Since Quake 3, the function that's responsible for game physics (amongst other things) is named Pmove(). Normally, this function is run once every client frame, and this is where framerate-dependant physics (most notably jump heights) come from. pmove_fixed is an attempt to level the playing field by running this function at a fixed rate (every pmove_msec milliseconds.) The default pmove_msec of 8 is equivalent to normal physics at 1000 / 8 = 125 frames/second.

pmove_fixed can be set on either the client or the server--the client will default to the value on the server (And, on non-ETPro servers, it will be reset to the value of the server quite frequently due to some silliness in the code.) pmove_msec can only be set on the server side.

note: setting pmove_fixed on etpro can still have some undesired side effects. In particular, there is still a bug when using weapons that recoil which makes it very difficult to hit your target.

b_fixedphysics is a server-side setting with similar goals to pmove_fixed. b_fixedphysics 1 avoids rounding the velocity at all (Removing the framerate-dependent behavior in movement); however, it offers b_fixedphysicsfps to adjust the jump velocity to emulate the old framerate-dependent behavior. This is the best of both worlds—no movement speed problems (as several people observed at 333fps) nor any of the problems that pmove_fixed brings with it (some parts of the game behave poorly at high fps, most notably mounted MG42 and mortar aiming), but the same jump heights trickjumpers would kill the etpro team for taking away. b_fixedphysics 2 is a cross between normal behavior and fixed behavior, which effectively just caps the maximum framerate-dependent behavior to 166fps.

b_optimizeprediction is a client-side setting that (theoretically) has no effect other than increasing performance. Basically, etmain (and RTCW, and Q3) will re-do all the physics computations for up to the previous 64 frames in order to figure out where you should be for the current one. b_optimizeprediction will store the result of the previous computations and reuse them, although only if the results look acceptable based on the latest data the client has from the server, instead of doing all the math again. This provides a rather dramatic performance boost, especially if you have a high ping.