NAME

gma go map-console – GMA map service console (Go version)

SYNOPSIS

(If using the full GMA core tool suite)

`gma go map-console` [options as described below...]

(Otherwise)

`map-console -h`

`map-console -help`

`map-console` [`-Dm`] [`-C` *configfile*] [`-c` *calendar*] [`-H` *host*] [`-l` *logfile*] [`-list-profiles`] [`-P` *password*] [`-p` *port*] [`-S` *profile*] [`-u` *user*]

`map-console` [`-calendar` *calendar*] [`-config` *configfile*] [`-debug`] [`-help`] [`-host` *host*] [`-list-profiles`] [`-log` *logfile*] [`-mono`] [`-password` *password*] [`-port` *port*] [`-select` *profile*] [`-username` *user*]

DESCRIPTION

`Map-console` provides a way to interact directly with the GMA game server. It will print any server messages it receives in a colorized text representation. Commands typed into `map-console` are sent to the server as described in detail below.

This tool is designed primarily for debugging the server. Its input and output is not designed to be user-friendly, but rather to make it possible for someone familiar with the server's operation and network protocol to manually manipulate it.

When `map-console` starts, it will look for a GMA Mapper preferences file at `~/.gma/mapper/preferences.json`. If present, that file will be loaded to obtain the connection information. If no such file was found, it will fall back to the legacy behavior by reading an old-style Mapper configuration file at `~/.gma/mapper/mapper.conf`.

If an old-style configuration file is named explicitly via the `-config` CLI option, it will be loaded after `preferences.json` (such that its settings override the preferences file's information).

OPTIONS

The command-line options described below have a long form (e.g., `-port`) and a short form (e.g., `-p`) which are equivalent. In either case, the option may be introduced with either one or two hyphens (e.g., `-port` or `--port`). Options which take parameter values may have the value separated from the option name by a space or an equals sign (e.g., `-port=2323` or `-port 2323`), except for boolean flags which may be given alone (e.g., `-m`) to indicate that the option is set to "true" or may be given an explicit value which must be attached to the option with an equals sign (e.g., `-m=true` or `-m=false`).

You **may not** combine multiple single-letter options into a single composite argument, (e.g., the options `-r` and `-m` would need to be entered as two separate options, not as `-rm`).

Any of these options will override the settings read from `preferences.json` or `mapper.conf`.

`-c`, `-calendar` *name*
: This specifies the name of the calendar system in use for the campaign. If not specified, it defaults to "`golarion`". Normally, the server should be configured to tell all clients (including `map-client`) what calendar is in use, in which case if you also provide this option it will override the server's advertised calendar in favor of the one you are explicitly setting here.

`-C`, `-config` *file*
: The named *file* is read to set the same options as documented here for command-line parameters. The only difference is that the long name of the option must be used with no leading hyphens, and with equals signs between the option name and value, one option per line. For example, the file could contain a line reading

  `host=example.com`

but not lines like

```
-host=example.com
--host example.com
h=example.com
```

Any options set on the command line override those read from the configuration file. To set a boolean flag to be true, simply name it on a line of the file. For example:

```
mono
```

Lines beginning with an octothorpe ("#") are ignored as comments.

`-D`, `-debug` *flags*
: This adds debugging messages to `map-console`'s output. The *flags* value is a comma-separated list of debug flag names, which may be any of the following:

  `all` Enable all possible debugging flags.

  `none` If you want to explicitly disable debugging (e.g., if debugging is enabled in your configuration file but you want to turn it off via command-line option), you can specify `none` for the *flags* value to override the previous flag list and effectively disable debugging. If `none` appears in a comma-separated flag list, it cancels all the previously-set flags, but any other flag names which occur after it will be set.

  `auth` Authentication operations

  `binary` When printing certain values such as data transmitted over the network, include a hex dump of the actual binary data in addition to other debugging messages.

  `events` Show background events such as expiring timers and received process signals.

  `i/o` Show input/output operations used to get data in and out of the client.

  `messages` Show the server messages sent and received.

  `misc` Show miscellaneous debugging messages

`-H`, `-host` *host*
: Specifies the server's hostname.

`-l`, `-log` *file*
: Directs log messages (including any debugging output) to the named *file* instead of the standard output. You may explicitly specify the standard output by using a single hyphen as the file name (e.g., "`-log=-`").

`-list-profiles`
: Prints a list of all the profiles defined in the mapper preferences and exits.

`-m`, `-mono`
: Prevent `map-console` from using ANSI escape codes to colorize the output. With this flag set, only plain text output will be emitted.

`-P`, `-password` *password*
: If the server has authentication enabled, this specifies the password to be sent to log in. If the GM password is given, then the `map-console` user will have GM privileges on the server; otherwise normal player privileges will be granted, as with any client.

`-p`, `-port` *port*
: Specifies the server's TCP port number. The GMA map server's default port, 2323, will be assumed by default.

`-S`, `-select` *profile*
: Use the named *profile* from the mapper preferences for the connection information instead of the one currently designated as the mapper's current profile.

`-u`, `-username` *user*
: This specifies the user name by which the server will know you. If you log in with the GM credentials, the server will assign you the name "`GM`" regardless of what you request here. If you don't use the GM credentials, you may not ask for the name "`GM`". If you do not specify a username, it will default to your local system username if possible.

COMMANDS

Commands typed into the standard input of `map-console` are sent to the server as described here.

Obviously, this should be done with caution by someone intimately familiar with the protocol and who understands the implications of injecting commands into the working system like this.

Pre-Defined Commands

The following commands are recognized with a simple interface which should be easier to type than a full JSON string would be. They may not suffice for every possible set of operations; they are designed to handle common cases conveniently.

Note that the command names may be typed in either upper- or lower-case, but the values are taken exactly as typed. The entire input line must conform to the syntax of a Tcl list string. This means, in a nutshell, that the command and its arguments are separated by spaces, and that multi-word values need to be enclosed in curly braces so they are interpreted as a single value. Braces must be balanced. An empty string value may be typed as "`{}`".

`AI` *name size file*
: Upload an image from a local named file, for clients to access with the given *name* and zoom factor *size* (the latter expressed as a real number with 1.0 meaning the normal zoom setting).

  This is deprecated. Instead, images should be prepared using the `gma rendersizes`(6) program and uploaded to the server directly.

`AI?` *name size*
: Ask the server and/or other connected clients if they know where an image file with the given *name* and zoom factor *size* may be found.

`AI@` *name size id*
: Inform the server that it should advertise the location of a stored image file with the given *name* and *size* as the server storage name *id*. (Refer to the full documentation for an explanation of what that actually means.)

`AV` *label x y*
: Adjust the view of all clients so that grid label *label*, or if that is empty or unable to be understood, scroll so that the display is the fraction *x* of the way to the right and *y* of the way down, where *x* and *y* are numbers ranging from 0.0 (far left or top) to 1.0 (far right or bottom).

`CC` *silent? target*
: Tell the server to clear the chat history. If *silent?* is true, do so without announcement. The *target* is negative, all messages are deleted except for the most recent −*target* messages. Otherwise, all messages with IDs less that *target* are deleted.

`CLR` *id*
: Remove the specified object from the map clients. The *id* may be an object ID number, name as known to the mapper (e.g., "`Bob`" or "`goblinMimg=Goblin`"), or the values `*`, `E*`, `M*`, or `P*`, to remove all objects, all map elements, all monsters, or all player tokens, respectively.

`CLR@` *id*
: Clears all elements that are mentioned in the server-side map file with the specified *id*.

`CO` *enabled?*
: Enter combat mode if *enabled?* is true, otherwise exit to normal play mode.

`D` *recipients rollspec* [*id*]
: Ask the server to roll the dice as specified in *rollspec* with the results being sent to the list of names in *recipients* (which is itself a brace-enclosed, space separated Tcl list). The special

recipient names `*` and `%` may be used to send the results to all clients, or blindly send them only to the GM, respectively. In this case the `*` or `%` must be the only thing in the recipient list. You may optionally provide an arbitrary *id* which will be sent back with the die-roll results from the server.

`DD` *list* Set your server-side die-roll preset list to *list*, which is a brace-enclosed list of presets, each of which is a brace-enclosed list of three values: preset name, description, and the die-roll spec for that roll.

`DD+` *list*
Just like `DD` but adds the contents of *list* to your existing set of presets rather than replacing them.

`DD/` *re*
Delete all the die-roll presets stored for you whose names match the regular expression *re*.

`DR` Request that the server send you all your die-roll presets.

`EXIT` Exit the `map-console` program.

`HELP` Prints out a command summary.

`L` *filename*
Load the map file stored in a local file onto all the connected map clients. This replaces any existing elements on the map previously.

`L@` *id* Load the map file stored on the server under the given *id* onto all the connected map clients. This replaces any existing elements on the map previously.

`M` *filename*
Like `L` but merges the contents of the map with the existing contents of the map instead of replacing them.

`M?` *id* Tells clients to pre-fetch and cache a copy of the server's map file stored under the given *id*.

`M@` *id* Like `L@` but merges the map contents with the existing contents of the map.

`MARK` *x y*
Visibly mark the given (*x*,*y*) coordinates on the map for a second.

`OA` *id kvlist*
Set one or more attributes of the object with the given *id* to those in *kvlist*. The latter is a brace-enclosed, space-separated Tcl list where the first value is the name of an attribute, the next is the value for that attribute, and so on for each pair of attribute names and values you need to change.

`OA+` *id attribute list*
For object attributes whose values are a list of strings, this command adds one or more values to that object's attribute.

`OA-` *id attribute list*
Like `OA+` but removes each of the values in *list* from the named attribute.

`POLO` Sends a client response to the server's `MARCO` ping message. `Map-client` automatically sends these every time the server pings it.

`PS` *id color name area size* `player|monster` *x y reach*
Place a creature token on the map.

`QUIT` Synonymous with `EXIT`.

`SYNC` Request that the server send a full dump of the game state to you.

`SYNC CHAT` [*target*]
Request that the server send a full dump of all chat messages in its history to you. If *target* is given, it limits the number of requested chat message. If it is negative, only the most recent −*target* messages are sent. Otherwise, only any messages with message IDs greater than *target* are sent.

`TO` *recipients message*
: Send a chat *message* to the users named in the *recipients* list. The latter value may be given as described above for the `D` command.

`/CONN` Request a list of all connected clients.

`?` Synonymous with `HELP`.

Generalized Command Entry

You may also type a full command with options by typing "`!`*command*" followed by a number of parameters in the following forms. The same rules apply as above, so values which contain spaces will need to be enclosed in braces.

key=*val*
: Include parameter *key* in the JSON payload for the command, with the value *val* as a character string.

key`#`*val*
: As above, but don't quote *val* as a string. Thus, *val* may be a number, `true`, `false`, or `null`.

key`:`*val*
: As *key*=*val* except that any underscore characters in *val* are converted to spaces, making it unnecessary to put braces around this parameter.

For example, typing the command:

```
!d RequestID=abc123 ToAll#true RollSpec:d20+2_acid
```

will send this command to the server:

```
D {"RequestID":"abc123","ToAll":true,"RollSpec":"d20+2 acid"}
```

The *command* name may be typed in any case but the parameters must be typed exactly as expected by the server protocol.

Raw Message Entry

Finally, it is also possible to simply type a literal string of characters which will be sent to the server AS-IS without further interpretation.

This is done by prefixing the string with a backquote character. Thus, the previous server command could have been typed into `map-console` literally as:

```
`D {"RequestID":"abc123","ToAll":true,"RollSpec":"d20+2 acid"}
```

SEE ALSO

`gma`(6), `gma-mapper`(5), `gma-mapper`(6).

AUTHOR

Steve Willoughby / steve@madscience.zone.

BUGS

This program blindly assumes that the user's terminal understands ANSI standard escape codes to produce colored text (although the `-mono` option will stop it from doing so.)

COPYRIGHT