# gfn

Generate fantasy names for games and stories. It uses the fine fantasyname module by s0rg, which implements the code created by the rinkworks fantasy name generator. The code itself is outlined here, or take a quick look at the reference guide.

In case the site vanishes some day, a copy of those documents is contained here in the repository.

## Installation

The tool does not have any dependencies. Just download the binary for your platform from the releases page and you're good to go.

### Installation using a pre-compiled binary

Go to the latest release page and look for your OS and platform. There are two options to install the binary:

Directly download the binary for your platform, e.g. `gfn-linux-amd64-0.0.2`, rename it to `kleingebaeck` (or whatever you like more!) and put it into your bin dir (e.g. `$HOME/bin` or as root to `/usr/local/bin`).

Be sure to verify the signature of the binary file. For this also download the matching `gfn-linux-amd64-0.0.2.sha256` file and:

```
cat gfn-linux-amd64-0.0.2.sha25 && sha256sum gfn-linux-amd64-0.0.2
```

You should see the same SHA256 hash.

You may also download a binary tarball for your platform, e.g. `gfn-linux-amd64-0.0.2.tar.gz`, unpack and install it. GNU Make is required for this:

```
tar xvfz gfn-linux-amd64-0.0.2.tar.gz
cd gfn-linux-amd64-0.0.2
sudo make install
```

### Installation from source

You will need the Golang toolchain in order to build from source. GNU Make will also help but is not strictly neccessary.

If you want to compile the tool yourself, use `git clone` to clone the repository. Then execute `go mod tidy` to install all dependencies. Then just enter `go build` or - if you have GNU Make installed - `make`.

To install after building either copy the binary or execute `sudo make install`.

## Usage

There are a bunch of pre compiled fantasy name codes builtin, you can get a list with:

```
% gfn -l
```

To use one of them and limit the number of words generated:

```
% gfn JapaneseNamesDiverse -c 24
yumufuchi  afuchin    keyu       amorekin   ekimuwo    ashihewani rosa       chireki
oterun     ruwahi     uwamine    emiyumu    temimon    yuwa       awayason   fuki
emiwa      nushiron   achihora   yomichi    saniyutan  kewaritsu  saroru     uhashi
```

You can also write a code yourself:

```
gfn '!sVm' -c 24
Quaoobunker    Emeemoopsie    Angeepookie    Osousmoosh     Umuisweetie    Ustoesnookum   Su
Skelaesnoogle  Echiapookie    Cereepoochie   Gariwuddle     Echaewookie    Tiaieschmoopie Qu
Undousnuggy    Urnuigooble    Mosoesnuggy    Eldoegooble    Denoipoochie   Mosoosmooch    Sh
```

A short outline of the code will be printed if you add the `-h` parameter:

```
This is gfn, a fantasy name generator cli.

Usage: gfn [-vld] [-c <config file>] [-n <number of names>] <name|code>

Options:
-c --config   Config file to use (optional)
-n --number   Number of names to generate
-l --list     List pre-compiled shortcuts
-d --debug    Show debugging output
-v --version  Show program version

pattern  syntax The  letters s,  v, V,  c, B,  C, i,  m, M,  D, and  d
represent different types of random replacements:

s - generic syllable
v - vowel
V - vowel or vowel combination
c - consonant
B - consonant or consonant combination suitable for beginning a word
C - consonant or consonant combination suitable anywhere in a word
i - insult
m - mushy name
M - mushy name ending
D - consonant suited for a stupid person's name
d - syllable suited for a stupid person's name (begins with a vowel)
Everything else is emitted literally.
```

All characters between parenthesis () are emitted literally. For example, the pattern s(dim), emits a random generic syllable followed by dim.

Characters between angle brackets <> emit patterns from the table above. Imagine the entire pattern is wrapped in one of these.

In both types of groupings, a vertical bar | denotes a random choice. Empty groups are allowed. For example, (foo|bar) emits either foo or bar. The pattern <c|v|> emits a constant, vowel, or nothing at all.

An exclamation point ! means to capitalize the component that follows it. For example, !(foo) will emit Foo and v!s will emit a lowercase vowel followed by a capitalized syllable, like eRod.

You can use a config file to store your own codes, once you found one you like.
A configfile is searched in these locations in this order:

- /etc/gfn.conf
- /usr/local/etc/gfn.conf
- $HOME/.config/gfn/config
- $HOME/.gfn

You may also specify a config file on the commandline using the -c flag.

You can add multiple codes, here's an example:

```
# example config file
[[Templates]]
morph = "!s(na|ha|ma|va)v"
morphium = "!s(na|ha|ma|va)v(ius|ium|aum|oum|eum)"
```

Config files are expected to be in the TOML format.

## Report bugs

Please open an issue. Thanks!

## License



## Author