人だけで、他の面子は壁にして自分の身を守ったり、タンバリンに見立てて叩いたり、何故か矢ではなく剣を連射したりと使い方は散々である。

*4 プリキュアがこのままでは敵に勝てないことを悟った時、一定周期で妖精から強力な武器が投入される。その強力な武器の正体は、何故か本編よりも先にタカラ○ミーのCMで白日の下に晒される訳だが、これを購入（課金）することで他のプリキュア候補生との差は大きく広がるだろう。これらをまとめて課金武器と言う。

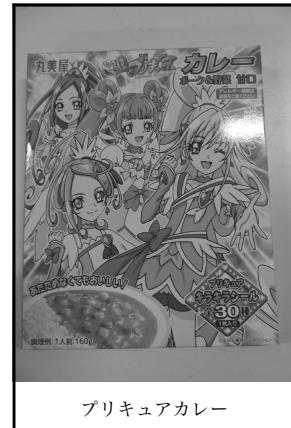
食料編

プリキュアとして自己の生活を見直すならば、まず食生活の改善が優先されるだろう。そのため○美屋などのメーカーからは、プリキュアとしての適切な食生活を送ることを目的とした、公式のプリキュア食料品が提供されている。我々もプリキュアを志す者として胃袋に磨きをかけていきたい。なお現在販売されているプリキュア食品は、「ドキドキ！プリキュア」に関するものが多い。

・プリキュアカレー

プリキュアとしての任務が忙しくなると、食事を準備する時間すら十分に確保できなくなる可能性も高い。そこでオススメしたいのが、手軽に用意できて栄養価も高い○美屋のプリキュアカレーだ。味はポーク＆野菜のカルシウム入り甘口で、お値段は1パック130円と大変お買い得。また小さいお子様でも食べられる一口サイズ版も販売されている。そしてこのプリキュアカレー、なんと温めなくても美味しく食べることができる。まさに忙しい現代社会を生きるプリキュアにはピッタリの一品だろう。

さらに特筆すべきは商品のおまけとしてプリキュアシールが1枚ついてくることだ。こちらは全部で30種類。厳しい道のりだが全てはプリキュアへの変身のためと考え、フルコンプを目指して頂きたい。



プリキュアカレー

・プリキュアふりかけ、お茶漬け、わかめスープ

プリキュアとしての強靭な肉体をキープするため、栄養価の面からも食事にこだわりたい。そこでオススメしたいのが○美屋のプリキュアふりかけだ。味はたまご、さけ、おかか、小魚の4種類から選べる上、もちろんカルシウムも含まれている。こちらは20袋入りで220円。他にも冷えた身体を温めたい時にはプリキュアお茶漬け、プリキュアわかめスープがオススメだ。こちらにはカルシウムどころかCPP（カゼインホスホペプチド）が含まれている。CPPって何やねんと思いつ調べたところ、体内にカルシウムを吸収しやすくさせる物質だということが分かった。カルシウムを貪ることで、より強靭なプリキュアとしての一歩を踏み出すことが出来るのは言うまでも無い。わかめスープの方にはアイちゃん⁵を型どったかまぼこが浮かんでいる。これらの商品も1つ購入する毎に、先ほどと同じ全30種プリキュアシールの内1枚が手に入る。



プリキュアふりかけ

⁵*5 ドキドキ!プリキュアにて、主人公の相田マナがよく面倒を見ている赤ん坊のこと。実は卵から生まれた正体不明の赤ん坊だが、正直中学生で1児の母親って日曜朝のアニメとしてどうなんだろうか。私は凄くいいと思います。

・プリキュアチョコスナック

プリキュアだって年頃の女の子、疲れた時には甘いお菓子が食べたいはず。そこでオススメしたいのがプリキュアチョコスナックだ。いちご味のスナックはまるでカールのように軽い触感、そして言うまでもなくカルシウムも含まれている。だがこのお菓子の真の目的はこんなスナックなどではない。このプリキュアスナックをひっくり返して裏面を見て欲しい。そこには2枚のカードがパックに封入されているのが見えるだろう。そう、これこそがプリキュアスナックの真骨頂、データカードダス・プリキュアオールスターズに必要なカードなのだ。プリキュアオールスターズとは現在日本全国のゲームセンター、デパートなどで絶賛稼働中のアーケードゲームだ。公式HPによると、「歴代のプリキュアをおしゃれコードでキメて、おでかけするゲーム」らしい。おしゃれコードというのは薬物か何かのことだろうか。とにかくプリキュアチョコスナックを購入し、その足でゲーセンへと向かい、その場でパックを開封して直ちにこのゲームで遊ぶ、ということができるのだ。お値段も1つ100円と1クレジット相当の安心価格なので、大人買いをしても財布への影響が少ない点も評価したい。



・プリキュアチョコイチゴボール、ドキドキプリティグミ、菓子パン各種

先述のプリキュアチョコスナック同様、パッケージがプリキュアのチョコボールやグミ、メロンパン、ホットケーキ、いちご蒸しケーキなどが販売されている。だがこれらの商品、只のプリキュア関連食品ではない。これらの対象商品から「えらんでマーク」を3枚集めて応募することで、なんと毎月 100×3 （種類）=300名様にプリキュアの「えらんでトート」がプレゼントされるのだ。あれ、全国のプリキュアファンの数を考えると、300人って結構少なくないかなどという邪推をしてはいけない。さらにプリキュアチョコイチゴボールに付随している「あつめてシール」を28枚集めることで、プリキュアの「あつめてトート」が応募者全員にプレゼントされる。このあつめてシールは残念ながらチョコイチゴボールからしか手に入らないので、 $84 \text{ 円} \times 28 = 2352 \text{ 円}$ の支払いを露骨に要求している……などと邪推してもいけない。プリキュアへの道は長く地道で険しいものなのだ。



プリキュアチョコボール

・プリキュアソーセージ

プリキュアたるものタンパク質の摂取も（以下略）。こちらはニッ○イオリジナル全32種類のプリキュアダブルシールが1枚付随している。お値段は150円前後。それでは以下WORD編集部員による生の感想をお聞きください。「至って普通の魚肉ソーセージだったが、よりプリキュアに近づけたような気がした」「オレタチプリキュア肉喰う。『リキア/カガ、モウ』」

日常生活編



プリキュアのソーセージ

日常生活の中でもプリキュアとしての精神鍛錬は可能である。いくつか例を紹介しよう。

・キュアラビーズ付き入浴剤

プリキュアとして一日の仕事を終えた後には、ひとつ風呂でも浴びてさっぱりしたいものだ。しかし風呂の中でプリキュアとしての鍛錬を怠ってはいけない。そこでオススメしたいのがこのキュアラビーズ付き入浴剤だ。4種類のパッケージはドキドキ！プリキュアの4人がそれぞれ対応しており、お好きなものから選べる。入浴剤の色は黄色で、香りはボカボカカモミール。カモミールというのはキク科の植物の一種で、古くから民間療法に使われてきた薬草だ。さらに名前の通り、プリキュアに変身するため必要なキュアラビーズと、オシャレアクセサリーがおまけとしてついてくる。お値段は420円。

ちなみに前作スマイルプリキュアの入浴剤では、びっくりたまごという方式を採用していた。これは卵状の入浴剤が溶け出すと、中から悪名高きキュアデコル⁶が出てくるという仕組みになっている。

・プリキュアぬりえ

プリキュアにも美的センスが求められる今日この頃。自分から色彩感覚の鍛錬を積みたいと考える時もあるはずだ。そこでオススメしたいのがプリキュアの「デッカぬりえ」だ。全部で25枚の塗り絵で用意された本書は、多彩なイラストの画集としても楽しむことができる。お値段は380円。



デッカぬりえ

⁶キュアデコルはスマイルプリキュアの敵、アカンベエを倒すことで得られるアイテム。しかし現実社会におけるキュアデコルは何故か至る所に分散しており、ガチャを回したりお菓子を買い占めなければコンプリートできない。詳しくは以下のURLを参照のこと。<http://www.yukawanet.com/archives/4157408.html>

・プリキュア携帯

プリキュアは毎年プリキュアオールスターズという映画にて全員集合し、互いの親睦を深め合っている。しかしあなたも新人のプリキュアとして、プライベートでも歴代プリキュアの先輩方と詳しく話したいと思うこともあるだろう。そこでオススメしたいのがプリキュア携帯「プリートフォン ドキドキプラス」だ。この商品は2012年に発売されたプリートフォンに、ドキドキ！プリキュアの要素が追加されたものだ。このプリートフォンはスマートフォンのような形状をしており、全て液晶タッチパネル上で操作できる。主な機能としては数種類のミニゲームと、某大人気SNSのようにプリキュア達のつぶやきを覗いたり、プリキュアのつぶやきに対して返信することができる。このプリキュア達のつぶやきはなかなかネタやウイットに富んだものとなっており、大きなお友だちにも大人気のことだ。ちなみに前作のプリートフォン、及び今作ドキドキプラスについては、無駄に詳しいAmazonのレビュー⁷がなかなか秀逸なので、購入の有無に迷わらず一読してみることをオススメする。

・プリキュアシール

庶民の強い味方こと100円ショップSeriaだが、なんとこんなところにもプリキュアグッズが隠されていた。それがプリキュアシールシリーズである。これは歴代のプリキュアがシールになって登場しているわけだが、その種類には単に侮れないものがある。例えばドキドキ！プリキュアは1袋5枚封入なので全50種類しか用意されていないが、スイートプリキュアでは1袋10枚封入の全71種類、ハートキャッチプリキュアに至っては全80種類近くものシールが用意されている。フルコンプまでは至難の業だろう。



・おせわしてきゅび！ちゅばちゅばアイちゃん

プリキュアだって可憐な女の子だ。素敵なあの人と電撃ゴールインからのベットイン、あわや1児の母なんてことがあってもおかしくはない。しかし戦闘民族として育ったプリキュアにとって、育児という不慣れな家業はいささか荷が重すぎるだろう。万が一我が子に愛を感じなくなってしまい、キュア虐待、キュアネグレクト、そのままキュア家庭裁判所へ直行となれば、元プリキュア逮捕とスポーツ紙の三面に掲載されるのが大概の落ちとなるだろう。そんな育児に不安を抱えるあなたにオススメしたいのが、こちらの「おせわしてきゅび！ちゅばちゅば

*7 <http://www.amazon.co.jp/dp/B006Q4SR90> と <http://www.amazon.co.jp/review/R1KAD0UF17I2CC> を見比べていただければ、新プリートフォンの変更、追加点などについて垣間見ることができる。他にもポンキネス氏はプリキュア製品について詳しいレビューを投稿しているので、購入する際の参考にしてみてはいかがだろうか。

「ゆばアイちゃん」だ。こちらの商品は先述の赤ん坊アイちゃんを型どつたぬいぐるみだが、なんと人間の赤ん坊同様哺乳瓶のミルク（付属品）を飲み、人語を話すのだ。さらにコミュニケーションを重ねるに連れこのアイちゃんぬいぐるみは語彙を増やしていく、最終的には「ママありがとうキュピ」と自らの母親に感謝の言葉をも口にする。また先ほど紹介にあげたラブリーコミューンと連動することで、中央のハート部分がピンピンに反応する。さらにはこれだけ赤ん坊としての機能が充実しているというのに、おしめを変える必要もなければ不平不満を口にすることもない。まさに育児未経験のプリキュアには、事前練習にぴったりの商品だろう。ちなみにプリキュア本編の合間に、小学生女児が「ママってたーのしい！」と歓喜するこの商品のCMがよく流れるが、母によると「楽しくねえよ！」とのことらしい。

3 終わりに

ここまでご覧になったプリキュア候補生の皆様は、プリキュアとしての自覚を高めることができただろうか。プリキュアに変身するため特別な修行などは必要なく、ただ一心不乱にプリキュアグッズを収集するだけでよいことがお分かり頂けたかと思う。

さて、ここまでプリキュアへの試練を淡々と述べてきた不肖みみずのひものは、ここで重大なことに気づいてしまった。私がプリキュアに変身できないのは、ひょっとして私にプリキュアの適正がないからではないのだろうか。現在までのプリキュアの平均年齢は15.66歳。現役プリキュア最高齢は17歳。そして性別は全て例外なく——女性だ。

……何ということだろうか。現在筑波大学生の私は、今の今になって己が女子中学生ではないことにとうとう気づいてしまったのだ。これでは来世で転生するまでプリキュアになることができない……。私は飯も満足に通らないほどの、深い絶望に包まれた。

「私は一生プリキュアになれない、とでもいうのか……？」

嗚呼、プリキュアに変身できない人生なんて何の意味があろうか。と、人生を諦観し始めたその時のことである。

キュアハッピーことこぼる氏（WORD編集部員）「そんなことないクル！まだチミにはコスチュームプレイという手段が残っているクル！」

「キュ、キュアハッピー？」

キュアビューティーことSantah氏（WORD編集部員）「さあ、チミもAmazonギフトカード1万円分をセットして、プリキュア、デポジットチャージって叫ぶクル！」

「キュアビューティーまで……うん、よく分からないけどやってみる！で、でも今はお金が……」

キュアサニーことmitty氏（WORD編集部員）「じゃあいつ変身するの？」

「今でしょ！」

一ヶ月後……



輝け、スマイルプリキュア！

*8 おわり

*8 WORD 編集部内ではキュアマーチちゃんが見つかりませんでした！自分こそが現世に転生した真のキュアマーチであるという方は、至急 WORD 編集部までお越しください！一緒に変身してアカンベエと闘いましょう！

The Latest C++

文 編集部 ArGxento

はじめに

C++は商用ソフトウェアやオープンソースソフトウェアでもっともよく使われるプログラミング言語の一つですが、登場が1983年頃^{*1}と比較的古いため、機能面でユーザが不満を感じる点がいくつもありました。そこでC++標準化委員会が中心となって改良を進め、数年ごとに新しい言語規格が発表されてきました。本稿執筆時点^{*2}での最新規格はISO/IEC 14882:2011、通称C++11とよばれています。

C++11は安全性の向上や初心者への配慮などの方針のもと、C++0xという仮称で策定が進められ、2011年8月にISOから標準規格として承認されました。規格承認後、長らく仕様に完全に準拠したコンパイラが現れませんでしたが、2013年4月にClang^{*3}がC++11への完全準拠を発表^{*4}し、GCCも4.8.1での完全準拠を予定^{*5}するなど、ようやく言語規格に沿ったコードが実際にコンパイルできるようになりました。

本稿では、C++に触れたことがあるが最新規格をまったく知らない、情報科学類の授業でC言語やJavaに触れたがC++に触れたことがない、C#やRubyなどを書いているがC++は古くさいイメージがあり触れたことがない、といった読者を対象に、C++11での新機能のうち比較的親しみやすいもの、便利なものを独断と偏見でピックアップし、具体的なサンプルコードとともに紹介していきます。

本稿のサンプルコードはGCC4.7.2^{*6}によって検証し、Microsoft Visual C++ Compiler Nov 2012 CTP^{*7}（以下VC++）でコンパイルできないコードについてはその原因を記しました^{*8}。また、内容は基本的に名前空間を省略せず、理解を妨げるような煩雑な書き方^{*9}を避けたつもりです。また、コードの断片ではなくコンパイル可能なコード全体を掲載しました。これらを写し、自分の手で色々試してみれば理解が深まることでしょう。

^{*1} <http://www.stroustrup.com/hopl2.pdf>

^{*2} 2013年5月

^{*3} <http://clang.llvm.org/>

^{*4} http://clang.llvm.org/cxx_status.html

^{*5} <http://gcc.gnu.org/gcc-4.8/changes.html#cxx>

^{*6} <http://gcc.gnu.org/>

^{*7} <http://www.microsoft.com/en-us/download/details.aspx?id=35515>

^{*8} 基本的にはvectorのinitializer_list未対応

^{*9} 特にconst地獄

新しい配列: array

C++には固定長配列^{*10}を実現する仕組みとして、C 言語由来の配列が用意されています。しかし、この C 言語由来の配列は C++で例外機構が導入される以前に設計されたため、範囲外アクセスを検知する方法がない、C++で導入された標準コンテナ^{*11}との互換性がないなどの欠点があります。

旧来の C++では `vector` を固定長配列の代わりに用いる手法が採られることが多かったのですが、`vector` は動的配列^{*12}であるため、固定長配列として用いた場合に不要な処理が発生する、要素数によって型を分けられないといった欠点があります。そこで、C++11 では新しい固定長配列として `array` が導入されました。

使い方

```
1 #include<array>
2 #include<iostream>
3 #include<exception>
4
5 int main() {
6     std::array<int, 4> ar1; // テンプレート引数に要素の型と要素数を渡す
7     std::array<int, 4> ar2 = {1, 2, 3, 4}; // 普通の配列のように初期化できる
8
9     std::cout << ar2[0] << ","; // 今まで通りの書き方でアクセスできる(範囲外アクセスチェックがないので注意)
10    std::cout << ar2.at(1) << ","; // こちらの方が安全
11
12    ar1 = ar2; // より直感的な代入や比較
13    if(ar1 == ar2) {std::cout << ar1.at(2) << ",";}
14
15    std::cout << ar1.back() << ","; // 配列に対する、標準コンテナ互換のいくつかのメソッド
16
17    try {
18        ar2.at(42) = 31337; // 範囲外アクセスをしても……
19    } catch(std::exception) {
20        std::cout << "caught an exception." << std::endl; // 例外として捕捉できる
21    }
22    return 0;
23 }
```

実行結果

```
1, 2, 3, 4, caught an exception.
```

*10 要素数を変更できない配列

*11 C++標準ライブラリが提供する、`vector` や `list` などのデータ構造群。統一されたインターフェースが定められている

*12 要素数を動的に変更できる配列

型推論

C++を書いていると、必ずといっていいほど異様なまでに長い型名に出会うことになります。それらを手書きするのは苦痛で非効率的なので、できる限り型名を自動的に推論してほしいものです。また、C++にはテンプレートという機能があり、それを使った関数の返値には人間には推測困難な型も少なくありませんし、後述するラムダ式など、そもそも型名をコード上に書けない構文も存在します。

そこで C++11 では、C 言語の時代から予約語になっていたものの、今ではほとんど意味のなくなってしまった^{*13} auto 予約語を再利用し、型推論のための仕組みを整備しました。

使い方

```

1 #include<iostream>
2 #include<vector>
3
4 int main() {
5     std::vector<int> vec = {1, 2, 3, 4}; // VC++では未対応
6
7     // vec の要素を順に出力する
8     // 今までの一般的な書き方
9     for(std::vector<int>::iterator itr = vec.begin(); itr != vec.end(); ++itr) {
10         std::cout << *itr << "\n";
11     }
12     std::cout << std::endl;
13     // std::vector<int>::iterator などの長い型名を毎回書くのは面倒
14
15     // auto を使った書き方
16     for(auto itr = vec.begin(); itr != vec.end(); ++itr) {
17         std::cout << *itr << "\n";
18     }
19     // 初期化のときに右辺から型が推定できるものはなんでもauto に置き換える可能
20     return 0;
21 }
```

実行結果

1 2 3 4

1 2 3 4

^{*13} 関数の呼び出し時に確保され、関数の終了時に破棄される、いわゆるローカル変数の宣言に使われていた。ローカル変数の使用が当たり前となった今では通常省略している

新しい for 文

配列やリスト構造、文字列などに対し、各要素を順番に参照するのに `for` 文がよく用いられてきました。しかし、よく用いられる割に書き方が冗長で面倒だという欠点があり、それを改善するため、`Boost.Foreach`^{*14}などのライブラリが開発されてきましたが、今回その「要素の集まりから各要素を順番に取り出す」という操作に特化した構文が（ようやく）追加されました。

この構文は各要素の型を記述する必要がありますが、`auto` を使うことにより、より簡潔に記述することができるようになります。

使い方

```
1 #include<iostream>
2 #include<array>
3 #include<string>
4
5 int main() {
6     std::array<int, 4> ary = {1,2,3,4};
7     int ar[] = {5, 6, 7, 8};
8     std::string str = "Test";
9
10    // ary, ar, str の要素を順に出力する
11    for(int i: ary) { // for(変数: 配列など)の形で書く
12        std::cout << i << ",";
13    }
14    std::cout << std::endl;
15
16    for(int & i: ar) { // C 言語由来の配列にも同じように使える
17        i *= 2; // 参照型に代入すれば要素を書き換えられる
18        std::cout << i << ",";
19    }
20    std::cout << std::endl;
21    // auto を使えば型名を自動的に推論してくれる
22    for(auto e: str) {std::cout << e << ",";}
23
24 }
```

実行結果

```
1, 2, 3, 4,
10, 12, 14, 16,
T, e, s, t,
```

*14 <http://www.boost.org/doc/html/foreach.html>

ラムダ式

C 言語や C++ の一般的な関数は、他の関数などの中では定義できず^{*15}、必ず名前を付けなければならないという制限があります。しかし、関数にちょっとした処理を渡すためだけに別の新しい関数を定義するのが大げさに見える、関数の定義と呼び出しが離れていて可読性が下がるといったことがよく起こります。それを解決するために、関数のようなものをより自由に作り出すための、ラムダ式と呼ばれる構文が導入されました。

ラムダ式は基本的に、

```
[ ]( /* 引数リスト */ ) { /* 関数の中身 */ }
```

という形で書き表します。返値の型は `return` に渡された値から自動的に決定されます。また、ここでは詳しく述べませんが、ローカル変数をキャプチャしたり、返値の型を明示したりする構文も備えています。興味のある方は調べてみてください。

使い方

```

1 #include<iostream>
2 #include<vector>
3 #include<string>
4 #include<algorithm>
5
6 struct Person {
7     std::string name;
8     int age;
9 };
10
11 // こんな所に比較関数が
12 bool compareByAge(Person const& a, Person const& b) {
13     return(a.age < b.age);
14 }
15
16 int main() {
17     std::vector<Person> members1 = { // VC++では未対応
18         // name          age
19         {"Sue_Thruster", 23},
20         {"James_Burton", 17},
21         {"Riga_Pratica", 20}
22     };
23     std::vector<Person> members2 = {
24         // name          age
25         {"Lloyd_Evansmann", 26},
26         {"Cenes_Crawford", 32},

```

^{*15} GCCなどのコンパイラ独自拡張を除く

```
27     {"Kei_Oread",      22}
28 };
29
30 // members1 と members2 を age でソートしたい
31
32 // 関数ポインタをもちいた方法
33 std::sort(members1.begin(), members1.end(), compareByAge);
34
35 // 比較関数が何をやっているか分かりにくい、そこでラムダ式をもちいると
36 std::sort(members2.begin(), members2.end(),
37 [](Person const& a, Person const& b){return(a.age < b.age);}
38 ); // 処理の内容が一目で分かる
39
40 auto printPerson = [](Person const& p) { // auto を使えば変数に代入可能
41     std::cout << p.name << ":" << p.age << "\n";
42 };
43
44 std::cout << "members1:" "\n";
45 std::for_each(members1.begin(), members1.end(), printPerson);
46 std::cout << "\n" "members2:" "\n";
47 std::for_each(members2.begin(), members2.end(), printPerson);
48 return 0;
49 }
```

実行結果

```
members1:
James Burton: 17, Riga Pratica: 20, Sue Thruster: 23,
members2:
Kei Oread: 22, Lloyd Evansmann: 26, Cenes Crawford: 32,
```

さいごに

本稿では、array や auto による型推論、新しい for 文、ラムダ式について紹介しました。これらの機能は他の言語に導入され、多くのユーザーに支持されたので C++に取り入れたと考えられるものばかりです。したがって、C++でも開発効率やコードの可読性を向上させる大きな助けとなるでしょう。

C++11 には他にも、文字列操作を格段に楽にする正規表現や、メモリ管理の煩わしさから解放してくれるスマートポインタなど、本稿で紹介できなかった便利な機能がたくさん追加されました。読者の皆さんも、次回 C++のコードを書くときには C++11 の新機能を使ってみてはいかがでしょうか。

GRな日々。 XVI

文 編集部 葡萄酒

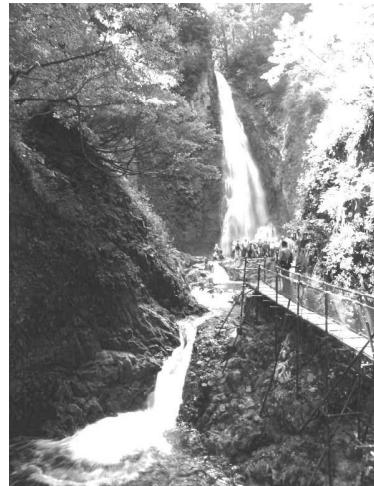
この「GRな日々。」も16回目、連載開始から早4年近い歳月が経とうとしている。少し唐突な気もするが、4年という区切りで幕を引くのも悪くないと思い、今回をもって本シリーズの連載を終了することに決めた。

思えばここまで、色々な写真を撮ってきた。そこで最終回となった今回は今まで掲載してきた写真の総集編ということで、各回から特に気に入った1枚ずつをコメントとともに再掲して行こうと思う。

第1回 「GRな日々。」

(WORD12号「WORDは一太郎を使っています号」, 2009年11月)

記念すべき第1回は、購入したばかりのGRD2を片手に東北の白神山地へ。途中で雨に降られたりもしたが、おかげで綺麗な虹が見られたのが印象深い。爽やかな夏の森の中、名所「暗門の滝」を撮影した。いかんせん買ったばかりのカメラなので、露出の感覚がイマイチ掴めていない。



第2回 「GRな日々。II ~さよなら能登/北陸号~」

(WORD13号「WORDは仕分けされませんでした号」, 2010年1月)



夜行列車の急行「能登」と寝台特急「北陸」の特集。すでに両方とも廃止になってしまったが、記事を読み返すと思い出が鮮やかに蘇ってくる。今になってから、金沢駅で両者が揃い踏みしている所を撮れなかったのが悔やまれる。

第3回 「GRな日々。III」

(WORD14号「WORDはSIMフリーです号」, 2010年5月)

富山県の名物である「雪の大谷」の撮影。冬の間降り積もった雪を削り、開通した道の両端には10メートルを超える巨大な壁がそり立つ。その姿はまさに圧巻と言う他無いだろう。ちなみにこれはゴールデンウィークに撮った写真である。地元民ながら日本の風景とは思えない。



第4回 「GRな日々。IV」

(WORD15号「非実在WORD編集部号」, 2010年9月)



初の巻頭記事となった割には、写真の技術的な話なので地味な回であった。しかしこうして改めて写真を見てみると、やはり GRD2 のレンズは優秀である。被写界深度、ボケ味とともに申し分ない。余談だが、後継機の GRD3 では更に開放側が広くなり、F値は怒涛の 1.9 という素晴らしいスペックに。

第5回 「GRな日々。V」

(WORD16号「侵略されたいでゲソ号」, 2010年11月)

もう1台の所有カメラである「ASAHI PENTAX SP」の紹介から、銀塗写真の簡単な入門まで。最近はめっきり使うことも無くなってしまったが、次の旅行にはコイツも連れて行ってやろうと思う。そろそろ広角のレンズも手に入れようか……。



第6回 「CX & GRな日々。~サンライズウサギ島~」

(WORD17号「WORDおせち特盛号」, 2011年2月)



他の編集部員との合同企画で、広島県にある「ウサギと廃墟の島」こと「大久野島」への旅行記。打ち捨てられた旧帝国陸軍の毒ガス製造施設や発電所の成れの果てを巡り、無常感を満喫した。廃墟好き・ウサギ好き（何と奇妙な組み合わせだろうか）なら行って間違いなく損は無い島。ちなみに行きの列車にコートを忘れて、島で潮風に震えていたのは内緒。

第7回 「GRな日々。VII」

(WORD18号「え～？WORD～(中略)ありますっ☆号」, 2011年6月)



奥多摩の水根貨物線跡の散策。朽ち果てた鉄橋やトンネルを越え、山道を進む進む。とは言え列車が走れる程度の傾斜なので、実は一般的な廃線跡は歩きやすかったりする。この取材以降、東京都は広いなと思いました。今度は東京都内の離島にも行ってみたい。

第8回 「真夏の夜の死闘 ~東日本縦断レース~ (feat. GR な日々。VIII)」

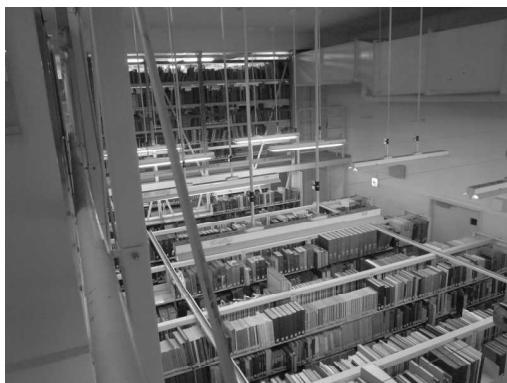
(WORD19号「一斗缶special edition号」, 2011年9月)



合同記事ふたたび。青春18きっぷの鉄道組と、高速道路を使わない車組との東京-札幌間でのデッドヒート。今だから暴露すると、企画時点の日程ではおそらく車組が大勝するであろう予定だったが、運命のいたずらによって鉄道組がギリギリ勝つという奇跡のどんでん返しとなった。詳しい経緯は本編を参照のこと。写真は青森で車組とすれ違ったときのものである。

第9回 「GRな日々。 IX - 図書館探訪」

(WORD20号「WORD燃やすぞ号」, 2011年12月)



中央図書館の中2階を紹介。アクセスの悪い場所にあるのと、利用者が全然いないため、秘密基地みたいでワクワクする。実は記事のネタが無くて悩んだ結果、苦肉の策として「お気に入りの場所」を紹介することに。妙な資料が大量に集約されているので、背表紙眺めて歩くだけでも結構面白い。

第10回 「GRな日々。 X - 日帰り温泉旅行-」

(WORD21号「このタイトルはIMAGINE THE FUTURE.されました号」, 2012年2月)

大糸線沿いの温泉を巡る話。正直なところ、雪の積もる中での露天風呂は死ぬほど寒かった。一旦お湯に浸かると、外気が冷たすぎて浴槽から出る気になれない。温泉巡りは大学生らしからぬ趣味のような気もするが、実際は大学生くらい時間が無いと色々回ってもいられないでの、就職するまでが華であろう。



第11回 「GRな日々。 XI」

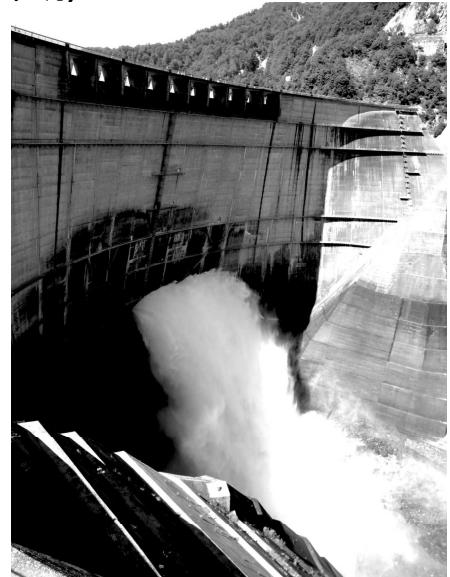
(WORD22号「学類誌と称する事実上の薄い本号」, 2012年5月)

茨城県の名所、また日本三名瀑の1つにもかぞえられる「袋田の滝」へ。ここ観瀑台から眺める滝は申し分ない絶景で、何度も行っていても楽しめる。やはりオススメは紅葉の時期だろうか。まだ凍った状態は見たことがないので、こちらもいつか行ってみたいものである。

第12回 「GRな日々。 XII」

(WORD23号「WORDには領土問題は存在しません号」, 2012年9月)

関西電力の黒部川発電所見学会へ参加。今まで立山に登ったことは何度もあるが、この見学会へ参加できたのは今回が初めてなので感動もひとしお。事前に「高熱隧道」を読んでいたので、より楽しめたように思う。適当な靴で山に登つたら、帰ってくる頃には靴が大破して穴だらけになっていたのはご愛嬌。山に登るときはちゃんと登山靴を履こう。



第13回 「GRな日々。 XIII」

(WORD24号「清掃車でもお茶がしたい！号」, 2012年12月)



わたらせ渓谷鐵道と足尾銅山の観光。非電化路線の写真は架線が無いぶん寂れた雰囲気が出て良いのだけれども、しっくりくる構図が取りづらいイメージ。逆に上から見下ろすときは全体がスッキリして見栄えの良い写真が撮りやすい気がする。この号からWORDのレイアウトが一新、より格好良いデザインになった。

第14回 「GRな日々。 XIV」

(WORD25号「60秒以内にWORDを用意出来なかったらWORD無料券をプレゼントします号」, 2013年2月)

蒸気機関車を求めて栃木県は真岡鐵道へ。雪の降る中、SLに揺られて田舎を走る、情緒豊かな旅路だった。転車台で車両を回している間は、他の観光客とともに頭に雪を積もらせつつ、ガタガタ震えながらの撮影である。手ブレを抑えるため、なかなかの根性が要求される撮影であった。



第15回 「GRな日々。XV」

(WORD26号「容赦ない締め切りによる無慈悲な鉄槌が下りました号」, 2013年4月)

おそらく今まで乗った鉄道の中でも一番辺鄙な大井川鐵道の紹介。何も無い場所とは聞いていたが、正直見くびっていたと思う。また、帰路はトラブルばかりだったのも、今になってみれば良い思い出である。あんなに長く感じた1日は未だかつて無いだろう。



こうして見てみると、このカメラと一緒に日本全国を回ってきたのだなあという感慨が沸き上がってくる。元々は写真記事のつもりで書き始めたはずが、今となっては完全に旅行エッセイとなってしまったが、連載している間は楽しかったので、まあこれで良かったのだろう。長期にわたって同じテーマで文章を書くという経験は初めてのことだったので、毎回ネタを探すのにも苦労したが、おかげで得られることも大きかった。記事の構成を考えながら写真を撮るコツも、こういった連載が無いと得難い技術であろう。

また、この連載の発端でもある GRD2 はよく頑張ってくれたと思う。もう 28mm の画角はしっかりと脳に刻み込まれているし、細かい操作でもボタンの配置を見ることはない。心躍る旅先の風景を鮮やかに切り取ってくれる、私にとってこの上ない相棒である。GR ユーザとして、このカメラの魅力を完全に引き出しきれたかというとあまり自信は無いが、「写真を撮る楽しみ」というテーマは今まで掲載した写真そのもので表現できたと思いたい。

さて、名残惜しいのは山々だが、あまりダラダラと後書きを綴っていても見苦しいので、このあたりで一旦筆を置こうと思う。さて、一休みしたあとは何の記事を書くことにしようか。

新入生の新入生による新入生の為のPC購入のススメ

文 編集部 Moneto

1. はじめに

初めまして。4月に情報科学類に入学しました、Monetoと申します。東京都内からの宅通民で、毎朝強烈な眠気を感じながらつくばエクスプレスと関鉄バスに揺られて通学しております。

何故本記事を書いたかと申しますと、最近、諸事情により様々な方面でPCの購入に関する相談に乗ることが多くなってきたということが理由の1つとして挙げられます。「大学のレポートが書けるくらいのPC」から「ホームビデオの編集ができるくらいのPC」、更には「情報科学類で4年間困らないPC」等といったものまで様々なケースに対応していく中で、幅広く共有した方が良さそうな事柄に関してまとめていこうと思います。

(Twitter:@Moneto_Tkに居りますので、Twitter-erの方は是非ともfollowをお願いします)

2. そもそもPCは必要なのか？

確かに情報科学類には全学サテライトとは別に専用の計算機群が存在し、24時間誰でも使用可能になっています（情報科学類以外にも、専用の計算機群を保有している学類はいくつかあります）。ですが、各講義の課題が重なる時期、或いはコンピュータリテラシ等の計算機室の計算機を利用する講義で多めの課題が出された直後など、計算機室がオーバーフローしてしまうことも少なくありません。

そのような場合、自分用のPCを持っていないと、一刻も早く課題をこなさなければならない状況であるにも関わらず計算機室が空くまで待たなければならなかったり、宅通であれば最悪つくばに泊まらざるを得ない状況になってしまったりすることも考えられます。これでは課題の消化の為に不要な労力を多く費やすことになってしまいますね。

それだけでなく、自分用のPCを持っていると何かと便利なことが多いです。授業で学んでいくプログラミングとは別に、自分のやってみたいことを実現するために別のプログラミング言語を学んでみたり（一通りの環境はcoins、或いは全学サテライトで揃いはしますが……ニッチな言語を学ぶ、或いは特殊な環境を構築しなければならない場合などは、機室の計算機では対応しきれません）、Skype等のテレビ電話サービスを利用して遠方の家族と連絡を取ったり、宅通民であれば移動中に講義ノートをまとめてみたり、或いはそれらを読み返してみたり… …と、様々な用途でPCは役に立つものです。

これらのことを考えると、PC はやはり「可能であれば持つておくべき」ものであると考えられます。確かに「絶対に」持つていなければならない、というものでは無いのですが、持つていなかつたために感じることになる不便は決して少なくないでしょう。

特に PC の中でも、持ち運びが可能で、委員会・サークル活動や講義ノート等、様々な局面で役に立つノート PC の利便性にはやはり目を見張るものがありますから、出来る限りノート PC を 1 台（以上）持つておくと便利なのではないでしょうか。

3. おすすめPCの紹介

さて、前項で「ノート PC があると便利」という説明をしましたが……いざノート PC を買おうかなと決断したとしても、現在の市場には無数のノート PC が出回っていますから、余り詳しくない人にとってはどれを買えばいいのかついつい混乱してしまいがちなものです。

そこで、筆者（及び、周辺の数人）の独断と偏見で選んだ「coins 新入生にオススメなノート PC」を以下にいくつか紹介します。一応、申し訳程度ではありますが coins 以外の学類の新入生が読むことも想定して内容を執筆していますので、coins 以外の方も安心してお読み頂ければと思います。

・ MacBook Air

- ・メーカー：Apple
- ・価格帯：85,000 円～ 125,000 円（学割適用時）
- ・サイズ：11inch・13inch
- ・スペック概要：ミドルエンド。薄型 PC の中では総じて高め。
- ・特記事項：学割を適用するといくらか安くなる。

言わずと知れた Apple 社の薄型ノート PC。本体メモリを後から増設できない等、拡張性にはやや難が残るもの、元々のスペックのお陰で殆ど不自由なく利用することができる。11inch モデルでは重量がほぼ 1kg と超軽量級であるため、持ち運びにも最適であろう。~~スクバでドヤ顔するなら間違いなくこれを選ぶべきであろうが、起動させるソフトウェアにはくれぐれも気を付けよう。~~

Macbook のユーザーにはありがちであるが、Bootcamp を利用して Windows をデュアルブートする場合、搭載する SSD の容量は出来る限り大きいものを選択しておこう。OS はかなりの容量を食ってしまうものであるから、ミドルクラスの 128GB モデルだと容量的にやや不便である。価格帯との兼ね合いにより、可能であれば 256GB が最適か（勿論、デュアルブートしなければ 128GB モデルでも不便は少ない）。

なお、11inch モデルは画面が狭いため、余り作業には適さないと感じているユーザーも少なくない。筐体の重量が少し増えるのを厭わず作業効率を重視するなら、少しでも画面の広い（解像度も大きい）13inch モデルを選択しておくのがよいだろう。

・MacBook Pro

- ・メーカー：Apple
- ・価格帯：100,000 円～ 237,000 円（学割適用時）
- ・サイズ：13inch・15inch
- ・スペック概要：ミドルエンド～ハイエンド。様々な性能のモデルがラインナップされている。
- ・特記事項：学割を適用するといくらか安くなる。

Apple 社の販売しているノートパソコンの中で Macbook Air に対して上位に位置する、ハイエンドノート PC 群である。15inch モデルでは nVIDIA 社製のグラフィックチップ（nVIDIA GeForce GT 650M 1GB GDDR5-Memory）が搭載されているなどの極めて高い性能を誇り、一般的な用途への利用だけでなく、ゲーム開発等の全体的に高いスペックが要求される作業も難なくこなすことが可能である。

130,000 円程度から Retina ディスプレイが搭載されているモデルが購入でき、15inch のディスプレイにおいて FullHD(1920 × 1080)を凌駕する 2880 × 1800 の超高解像度も体感可能であるが……それだけに価格も非常に高いので、余程の理由が無い限りは普通のディスプレイが搭載されているモデルを購入すれば良いだろう。

なお、最上位モデルの性能には恐ろしいものがある。2.7GHz Quad Core の Core i7 を搭載しているだけでなく、オンボード 16GB DDR3-RAM や 512GB SSD、nVIDIA GeForce GT 650M 1GB までも搭載しているのであるから、滅多なことではこの性能で不満が出ることは無いだろう。しかし、学割を適用しても 24 万円弱の価格なので、そうそう手が出せるユーザーも居ない、というのもまた確かなことであろう。

・ThinkPad Lシリーズ

- ・メーカー：lenovo
- ・価格帯：65,000 円～ 100,000 円
- ・サイズ：14.0 型・15.6 型
- ・スペック概要：ローエンド～ミドルエンド、構成により様々
- ・特記事項：クーポンが時々出ている。

高い堅牢性と安定した品質で名が知られている ThinkPad のベーシックなモデル。Apple 社の PC であることに特に拘らないのであれば、まずこのシリーズを選んでおけば間違いは無いだろう。

筆者が情報を参照した 2013 年 5 月 17 日時点では主に 3 つの構成が存在し（それ以外にも、カスタマイズによって様々な構成にすることが可能である）、Core i3・2GB メモリを搭載したローエンドモデルが 65,000 円、Core i5・4GB メモリを搭載したローミドルモデルが 80,000 円、Core i7・8GB メモリを搭載したミドルエンドモデルが 100,000 円といった具合になっている。

一般的な用途（Web サイトの閲覧、レポートの編集等）のみを想定する限りにおいては、ミドルエン

ドモデルの性能があれば十分だろう。自分でメモリを増設できる、或いはカスタマイズでメモリを増やすのであれば、ローエンドモデルを選択するのも良いのではなかろうか。

・ ThinkPad T430/430s

- ・メーカー：lenovo
- ・価格帯：100,000 円～ 200,000 円
- ・サイズ：14.0 型
- ・スペック概要：ミドルエンド～ハイエンド
- ・特記事項：クーポンが時々出ている。

ThinkPad の中でも、全体的に高性能を誇る T シリーズの 14.0 型モデル。T430 はスタンダードな筐体であるが、T430s は薄型・軽量化が図られており、同じ性能でありながら T430 よりも携帯性が向上しているモデルとなっている。ただしその分、T430s の方が価格は高くなっている。

L シリーズ同様、最も安価な構成における搭載メモリの容量は 2GB と少なめであり、またストレージ容量も他の構成と比較して少ないものとなっているため、中位モデル（110,000 円程度～、T430s は 130,000 円程度～）を選択するのが良いだろう。中には独立したグラフィックチップとして nVIDIA NVS 5200M Optimus(1GB) を搭載しているモデルもあるため、高度なグラフィック処理を必要とする 3D ゲームソフト、或いは簡易な動画編集程度であれば問題なくこなせると思われる。

なお、T シリーズには他に 15.6inch ディスプレイを搭載した T530 というモデルも存在する。こちらは T430/T430s とは異なり、解像度 1920 × 1080 の FullHD 液晶を選択することができるため、高解像度なディスプレイが欲しい場合はこちらを選択すると良いだろう（と言っても、解像度単体で見れば Retina ディスプレイの方が優秀なものではあるが……）。

・ ZENBOOK

- ・メーカー：ASUS（エイスース）
- ・価格帯：60,000 円～ 110,000 円
- ・サイズ：11.6inch・13.3inch
- ・スペック概要：中程度

どこぞの超薄型ノート PC によく似たデザインのウルトラブック。Apple 社製に拘らず、超薄型のウルトラブックが欲しいというのであれば先ず上位候補に挙がる製品であろう。

勿論ウルトラブックなので、CPU には超低電圧版のものが採用されている。そのため高パフォーマンスを必要とする作業には比較的不向きであるが、レポートの作成、簡易な画像編集等の一般的な作業であれば十分にこなすことが可能である。

いずれも非常に扱いやすい性能を誇り、外観・操作性等も洗練されてきたものではあります。購入前に必ず一度は実物を触って操作性を確認してみるべきです。大型家電量販店等であれば、上に挙げたようなノート PC が実際に展示されていますので、そのような場所を利用してみるのも手段の 1 つです（店によってはインターネット価格より安く販売しているところ、或いはポイント特典等が付加されているところもありますから、実店舗に行って買った方が交通費も含めて安上がりになる可能性もあります）。

4. ソフトウェアのススメ

上記に紹介しました PC には、何れにおいても標準構成で Office ソフトウェア・セキュリティソフトウェアがプリインストールされていない（勿論、構成次第ではプリインストールされていることがあるが……）ため、それらのソフトウェアを利用するためにはソフトウェアを単体で購入する必要があります。

・Office ソフトウェア

先ず Office ソフトウェアですが、性能面・互換性の問題からこれは決して安価な互換品で妥協してはいけません。Windows 向けの Office 2010 Academic（学割版）が 30,000 円弱で購入できますから、安価な互換ソフトウェアに頼らず、思い切って本家 Office を購入してみることを強くお勧めします。互換ソフトウェアと本家 Office との間で完全な表示の互換性は保証されていませんので、それに起因する無用なトラブルを避けるためには初めから本家 Office を利用しておくのが良いだろうと思われます（実際、図などを利用しているドキュメントでは大幅に表示が崩れことがあります。折角作ったレポートが、互換性の関係で正常に表示されず、採点されないという事態になったらとても悲しいですよね）。

・セキュリティソフトウェア

次にセキュリティソフトウェアですが、筆者の経験、及び周囲の評判より Symantec Norton Internet Security を強くお勧めします。なお、赤いパッケージのウイルスソフトは、誤検知、及びそれに起因する諸問題のためお勧めできません。

Symantec Norton Internet Security は 3 台に対して 1 年間 6,500 円、或いは 1 台に対して 3 年間 10,000 円、等といった様々な形態でのライセンスが提供されていますから、自分の PC の利用形態に応じて選択すると良いでしょう。

なお、筑波大学では学術情報メディアセンターにより、アンチウイルス・セキュリティ機能を備えた Symantec Endpoint Protection が 1 人 1 台分まで無料で配布されています。詳細は学術情報メディアセンター公式サイト内の Web ページ（<http://www.cc.tsukuba.ac.jp/computer/license/antivirus.html>）に掲載されていますので、興味のある方、或いはできる限りセキュリティソフトウェアにお金をかけたくないという方は見てみることをお勧めします。

5. 周辺機器のススメ

ノート PC を購入したら、次はノート PC で使用する周辺機器を購入しておくことを強くお勧めします。これは単純に利便性を向上させる為のものであり、以下に挙げるものを揃えておくと PC の利用が極めて快適になるだろうと思われます。

・USBフラッシュメモリ

- ・価格帯：4GB 500 円程度～、8GB 700 円程度～、16GB 1000 円程度～

~~断じて「USB」と略してはいけない。~~

オンラインストレージを使わずにデータをやり取りする為の最も基本的、かつ最も有用な手段である。最近は 16GB のものでも 1000 円程度と非常に安価で購入できるため、1 ～ 2 個持つておいて損は無いだろう。

・外付けハードディスクドライブ

- ・価格帯：500GB 6000 円程度～、750GB 8000 円程度～

USB フラッシュメモリでは足りないような大容量のデータ（ムービー、大量の写真等）をオフラインで持ち運ぶ場合に有用なデバイスである。転送速度の関係から、出来れば USB3.0 対応のものを購入することが望ましい。

なお、テレビの録画に対応しているモデルも存在するため、自分の生活パターンにも依るものではあるが、1 台買っておけば幅広い用途に対応することができるだろう。

・ワイヤレスマウス

- ・価格帯：2000 円程度～(Logicool 社)

ノートパソコンを購入すると標準で添付されていることも少なくないが、添付されていないモデルも少なからず存在する。そのような場合は、1 個だけでもワイヤレスのマウスを持っておくと、単純に操作性が向上するなど便利になるだろう。

市場には様々なメーカーの製品が出回っているが、個人的には Logicool 社製のマウスをお勧めする。ワイヤレスマウスでも電池の消耗が殆どないため、電池を交換せずに 1 年～ 1 年半持つということも少なくない上、レシーバの感度も非常に良好である為、操作性に全く難を感じずに使用することができるのだ。

・USB→LAN変換アダプタ (LANケーブル端子のないパソコンのみ)

- ・価格帯：1500 円程度～

例えば MacBook Air や MacBook Pro Retina ディスプレイモデルには、LAN ケーブルを接続するための端子が存在しない（完全に無線 LAN のみを利用してインターネットに接続することが想定されている）。しかしながら大

学宿舍のインターネット回線は有線での接続となるため、このままでは Macbook Air をインターネットに接続することができないのである。

そのような場合に有効となるのが、この変換アダプタである。USB-A オス端子と LAN ケーブル・メス端子が両端となっており、USB 端子をパソコンに接続することで、LAN ケーブルを利用してインターネットに接続することが可能になるのだ。

6. おわりに

改めて見てみると、本当に現在の市場には様々な性能の PC が出回っているものですね。秋葉原に行って特価品の PC を狙い撃ちしてくるも良し、性能をじっくり比較しながら自分だけの 1 台を探す（或いは、BTO で自分だけの 1 台を作る）というのもまた良しというものです。

この記事が少しでも、皆様のパソコンライフにとって有用なものになりますよう。

情報科学類誌 WORD 読者アンケート

題字 編集部 ふあい

文 編集部 ItosugI

1 あいさつ

五月病のみなさん、おはこんばんちは¹。そうでないみなさんもおはこんばんちは。11回目の集計では**4人**の方から回答をいただきました！ 今回は募集期間が短いにも前回と変わらない人数からいただけてうれしいです。

2 今回の粗品

新しい粗品が増えました！！ この機会に是非アンケート書いてください！！！ 詳しい情報は過去記事をご覧ください。手元に過去記事がない方は下の URL に今すぐアクセス！！

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

2.1 Intelロゴ入りポロシャツ（図1）

白と紺の2色がおりなすハーモニー！！ シンプルでいてかつ飽きさせないそんなデザイン！！ サイズはS、M、Lの3種をご用意しております。※ただしガスマスクは付属しません。

2.2 まるで本物！？ メモリ型定規（図2）

さりげなく情報科学類生をアピールできる定規です。たくさんのデザインを用意しております。ただし目盛りはついてません。自分の知識とカンで長さを測りましょう！ ~~洪山あるから早く持つて行ってほしい。~~



図2：メモリ型定規



図1：Intelロゴ入りポロシャツ

*1 おはこんばんちは：おはよう+こんばんは+こんにちはのこと。

2.3 琴浦さん名刺（図3） ←NEW!!

(>ワ<=>ワ<)コシコシコシで有名な琴浦さんと琴浦町のコラボ名刺です。デザインは10種類!! それぞれ10枚、計100枚準備しております!! ただしアンケート1枚につき1枚までとさせていただきます。(ゲス顔)



図3：琴浦さん名刺

2.4 シェフのきまぐれ粗品

ゴネれば他の~~まぐれ~~粗品が出てくるかもしれません。チャレンジャーなそこの君!! 挑戦待ってるぞ!!

3 アンケート集計

3.1 Q1：所属を教えてください。

- ・cosys 学類：1人
 - ・(IMAGINE THE FUTURE.されました)¹² 学類：1人
 - ・iit 専攻：1人
 - ・ごかぼうプロ：1人

WORD の読者の半分はまともな人間でできています。後半分は……。iit 専攻は知能機能システム専攻のことなんですね！ 初めて知りました。ごかばうプロとはいつたい何なんだ……。

3.2 Q2：性別を教えてください。

- ・男：2人
 - ・オコジョ：1人
 - ・マナの奥さんケル：1人

語尾が『ケル』ということはプリキュアの淫獸^{*3}ですね！ わかります。

3.3 Q3：年齢を教えてください。

3.4 Q4：良かったと思う記事などがあれば教えてください。

前号 WORD 25 号のタイトルなどの一覧は以下の通りです。

- 1.表紙 2.号名 3.目次 4.mbed系男子になろう！ 5.そうだ、琴浦町に行こシコシコシコシ
 - 6.プログラミング言語探訪記Forth編 7.GRな日々。XV 8.SECCON CTF全国大会 9.引越しを完了させよう！
 - 10.書籍紹介 11.つくペディア2013年度版 12.WORD読者アンケート 13.次回予告 14.編集後記 15.裏表紙
 - 16.アンケート用紙 17.配布場所 18.配布時期 19.配布媒体 20.冊子の厚さ

- ・5. そうだ、琴浦町に行こシコシコシコシコシ：1票
 - ・9. 引越しを完了させよう！：1票
 - ・11. つくペディア 2013 年度版：1票
 - ・12. WORD 読者アンケート：1票
 - ・20. 冊子の厚さ：1票

アンケートに票が入ってるひやっほーい！！ アンケート万歳！！ アンケート万歳！！！ 後はどうでもいいわ！！！！ では、回答晒し上げコーナーへ参ります。

*2 (IMAGINE THE FUTURE.されました)：筆者の独断と偏見で不適切な表現と判断されました。

*3 淫獣：マスコットキャラクターのこと。

12.



【cosys 学類 kmk さん (20 歳)】

20. ガンガンを超える日を楽しみにしています。

【ごかぼうプロ IMAGINE THE FUTURE.さん^{*4}(20 → 21 ってなんか悲しい歳)】

月まで届け僕らの WORD !! しかし 100 ページを超えると今あるホチキスでは留められなくなってしまいま
す。なので是非とも強力なホチキスを我々にください！！

(>ワ<ミ>ワ<)コシコシコシ(>ワ<ミ>ワ<)コシコシコシ(>ワ<ミ>ワ<)
コシコシコシ(>ワ<ミ>ワ<)コシコシコシ(>ワ<ミ>ワ<)コシコシコシ(>ワ<ミ>ワ<)
コシコシコシ(>ワ<ミ>ワ<)コシコシコシ(>ワ<ミ>ワ<)コシコシコシ(>ワ<ミ>ワ<)

【(IMAGINE THE FUTURE.されました) 学類 IMAGINE THE FUTURE.さん(100 歳)】

「(^o^)」モリ! 「(^o^)」モリ!

11.

是非、井上サイクルの評判についても言及してほしい。

【iit 専攻 IMAGINE THE FUTURE.さん(約 7 π 歳)】

「筑波大学生みたいな偏差値高い人以外の客の仕事は受けたくない」と店主さんがおっしゃっていたという情
報が入っております。また「うちで買った自転車しか見ないよ」ともおっしゃってたそうです。以上の証言から
各自判断してください。

*4 IMAGINE THE FUTURE.さん：アンケートの名前欄が NULL だった回答は、IMAGINE THE FUTURE.さんとして掲載
しています。

3.5 Q5：良くなかったと思う記事などがあれば教えてください。

- ・9.引越しを完了させよう！：1票
 - ・11.つくペディア2013年度版：1票

何が悪かったんですかね？回答晒し上げコーナーで詳しく見てみましょう。

11. 特別コラムを読んでから夜にトイレいけなくなつた

【cosys 学類 kmk さん (20 歳)】

ちょっと怖い話でしたね。解決策として、多い日でも安心夜用をつけて寝るといいですよ。

9.

Emacs 派が黙っちゃいない

【iit 専攻 IMAGINE THE FUTURE.さん(約 7 歳)】

本誌 WORD は宗教的にニュートラルな立場を採用しています。

3.6 Q6：過去の記事に関する感想を教えてください。

アメリカンパワーフードまた見たいです。

【cosys 学類 kmk さん (20 歳)】

今号でもアメリカンパワーフードやりました！！是非読んでください。旧記事であればWORD 18号⁵ですね。

http://www.word-ac.net/?page_id=721 こちらをご覧ください！！

もっと顔文字の記事を！！！！！！

顔文字の記事といえばはろぺり氏ですね。もっと書くように伝えておきます！！

*5 WORD 18号:『え～？WORD～？何それ何それ知りた～い><最新のWORD19号とか全然読めなくてえ～ふんふくり～ん（怒）ピヨピヨとすら鳴けないんですよ。えつWORDの最新号は18号で、女子力がアップするんですね！おぼえたぞお～メモメモ！キュンキュンキュン！キュンキュンキュン！私のハードディスクに記録しているのでありますっ☆号』のこと。

WORD娘（イカ的なやつ）はどこへ……

【iit 専攻 IMAGINE THE FUTURE.さん(約 7 ピース)】

あなたの心と WORD 16 号⁶ http://www.word-ac.net/?page_id=650 にい
ますよ！ そのほかの記事も絶賛公開中！！ 今すぐ WORD Press に急げ！！

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

3.7 Q7：自由兵团。

例のごとくそのまま晒し上げます。

ドナキチだ！ ミスターードーナツの回し者ですね。やっぱり課金してるんですかね？

【ごかぼうプロ IMAGINE THE FUTURE.さん(20→21ってなんか悲しい歳)】

通常号では難しいですね。イベントの際はやるかもしれません。楽しみにしてね！！

あと、Word じゃなくて WORD です！！

『変態王子と笑わない猫』ですかね？私はアニメも小説も見てないでよくわからないんですね。しかし一時期ネットニュースで合法的にペロペロできる飴とケーキなるものを見たときは日本はまだまだ終わっちゃいないなと思いました（小学生並みの感想）。



【iit 専攻 IMAGINE THE FUTURE. さん(約 7 歳)】



*6 WORD 16号：『侵略されたいでゲソ号』のこと。



【cosys 学類 kmk さん（20歳）】

貴様…合成したのか…。なぜだ…。なぜ今……サシャとゴジラを悪魔合体させた？ イヤわからないな。なぜ貴様は悪魔合体させた？

死ぬ寸前まで走れ！！

4 おわりに

以上で 11 回目のアンケート集計は終了です。WORD の新デザインに関する意見をじんじんバリバリ募集しております。ということで、今号でもアンケートを実施いたします。アンケートの回収 BOX は前回と変わらず、WORD 編集部室前（3C212、情報科学類生ラウンジ横の怪しい部屋）、学類計算機室前（3C113、3C205）に設置しています。ご協力お願いします。回答数が多い場合はすべてを掲載しないかもしれません。ご了承ください。

情報科学類誌

WORD

From College of Information Science

あ…(冊子) 号

発行者

情報科学類長

編集長

吉村優

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)

2013年6月20日 初版第一刷発行

(512部)