Q2PRO CLIENT MANUAL: ***http://skuller.net/q2pro/nightly/client.html***

Andrey Nazarov ***skuller@skuller.net***

About

Q2PRO is an enhanced, multiplayer oriented Quake 2 client, compatible with existing Quake 2 ports and licensed under GPLv2. This document provides descriptions of console variables and commands added to or modified by Q2PRO since the original Quake 2 release. Cvars and commands inherited from original Quake 2 are not described here (yet).

VARIABLES

NETCODE

Q2PRO client supports separation of outgoing packet rate, physics frame rate and rendering frame rate. Separation of physics and rendering frame rates is accomplished in R1Q2 'cl_async' style and is enabled by default.

In addition to this, Q2PRO network protocol is able to pack several input commands into the single network packet for outgoing packet rate reduction. This is very useful for some types of network links like Wi-Fi that can't deal with large number of small packets and cause packet delay or loss. Q2PRO protocol is only in use when connected to a Q2PRO server.

For the default Quake 2 protocol and R1Q2 protocol a hacky solution exists, which exploits dropped packet recovery mechanism for the purpose of packet rate reduction. This hack is disabled by default.

cl_protocol

Specifies preferred network protocol version to use when connecting to servers. If the server doesn't support the specified protocol, client will fall back to the previous supported version. Default value is 0.

0 — automatically select the highest protocol version supported
34 — use default Quake 2 protocol
35 — use enhanced R1Q2 protocol
36 — use enhanced Q2PRO protocol

cl_maxpackets

Number of packets client sends per second. 0 means no particular limit. Unless connected using Q2PRO protocol, this variable is ignored and packets are sent in sync with client physics frame rate, controlled with 'cl_maxfps' variable. Default value is 30.

cl_fuzzhack
Enables 'cl_maxpackets' limit even if Q2PRO protocol is not in use by dropping packets. This is not a generally recommended thing to do, but can be enabled if nothing else helps to reduce ping. Default value is 0 (disabled).

cl_packetdup
Number of backup movement commands client includes in each new packet, directly impacts upload rate. Unless connected using Q2PRO protocol, hardcoded value of 2 backups per packet is used. Default value is 1.

cl_instantpacket
Specifies if important events such as pressing '+attack' or '+use' are sent to the server immediately, ignoring any rate limits. Default value is 1 (enabled).

cl_async
Controls rendering frame rate and physics frame rate separation. Default value is 1. Influence of 'cl_async' on client framerates is summarized in the table below.

0 — run synchronous, like original Quake 2 does
1 — run asynchronous
2 — run asynchronous, limit rendering frame rate to monitor's vertical retrace frequency (supported only by X11/GLX drivers)

Table 1. Rate limits depending on 'cl_async' value

Value of 'cl_async'	Rendering	Physics	Main loop
0	cl_maxfps	cl_maxfps	cl_maxfps
1	r_maxfps	cl_maxfps	unlimited
2	vertical refresh	cl_maxfps	unlimited

r_maxfps
Specifies maximum rendering frame rate if 'cl_async' is set to 1, otherwise ignored. Default value is 0, which means no particular limit.

cl_maxfps
Specifies client physics frame rate if 'cl_async' 1 or 2 is used. Otherwise, limits both rendering and physics frame rates. Default value is 60.

cl_gibs
Controls rendering of entities with 'EF_GIB' flag set. When using Q2PRO protocol, disabling this saves some bandwidth since the server stops sending these entities at all. Default value is 1 (enabled).

cl_gun
Controls rendering of the player's own gun model. When using R1Q2 or Q2PRO protocol, disabling this saves some bandwidth since the server stops sending gun updates at all. Default value is 1 (enabled).

cl_footsteps
Controls footstep sounds. When using Q2PRO protocol, disabling this saves some bandwidth since the server stops sending footstep events at all. Default value is 1 (enabled).

cl_updaterate
Specifies the perferred update rate requested from Q2PRO servers. Only used when server is running in variable FPS mode, otherwise default rate of 10 packets per second is used. Specified rate should evenly divide native server frame rate. Default value is 0, which means to use the highest update rate available (that is, native server frame rate).

NETWORK

net_enable_ipv6
Enables IPv6 support. Default value is 1 on systems that support IPv6 and 0 otherwise.

0 — disable IPv6, use IPv4 only
1 — enable IPv6, but prefer IPv4 over IPv6 when resolving host names with multiple addresses
2 — enable IPv6, use normal address resolver priority configured by OS

net_ip
Specifies network interface address client should use for outgoing UDP connections using IPv4. Default value is empty, which means that default network interface is used.

net_ip6
Specifies network interface address client should use for outgoing UDP connections using IPv6. Default value is empty, which means that default network interface is used. Has no effect unless 'net_enable_ipv6' is set to non-zero value.

net_clientport
Specifies UDP port number client should use for outgoing connections (using IPv4 or IPv6). Default value is -1, which means that random port number is chosen at socket creation time.

net_maxmsglen
Specifies maximum server to client packet size client will request from servers. 0 means no hard limit. Default value is conservative 1390 bytes. It is nice to have this variable as close to your network link MTU as possible (accounting for headers). Thus for normal Ethernet MTU of 1500 bytes 1462 can be specified (10 bytes quake header, 8 bytes UDP header, 20 bytes IPv4 header). Higher values may cause IP fragmentation which is better to avoid. Servers will cap this variable to their own maximum values. Please don't change this variable unless you know exactly what you are doing.

net_chantype
Specifies if enhanced Q2PRO network channel implementation is enabled when connecting to Q2PRO servers. Q2PRO netchan supports application-level fragmentation of datagrams that results is better gamestate compression ratio and faster map load times. Default value is 1 (enabled).

TRIGGERS

cl_beginmapcmd
Specifies command to be executed each time client enters a new map. Default value is empty.

cl_changemapcmd
Specifies command to be executed each time client begins loading a new map. Default value is empty.

cl_disconnectcmd
Specifies command to be executed each time client disconnects from the server. Default value is empty.

See also 'trigger' client command description.

EFFECTS

COLOR SPECIFICATION

Colors can be specified in one of the following formats:

#RRGGBBAA, where R, G, B and A are hex digits
#RRGGBB, which implies alpha value of FF
#RGB, which is expanded to #RRGGBB by duplicating digits

one of the predefined color names (black, red, etc)

cl_railtrail_type
Defines which type of rail trail effect to use. Default value is 0.

0 — use original effect
1 — use alternative effect, draw rail core only
2 — use alternative effect, draw rail core and spiral

Note
Rail trail variables listed below apply to the alternative effect only.

cl_railtrail_time
Time, in seconds, for the rail trail to be visible. Default value is 1.0.

cl_railcore_color
Color of the rail core beam. Default value is "red".
cl_railcore_width
Width of the rail core beam. Default value is 3.

cl_railspiral_color
Color of the rail spiral. Default value is "blue".

cl_railspiral_radius
Radius of the rail spiral. Default value is 3.

cl_disable_particles
Disables rendering of particles for the following effects. This variable is a bitmask. Default value is 0.

1 — grenade explosions
2 — grenade trails
4 — rocket explosions
8 — rocket trails

Bitmasks
Bitmask cvars allow multiple features to be enabled. To enable the needed set of features, their values need to be summed.

cl_disable_explosions
Disables rendering of animated models for the following effects. This variable is a bitmask. Default value is 0.

1 — grenade explosions
2 — rocket explosions

cl_noglow
Disables the glowing effect on bonus entities like ammo, health, etc. Default value is 0 (glowing enabled).

cl_gunalpha
Specifies opacity level of the player's own gun model. Default value is 1 (fully opaque).

SOUND SUBSYSTEM

s_enable
Specifies which sound engine to use. Default value is 1.

0 — sound is disabled
1 — use DMA sound engine
2 — use OpenAL sound engine

s_ambient
Specifies if ambient sounds are played. Default value is 1.

0 — all ambient sounds are disabled
1 — all ambient sounds are enabled
2 — only ambient sounds from visible entities are enabled (rocket flybys, etc)
3 — only ambient sounds from player entity are enabled (railgun hum, hand grenade ticks, etc)

s_auto_focus
Specifies the minimum focus level main Q2PRO window should have for sound to be activated. Default value is 0.

0 — sound is always activated
1 — sound is activated when main window is visible, and deactivated when it is iconified, or moved to another desktop
2 — sound is activated when main window has input focus, and deactivated when it loses it

s_swapstereo
Swap left and right audio channels. Only effective when using DMA sound engine. Default value is 0 (don't swap).

al_driver
Specifies the name of OpenAL driver to use. Default value is 'openal32' on Windows, and 'libopenal.so.1' on Linux.

al_device
Specifies the name of OpenAL device to use. Format of this value depends on your OpenAL implementation. Default value is empty, which means default sound output device is used.

Tip
On Windows, there are two well-known OpenAL implementations available: OpenAL32 from Creative, with support for harware acceleration on certain audio cards, and an open source software implementation named OpenAL Soft. Both should work with Q2PRO, but to get the results most perceptually close to original Quake 2 sound, I recommend using OpenAL Soft. Creative's implementation seems to perform some default effects processing even when not requested, and that makes it sound somewhat differently. With OpenAL Soft in stereo configuration I can't really tell if I'm using OpenAL or default Quake 2 sound engine. Of course you can install both implementations and switch between them by changing 'al_driver' variable between 'openal32' and 'soft_oal'.

GRAPHICAL CONSOLE

con_clock
Toggles drawing of the digital clock at the lower right corner of console. Default value is 0 (disabled).

con_height
Fraction of the screen in-game console occupies. Default value is 0.5.

con_alpha
Opacity of in-game console background. 0 is fully transparent, 1 is opaque. Default value is 1.

con_scale
Scaling factor of the console text. Takes effect in OpenGL mode only. Default value is 1. Automatically scales depending on current display resolution when set to 0.

con_font
Font used for drawing console text. Default value is "conchars".

con_background
Image used as console background. Default value is "conback".

con_notifylines
Number of the last console lines displayed in the notification area in game. Default value is 4.

con_history
Specifies how many lines to save into console history file before exiting Q2PRO, to be reloaded on next startup. Maximum number of history lines isDefault value is 0.

con_scroll
Controls automatic scrolling of console text when some event occurs. This variable is a bitmask. Default value is 0.

1 — when new command is entered
2 — when new lines are printed

GAME SCREEN

scr_draw2d
Toggles drawing of 2D elements on the screen. Default value is 2.

0 — do not draw anything
1 — do not draw stats program
2 — draw everything

scr_showturtle
Toggles drawing of various network error conditions at the lower left corner of the screen. Default value is 1 (draw all errors except of SUPPRESSED, CLIENTDROP and SERVERDROP). Values higher than 1 draw all errors.

TYPES OF NETWORK ERROR

SERVERDROP
Packets from server to client were dropped by the network.

CLIENTDROP
A few packets from client to server were dropped by the network. Server recovered player's movement using backup commands.

CLIENTPRED
Many packets from client to server were dropped by the network. Server ran out of backup commands and had to predict player's movement.

NODELTA
Server sent an uncompressed frame. Typically occurs during a heavy lag, when a lot of packets are dropped by the network.

SUPPRESSED
Server suppressed packets to client because rate limit was exceeded.

BADFRAME
Server sent an invalid delta compressed frame.

OLDFRAME
Server sent a delta compressed frame that is too old and can't be recovered.

OLDENT
Server sent a delta compressed frame whose entities are too old and can't be recovered.

scr_demobar
Toggles drawing of progress bar at the bottom of the screen during demo playback. Default value is 1.

0 — do not draw demo bar
1 — draw demo bar and demo completion percentage
2 — draw demo bar, demo completion percentage and current demo time

scr_showpause
Toggles drawing of pause indicator on the screen. Default value is 1.

0 — do not draw pause indicator
1 — draw pic in center of the screen
2 — draw text in demo bar (visible only during demo playback)

scr_scale
Scaling factor of the HUD elements. Takes effect in OpenGL mode only. Default value is 1. Automatically scales depending on current display resolution when set to 0.

scr_alpha
Opacity of the HUD elements. 0 is fully transparent, 1 is opaque. Default value is 1.

scr_font
Font used for drawing HUD text. Default value is "conchars".

scr_lag_draw
Toggles drawing of small (48x48 pixels) ping graph on the screen. Default value is 0.

0 — do not draw graph
1 — draw transparent graph
2 — overlay graph on gray background

scr_lag_x
Absolute value of this cvar specifies horizontal placement of the ping graph, counted in pixels from the screen edge. Negative values align graph to the right edge of the screen instead of the left edge. Default value is -1.

scr_lag_y
Absolute value of this cvar specifies vertical placement of the ping graph, counted in pixels from the screen edge. Negative values align graph to the bottom edge of the screen intead of the top edge. Default value is -1.

scr_lag_min
Specifies ping graph offset by defining the minimum value that can be displayed. Default value is 0.

scr_lag_max
Specifies ping graph scale by defining the maximum value that can be displayed. Default value is 200.

scr_chathud
Toggles drawing of the last chat lines on the screen. Default value is 0.

0 — do not draw chat lines
1 — draw chat lines in normal color
2 — draw chat lines in alternative color

scr_chathud_lines
Specifies number of the last chat lines drawn on the screen. Default value is 4. Maximum value is 32.

scr_chathud_time
Specifies visibility time of each chat line, counted in seconds. Default value is 0 (lines never fade out).

scr_chathud_x
Absolute value of this cvar specifies horizontal placement of the chat HUD, counted in pixels from the screen edge. Negative values align graph to the right edge of the screen instead of the left edge. Default value is 8.

scr_chathud_y
Absolute value of this cvar specifies vertical placement of the chat HUD, counted in pixels from the screen edge. Negative values align graph to the bottom edge of the screen intead of the top edge. Default value is -64.

ch_health
Enables dynamic crosshair coloring based on the health statistic seen in the player's HUD. Default value is 0 (use static color).

ch_red
ch_green
ch_blue
These variables specify the color of crosshair image. Default values are 1 (draw in white color). Ignored if 'ch_health' is enabled.

ch_alpha
Opacity level of crosshair image. Default value is 1 (fully opaque).

ch_scale
Scaling factor of the crosshair image. Default value is 1 (original size).

ch_x
ch_y
These variables specify the crosshair image offset, counted in pixels from the default position in center of the game screen. Default values are 0 (draw in center).

VIDEO MODES

Hard coded list of the fullscreen video modes is gone from Q2PRO, you can specify your own list in configuration files. Vertical refresh frequency freq and bit depth bpp can be specified individually for each mode.

Video mode change no longer requires ‘vid_restart’ and is nearly instant. In windowed mode, size as well as position of the main window can be changed freely.

vid_modelist
Space separated list of fullscreen video modes. Both freq and bpp parameters are optional. Full syntax is: WxH[@freq][:bpp] […]. Default value is "640x480 800x600 1024x768". On Linux, freq parameter is currently ignored. Special keyword ‘desktop’ means to use default desktop video mode.

vid_fullscreen
If set to non zero value, run in the specified fullscreen mode. This way, value acts as index into the list of video modes specified by ‘vid_modelist’. Default value is 0, which means to run in windowed mode.

vid_geometry
Size and optional position of the main window on virtual desktop. Full syntax is: WxH[+X+Y]. Default value is "640x480".

vid_flip_on_switch
On Windows, specifies if original video mode is automatically restored when switching from fullscreen Q2PRO to another application or desktop. Default value is 0 (don’t switch video modes).

vid_hwgamma
Instructs the video driver to use hardware gamma correction for implementing 'vid_gamma'. Default value is 0 (use software gamma).

Example 1. Setting video modes
The following lines define 2 video modes: 640x480 and 800x600 at 75 Hz vertical refresh and 32 bit framebuffer depth, and select the last 800x600 mode.

/set vid_modelist "640x480@75:32 800x600@75:32"
/set vid_fullscreen 2

WINDOWS SPECIFIC

The following variables are specific to the Windows port of Q2PRO.

win_noalttab
Disables the Alt-Tab key combination to prevent it from interfering with game when pressed. Default is 0 (don't disable).

win_disablewinkey
Disables the default Windows key action to prevent it from interfering with game when pressed. Default is 0 (don't disable).

win_noresize
Prevents the main window from resizing by dragging the border. Default is 0 (allow resizing).

win_notitle
Hides the main window title bar. Default is 0 (show title bar).

win_alwaysontop
Puts the main window on top of other windows. Default is 0 (main window can be obscured by other windows).

win_xpfix
Temporary disables mouse acceleration setting applied by the OS. Only effective when legacy Windows mouse input is in use, otherwise ignored. Default value is 0 (don't modify OS setting).

win_rawmouse
Enables raw mouse input instead of legacy Windows mouse input. Default value is 1 (use raw input).

OPENGLRENDER

gl_gamma_scale_pics
Apply software gamma scaling not only to textures and skins, but to HUD pictures also. Default value is 0 (don't apply to pics).

gl_noscrap
By default, OpenGL renderer combines small HUD pictures into the single texture called scrap. This usually speeds up rendering a bit, and allows pixel precise rendering of non power of two sized images. If you don't like this optimization for some reason, this cvar can be used to disable it. Default value is 0 (optimize).

gl_bilerp_chars
Enables bilinear filtering of charset images. Default value is 0 (disabled).

gl_bilerp_pics
Enables bilinear filtering of HUD pictures. Default value is 1.

0 — disabled for all pictures
1 — enabled for large pictures that don't fit into the scrap
2 — enabled for all pictures, including the scrap texture itself

gl_upscale_pcx
Enables upscaling of PCX images using HQ2x and HQ4x filters. This improves rendering quality when screen scaling is used. Default value is 0.

0 — don't upscale
1 — upscale 2x (takes 5x more memory)
2 — upscale 4x (takes 21x more memory)

gl_texture_non_power_of_two
Enables use of non power-of-two sized textures without resampling on OpenGL 3.0 and higher compliant hardware. Default value is 1.

gl_downsample_skins
Specifies if skins are downsampled just like world textures are. When disabled, 'gl_round_down', 'gl_picmip' cvars have no effect on skins. Default value is 1 (downsampling enabled).

gl_drawsky
Enable skybox texturing. 0 means to draw sky box in solid black color. Default value is 1 (enabled).

gl_fontshadow
Specifies font shadow width, in pixels, ranging from 0 to 2. Default value is 0 (no shadow).

gl_partscale
Specifies minimum size of particles. Default value is 2.

gl_partstyle
Specifies drawing style of particles. Default value is 0.

0 — blend colors
1 — saturate colors

gl_celshading
Enables drawing black contour lines around 3D models (aka 'celshading'). Value of this variable specifies thickness of the lines drawn. Default value is 0 (celshading disabled).

gl_dotshading
Enables dotshading effect when drawing 3D models, which helps them look truly 3D-ish by simulating diffuse lighting from a fake light source. Default value is 1 (enabled).

gl_saturation
Enables grayscaling of world textures. 1 keeps original colors, 0 converts textures to grayscale format (this may save some video memory and speed up rendering a bit since textures are uploaded at 8 bit per pixel instead of 24), any value in between reduces colorfulness. Default value is 1 (keep original colors).

gl_invert
Inverts colors of world textures. In combination with 'gl_saturation 0' effectively makes textures look like black and white photo negative. Default value is 0 (do not invert colors).

gl_anisotropy
When set to 2 and higher, enables anisotropic filtering of world textures, if supported by your OpenGL implementation. Default value is 0 (anisotropic filtering disabled).

gl_brightness
Specifies a brightness value that is added to each pixel of world lightmaps. Positive values make lightmaps brighter, negative values make lightmaps darker. Default value is 0 (keep original brightness).

gl_coloredlightmaps
Enables grayscaling of world lightmaps. 1 keeps original colors, 0 converts lightmaps to grayscale format, any value in between reduces colorfulness. Default value is 1 (keep original colors).

gl_modulate
Specifies a primary modulation factor that each pixel of world lightmaps is multiplied by. This cvar affects entity lighting as well. Default value is 1 (identity).

gl_modulate_world
Specifies an secondary modulation factor that each pixel of world lightmaps is multiplied by. This cvar does not affect entity lighting. Default value is 1 (identity).

gl_modulate_entities
Specifies an secondary modulation factor that entity lighting is multiplied by. This cvar does not affect world lightmaps. Default value is 1 (identity).

Tip
An old trick to make entities look brighter in Quake 2 was setting 'gl_modulate' to a high value without issuing 'vid_restart' afterwards. This way it was possible to keep 'gl_modulate' from applying to world lightmaps, but only until the next map was loaded. In Q2PRO this trick is no longer needed (and it won't work, since 'gl_modulate' is applied dynamically). To get the similar effect, set the legacy 'gl_modulate' variable to 1, and configure 'gl_modulate_world' and 'gl_modulate_entities' to suit your needs.
gl_doublelight_entities
Specifies if combined modulation factor is applied to entity lighting one more time just before final lighting value is calculated, to simulate a bug (?) in the original Quake 2 renderer. Default value is 1 (apply twice).

ENTITY LIGHTING

Entity lighting is calculated based on the color of the lightmap sample from the world surface directly beneath the entity. This means any cvar affecting lightmaps affects entity lighting as well (with exception of 'gl_modulate_world'). Cvars that have effect only on the entity lighting are 'gl_modulate_entities' and 'gl_doublelight_entities'. Yet another cvar affecting entity lighting is 'gl_dotshading', which typically makes entities look a bit brighter. See also 'cl_noglow' cvar which removes the pulsing effect (glowing) on bonus entities.

gl_dynamic
Controls dynamic lightmap updates. Default value is 2.

0 — all dynamic lighting is disabled
1 — all dynamic lighting is enabled
2 — most dynamic lights are disabled, but lightmap updates are still allowed for switchable lights to work

Note

Dynamic lights may noticeably hurt rendering performance on some video cards and drivers, therefore they are disabled by default.

gl_dlight_falloff

Makes dynamic lights look a bit smoother, opposed to original jagged Quake 2 style. Default value is 1 (enabled).

gl_fragment_program

Enables 'GL_ARB_fragment_program' extension, if supported by your OpenGL implementation. Currently this extension is used only for warping effect when drawing liquid surfaces. Default value is 1 (enabled).

gl_vertex_buffer_object

Enables 'GL_ARB_vertex_buffer_object' extension, if supported by your OpenGL implementation. This extension allows world surfaces to be stored in high-performance video memory, which usually speeds up rendering. Default value is 1 (enabled).

gl_video_sync

On X11/GLX, enables 'GLX_SGI_video_sync' extension. This extension allows synchronizing rendering framerate to monitor vertical retrace frequency. Default value is 1 (enabled). See also 'cl_async' variable.

gl_colorbits

Specifies desired size of color buffer, in bits, requested from OpenGL implementation (should be typically 0, 24 or 32). Default value is 0 (determine the best value automatically).

gl_depthbits

Specifies desired size of depth buffer, in bits, requested from OpenGL implementation (should be typically 0 or 24). Default value is 0 (determine the best value automatically).

gl_stencilbits

Specifies desired size of stencil buffer, in bits, requested from OpenGL implementation (should be typically 0 or 8). Currently stencil buffer is used only for drawing projection shadows. Default value is 8. 0 means no stencil buffer requested.

gl_multisamples

Specifies number of samples per pixel used to implement multisample anti-aliasing, if supported by OpenGL implementation. Values 0 and 1 are equivalent and disable MSAA. Values from 2 to 32 enable MSAA. Default value is 0.

gl_texturebits
Specifies number of bits per texel used for internal texture storage (should be typically 0, 8, 16 or 32). Default value is 0 (choose the best internal format automatically).

gl_screenshot_format
Specifies image format ‘screenshot’ command uses. Possible values are "png", "jpg" and "tga". Default value is "jpg".

gl_screenshot_quality
Specifies image quality of JPG screenshots. Values range from 0 (worst quality) to 100 (best quality). Default value is 100.

gl_screenshot_compression
Specifies compression level of PNG screenshots. Values range from 0 (no compression) to 9 (best compression). Default value is 6.

r_override_textures
Enables automatic overriding of palettized textures (in WAL or PCX format) with truecolor replacements (in PNG, JPG or TGA format) by stripping off original file extension and searching for alternative filenames in the order specified by ‘r_texture_formats’ variable. Default value is 1 (enabled).

r_texture_formats
Specifies the order in which truecolor texture replacements are searched. Default value is "pjt", which means to try ‘.png’ extension first, then ‘.jpg’, then ‘.tga’.

MD2 model overrides
When Q2PRO attempts to load an alias model from disk, it determines actual model format by file contents, rather than by filename extension. Therefore, if you wish to override MD2 model with MD3 replacement, simply rename the MD3 model to ‘tris.md2’ and place it in appropriate packfile to make sure it gets loaded first.

DOWNLOADS

These variables control automatic client downloads (both legacy UDP and HTTP downloads).

allow_download
Globally allows or disallows client downloads. Remaining variables listed below are effective only when downloads are globally enabled. Default value is 1.

-1 — downloads are permanently disabled (once this value is set, it can't be modified)
0 — downloads are disabled
1 — downloads are enabled

allow_download_maps
Enables automatic downloading of maps. Default value is 1.

allow_download_models
Enables automatic downloading of non-player models, sprites and skins. Default value is 1.

allow_download_sounds
Enables automatic downloading of non-player sounds. Default value is 1.

allow_download_pics
Enables automatic downloading of HUD pictures. Default value is 1.
allow_download_players
Enables automatic downloading of player models, skins, sounds and icons. Default value is 1.

allow_download_textures
Enables automatic downloading of map textures. Default value is 1.

HTTP DOWNLOADS

cl_http_downloads
Enables HTTP downloads, if server advertises download URL. Default value is 1 (enabled).

cl_http_filelists
When a first file is about to be downloaded from HTTP server, send a filelist request, and download any additional files specified in the filelist. Filelists provide a 'pushing' mechanism for server operator to make sure all clients download complete set of data for the particular mod, instead of requesting files one-by-one. Default value is 1 (request filelists).

cl_http_max_connections
Maximum number of simultaneous connections to the HTTP server. Default value is 2.

cl_http_proxy
HTTP proxy server to use for downloads. Default value is empty (direct connection).

LOCATIONS

Client side location files provide a way to report player's position on the map in team chat messages without depending on the game mod. Locations are loaded from 'locs/<mapname>.loc' file. Once location file is loaded, 'loc_here' and 'loc_there' macros will expand to the name of location closest to the given position. Variables listed below control some aspects of location selection.

loc_trace
When enabled, location must be directly visible from the given position (not obscured by solid map geometry) in order to be selected. Default value is 0, which means any closest location will satisfy, even if it is placed behind the wall.

loc_dist
Maximum distance to the location, in world units, for it to be considered by the location selection algorithm. Default value is 500.

loc_draw
Enables visualization of location positions. Default value is 0 (disabled).

MOUSE INPUT

in_direct
On Linux, enables Evdev interface for direct mouse input. Otherwise, standard input facilities provided by the window system are used. Default value is 1 (use direct input).

in_device
On Linux, specifies device file to use for direct mouse input. Normally, it should be one of '/dev/input/eventX' files (reading permissions are required). Default value is empty and needs to be filled by user.

in_grab
Specifies mouse grabbing policy in windowed mode. Normally, mouse is always grabbed in-game and released when console or menu is up. In addition to that, smart policy mode automatically releases the mouse when its input is not needed (playing a demo, or spectating a player). Default value is 1.

0 — don't grab mouse
1 — normal grabbing policy
2 — smart grabbing policy

m_autosens
Enables automatic scaling of mouse sensitivity proportional to the current player field of view. Values between 90 and 179 specify the default FOV value to scale sensitivity from. Zero disables automatic scaling. Any other value assumes default FOV of 90 degrees. Default value is 0.

m_accel
Specifies mouse acceleration factor. Default value is 0 (acceleration disabled).

m_filter
When enabled, mouse movement is averaged between current and previous samples. Default value is 0 (filtering disabled).

lirc_enable
On Linux, enables input from the LIRC daemon, which allows menu navigation and command execution from your infrared remote control device. Default value is 0 (disabled).

lirc_config
On Linux, specifies LIRC configuration file to use. Default value is empty, which means to use the default '~/.lircrc' file. This variable may only be set from command line. See README.lirc file for command syntax description.

MISCELLANEOUS

cl_chat_notify
Specifies whether to display chat lines in the notify area. Default value is 1 (enabled).

cl_chat_sound
Specifies sound effect to play each time chat message is received. Default value is 1.

0 — don't play chat sound
1 — play normal sound ('misc/talk.wav')
2 — play alternative sound ('misc/talk1.wav')

cl_chat_filter
Specifies if unprintable characters are filtered from incoming chat messages, to prevent common exploits like hiding player names. Default value is 0 (don't filter).

cl_noskins
Restricts which models and skins players can use. Default value is 0.

0 — no restrictions, if skins exists, it will be loaded
1 — do not allow any skins except of 'male/grunt'
2 — do not allow any skins except of 'male/grunt' and 'female/athena'

Tip
With 'cl_noskins' set to 2, it is possible to keep just 2 model/skin pairs ('male/grunt' and 'female/athena') to save memory and reduce map load times. This will not affect model-based TDM gameplay, since any male skin will be replaced by 'male/grunt' and any female skin will be replaced by 'female/athena'.

cl_rollhack
Default OpenGL renderer in Quake 2 contained a bug that caused 'roll' angle of 3D models to be inverted during rotation. Due to this bug, player models did lean in the opposite direction when strafing. New Q2PRO renderer doesn't have this bug, but since many players got used to it, Q2PRO is able to simulate original behavior. This cvar chooses in which direction player models will lean. Default value is 1 (invert 'roll' angle).

cl_adjustfov
Specifies if horizontal field of view is automatically adjusted for screens with aspect ratio different from 4/3. Default value is 0 (don't adjust FOV).

cl_demosnaps
Specifies time interval, in seconds, between saving 'snapshots' in memory during demo playback. Snapshots enable backward seeking in demo (see 'seek' command description), and speed up repeated forward seeks. Setting this variable to 0 disables snapshotting entirely. Default value is 10.

cl_demomsglen
Specifies default maximum message size used for demo recording. Default value is 1390. See 'record' command description for more information on demo packet sizes.

cl_demowait
Specifies if demo playback is automatically paused at the last frame in demo file. Default value is 0 (finish playback).

cl_autopause
Specifies if single player game or demo playback is automatically paused once client console or menu is opened. Default value is 1 (pause game).

ui_open
Specifies if menu is automatically opened on startup, instead of full screen console. Default value is 1 (open menu).

ui_background
Specifies image to use as menu background. Default value is empty, which just fills the screen with solid black color.

ui_scale
Scaling factor of the UI widgets. Takes effect in OpenGL mode only. Default value is 1. Automatically scales depending on current display resolution when set to 0.

ui_sortdemos
Specifies default sorting order of entries in demo browser. Default value is 1. Negate the values for descending sorting order instead of ascending.

0 — don't sort
1 — sort by name
2 — sort by date
3 — sort by size
4 — sort by map
5 — sort by POV

ui_listalldemos
List all demos, including demos in packs and demos in base directories. Default value is 0 (limit the search to physical files within the current game directory).

ui_sortservers
Specifies default sorting order of entries in server browser. Default value is 0. Negate the values for descending sorting order instead of ascending.

0 — don't sort
1 — sort by hostname
2 — sort by mod
3 — sort by map
4 — sort by players
5 — sort by RTT

ui_colorservers
Enables highlighting of entries in server browser with different colors. Currently, this option grays out password protected and anticheat enforced servers. Default value is 0 (disabled).

ui_pingrate
Specifies the server pinging rate used by server browser, in packets per second. Default value is 0, which estimates the default pinging rate based on ‘rate’ client variable.

com_time_format
Time format used by ‘com_time’ macro. Default value is "%H.%M" on Win32 and "%H:%M" on UNIX. See strftime(3) for syntax description.

com_date_format
Date format used by ‘com_date’ macro. Default value is "%Y-%m-%d". See strftime(3) for syntax description.

MACROS

Macros behave like automated console variables. When macro expansion is performed, macros are searched first, then console variables.

Example 2. Macro expansion syntax
Each of the following examples are valid and produce the same output:

/echo $loc_here
/echo loc_here
/echo ${loc_here}
/echo ${$loc_here}

List of client macros

cl_armor
armor statistic seen in the HUD

cl_ammo
ammo statistic seen in the HUD

cl_health
health statistic seen in the HUD

cl_weaponmodel
current weapon model

cl_timer
time since level load

cl_demopos
current position in demo, in timespec syntax

cl_server
address of the server client is connected to

cl_mapname
name of the current map

loc_there
name of the location player is looking at

loc_here
name of the location player is standing at

cl_ping
average round trip time to the server
cl_lag
incoming packet loss percentage

cl_fps
main client loop frame rate [1]

cl_mps
movement commands generation rate in movements per second [2]

cl_pps
movement packets transmission rate in packets per second

cl_ups
player velocity in world units per second

r_fps
rendering frame rate

com_time
current time formatted according to ‘com_time_format’

com_date
current date formatted according to ‘com_date_format’

com_uptime
engine uptime in short format

net_dnrate
current download rate in bytes/sec

net_uprate
current upload rate in bytes/sec

random
expands to the random decimal digit

List of special macros

qt
expands to double quote

sc
expands to semicolon

$
expands to dollar sign

COMMANDS

CLIENT DEMOS

demo [/]<filename[.ext]>
Begins demo playback. This command does not require file extension to be specified and supports filename autocompletion on TAB. Loads file from 'demos/' unless slash is prepended to filename, otherwise loads from the root of quake file system. Can be used to launch MVD playback as well, if MVD file type is detected, it will be automatically passed to the server subsystem. To stop demo playback, type 'disconnect'.

seek [+-]<timespec>
Seeks the given amount of time during demo playback. Prepend with '+' to seek forward relative to current position, prepend with '-' to seek backward relative to current position. Without prefix, seeks to an absolute position within the demo file. See below for timespec syntax description. Initial forward seek may be slow, so be patient.

Note

The 'seek' command actually operates on demo frame numbers, not pure server time. Therefore, 'seek +300' does not exactly mean 'skip 5 minutes of server time', but just means 'skip 3000 demo frames', which may account for more than 5 minutes if there were dropped frames. For most demos, however, correspondence between frame numbers and server time should be reasonably close.

Demo time specification

Absolute or relative demo time can be specified in one of the following formats:

.FF, where FF are frames
SS, where SS are seconds
SS.FF, where SS are seconds, FF are frames
MM:SS, where MM are minutes, SS are seconds
MM:SS.FF, where MM are minutes, SS are seconds, FF are frames

record [-hzes] <filename>

Begins demo recording into 'demos/filename.dm2', or prints some statistics if already recording. If neither '--extended' nor '--standard' options are specified, this command uses maximum demo message size defined by 'cl_demomsglen' cvar.

-h | --help

display help message

-z | --compress

compress demo with gzip

-e | --extended

use extended packet size (4086 bytes)

-s | --standard

use standard packet size (1390 bytes)

Tip

With Q2PRO it is possible to record a demo while playing back another one.
stop
Stops demo recording and prints some statistics about recorded demo.

suspend

Pauses and resumes demo recording.

Demo packet sizes

Packet size options limit maximum demo message size and thus define compatibility level of the recorded demo. Original Quake 2 supports just 1390 bytes ('standard' size), while Q2PRO and R1Q2 support message sizes up to 4086 bytes ('extended' size). When Q2PRO or R1Q2 protocols are in use, demo written to disk is automatically downgraded to protocol 34. This can result in dropping of large frames that don't fit into standard protocol 34 limit. Demo packet size can be extended to overcome this, but the resulting demo will be playable only by Q2PRO and R1Q2 clients and will be incompatible with other Quake 2 clients or demo editing tools. By default, 'standard' packet size is used. This default can be changed using 'cl_demomsglen' cvar.

CVAR OPERATIONS

toggle <cvar> [value1 value2 …]

If values are omitted, toggle the specified cvar between 0 and 1. If two or more values are specified, cycle through them.

inc <cvar> [value]

If value is omitted, add 1 to the value of cvar. Otherwise, add the specified floating point value.

dec <cvar> [value]

If value is omitted, subtract 1 from the value of cvar. Otherwise, subtract the specified floating point value.

reset <cvar>

Reset the specified cvar to it's default value.

resetall

Resets all cvars to their default values.

set <cvar> <value> [u|s|…]

If 2 arguments are given, sets the specified cvar to value. If 3 arguments are given, and the last argument is 'u' or 's', sets cvar to value and marks the cvar with 'userinfo' or 'serverinfo' flags, respectively. Otherwise, sets cvar to value, which is handled as consisting from multiple tokens.

setu <cvar> <value> […]

Sets the specified cvar to value, and marks the cvar with 'userinfo' flag. Value may be composed from multiple tokens.

sets <cvar> <value> […]

Sets the specified cvar to value, and marks the cvar with 'serverinfo' flag. Value may be composed from multiple tokens.

seta <cvar> <value> […]
Sets the specified cvar to value, and marks the cvar with ‘archive’ flag. Value may be composed from multiple tokens.

cvarlist [-achlmnrstuvw:]
Display the list of registered cvars and their current values with filtering by cvar name or by cvar flags. If no options are given, all cvars are listed. Supported options are reproduced below.

-a | --archive
list archived cvars

-c | --cheat
list cheat protected cvars

-h |--help
display help message

-l | --latched
list latched cvars

-m | --modified
list modified cvars

-n | --noset
list command line cvars

-r | --rom
list read-only cvars

-s | --serverinfo
list serverinfo cvars

-t | --custom
list user-created cvars

-u | --userinfo
list userinfo cvars

-v | --verbose
display flags of each cvar

-w | --wildcard=<string>
list cvars matching wildcard string

macrolist
Display the list of registered macros and their current values.

MESSAGE TRIGGERS

Message triggers provide a form of automatic command execution when some game event occurs. Each trigger is composed from a command string to execute and a match string. When a non-chat message is received from server, a list of message triggers is examined. For each trigger, match is macro expanded and wildcard compared with the message, ignoring any unprintable characters. If the message matches, command is stuffed into the command buffer and executed.

trigger [<command> <match>]
Adds new message trigger. When called without arguments, prints a list of registered triggers.

untrigger [all] | [<command> <match>]
Removes the specified trigger. Specify all to remove all triggers. When called without arguments, prints a list of registered triggers.

CHAT FILTERS
Chat filters allow messages from annoying players to be ignored. Each chat filter is composed from a match string. When a chat message is received from server, a list of chat filters is examined. For each filter, match is wildcard compared with the message, ignoring any unprintable characters. If the message matches, it is silently dropped.

ignoretext [match …]
Adds new chat filter. When called without arguments, prints a list of registered filters.

unignoretext [all] | [match …]
Removes the specified chat filter. Specify all to remove all filters. When called without arguments, prints a list of registered filters.

ignorenick [nickname]
Automatically composes and adds two chat filters: ‘nickname: *’ and ‘(nickname): *’. This command supports nickname completion. When called without arguments, prints a list of registered filters.

unignorenick [nickname]
Automatically composes and removes two chat filters: ‘nickname: *’ and ‘(nickname): *’. This command supports nickname completion. When called without arguments, prints a list of registered filters.

DRAW OBJECTS

Draw objects provide a uniform way to display values of arbitrary cvars and macros on the game screen. By default, text is positioned relative to the top left corner of the screen, which has coordinates (0, 0). Use negative values to align text to the opposite edge, e.g. point with coordinates (-1, -1) is at the bottom right corner of the screen. Absolute value of each coordinate specifies the distance from the corresponding screen edge, counted in pixels.

draw <name> <x> <y> [color]
Add console variable or macro identified by name (without the ‘$’ prefix) to the list of objects drawn on the screen at position (x, y), drawn in optional color.

undraw [all] | <name>
Remove object identified by name from the list of objects drawn on the screen. Specify all to remove all objects.

Example 3. Drawing FPS and a clock
/draw cl_fps -1 -1 // bottom right
/draw com_time 0 -1 // bottom left

SCREENSHOTS

screenshot [format]
Standard command to take a screenshot. If format argument is given, takes the screenshot in this format. Otherwise, takes in the format specified by ‘gl_screenshot_format’ variable. File name is picked up automatically from the ‘screenshots/quakeNNN.EXT’ template.

screenshotpng [filename] [compression]
Takes the screenshot in PNG format. If filename argument is given, saves the screenshot into ‘screenshots/filename.png’. Otherwise, file name is picked up automatically. If compression argument is given, saves with this compression level. Otherwise, saves with ‘gl_screenshot_compression’ level.

screenshotjpg [filename] [quality]
Takes the screenshot in JPG format. If filename argument is given, saves the screenshot into 'screenshots/filename.jpg'. Otherwise, file name is picked up automatically. If quality argument is given, saves with this quality level. Otherwise, saves with 'gl_screenshot_quality' level.

screenshottga [filename]
Takes the screenshot in TGA format. If filename argument is given, saves the screenshot into 'screenshots/filename.tga'. Otherwise, file name is picked up automatically.

LOCATIONS

loc_add <name …>
Adds new location with the specified name at current player position.

loc_delete
Deletes location closest to player position.

loc_update <name …>
Changes name of location closest to player position.

loc_write
Saves current location list into 'locs/<mapname>.loc' file.

Note
Edit locations on a local server and don't forget to execute 'loc_write' command once you are finished. Otherwise all changes to location list will be lost on map change or disconnect.

MISCELLANEOUS

vid_restart
Perform complete shutdown and reinitialization of the renderer and video subsystem. Rarely needed.

fs_restart
Flush all media registered by the client (textures, models, sounds, etc), restart the file system and reload the current level.

r_reload
Flush and reload all media registered by the renderer (textures and models). Weaker form of 'fs_restart'.

Tip

In Q2PRO, you don't have to issue 'vid_restart' after changing most of the settings, a 'fs_restart' or 'r_reload' usually suffice. This helps to avoid main window recreation and changing video modes back and forth, and is much faster.

PASSIVE

Toggle passive connection mode. When enabled, client waits for the first 'passive_connect' packet from server and starts usual connection procedure once this packet is received. This command is useful for connecting to servers behind NATs or firewalls. See 'pickclient' command for more details.

serverstatus [address]

Request the status string from the server at specified address, display server info and list of players sorted by frags. If connected to the server, address may be omitted, in this case current server is queried.

followip [count]

Attempts to connect to the IP address recently seen in chat messages. Optional count argument specifies how far to go back in message history (it should be positive integer). If count is omitted, then the most recent IP address is used.

Incompatibilities

Q2PRO client tries to be compatible with other Quake 2 ports, including original Quake 2 release. Compatibility, however, is defined in terms of full file format and network protocol compatibility. Q2PRO is not meant to be a direct replacement of your regular Quake 2 client. Some features are implemented differently in Q2PRO, some may be not implemented at all. You may need to review your config and adapt it for Q2PRO. This section tries to document most of these incompatibilities so that when something doesn't work as it used to be you know where to look. The following list may be incomplete.

Q2PRO has a built-in renderer and doesn't support run-time loading of external renderers. Thus, 'vid_ref' cvar has been made read-only and exists only for informational purpose.

Default value of 'intensity' variable has been changed from 2 to 1. This means textures will appear darker by default.

Default value of 'gl_dynamic' variable has been changed from 1 to 2. This means dynamic lights will be disabled by default.

Changes to ‘gl_modulate’ variable in Q2PRO take effect immediately. To set separate modulation factors for world lightmaps and entities please use ‘gl_modulate_world’ and ‘gl_modulate_entities’ variables.

Default value of R1GL-specific ‘gl_dlight_falloff’ variable has been changed from 0 to 1.

‘gl_particle_*’ series of variables are gone, as well as ‘gl_ext_pointparameters’ and R1GL-specific ‘gl_ext_point_sprite’. For controlling size of particles, which are always drawn as textured triangles, Q2PRO supports it’s own ‘gl_partscale’ variable.

‘ip’ variable has been renamed to ‘net_ip’.

‘clientport’ variable has been renamed to ‘net_clientport’, and ‘ip_clientport’ alias is no longer supported.

‘demomap’ command has been removed in favor of ‘demo’ and ‘mvdplay’.

Q2PRO works only with virtual paths constrained to the quake file system. All paths are normalized before use so that it is impossible to go past virtual filesystem root using ‘../’ components. This means commands like these are equivalent and all reference the same file: ‘exec ../global.cfg’, ‘exec /global.cfg’, ‘exec global.cfg’. If you have any config files in your Quake 2 directory root, you should consider moving them into ‘baseq2/’ to make them accessible.

Likewise, ‘link’ command syntax has been changed to work with virtual paths constrained to the quake file system. All arguments to ‘link’ are normalized.

Cinematics are not supported.

Joysticks are not supported.

Single player savegame format has been rewritten from scratch for better robustness and portability. Only the ‘baseq2’ game library included in Q2PRO distribution has been converted to use the new improved savegame format. Q2PRO will refuse to load and save games in old format for security reasons.

CD music is not supported.

1. This is not the framerate ‘cl_maxfps’ limits. Think of it as an input polling frame rate, or a ‘master’ framerate.
2. Can be also called ‘physics’ frame rate. This is what ‘cl_maxfps’ limits.