

NAME

qam — weather information for the terminal, hence morse request ‘QAM’

SYNOPSIS

qam [**-l** *Cityname*] [**-f** | **-F**] [**-e**] [**-a**] [**-u**] [**-p**] [**-s** *speed*] [**-d** *delay*]

qam **-h** | **--help**

qam **-v** | **--version**

DESCRIPTION**Overview**

qam implements a simple weather information widget for the terminal. It derives its name from the classic “QAM” morse request of the Q-codes to receive a weather report via radio from another station. In case of a storm, the program will issue visible warnings. Output will be written to *stdout*(4) in ANSI-sequenced rich text by default. Execution can be terminated any time by entering **^C**.

Details

Firstly, if no location is stated by **-l** with a *Cityname* argument, it reads from */dev/urandom* and forms random numbers to choose the location to be printed out for the run. By default, **qam** retrieves an actual weather report for the given location. A detailed forecast in three-hour-steps for the next five days can be obtained by stating **-f** and a daily one for the upcoming seven days by **-F**. Playback speed of the output can be changed by **-s** with its corresponding *speed* argument. **-d** with its corresponding *delay* argument enables a screensaver mode, which prints out the contents by default with a *speed* of 600 Baud and a loop return *delay* of 5 seconds. Please note, that both values must not be set to 0 simultaneously, because this would end up in an infinite loop, which **qam** will detect and abort operation with a proper error message.

After evaluation of the command line, **qam** secondly retrieves weather information from a predefined online resource as a **JSON** and extract the variables from the stream by a sequence of filtering by *sed*(1). Some additional values will be calculated locally from the processed data.

The data will be formatted in rich text by ANSI sequences, unless deactivated by **-a** and printed out one character after another in the desired *speed* as stated with **-s**. If **qam** would be run as a screensaver with **-d**, the screen will be cleared after a certain amount of time using *sleep*(1) and *clear*(1) to start over after finishing the loop.

Options

The options – besides arguments – are as follows:

-h | **--help**

Print out a short help to *stdout*(4). No other option is viable with **-h** / **--help**.

In case of an error, this short help will be printed out to *stderr*(4) with a brief description of the nature of the error.

-v | **--version**

Print out information about version, date, author, copyright and contact data to *stdout*(4). No other option is viable with **-v** / **--version**.

The following options are discretionary:

-l *Cityname*

Location for actual or forecast weather to retrieve stated by the *Cityname* argument. *Cityname* can be any single location, usual a city’s name, like *Jerusalem* or specified in more detail by adding a country marker with a leading comma like *Jerusalem, IL*.

-f Retrieve weather forecast for a given location on a three-hour-basis for the next five days.

Or alternatively:

-F Retrieve weather forecast for a given location for the next upcoming days.

-e Export (forecast) weather data to files into a default export directory, usual *qamdata*. This directory will be created into the parent working directory if not existant. The filenames within that directory represent a mnemonic set of the JSON arrays of the source material. They contain the space separated values of each array element. Existing data will be overwritten without warning. If creation of the directory or writing to the files is impossible, **qam** will issue a warning to *stderr*(4), but will continue operation.

- a Disable ANSI sequences. This option reduces output to pure *ascii*(7) character set without any escape sequences, hence reducing the rich text into plain text.
- u Disable solar information, including UV index, as well as air pollution and radiation information. By default, **qam** prints out information about sunrise and sunset including an UV index about the strength of the sun, as well as air pollution and radiation information for the given location.
- p Expand air pollution and radiation information; default: abbreviate. By default, **qam** prints out an abbreviated form of the air pollution and radiation information for the given location. If all values are within non-hazardous limits, the printout stays brief as possible to avoid unnecessary cluttering. So, if one or more values exceed hazardous limits, they will be printed out and a warning will be issued for each value in question. By stating -p, the pollution information can be expanded to see all values in details all the time.
- s *speed*
Set playback speed in Baud. A *speed* of 0 prints out the contents immediately all at once. If -s and -d are set both to 0, the program will terminate with an error message, because this setup would state an infinite loop. The default *speed* value, if not stated otherwise, is 600 Baud.
- d *delay*
Set **qam** into screensaver mode. The program will loop according to the given values stated in other options or, if absent, defined by default. -d sets the loop return delay in seconds. A *delay* of 0 means no delay, meaning, after finishing the loop, the next one will be executed immediately. If -d and -s are set both to 0, the program will terminate with an error message, because this setup would state an infinite loop. The default *delay* value, if not stated otherwise, is 5 seconds.

IMPLEMENTATION

qam has been implemented in POSIX Bourne Shell *sh*(1), relying mostly on programs and constructs of a standard POSIX compliant system. The retrieval of weather data is executed by using *curl*(1), alternatively *wget2*(1), *wget*(1), *fetch*(1) or *ftp*(1), in that row.

EXIT STATUS

The **qam** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

1. Prints out wather information about a random location:

```
qam
```

2. Prints out weather information for *Jerusalem*:

```
qam -l Jerusalem
```

3. As 2, but stating the playback *speed* to 1200 Baud:

```
qam -l Jerusalem -s 1200
```

4. As 2, but disables ANSI sequences, stating the reloop *delay* to 10 seconds and stating the playback *speed* to 1200 Baud as a screensaver:

```
qam -l Jerusalem -a -s 1200 -d 10
```

5. Prints out a weather forecast for *Jerusalem* on a three-hour basis for the next five days:

```
qam -l Jerusalem -f
```

6. Prints out a daily weather forecast for *Jerusalem* for the next seven days:

```
qam -l Jerusalem -F
```

7. As 6, but additionally exports the data into the default export directory, usually *qamdata*:

```
qam -l Jerusalem -F -e
```

SEE ALSO

clear(1), *curl(1)*, *fetch(1)*, *ftp(1)*, *sed(1)*, *sh(1)*, *sleep(1)*, *srl(1)*, *wget(1)*, *wget2(1)*, *scrsvr(6)*, *ascii(7)*, *rnd(4)*, *stderr(4)*

STANDARDS

qam is expected to be written in valid IEEE Std 1003.2 (“POSIX.2”).

Besides that **qam** is designed to limit its own printouts to a width of 72 characters per line in pure *ascii(7)* to stay compatible with even the oldest glass terminals. The rich text output uses ANSI escape sequences to embolden or underlining certain texts.

HISTORY

A **qam** utility exists since June, 10th, 2026 and based initially on *scrsvr(6)* from 2023, but eventually emerged into a seperate program.

COPYRIGHT

Copyright (C) 2026, *Marc Fege*.

This program and this documentation are provided by the terms of the “GNU GPL” in version 2. The original text of the licence could be obtained from:

<http://www.gnu.org/licenses/old-licenses/gpl-2.0>

AUTHOR

Marc Fege designed and implemented this program and wrote this man page.

Ressources	Address
Internet	https://www.fege.net/software
Mailbox	None.

Contact	Connection
E-mail	<i>13mdf@fege.net</i>
QRA (CB)	13MDF
QRA (HAM)	DN9MF

BUGS

In **NetBSD**, for some reason, *sh(1)* does not scope the environment for *printf(1)* correctly, resulting in an unprecise display, where all decimals after the dot are nulled. This can avoided by executing the program with *ksh(1)* instead of *sh(1)* and change the first line of **qam** respectively as a workaround.