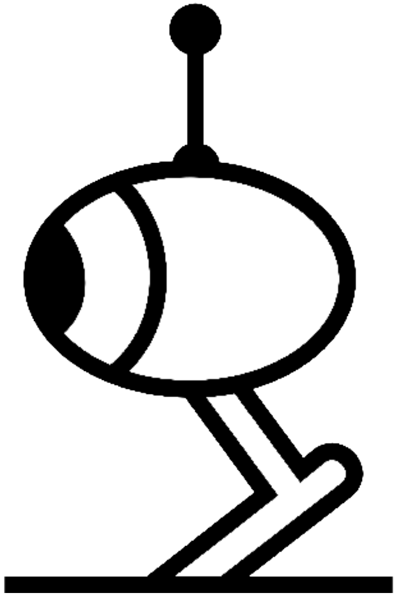
Haskell
et
Minitel

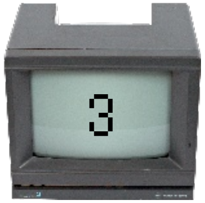
Frédéric Bisson

- Webmaster
- PHP, HTML, CSS, JavaScript
- Python, Bash etc.



Un jour

Pourquoi ne pas essayer
la programmation fonctionnelle ?



Oui, mais

Suivre un tutoriel donne
une impression (fausse) de facilité

Et si

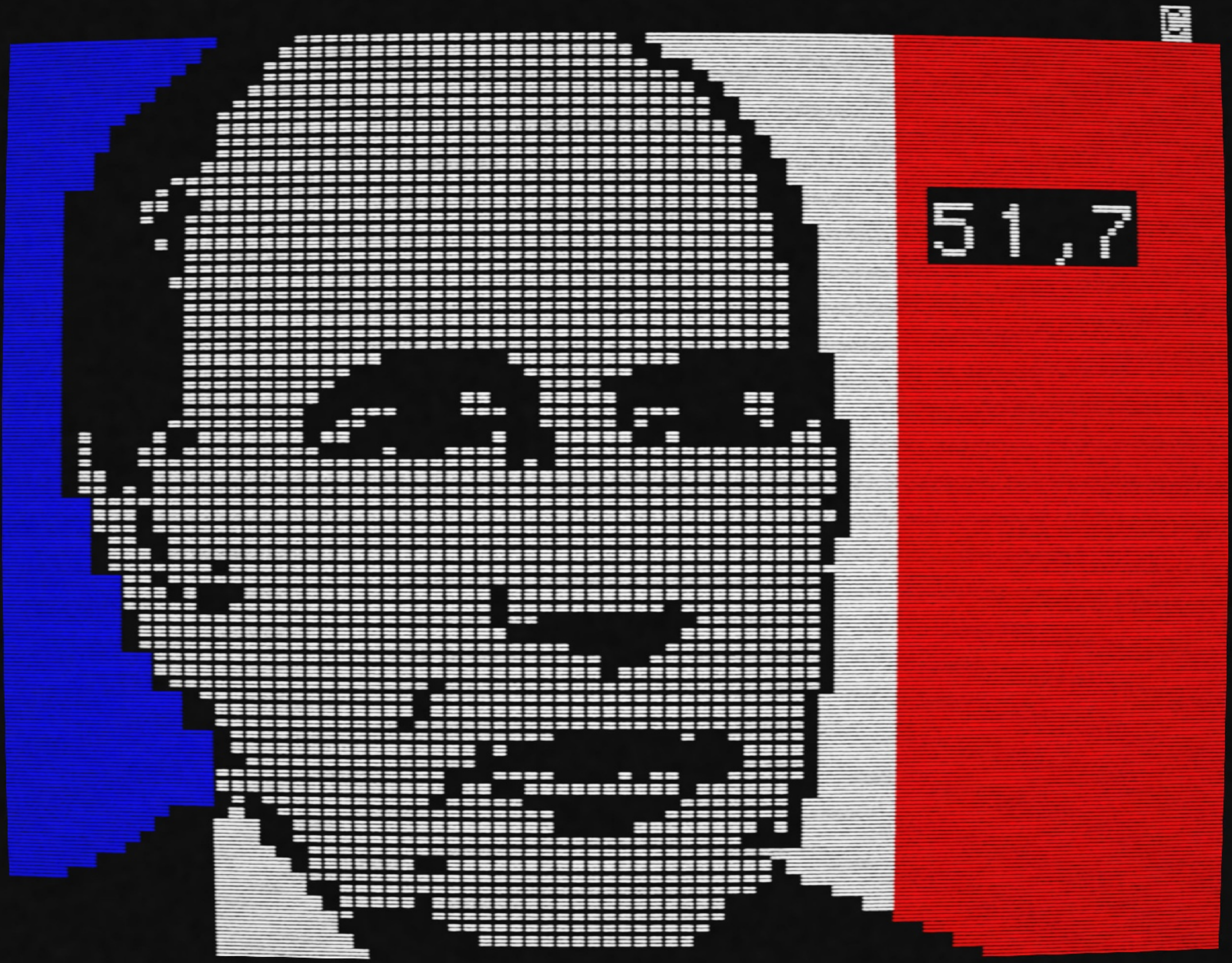
On contrôlait un Minitel branché sur un Raspberry Pi avec Haskell ?

+

+

Un Minitel ?

En 1981
c’est lui qui a appris à vos (grands-)parents
qui était leur nouveau président…


51,7

Minitel 1980-2012

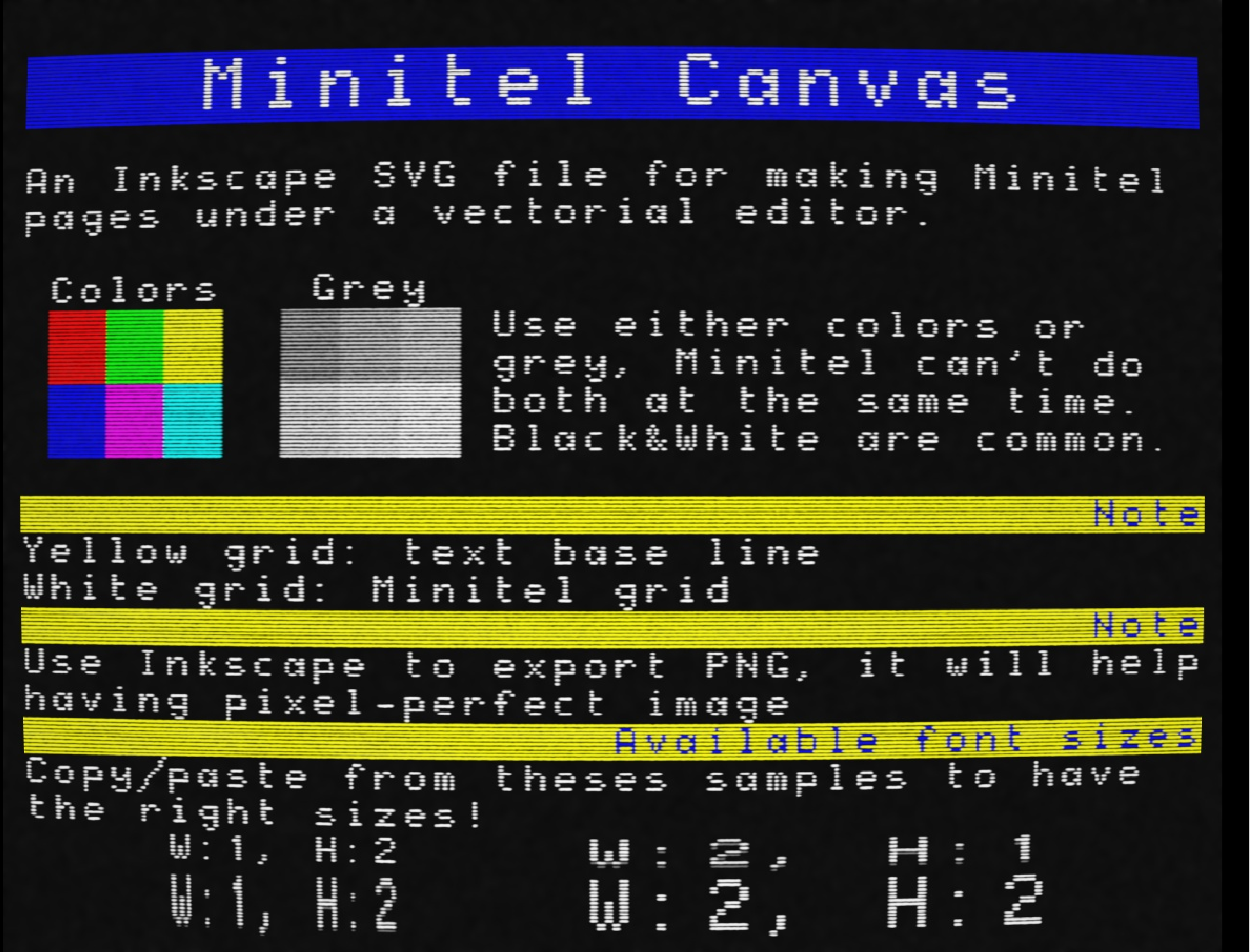
Terminal passif embarquant un modem

1200 bps

```
ABCDEFGHIJKLMNOPQRSTUVWXYZab
cdefghijklmnopqrstuvwxyzABCD
EFGHIJKLMNOPQRSTUVWXYZabcdef
ghijklmnopqrstuvyxyzABCDEFGH
IJKLMNOPQRSTUVWXYZabcdefghij
```

75 bps

```
ABCDEFG
```



Minitel Canvas

An Inkscape SVG file for making Minitel pages under a vectorial editor.

Colors

Grey

Use either colors or grey, Minitel can't do both at the same time. Black&White are common.

Note

Yellow grid: text base line
White grid: Minitel grid

Note

Use Inkscape to export PNG, it will help having pixel-perfect image

Available font sizes

Copy/paste from theses samples to have the right sizes!

W:1, H:2   W:2, H:1
W:1, H:2   W:2, H:2

Haskell ?

- Langage de programmation fonctionnelle

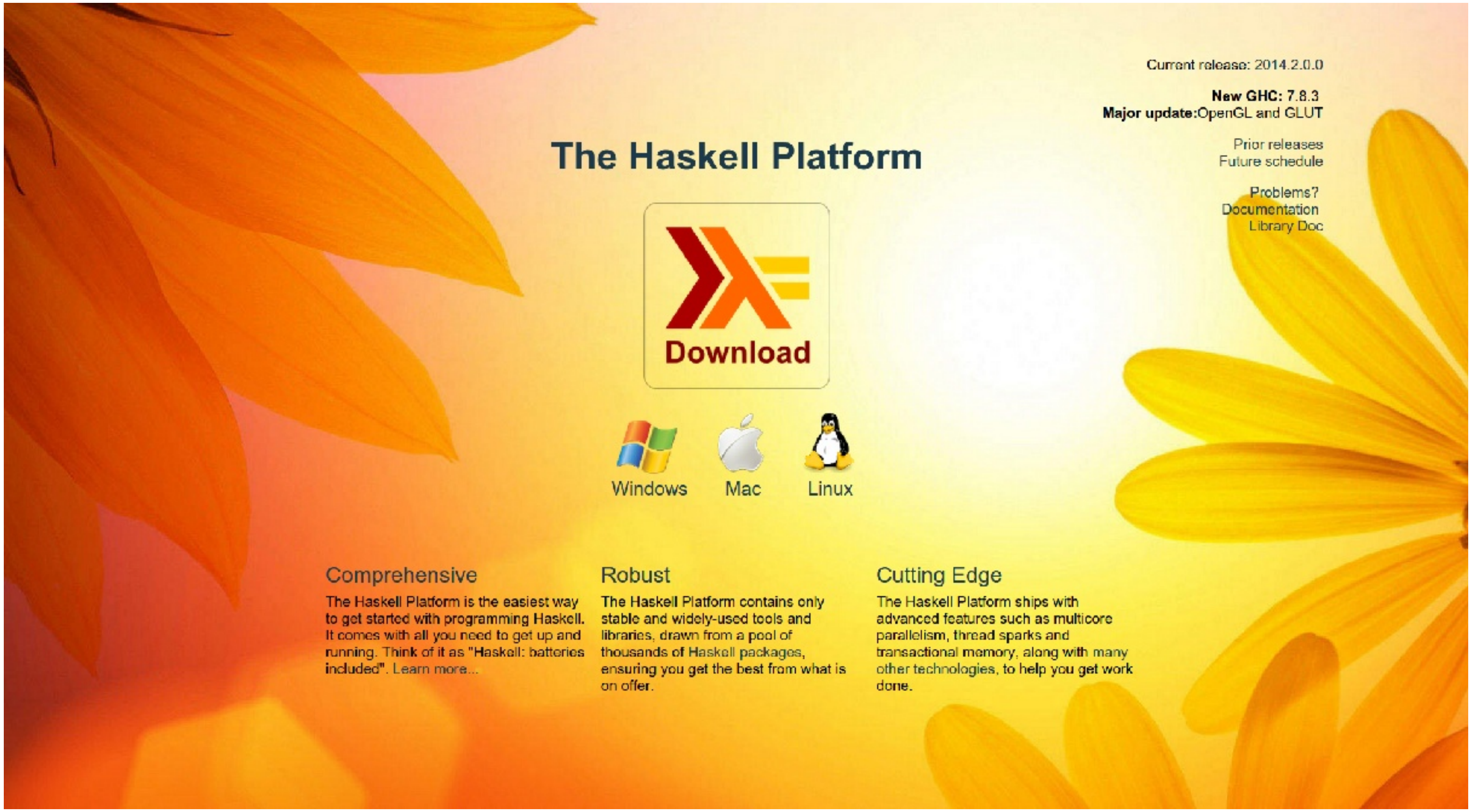
  fonctionnel ≠ procédural
- Typage statique fort
- Influencé par FP, Lisp, ML...
- Multiplate-forme

Ecosystème

- Compilation → GHC, runhaskell, GHCi
- Fabrication → Cabal
- IDE → Leksah, EclipseFP...
- Diffusion → Hackage
- Recherche → Hoogle, Hayoo
- Tests → QuickCheck...
- Analyse → Hlint, SourceGraph

GHC

- Installation recommandée : https://www.haskell.org/platform/



Current release: 2014.2.0.0

New GHC: 7.8.3
Major update: OpenGL and GLUT

Prior releases
Future schedule

Problems?
Documentation
Library Doc

The Haskell Platform

Comprehensive

The Haskell Platform is the easiest way to get started with programming Haskell. It comes with all you need to get up and running. Think of it as "Haskell: batteries included". Learn more...

Robust

The Haskell Platform contains only stable and widely-used tools and libraries, drawn from a pool of thousands of Haskell packages, ensuring you get the best from what is on offer.

Cutting Edge

The Haskell Platform ships with advanced features such as multicore parallelism, thread sparks and transactional memory, along with many other technologies, to help you get work done.

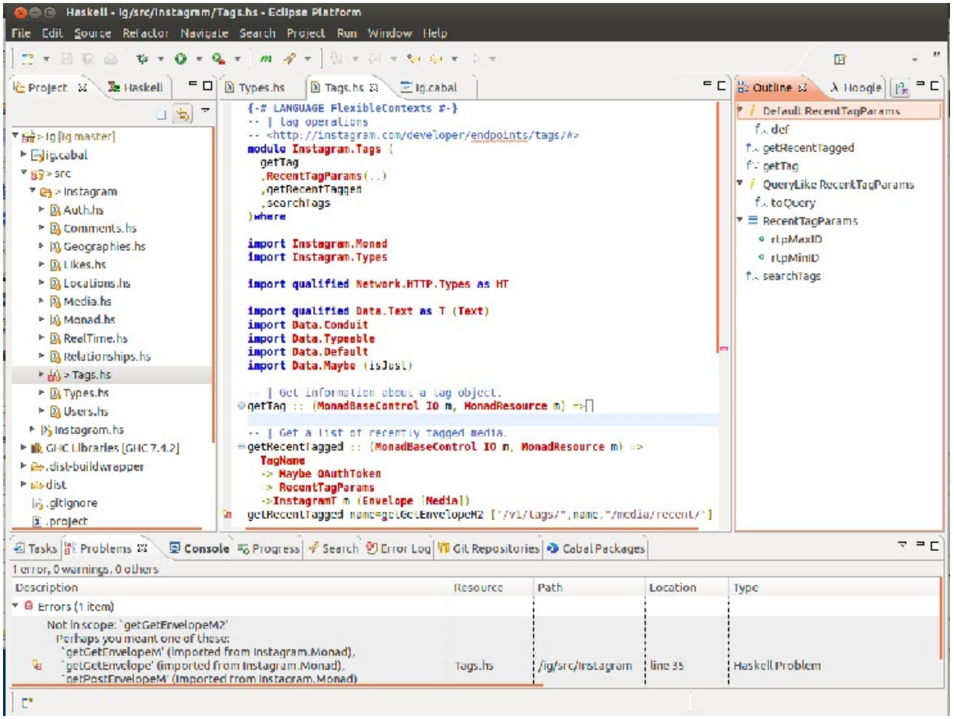
Cabal

- Construction (bibliothèque, exécutable)
- Dépendances
- Tests
- Bacs à sable
- Documentation (Haddock)
- Distribution

Common
Architecture for
Building
Applications and
Libraries

IDE

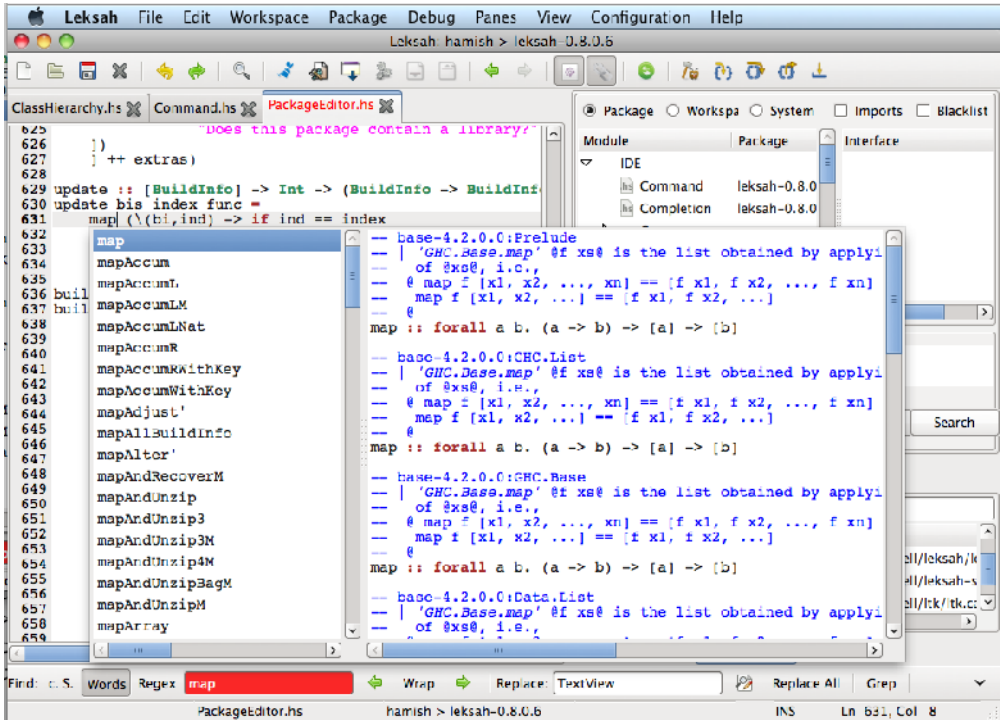
- EclipseFP : plug-in pour Eclipse
- Leksah : IDE dédié



```haskell
{-# LANGUAGE FlexibleContexts #-}
-- | Tag operations
-- <http://instagram.com/developer/endpoints/tags/#>
module Instagram.Tags (
  getTag
  ,RecentTagParams(..)
  ,getRecentTagged
  ,searchTags
)where

import Instagram.Monad
import Instagram.Types

import qualified Network.HTTP.Types as HT

import qualified Data.Text as T (Text)
import Data.Conduit
import Data.Typeable
import Data.Default
import Data.Maybe (isJust)

-- | Get information about a tag object.
getTag :: (MonadBaseControl IO m, MonadResource m) =>

-- | Get a list of recently tagged media.
getRecentTagged :: (MonadBaseControl IO m, MonadResource m) =>
  TagName
  -> Maybe OAuthToken
  -> RecentTagParams
  ->InstagramT m (Envelope [Media])
getRecentTagged name=getGetEnvelopeM2 ["/v1/tags/",name,"/media/recent/"]
```

1 error, 0 warnings, 0 others

Description	Resource	Path	Location	Type
Errors (1 item)				
Not in scope: `getGetEnvelopeM2' Perhaps you meant one of these: `getGetEnvelopeM' (imported from Instagram.Monad), `getGetEnvelope' (imported from Instagram.Monad), `getPostEnvelopeM' (imported from Instagram.Monad)	Tags.hs	/ig/src/Instagram	line 35	Haskell Problem

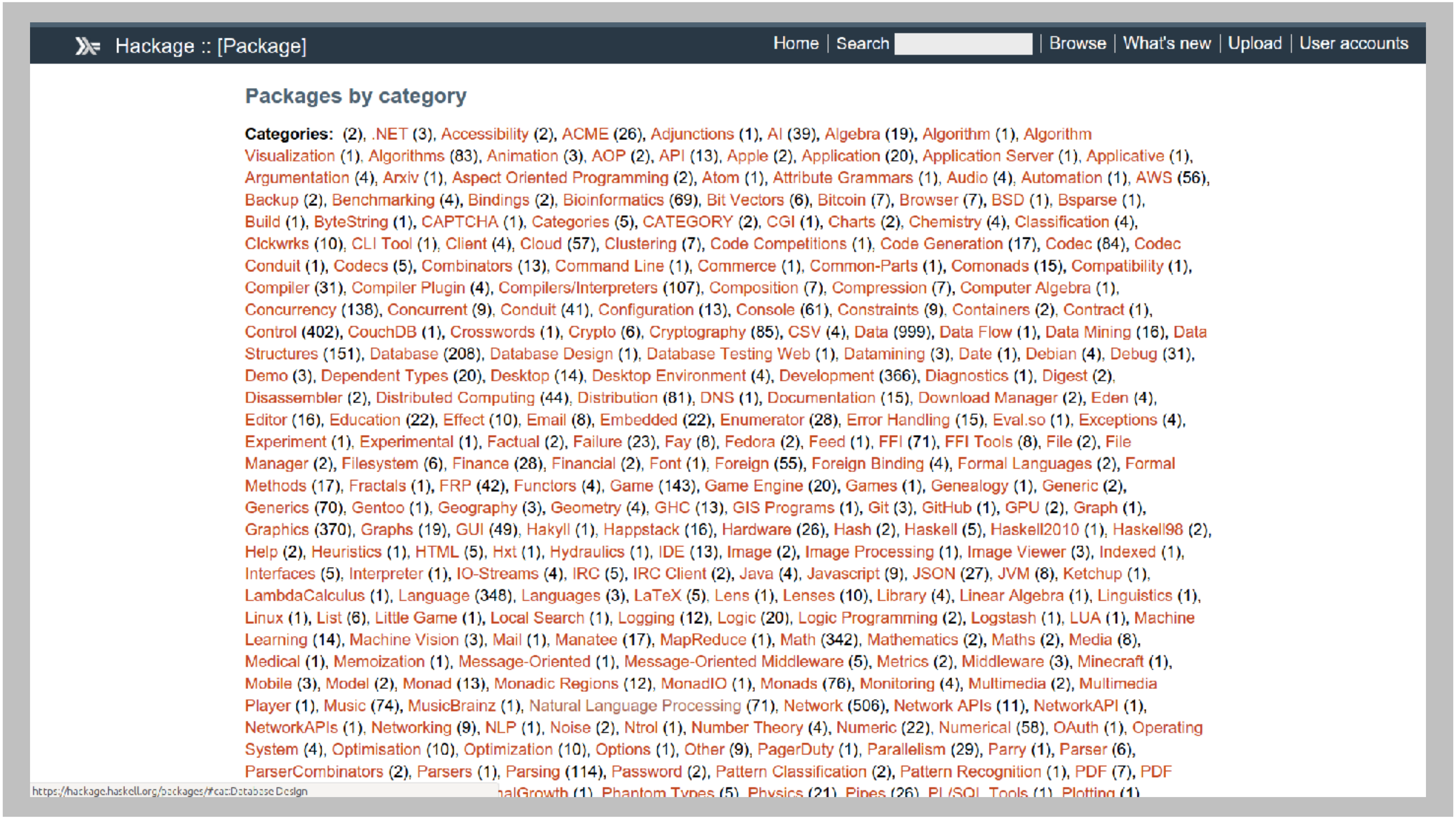

Leksah File Edit Workspace Package Debug Panes View Configuration Help
Leksah hamish > leksah-0.8.0.6
ClassHierarchy.hs
Command.hs
PackageEditor.hs
○ Package ○ Workspa ○ System □ Imports □ Blacklist
Module Package Interface
IDE
Command leksah-0.8.0
Completion leksah-0.8.0
Search
Find: Words Regex map Wrap Replace: TextView Replace All Grep
PackageEditor.hs hamish > leksah-0.8.0.6 INS Ln 631, Col 8

```haskell
                   "Does this package contain a library?"
      ])
      ] ++ extras)

update :: [BuildInfo] -> Int -> (BuildInfo -> BuildInf
update bis index func =
    map (\(bi,ind) -> if ind == index
```

```
map
mapAccum
mapAccumL
mapAccumLM
mapAccumLNat
mapAccumR
mapAccumRWithKey
mapAccumWithKey
mapAdjust'
mapAllBuildInfo
mapAlter'
mapAndRecoverM
mapAndUnzip
mapAndUnzip3
mapAndUnzip3M
mapAndUnzip4M
mapAndUnzipBagM
mapAndUnzipM
mapArray
```

```haskell
-- base-4.2.0.0:Prelude
-- | 'GHC.Base.map' @f xs@ is the list obtained by applyi
--   of @xs@, i.e.,
--  @ map f [x1, x2, ..., xn] == [f x1, f x2, ..., f xn]
--    map f [x1, x2, ...] == [f x1, f x2, ...]
--  @
map :: forall a b. (a -> b) -> [a] -> [b]

-- base-4.2.0.0:GHC.List
-- | 'GHC.Base.map' @f xs@ is the list obtained by applyi
--   of @xs@, i.e.,
--  @ map f [x1, x2, ..., xn] == [f x1, f x2, ..., f xn]
--    map f [x1, x2, ...] == [f x1, f x2, ...]
--  @
map :: forall a b. (a -> b) -> [a] -> [b]

-- base-4.2.0.0:GHC.Base
-- | 'GHC.Base.map' @f xs@ is the list obtained by applyi
--   of @xs@, i.e.,
--  @ map f [x1, x2, ..., xn] == [f x1, f x2, ..., f xn]
--    map f [x1, x2, ...] == [f x1, f x2, ...]
--  @
map :: forall a b. (a -> b) -> [a] -> [b]

-- base-4.2.0.0:Data.List
-- | 'GHC.Base.map' @f xs@ is the list obtained by applyi
--   of @xs@, i.e.,
```

Hackage



Hackage :: [Package]
Home | Search | Browse | What's new | Upload | User accounts

Packages by category

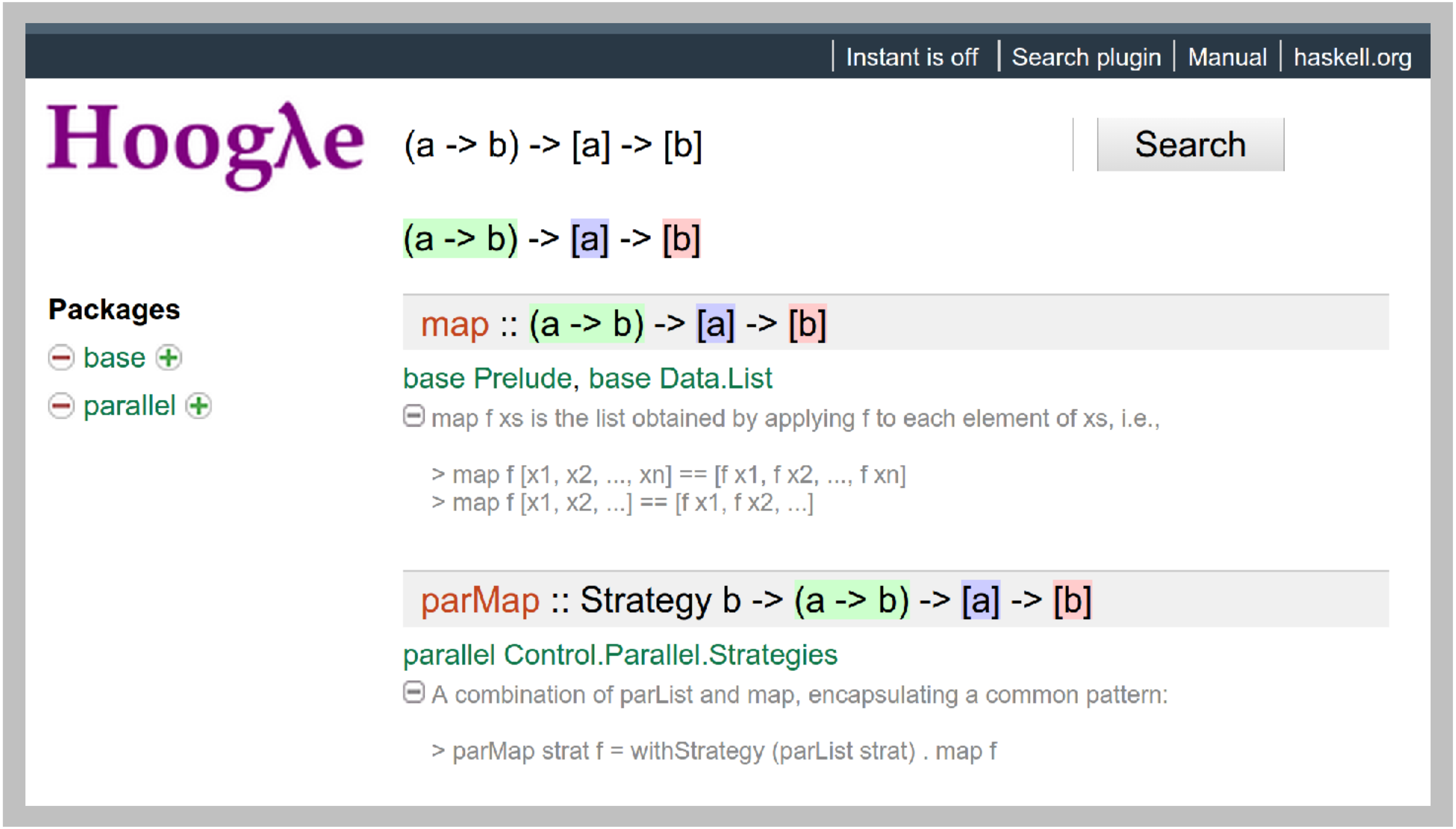
Categories: (2), .NET (3), Accessibility (2), ACME (26), Adjunctions (1), AI (39), Algebra (19), Algorithm (1), Algorithm Visualization (1), Algorithms (83), Animation (3), AOP (2), API (13), Apple (2), Application (20), Application Server (1), Applicative (1), Argumentation (4), Arxiv (1), Aspect Oriented Programming (2), Atom (1), Attribute Grammars (1), Audio (4), Automation (1), AWS (56), Backup (2), Benchmarking (4), Bindings (2), Bioinformatics (69), Bit Vectors (6), Bitcoin (7), Browser (7), BSD (1), Bsparse (1), Build (1), ByteString (1), CAPTCHA (1), Categories (5), CATEGORY (2), CGI (1), Charts (2), Chemistry (4), Classification (4), Clckwrks (10), CLI Tool (1), Client (4), Cloud (57), Clustering (7), Code Competitions (1), Code Generation (17), Codec (84), Codec Conduit (1), Codecs (5), Combinators (13), Command Line (1), Commerce (1), Common-Parts (1), Comonads (15), Compatibility (1), Compiler (31), Compiler Plugin (4), Compilers/Interpreters (107), Composition (7), Compression (7), Computer Algebra (1), Concurrency (138), Concurrent (9), Conduit (41), Configuration (13), Console (61), Constraints (9), Containers (2), Contract (1), Control (402), CouchDB (1), Crosswords (1), Crypto (6), Cryptography (85), CSV (4), Data (999), Data Flow (1), Data Mining (16), Data Structures (151), Database (208), Database Design (1), Database Testing Web (1), Datamining (3), Date (1), Debian (4), Debug (31), Demo (3), Dependent Types (20), Desktop (14), Desktop Environment (4), Development (366), Diagnostics (1), Digest (2), Disassembler (2), Distributed Computing (44), Distribution (81), DNS (1), Documentation (15), Download Manager (2), Eden (4), Editor (16), Education (22), Effect (10), Email (8), Embedded (22), Enumerator (28), Error Handling (15), Eval.so (1), Exceptions (4), Experiment (1), Experimental (1), Factual (2), Failure (23), Fay (8), Fedora (2), Feed (1), FFI (71), FFI Tools (8), File (2), File Manager (2), Filesystem (6), Finance (28), Financial (2), Font (1), Foreign (55), Foreign Binding (4), Formal Languages (2), Formal Methods (17), Fractals (1), FRP (42), Functors (4), Game (143), Game Engine (20), Games (1), Genealogy (1), Generic (2), Generics (70), Gentoo (1), Geography (3), Geometry (4), GHC (13), GIS Programs (1), Git (3), GitHub (1), GPU (2), Graph (1), Graphics (370), Graphs (19), GUI (49), Hakyll (1), Happstack (16), Hardware (26), Hash (2), Haskell (5), Haskell2010 (1), Haskell98 (2), Help (2), Heuristics (1), HTML (5), Hxt (1), Hydraulics (1), IDE (13), Image (2), Image Processing (1), Image Viewer (3), Indexed (1), Interfaces (5), Interpreter (1), IO-Streams (4), IRC (5), IRC Client (2), Java (4), Javascript (9), JSON (27), JVM (8), Ketchup (1), LambdaCalculus (1), Language (348), Languages (3), LaTeX (5), Lens (1), Lenses (10), Library (4), Linear Algebra (1), Linguistics (1), Linux (1), List (6), Little Game (1), Local Search (1), Logging (12), Logic (20), Logic Programming (2), Logstash (1), LUA (1), Machine Learning (14), Machine Vision (3), Mail (1), Manatee (17), MapReduce (1), Math (342), Mathematics (2), Maths (2), Media (8), Medical (1), Memoization (1), Message-Oriented (1), Message-Oriented Middleware (5), Metrics (2), Middleware (3), Minecraft (1), Mobile (3), Model (2), Monad (13), Monadic Regions (12), MonadIO (1), Monads (76), Monitoring (4), Multimedia (2), Multimedia Player (1), Music (74), MusicBrainz (1), Natural Language Processing (71), Network (506), Network APIs (11), NetworkAPI (1), NetworkAPIs (1), Networking (9), NLP (1), Noise (2), Ntrol (1), Number Theory (4), Numeric (22), Numerical (58), OAuth (1), Operating System (4), Optimisation (10), Optimization (10), Options (1), Other (9), PagerDuty (1), Parallelism (29), Parry (1), Parser (6), ParserCombinators (2), Parsers (1), Parsing (114), Password (2), Pattern Classification (2), Pattern Recognition (1), PDF (7), PDF

https://hackage.haskell.org/packages/#cat:Database Design

alGrowth (1), Phantom Types (5), Physics (21), Pipes (26), PL/SQL Tools (1), Plotting (1),

https://hackage.haskell.org/packages/

Hoogle



(a -> b) -> [a] -> [b]

(a -> b) -> [a] -> [b]

Packages

- base
- parallel

map :: (a -> b) -> [a] -> [b]

base Prelude, base Data.List

map f xs is the list obtained by applying f to each element of xs, i.e.,

```
> map f [x1, x2, ..., xn] == [f x1, f x2, ..., f xn]
> map f [x1, x2, ...] == [f x1, f x2, ...]
```

parMap :: Strategy b -> (a -> b) -> [a] -> [b]

parallel Control.Parallel.Strategies

A combination of parList and map, encapsulating a common pattern:

```
> parMap strat f = withStrategy (parList strat) . map f
```

Recherche par nom de fonction ou signature – https://www.haskell.org/hoogle/

QuickCheck

- Soit une fonction mSize qui prend :
  - Une largeur de caractère
  - Une hauteur de caractère
- Et retourne une chaîne Minitel

```haskell
mSize :: CharWidth -> CharHeight -> Mstring
mSize SimpleWidth SimpleHeight = [esc, 0x4c + 0]
mSize SimpleWidth DoubleHeight = [esc, 0x4c + 1]
mSize DoubleWidth SimpleHeight = [esc, 0x4c + 2]
mSize DoubleWidth DoubleHeight = [esc, 0x4c + 3]
```

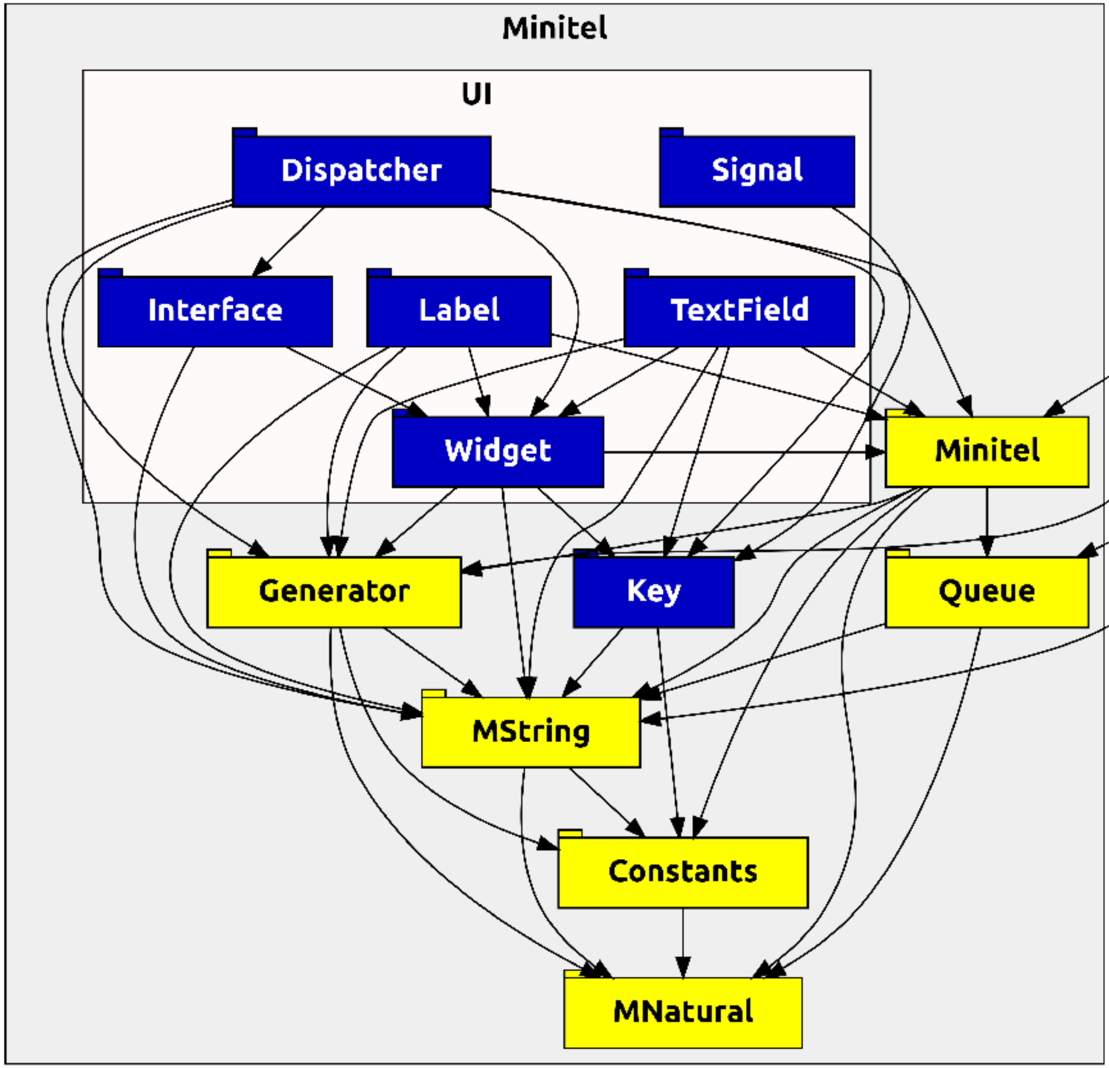
QuickCheck

- Pour la tester, QuickCheck permet d'écrire :

```haskell
instance Arbitrary CharWidth where
    arbitrary = elements [SimpleWidth, DoubleWidth]

instance Arbitrary CharHeight where
    arbitrary = elements [SimpleHeight, DoubleHeight]

prop_msize l h = length (mSize l h) == 2
```

SourceGraph

- Analyse statique du code
- Complexité cyclomatique
- Conseils
- Etc.



Et le code ?

Oui, et le code ?
Ça ressemble à quoi Haskell ?

“Learning a new language is easy
Learning a new paradigm is hard”
Functional Thinking

Neal Ford – Jax Conference – 07.2012
https://www.youtube.com/watch?v=JeK979aqqqc

Fonctionnel ?

- Évaluation de fonctions mathématiques
  - Une fonction retourne toujours le même résultat pour des arguments donnés
  - Une fonction n’a pas d’effet de bord
- Rejet du changement d'état
- Rejet de la mutation des données

Haskell n'a pas

- Variable
- Boucle `for`, `while`, `until`, `repeat`…
- Objet

Marteau : Evan Amos

Un marteau à la main, les problèmes deviennent des clous

Evaluation paresseuse

Liste infinie

```
myList = [1..]
print (head myList)
```

Spark

process

thread

spark

Poids: Coyau@WikimediaCommons CC-BY-SA 3.0

Plume: Pybalo CC-BY-SA 3.0

Spark

- Création simple
- Programmation parallèle

```
sendThread <- forkIO sendLoop
recvThread <- forkIO recvLoop
```

Inférence de type

```
isL c = c == 'l'
```

```
isL :: Char -> Bool
```

Filtrage par motif

```haskell
complete :: MString -> Bool
complete []           = False
complete [27]         = False
complete [27, 91]     = False
complete [27, 91, 50] = False
complete [27, 91, 52] = False
complete _            = True
```

Compréhension de liste

```haskell
evens = [x | x <- [1..], even x]
print (take 5 evens)
```

```
[2,4,6,8,10]
```

Opérateurs

- Haskell autorise la création d'opérateurs
- Un opérateur est une fonction
- Une fonction binaire peut être utilisée comme un opérateur

`>>=`   `$`   `<<`   `.`

`<|>`   `.&.`   `/!\`

Récursivité

- Support poussé de la récursivité

```
recursive 10000000 = 1
recursive i = 1 + recursive (i + 1)
print (recursive 1)
```

Classes de type

- Similaires aux interfaces de Java

```
class Sendable a where
    (<<<) :: Minitel -> a -> IO ()

instance Sendable MString where
    (<<<) minitel = putM (output minitel)

instance Sendable [MString] where
    (<<<) minitel = mapM_ (minitel <<<)
```

“If the code compiles, it works! (almost all the time)”

http://www.haskell.org/haskellwiki/Why_Haskell_just_works

“When [the] Blub programmer looks [...] up
the power continuum,
he doesn’t realize he’s looking up.

What he sees are merely weird languages.
He probably considers them about equivalent
in power to Blub [...].

Blub is good enough for him,
because he thinks in Blub.”

Beating the averages
Paul Graham – 04.2003

http://www.paulgraham.com/avg.html

Try Haskell

Christopher Done – 12.2013
http://tryhaskell.org/

Learn you a Haskell for great good!

Miran Lipovača – 04.2011
http://learnyouahaskell.com/

Real world Haskell

Bryan O’Sullivan, Don Stewart, John Goerzen – 2007-2008
http://book.realworldhaskell.org/read/

What I wish I knew when learning Haskell

Stephen Diehl – 08.2014

http://dev.stephendiehl.com/hask/

Functional principles for object-oriented development

Jessica Kerr – GOTO Conferences – 05.2014

https://www.youtube.com/watch?v=GpXsQ-NIKXY

“Every successful technology eventually becomes dominated by mechanisms for solving problems that only it creates. […]

When experts from two competing technologies look at each other, they see the complexity inherent in the other’s tools.

Thus, we get tribes arguing past each other, e.g. Functional vs OOP programmers, each convinced the other’s tools is needlessly complex.”

Reginald Braithwaite – Twitter – 21.11.2014

https://twitter.com/raganwald/status/535852629264326656