Haskell Cheat Sheet

BASICS

Comments

All comments start with two hyphens

```
-- This is a comment in the code
```

Data Types

Haskell uses various data types, all of them starts by a capital letter:

- `Int`: Integer number with fixed precision
- `Integer`: Integer number with virtually no limits
- `Float`: Floating number
- `Bool`: Boolean. Takes two values: `True` or `False`.
- `Char`: Character. Any character in the code is placed between quotes (`'`).
- `String`: Strings (In fact, a list of Chars).

Conditionals

Identity: `==`, non identity `/=`

Comparatives (type must be a subclass of Ord) :

`>`, `>=`, `<`, `<=`

```
if conditional then truePart else falsePart
```

Scripts types

Classic (.hs) *the code is more important*

Litterals (.lhs) *lines with code starts with >, lines with comments **don't** start by two hyphens.*

Hugs basics

Load file `:load filename`

reload file `:reload`

Launch file editor with current file `:edit`

Get information about a function `:info command`

FUNCTIONS

Declare a new function

Start with explicit type declaration (optional)

```
functionName::inputType1[->inputTypeN]->outputType
```

Declare function with pattern matching (sample)

```
intToChar 1 = "One"
intToChar 2 = "Two"
```

Declare function with guards

```
intToChar x
        | x==1 = "One"
        | x==2 = "Two"
```

Type redefinition

```
Type NewTypeName = TypeValue
```

Sample : `Type String = [Char]`

LISTS

Tuples

Tuples are designed to group data (multiple types allowed).

`(El1,El2,[Elx…])`

Sample: `("James",41,1.85)`

Basic list creation

Lists are between [], elements are separated by comma.

Sample : `[1,2,3,4,5]` `["John","Paul","Andy"]`

You can create lists by populate them with a range:

`[1..5]=[1,2,3,4,5]` `[1..]=[1,2,3,4,5,6,…` *(infinite)*

Comprehension lists

= creating liste using arithmetic operations or functions. `[ body | generator ].`

Samples :

```
[2*a | a <- [1..3]] = [2,4,6]
[x*y | x <- [1..3], y <- [3..6] ]
[x | x <- [1,5,12,3,23,11,7,2], x>10]
[(x,y) | x <- [1,3,5], y <- [2,4,6], x<y]
```

Using lists

An empty list is designed by `[]`

In (h:q), h stands for the first element of the list, and q for the rest. With `(f:s:t:q)`, you can directly gets the first (`f`), second (`s`) and third (`t`) element of the list.

You can add the element `e` to the list `l` with `e:l`

Predefined operations

Append two lists	`list1++list2`
Return element n	`list!!n`
Get the first/last element	`head/last list`
Get the sum of all list elements	`sum list`
Get the product of all list elements	`product list`

USE HASKELL

Interpreter

Hugs : Available for Windows, Linux, FreeBSD and MacOs X. http://www.haskell.org/hugs/

Compiler

GHC (*Glasgow Haskell Compiler*) : Available for Windows, Linux.

http://www.haskell.org/ghc/download.html

Editors

- Any good text editor :)
- Visual Studio Haskell.

http://www.haskell.org/visualhaskell

Documentation

- Haskell API search : http://www.haskell.org/hoogle/
- Haskell reference : ftp://ftp-developpez.com/c-maneu/langages/haskell/haskell-reference.zip