NAME

i3blocks – A flexible scheduler for your i3bar blocks

SYNOPSIS

i3blocks [*options*]

DESCRIPTION

i3blocks allows one to easily describe blocks in a simple format, and generate a status line for i3bar(1). It handles clicks, signals and time interval for user scripts.

OPTIONS

–c Specifies an alternate configuration file path. By default, i3blocks looks for configuration files in the following order (note that /etc may be prefixed with /usr/local depending on the compilation flags):

1. ~/.config/i3blocks/config (or $XDG_CONFIG_HOME/i3blocks/config if
2. ~/.i3blocks.conf
3. /etc/xdg/i3blocks/config (or $XDG_CONFIG_DIRS/i3blocks/config if s
4. /etc/i3blocks.conf

–v Log level. This option is cumulative. By default, error messages are displayed on stderr. Passed once, a failure during an update is shown within the block. Passed twice enables the debug messages on stderr.

–V Print the version and exit.

–h Print the help message and exit.

CONFIGURATION

The configuration file is an ini file. Each section describes a new block. A line beginning with a # sign is a comment, and empty lines are ignored. A property is a `key=value` pair per line, with no space around the equal sign. Properties declared outside a block (i.e. at the beginning of the file) describe global settings.

Here is an example config file:

```
# This is a comment
interval=5
color=#00FF00

[weather]
command=~/bin/weather.pl
interval=1800

[time]
command=date +%T
```

To use i3blocks as your status line, define it in a *bar* block of your `~/i3/config` file:

```
bar {
  status_command i3blocks
}
```

BLOCK

The properties used to describe a block are the keys specified in the i3bar protocol http://i3wm.org/docs/i3bar−protocol.html , plus additional properties used by **i3blocks** to describe when and how to update a block. All the supported properties are described below.

The following keys are standard, see http://i3wm.org/docs/i3bar−protocol.html for details.

- `full_text`
- `short_text`

- `color`
- `min_width`
- `align`
- `name`
- `instance`
- `urgent`
- `separator`
- `separator_block_width`
- `markup`

The following keys are specific to **i3blocks**.

`command`
: The command executed by a shell, used to update the block. The expected behavior is described below, in the **COMMAND** section.

`interval`
: If it is a positive integer, then the block is spawned on startup and the value is used as a time interval in seconds to schedule future updates. If unspecified or 0, the block won't be executed on startup (which is useful to simulate buttons).

  If "*once*" (or −1), the block will be executed only on startup (note that a click or signal will still trigger an update).

  If "*repeat*" (or −2), the block will be spawned on startup, and as soon as it terminates (useful to repeat blocking commands). Use with caution!

  If "*persist*" (or −3), the block will be executed only on startup, and updated as soon as it outputs a line. Thus limited to single line updates.

`signal`
: The signal number used to update the block. All the real−time (think prioritized and queueable) signals are available to the user. The number is valid between 1 and N, where SIGRTMIN+N = SIGRTMAX. (Note: there are 31 real−time signals in Linux.) For instance, `signal=10` means that this block will be updated when **i3blocks** receives SIGRTMIN+10.

`label` An optional label to preprend to the `full_text` after an update.

`format`
: This property specifies the format of the output text. The default format is plain text, as described in the **COMMAND** section. If "json" (or 1) is used, the block output is parsed as JSON.

COMMAND

The value of the `command` key will be passed and executed as is by a shell.

The standard output of the command line is used to update the block content. Each non−empty line of the output will overwrite the corresponding property:

1. full_text
2. short_text
3. color

For example, this script sets the `full_text` in blue but no `short_text`:

```
echo "Here's my label"
echo
echo \#0000FF
```

If the command line returns 0 or 33, the block is updated. Otherwise, it is considered a failure and the first

line (if any) is still displayed. Note that stderr is ignored. A return code of 33 will set the `urgent` flag to true.

For example, this script prints the battery percentage and sets the urgent flag if it is below 10%:

```
BAT=`acpi -b | grep -E -o '[0-9][0-9]?%'`
echo "BAT: $BAT"
test ${BAT%?} -le 10 && exit 33 || exit 0
```

When forking a block command, **i3blocks** will set the environment with some `BLOCK_*` variables. The following variables are always provided, with eventually an empty string as the value.

`BLOCK_NAME`
: The name of the block (usually the section name).

`BLOCK_INSTANCE`
: An optional argument to the script.

`BLOCK_BUTTON`
: Mouse button (1, 2 or 3) if the block was clicked.

`BLOCK_X` **and** `BLOCK_Y`
: Coordinates where the click occurred, if the block was clicked.

Here is an example using the environment:

```
[block]
command=echo name=$BLOCK_NAME instance=$BLOCK_INSTANCE
interval=1

[clickme]
full_text=Click me!
command=echo button=$BLOCK_BUTTON x=$BLOCK_X y=$BLOCK_Y
min_width=button=1 x=1366 y=768
align=left
```

Note that **i3blocks** provides a set of optional scripts for convenience, such as network status, battery check, cpu load, volume, etc.

EXAMPLES

As an example, here is a close configuration to i3status(1) default settings:

TODO

```
interval=5
signal=10

[ipv6]

[free]

[dhcp]

[vpn]

[wifi]

[ethernet]
min_width=E: 255.255.255.255 (1000 Mbit/s)

[battery]
```

```
[cpu]

[datetime]
```

The following block shows the usage of `signal` with some i3(1) bindings which adjust the volume, before issuing a `pkill -RTMIN+1 i3blocks`:

```
[volume]
command=echo -n 'Volume: '; amixer get Master | grep -E -o '[0-9][0-9]?%'
interval=once
signal=1
# no interval, only check on SIGRTMIN+1
```

Here is an example of a very minimalist config, assuming you have a bunch of scripts under `~/bin/blocks/` with the same name as the blocks:

```
command=~/bin/blocks/$BLOCK_NAME
interval=1

[free]
[wifi]
[ethernet]
[battery]
[cpu]
[datetime]
```

SEE ALSO

The development of i3blocks takes place on Github (https://github.com/vivien/i3blocks).

The wiki (https://github.com/vivien/i3blocks/wiki) is a good source of examples for blocks and screenshots.

`i3` (1), `i3bar` (1), `i3status` (1)

Reporting Bugs

Please report bugs on the issue tracker (https://github.com/vivien/i3blocks/issues).

Known Bugs

None.

AUTHOR

Written by Vivien Didelot <vivien.didelot@gmail.com>.

COPYRIGHT