Fu Bar

Marvin Cohrs

November 6, 2011

Contents

1 Skills

1.1 Operators + Functions

Fu Bar knows several operators and functions:

- Addition: $x + y$
  Adds two floating-point numbers.
  e.g. $4.1 + 5.8 = 9.9$
- Subtraction: $x - y$
  Subtracts a floating-point number from another one.
  e.g. $9.9 - 5.8 = 4.1$

- Multiplication: $x * y$
  Multiplies two floating-point numbers.
  e.g. $1.1 * 9.0 = 1.1 \times 9.0 = 9.9$

- Division: x/y
  Divides a floating-point number by another one.
  e.g. $9.9/1.1 = \frac{9.9}{1.1} = 9.0$

- Integer division: $n\backslash m$
  Divides an integer by another one and returns the integer quotient.
  e.g. $20\backslash 6 = 3$
  because $6 \times 3 = 18$ and $6 \times 4 = 21$

- Modulo: $n\#m$
  Divides an integer by another one and returns the modulo.
  e.g. $20\#6 = 2$
  because $6 \times 3 + 2 = 20$

- Exponentiation: $x\hat{}n$
  Exponentiates a floating-point base by an integer exponent.
  e.g. $2\hat{}8 = 2^8 = 256$

- Paranthenses: $(expression)$
  An expression in paranthenses has got a higher priority.
  e.g. $2 * (4 + 5) = 2 * 9 = 18$
  but $2 * 4 + 5 = 8 + 5 = 13$

- Absolute Value: $|expression|$, $abs(expression)$
  Gets the absolute value of an expression.
  e.g. $|+3| = 3$
  and $|-3| = 3$, too!

- Negation: $-x$, $neg(x)$
  Negates a floating-point number.
  e.g. $-(3 * 2) = -6$

- Reciprocal: $1/x$, $rec(x)$
  Divides $\frac{1}{x}$, resulting the reciprocal.
  e.g. $rec(4) = \frac{1}{4} = 0.25$

- Factorial: $n!$, $fac(n)$
  Computes the factorial of an integer.
  e.g. $4! = 24$

- Binomial Coefficient: k from n, $n\mathrm{C}k$
  Computes the binomial coefficient for $\binom{n}{k}$
  e.g. 2 from $3 = 3\mathrm{C}2 = \binom{3}{2} = 3$

- Square root: $sqrt(x)$, $SquareRoot(x)$
  Computes the square root of a floating-point number.
  e.g. $sqrt(9) = \sqrt{9} = 3$

- Cube root: $cbrt(x)$, $CubeRoot(x)$
  Computes the cube root of a floating-point number.
  e.g. $cbrt(27) = \sqrt[3]{27} = 3$

- nth root: $nrt(x;n)$, $Root(x;n)$
  Computes the nth root of a floating-point number.
  e.g. $nrt(4096;6) = \sqrt[6]{4096} = 4$

- Integer reduction: $int(x)$, $x\backslash 1$
  Reduces a floating-point number to an integer:
  e.g. $int(tau) = 6$

- Cut: $frac(x)$
  Cut's the number at the floating-point, resulting fraction part.
  e.g. $frac(23.45) = 0.45$

- Binary conjunction: n and m
  Conjuncts two integers.
  e.g. 7 and $11 = 3$

- Binary disjunction: n or m
  Disjuncts two integers.
  e.g. 7 or $11 = 15$

- Binary contravalence: n xor m
  Contravalences two integers.
  e.g. 7 or $11 = 12$

- Greatest common divisor: $gcd(n;m)$
  Computes the greatest common divisor of two integers.
  e.g. $gcd(21;28) = 7$

- Least common multiple: $lcm(n;m)$
  Computes the least common multiple of two integers.
  e.g. $lcm(3;4) = 12$

- Sinus / Cosinus: $sin(x)$, $cos(x)$
  Computes the sinus / cosinus of an angle.
  Don't forget to specify the input unit (otherwise I'll use radians).
  The output unit is always radian.
  e.g. $sin(90\,\text{deg}) = \sin\frac{1}{4}\tau = 1$

- Conversions:
  - $rad(x\ \text{deg})$: Degree ⇒ Radians
  - $rad(x\ \text{grad})$: Gradian ⇒ Radians
  - $rad(x\ \text{turn})$: τ-Radians ⇒ Radians
  - $deg(x\ [\text{rad}])$: Radians ⇒ Degree
  - $deg(x\ \text{grad})$: Gradian ⇒ Degree
  - $deg(x\ \text{turn})$: τ-Radians ⇒ Degree

  - $grad(x$ [rad]): Radians ⇒ Gradian
  - $grad(x$ deg): Degree ⇒ Gradian
  - $grad(x$ turn): τ-Radians ⇒ Gradian

- Iterations: $iterate(firstx : expression)$
  Runs the given iteration, resulting the fix point.
  e.g. $iterate(4 : (xn + (9/xn))/2) = iterate(4 : \frac{x_n + \frac{9}{x_n}}{2}) = \sqrt{9} = 3$
  expresses $x_{n+1} = \frac{x_n + \frac{9}{x_n}}{2}$

- Short sum: $sum(var : start < end$ [\$ $step]; expression)$
  Sums the expression several times, iterating var from $start$ to end.
  e.g. $sum(i : 1 < 4; i) = \sum_{i=1}^{4} i = 1 + 2 + 3 + 4 = 10$

- Short product: $product(var : start < end$ [\$ $step]; expression)$
  Multiplies the expression several times, iterating var from $start$ to end.
  e.g. $product(i : 1 < 5; i) = \prod_{i=1}^{5} i = 5! = 1 \times 2 \times 3 \times 4 \times 5 = 120$

- Zero: $zero(var : expr)$
  Finds the zero of a function.
  e.g. $zero(a : 5 * a - 55) = 11$
  because $5 \times 11 - 55 = 0$

- Solve: $solve(var : expr1 = expr2)$
  Solves the equation.
  e.g. $solve(a : a\hat{}2 = 49) = 7$
  because $7^2 = 49$

1.2 Constants

Fu Bar knows these constants:

- tau: τ, a full radian turn, about 6.28
- pi: π, $\frac{\tau}{2}$, a half radian turn, about 3.14
- phi: Φ, the golden ratio, about 1.62
- full: A whole, 1
- half: $\frac{1}{2} = 0.5$
- quarter: $\frac{1}{4} = 0.25$

1.3 Units

- deg: Degrees, full turn = 360
- grad: Gradians, full turn = 400
- rad: Radians, full turn = τ
- turn, tau: τ-Radians, full turn = 1
- pi: π-Radians, full turn = 2

1.4 Comparison

You can simply compare two expressions using an equation operator:
$expression_1 = expression_2$

Fu Bar will tell you the result, and whether the terms are equal or not.
If not, it will tell you, which term is the greater one, and output the difference and the factor.

1.5 Custom variables

You can simply save results into your own variables:
$[\,name\,]\ expression$

Fu Bar will compute the term and save the result to a variable of your choice.
To mark it as a constant, insert an exclamation mark before the name:
$[\,!name\,]\ expression$

Note: Fu Bar won't overwrite constants!

1.6 Custom functions

Of course, you can define custom functions, too:
$[\,name(\,var\,)\,]\ expression$

To mark it as read-only, insert an exclamation mark before the name:
$[\,!name(\,var\,)\,]\ expression$

To ensure, that the argument is not zero, add 'nonzero' before the var name:
$[\,name(\,\text{nonzero}\,var\,)\,]\ expression$

To assign a separate value to $name(0)$, enter:
$[\,name(0)\,]\ expression$

For example:

$[\, f(x) \,]\ 25x$ $\Rightarrow 0$
$f(4)$ $\Rightarrow 100$

$[\, f(\text{nonzero}\, x) \,]\ 1/x$ $\Rightarrow 1$
$f(4)$ $\Rightarrow 0.25$
$f(0)$ $\Rightarrow$ Function not defined for $x = 0$

$[\, f(0) \,]\ 0$ $\Rightarrow 0$
$f(4)$ $\Rightarrow 0.25$
$f(0)$ $\Rightarrow 0$

To ensure, that the argument is under a specific set of numbers, there are the following directives:

complex The full complex set.

real As a computer cannot calculate with irrational numbers, *real* equals *rational*.

rational The imaginary part is dropped, a rational number remains.

integer The fraction is truncated, an integer remains.

natural The sign is dropped, a natural number remains.

Each of these directives may be combined with *nonzero*.
If no set is specified, Fu Bar uses *complex*.

1.7 Explanations

Fu Bar can tell you the steps, how to calculate a term.
: *explain* $32 * (1 + 5)$ will result:
$1 + 5 = 6$
$32 * 6 = 192$

Come on, try it!

2 Plugins

Fu Bar provides a plugin interface. Plugins are invoked by one of these syntaxes:

```
:invoke plugin-name param-list
/plugin-name param-list
```

2.1 Writing own plugins

Fu Bar wants its plugins to have the following exported procedures:

void __cdecl fubar_plugin_transmit(char *params) Receives the arguments. Do not implement the main skills here! Always remember, that this procedure may be called multiple times before invocation.

void __cdecl fubar_plugin_invoke() Implement all the plugin's functionality here.

A short example in Pascal:

```
library example;

var params: string;

procedure transmit(const s: PChar); cdecl;
begin
  params := params + ' ' + string(s);
end;

procedure invoke; cdecl;
begin
  Writeln('Hello World!');
  Writeln(params);
end;

exports
  transmit name 'fubar_plugin_transmit',
  invoke name 'fubar_plugin_invoke';

initialization
  params := '';
end.
```

Compile it, and you'll receive a file named *libexample.so* (Unix) or *example.dll* (Windows). Now add your plugin to *libplugs.dat*:

```
example: /etc/fubar/libexample.so
```

If possible, the plugin should be in the global configuration path of Fu Bar. On Linux, it's /etc/fubar/ in the most cases. Now don't forget to reinstall Fu Bar with the new *libplugs.dat* (`sudo make install`) and erase your local copy (usually in ∼/.config/fubar/) using `make erasecfg`.

2.2 tcal - the integrated task-based calendar

tcal is a task-based calendar, shipped as a plugin for fubar. To queue a task, use

```
/tcal queue task param-list
/queue task param-list
/task param-list
```

For example, the *dayofweek* task can be queued by

```
/tcal queue dayofweek 2011-11-11
/queue dayofweek 2011-11-11
/dayofweek 2011-11-11
```

To run all queued tasks, enter

```
/tcal apply
/apply
```

3 Get

3.1 Using Git

Simply clone the repository:

```
git clone git://github.com/implementor/Fu-Bar.git fubar
```

Later, you can easily check for updates:

```
cd fubar
git pull origin master
```

3.2 Packed and compressed

Get it here: `https://github.com/implementor/Fu-Bar/zipball/master`

4 Build & Install

On Linux, you can easily build Fu Bar using GNU `make`. Other Unixes may also work, but aren't tested. The packages `make` and `fpc` are required.

make build Build Fu Bar.

make clean Clean up the directory.

make erasecfg Erase your local configuration.

sudo make install Installs Fu Bar to $(INST_DEST)

sudo make uninstall Uninstall Fu bar from $(INST_DEST)

make run Run Fu Bar from the scratch.

If `wine` is available, and you have installed `FPC for Win32` using it, the following targets are also usable:

make build4win Build Fu Bar for Win32

make buildX Build Fu Bar for Win32 AND your current platform.

make run4win Run the Win32 executable

5 Run

After installation, you should be able to call `fubar` from the command-line.
`fubar --help` reveals this:

```
Fu Bar, a term-parsing calculator.
Copyright (c) 2011 Marvin Cohrs
Distributed under the terms of the GNU General Public License.

Usage: fubar [-i|--instant] [-q|--quiet] [--noman] [[-t|--term] TERM]
       fubar --safe [[-t|--term] TERM]
       fubar --about TOPIC
       fubar --help

  -i --instant      Do not load my ~/.config/fubar/autoload
  -q --quiet        Be quiet.
  -t --term TERM    Calculate the given term.
     --noman        Disable built-in manual.
     --safe         Safe mode, does not read or modify any file.
     --help         This help screen.
     --about TOPIC  Displays help about a specific topic.

--term implies --quiet
--safe implies --quiet, --instant, --noman
--help, --about deny --safe, --noman
```

Fu Bar also provides a manpage. Try `man fubar` ;)