



HaskellとDebianの辛くて甘い関係

Kiwamu Okabe



自己紹介

- ☆ twitter: @master_q
- ☆ http://www.masterq.net/
- ☆ 職業: コピペプログラマ
- ☆ Debianのおかげで結婚できました
- ☆ Haskell忍者になるべく修行中
- ☆ (NINJA:=No Income No Job or Asset)
-  Perl忍者リスペクト

Haskellを知っていますか？

いくつもの特徴を持った関数型言語です。
ちょっとだけ解説します。

(注: ぼくはHaskell初心者です)



静的型付け

- ☆ 全てのデータ/関数には型がついている
- ☆ 型が合わないとエラー
- ☆ 暗黙の型変換なんてもんはない
- ☆ "型による設計"
- ☆ 今まで動作時エラーだったものが、..
- ☆ コンパイル時エラーになる。やった!



型推論

- ☆ 全部の関数に型書かなくてもOK
- ☆ たまーに推論失敗するけど。。。
- ☆ 公開関数には型書こう
- ☆ where内の非公開関数は省略がいいかも
- ☆ hlintの言うことは聞いとけ



型クラス

returnとか作れるよ

```
class Functor f => Applicative f where  
    return :: a -> f a
```

```
instance Applicative [] where  
    return a = [a]
```

```
instance Applicative Maybe where  
    return a = Just a
```

おつしやれー



パターンマッチ

☆ 型 + パターンマッチ = 表現力∞

もうこんなの嫌

```
switch (l->l_stat) {  
    case LSONPROC:  
        break;  
  
    case LSRUN:  
        if (l->l_swttime > outpri2) {  
            outl2 = l;  
            outpri2 = l->l_swttime;  
        }  
        break;  
    ...
```



遅延評価

- ☆ 本当に必要になるまで評価されない
- ☆ 無限再帰構造を持つ純粹世界は作れる
- ☆ 現実世界(=IOモナド)が純粹世界を手招き
- ☆ 手招きされた分のみ純粹世界が評価される
- ☆ 使用上の注意をよく読み用法用量を守って



コンパイルして実行

- ☆ runhaskellでインタプリタ的にも使える
- ☆ でもコンパイルしてしまえば環境を選ばない
- ☆ Haskell環境のないサーバに直バイナリOK
- ☆ コンパイラだから最適化によっては速いかも



読みやすく、書きやすい文法

where厨になることうけあいです

-- http://hackage.haskell.org/packages/archive/containers/
-- latest/doc/html/Data-Tree.html から抜粋

```
data Tree a = Node {  
    rootLabel :: a,           -- ^ label value  
    subForest :: Forest a   -- ^ zero or more child trees  
}
```

```
type Forest a = [Tree a]
```

-- | The elements of a tree in pre-order.

```
flatten :: Tree a -> [a]
```

```
flatten t = squish t []
```

```
where squish (Node x ts) xs = x:Prelude.foldr squish xs ts
```



ghciでインタラクティブ ラブ

```
$ sudo apt-get install haskell-platform  
$ rehash  
$ ghci  
GHCi, version 7.0.4: http://www.haskell.org/ghc/ :? for help  
Loading package ghc-prim ... linking ... done.  
Loading package integer-gmp ... linking ... done.  
Loading package base ... linking ... done.  
Prelude> fmap (foldr (++) "") . flip replicate "hoge" [1..3]  
["hoge","hogehoge","hogehogehoge"]
```

```
$ irb  
irb(main)> (1..3).collect{|a|s="" ; a.times{ s+="hoge"} ; s}  
=> ["hoge", "hogehoge", "hogehogehoge"]
```

似てるー



cabalを使えばよりどり緑

- ☆ Hackage := Haskellのライブラリ
- ☆ Ruby gemみたいな感じ
- ☆ 使い方: "cabal install ライブラリ名"
- ☆ 依存したHackageを芋蔓式にインストール



Debianならcabal使うのも簡単!

```
$ sudo apt-get install cabal-install  
$ rehash  
$ cabal update  
$ cabal install caretta  
# がりがりっとコンパイルされる  
$ ~/.cabal/bin/caretta  
caretta version 0.0.4
```

haskell-platformをインストールすれば
cabal-installは自動でインストールされるけど



でもcaballには色々不都合が…

Ruby gemとか使ってればよくある日常

```
$ sudo gem update  
$ sudo gem install earchquake  
# 月日は流れ、、そしてある日、  
$ sudo gem update
```

これで以前インストールしていたearchquake
パッケージは依存ライブラリを含めて最新版に
なる



yesod hackageのあるある (続く)

```
$ cabal update # ローカルの Hackage データベースを更新  
$ cabal install yesod  
# 後日yesodを最新版に更新しようと思いつつ  
$ cabal upgrade  
--snip--
```

The 'cabal upgrade' command has been removed
because people found it confusing and it often
led to broken packages.

--snip--

なにこれ————!?



yesod packageのあるある (完)

しようがない、必要なパッケージだけ更新しよう

```
$ cabal install yesod
# yesodが動作しない or 依存関係をcabalが自動解決しない
# とりあえずcabalでインストールしたHackageを全部消そう
$ rm -rf ~/.ghc ~/.cabal
$ cabal update
$ cabal install yesod
```

さっきのyesodのバグが再現しない。

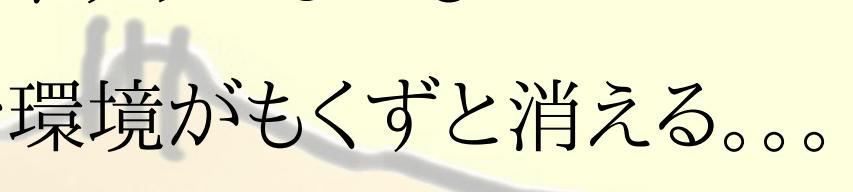
ふつーに動いとる。

なぜだ——————!?



これじやあ、、、 #orz

せっかくセットアップしても

経年変化で環境がもくずと消える。。。 

☆ orz

☆ orz orz

☆ orz orz orz

☆ orz orz orz orz

☆ orz orz orz orz orz



どうしてcabalはこんななの？

それはそれはいくつもの問題があるんじやよ

☆ cabalのしくみの問題

☆ Hackage作者達の文化の問題

の2つに大別されます。



Hackage 作成の文化的問題

```
$ cabal info yesod
```

```
--snip--
```

```
Versions available: 0.6.7, 0.7.2, 0.7.3, 0.8.0, 0.8.1, 0.8.2,  
0.8.2.1, 0.9.1, 0.9.1.1 (and 35 others)
```

```
--snip--
```

```
Dependencies: yesod-core >=0.9.1.1 && <0.10,  
yesod-auth ==0.7.*, yesod-json ==0.2.*,  
yesod-persistent ==0.2.*, yesod-form ==0.3.*,  
monad-control ==0.2.*, ...
```

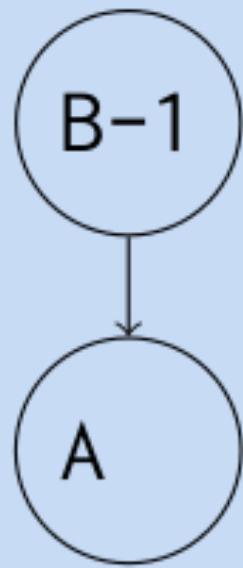
上限バージョンを決めてしまうんだ。。。#orz

未来は誰にもわからないんじやないのか？



cabal の実装上の問題 #1

B-1 packageがA packageに依存している

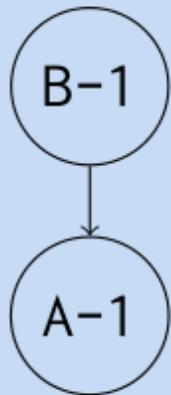


Assume B-1 depends on any version of A.



cabal の実装上の問題 #2

B-1をインストールするとA-1も一緒にに入る

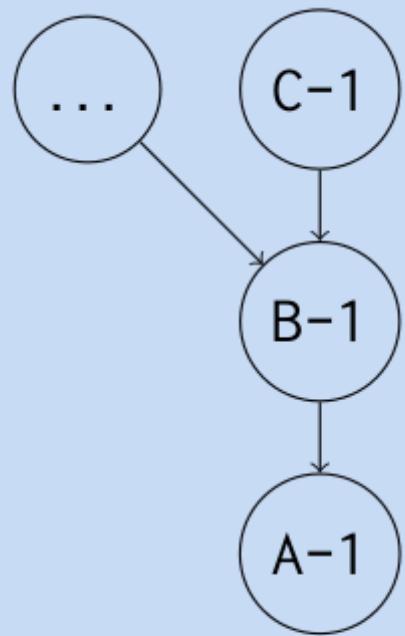


You install B-1 on your system, fixing the dependency to A-1.



cabal の実装上の問題 #3

B-1に依存したHackage群をインストール

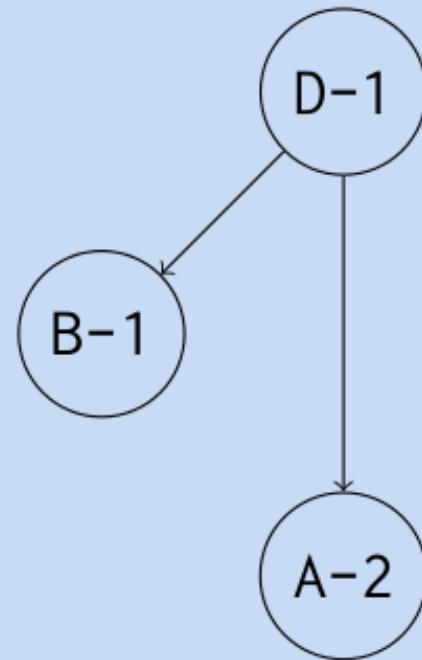
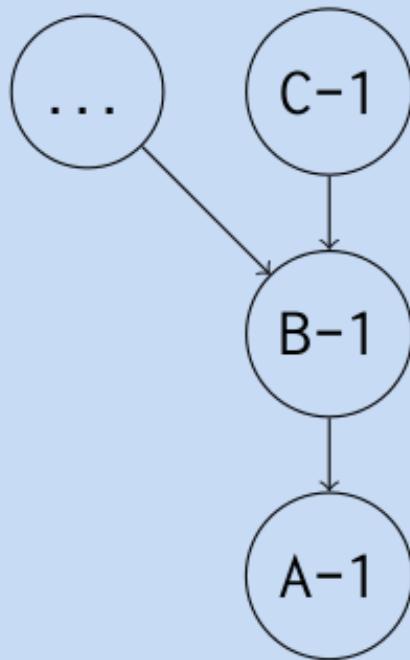


Many other packages that depend on B-1 are installed later.



cabal の実装上の問題 #4

A-2に依存しているD-1をインストールしよう

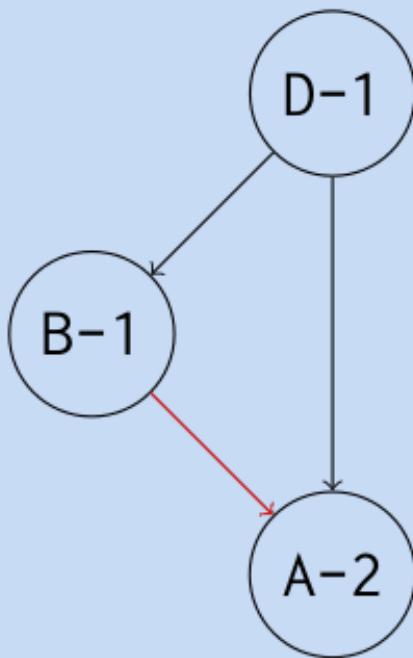
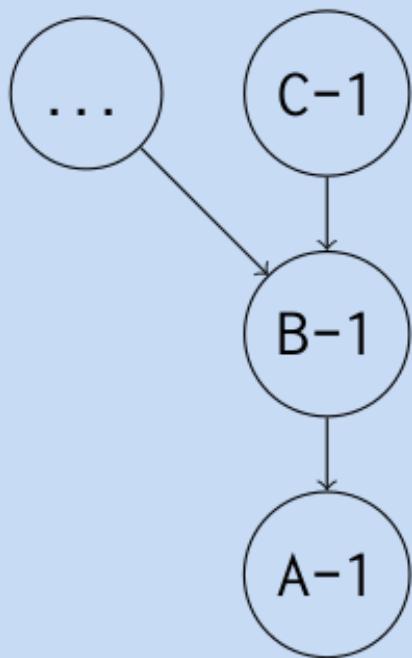


Now we want to install D which depends on A-2 (!) and B.



cabal の実装上の問題 #5

A-1のかわりにA-2をインストールするハメに

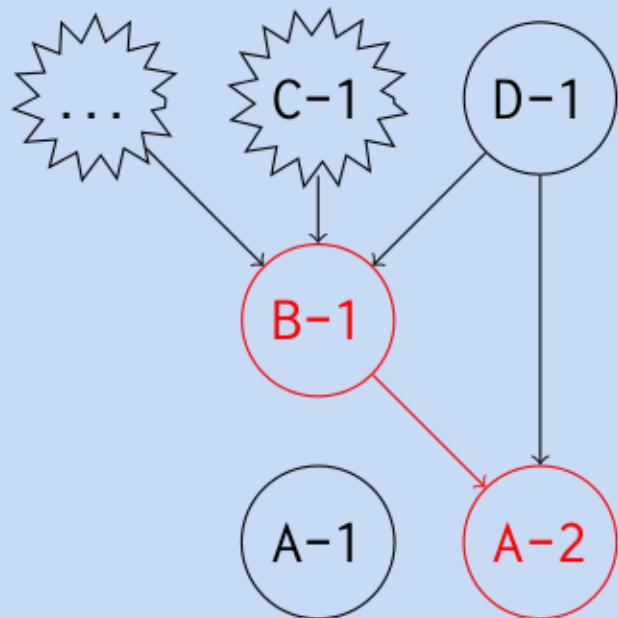


Since B still depends on A-1, the install plan selects A-2.



cabal の実装上の問題 #6

B-1に依存していたHackageが依存が壊れる



Upon actual installation, the old B-1 is destructively updated ...



Haskellの外をcabalは感知しない

hcwiid packageを例に取ると…

- ☆ hcwiid packageはlibcwiid-devに依存
- ☆ cabal install hcwiidしても…
- ☆ 自動でapt-get install libcwiid-devする？
- ☆ するワケない。。。auto-aptたん。。。orz



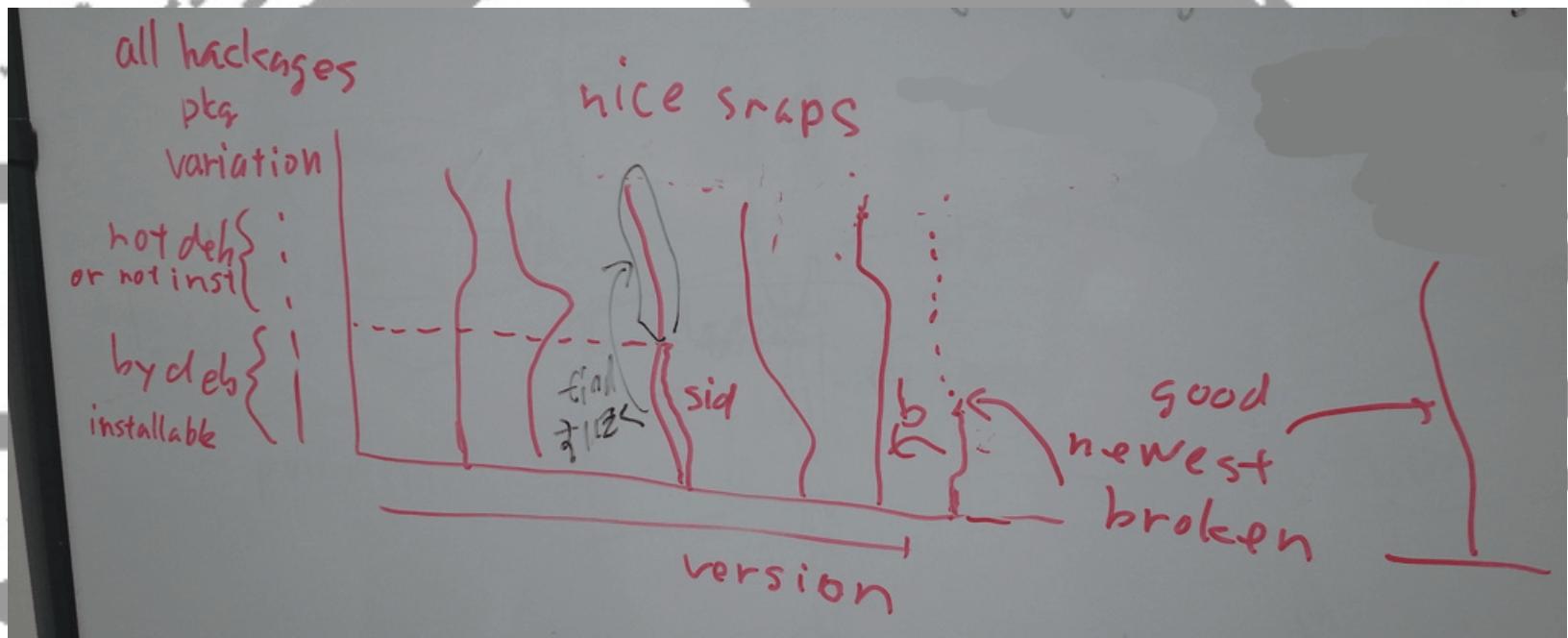
Hackage群全てを最新にはできない

yesod, hakyll, hamletを例に取ると...

- ☆ yesod-0.9.2はhamlet-0.10.*に依存
- ☆ hakyll-3.2.0.8はhamlet-0.{7,8}.*に依存
- ☆ 理由:hamletのAPI変更にhakyllが追従×
- ☆ yesodとhakyllを同時に使えない?
- ☆ orz (今は改善されました)



妄想: @khibinoさんのアイデア



可能性の中から最新を選択してくれたらイイナ
したらcabalさん最強!



そこでDebian DEATHよ!

- ☆ 最強のcabalができるまでどうすれば、..
- ☆ cabalダメならdeb化しちゃえばイイじやない
- ☆ Haskell以外のライブラリに紐づけ可だし



Hackageのdeb化 #1

Haskellパッケージ化環境整備

```
$ sudo apt-get install \  
  haskell-debian-utils haskell-devscripts
```

debhelperなにかがインストールされる。



Hackageのdeb化 #2

cabal-debianでdebianディレクトリ作成

```
$ wget http://hackage.haskell.org/packages/archive/\
  hcwiid/0.0.1/hcwiid-0.0.1.tar.gz
$ tar xfz hcwiid-0.0.1.tar.gz
$ cd hcwiid-0.0.1/
$ cabal-debian --debianize --ghc \
  --maintainer="Kiwamu Okabe <kiwamu@debian.or.jp>""
$ ls debian
changelog compat control copyright rules
```



Hackageのdeb化 #3

```
$ debuild -rfakeroot -us -uc  
$ ls ../*hcwiid*deb  
..../libghc-hcwiid-dev_0.0.1-1~hackage1_amd64.deb  
..../libghc-hcwiid-doc_0.0.1-1~hackage1_all.deb  
..../libghc-hcwiid-prof_0.0.1-1~hackage1_amd64.deb
```

- ☆ 通常使用するライブラリ
- ☆ Haddockで生成されたドキュメント
- ☆ プロファイラ対応ライブラリ
がでけた!



どうしてこんなに簡単なの？

debhelperの力です。

```
$ cat debian/rules
#!/usr/bin/make -f
include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/class/hlibrary.mk
$
```

ご覧の通りincludeしかないです。



hlibrary.mk #build

```
# build
DEB_SETUP_BIN_NAME ?= debian/hlibrary.setup
BUILD_GHC := $(DEB_SETUP_BIN_NAME) build

$(DEB_SETUP_BIN_NAME):
    if test ! -e Setup.lhs -a ! -e Setup.hs; then echo "No setup script found!"; exit 1; fi
    for setup in Setup.lhs Setup.hs; do if test -e $$setup; then ghc --make $>
$> $$setup -o $(DEB_SETUP_BIN_NAME); \
    exit 0; fi; done

build/libghc-$(CABAL_PACKAGE)-prof build/libghc-$(CABAL_PACKAGE)-dev::: build-ghc-stamp

build-ghc-stamp: dist-ghc
    $(BUILD_GHC) --builddir=dist-ghc
    touch build-ghc-stamp
```

cabalが普段やっていることと同じ



hlibrary.mk #install

```
# install
debian/tmp-inst-ghc: ${DEB_SETUP_BIN_NAME} dist-ghc
    ${DEB_SETUP_BIN_NAME} copy --builddir=dist-ghc --destdir=debian/tmp-ins
st-ghc

install/libghc-${CABAL_PACKAGE}-dev:: debian/tmp-inst-ghc debian/extrdepends
    cd debian/tmp-inst-ghc ; find usr/lib/haskell-packages/ghc/lib/ \
        !( ! -name "*_p.a" ! -name "*.p_hi" ) \
        -exec install -Dm 644 '{}' ../${notdir $@}'{}' '';
pkg_config='${DEB_SETUP_BIN_NAME} register --builddir=dist-ghc --gen-pk
g-config | sed -r 's,.*:,,'; \
$(if ${HASKELL_HIDE_PACKAGES},sed -i 's/^exposed: True$$/expose
d: False/' $$pkg_config;) \
    install -Dm 644 $$pkg_config debian/${notdir $@}/var/lib/ghc/pa
ckage.conf.d/$$pkg_config; \
    rm -f $$pkg_config
if [ 'z${DEB_GHC_EXTRA_PACKAGES}' != 'z' ] ; then \
    echo '${DEB_GHC_EXTRA_PACKAGES}' > \
    debian/${notdir $@}/usr/lib/haskell-packages/ghc/lib/${CABAL_PA
CKAGE}-${CABAL_VERSION}/extra-packages ; \
fi
dh_haskell_provides -p${notdir $@}
dh_haskell_depends -p${notdir $@}
dh_haskell_shlibdeps -p${notdir $@}
```



どーせなら本家にdebをアップロード

- ☆ 複数台PCの環境同期めんどい
- ☆ そのうちubuntuも取り込むかもしれんし
- ☆ やつちまえ! やつちまえ!



とりあえずDMになりましょう

- ☆ DDにならなくてもできることはアル
- ☆ 結構簡単になれる
- ☆ こちらからドゾ

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

後日談) JoachimからメールがあってDDやDMでなくとも
pkg-haskellチームになれるそうです！

>> [Q2] Can the person as not DM (Debian Maintainer) join
>> pkg-haskell team? Or they should become DM, first?
> No need to be a DM, as there are DDs around that can do
> the sponsoring."



aliothのアカウントを作りましょう

- ☆ アカウントの作り方は日記に書いた
- ☆ 読んでちょ

<http://d.masterq.net/?date=20100325>

(あんま詳しくないかも。。。)



pkg-haskellチームにjoinセヨ!

ボスは

☆ Joachim Breitner

☆ E-mail: nomeata@debian.org

debian-haskell@lists.debian.org 常駐？

頼まなくても活動してると勝手に登録される



さて作りますか

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

をまずは熟読のこと。とりあえずITPメール。

Package: wnpp

Severity: wishlist

Owner: Kiwamu Okabe <kiwamu@debian.or.jp>

* Package name : haskell-ansi-wl-pprint

Version : 0.6.3

Upstream Author : Daan Leijen, Max Bolingbroke
<batterseapower@hotmail.com>

* URL : <http://github.com/batterseapower/ansi-wl-pprint>

Vcs-Browser :

<http://anonscm.debian.org/gitweb/?p=collabaint/haskell-ans>

* License : BSD3



cabal-debianコマンドでdeb化

この時、debian/controlをpkg-haskell風に

```
$ vi debian/control
```

Maintainer: Debian Haskell Group \

<pkg-haskell-maintainers@lists.alioth.debian.org>

Uploaders: Kiwamu Okabe <kiwamu@debian.or.jp>

Vcs-Darcs: \

<http://darcs.debian.org/pkg-haskell/haskell-ansi-wl-pprint>

Vcs-Browser: [http://darcs.debian.org/cgi-bin/darcsweb.cgi?r=](http://darcs.debian.org/cgi-bin/darcsweb.cgi?r=(pkg-haskell/haskell-ansi-wl-pprint))(pkg-haskell/haskell-ansi-wl-pprint)

DM-Upload-Allowed: yes

DMでもdput可能にしておこう



debian/changelogにも注意

リリースしていないバージョンには

とりあえずUNRELEASEDマークをつける

haskell-ansi-wl-pprint (0.6.3-2) UNRELEASED; urgency=low

- * repo is moved to darcs.
- * change Vcs-* lines on debian/control.

-- Kiwamu Okabe <kiwamu@debian.or.jp> Wed, 12 Oct 2011

haskell-ansi-wl-pprint (0.6.3-1) UNRELEASED; urgency=low

- * Debianization generated by cabal-debian

-- Kiwamu Okabe <kiwamu@debian.or.jp> Wed, 05 Oct 2011

darcsリポジトリを作る

```
$ sudo apt-get install darcs  
$ pwd  
/home/kiwamu/deb/haskell-ansi-wl-pprint/debian  
$ darcs init --darcs-2  
$ darcs record -a -l -m "Initial Check-In"  
Finished recording patch 'Initial Check-In'  
$ darcs put kiwamu-guest@darcs.debian.org:/darcs\  
 /pkg-haskell/haskell-ansi-wl-pprint  
Finished applying...  
Put successful.
```

debianディレクトリだけ管理って。。。

どんなGentooだよwwwww



darcs フックを設定

```
$ ssh kiwamu-guest@darcs.debian.org \
/darcs/pkg-haskell/tools/add-hooks.sh \
haskell-ansi-wl-pprint
```

これでコミットログが

pkg-haskell-commits@lists.alioth.debian.org

に流れるようになる。



リリースを前提にしたお付き合い

リリース対象バージョンを決める。

debian/changelogの最新行をunstableに

```
$ dch # エディタが起動される
```

バージョンが一つ上げた。そしてdarcs push。

```
$ darcs record -a
```

```
$ darcs push
```

```
Sending mail to pkg-haskell-commits@lists.alioth.debian.org...
```

さつきのフックで通報されるはず。



Package Entropy Trackerが検出!

<http://pkg-haskell.alioth.debian.org/cgi-bin/pet.cgi>

"Ready for upload"状態になる。

With RC bugs (1)		
Package	Repository	
darcs-monitor	[more] 0.4.0-1 (0.4.0-2) [Tag: 0.4.0-1]	0.4. (exp)
Ready for upload (3)		
Package	Repository	
haskell-ansi-wl-pprint	[more] 0.6.3-3	
haskell-http-enumerator	[more] 0.7.1.2-1	0.7.



A blurred background image of a person in a green shirt sitting at a desk with a computer monitor in the background.

sponsor uploadを誰かにお願い

debian-haskell@lists.debian.org

にお願いメールすると、..、

..、たぶんJoachimが反応する。

無事dputされてしまえば、次回からは

自分でdputできますね :)



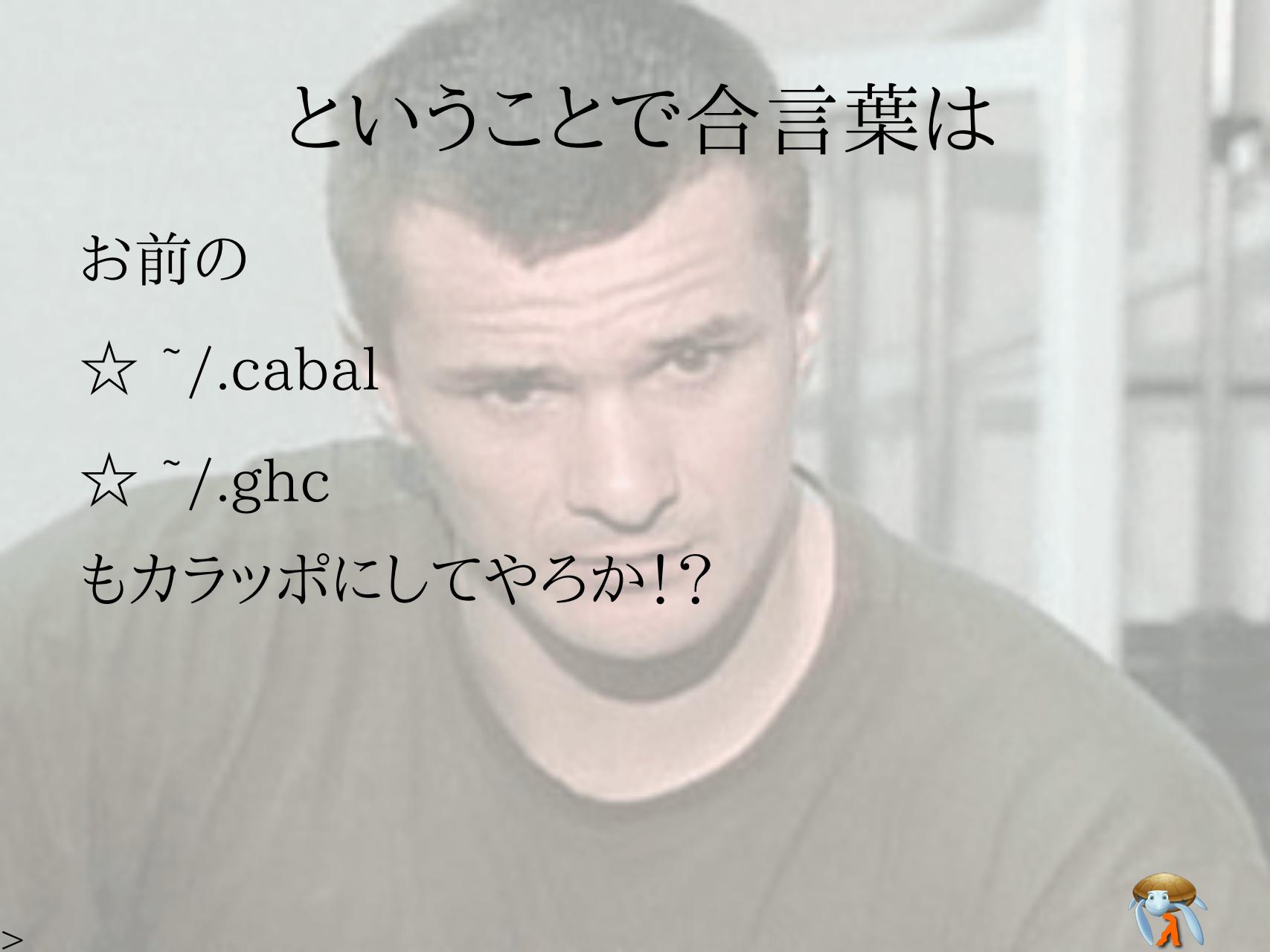
sponsorはこんなことしてるらしい

```
$ darcs get darcs.debian.org:/darcs/pkg-haskell/tools  
$ tools/pkg-haskell-checkout haskell-ansi-wl-pprint  
$ cd haskell-ansi-wl-pprint/  
$ debuild -i -I  
$ deborelease  
$ debuild clean  
$ cd debian/  
$ darcs tag $(dpkg-parsechangelog -lchangelog |  
    grep-dctrl -n -s Version .)  
$ darcs push -a
```

pkg-haskell-checkout失敗するような気が。。。

後darcsはhttp経由だととバグる sshでどぞ





ということで合言葉は

お前の

☆ ~/cabal

☆ ~/ghc

もカラッポにしてやろか!?



宣传: プrezentツール作ってます

- ☆ <http://carettah.masterq.net/>
- ☆ Haskell製
- ☆ <http://rabbit-shockers.org/> のパクリ
- ☆ このプレゼンもCarettah使ってます!
- ☆ そのうちapt-getできるようにしたる!

