

WARBIRDS COMMANDS REFERENCE

Version 1.00
May, 2003

Section I: Dot Commands

Part I: Front End Dot Commands

Format of listings:

DOTCOMMAND <arguments> Description *modes/builds*
modes/builds: *all* = available in all builds, online and offline
offline = available in release builds offline only
online = available in release builds online only
cmtool = available in cmtool builds only
internal = available in internal builds only
edittools = available in edit tools build
flightdata = available in flight data build
macintosh = mac only

.%%<script text> %% Marks beginning, end of embedded **SCRIPT** text. Text in between is submitted to the script parser. See **Script Specification** below. *offline,cmtool*
@<id>@ Inline expansion of global named variable <id>. *offline/cmtool*
.ACMPPLAYBACK/.ACMPPLAY <acmcam file> [LOOP] Start playback of a acmcam file. If optional LOOP arg, given, repeat playback indefinitely. *offline*
.ACMREC [<acmcam file>] Start acmcam recording to <acmcam file>. If <acmcam file> is blank, stops current recording. *online*
.ACMRECALL [<acmcam file>] Same as **.ACMREC**, accept all arena packets are recorded, including radio messages. Requires MODESET 7 (CM privileges). *online*
.AILIST Lists drone units. *online*
.AIRCRAFTLOD <distance> Sets maximum <distance> at which full LOD rendering of vehicles is applied. *all*
.APPENDTAB
.ARNAFLAGS Displays arena flags (ARNAFLAGS,ARNAFLAGS2), showing bit values and a short description. *online*
.ATTACH <drone> Attach to <drone>. *offline*
.ATTACHMO <mission> <slot> <station> For offline attached flight, sets player to be attached as gunner <station> on drone at <slot> of drone mission <mission>. *offline*
.AUTOARENA <arena description> Sets automatic login to the arena identified by <arena description>. *all*
.AUTOPANRATE <degrees> Set auto pan rate to <degrees> per second. *internal*
.AUTORELOG [<wait time>] If disconnected, attempt relog every <wait time> seconds. If <wait time> is blank, disable autorelog. *cmtool*
.AUTOTAKEOFF Sets flight to automatically takeoff if starting on runway. *all*
.AUTOTARGET Automatically triggers LCOS on first nme drone in range. *offline*
.AUTOTARGETOFF Disables AUTOTARGET trigger. *offline*
.AXISCAL *macintosh*
.BGLODRENDERRANGE
.BGMINRENDERRANGE/MINRENDER <distance> Sets the distance at which other vehicles are rendered using minimal drawing (aka green-headed fly).
.BINARYEFFECTS
.BINARYTERRAIN <path> Use binary file <path> for terrain. *internal*
.BITADD <n> Accumulator += <n>. *offline/cmtool*
.BITAND <n> Accumulator &= <n>. *offline/cmtool*
.BITNAND <n> Accumulator &= ~<n>. *offline/cmtool*
.BITOR <n> Accumulator |= <n>. *offline/cmtool*
.BITRES Displays accumulator in decimal. *offline/cmtool*

.BITRESHEX Displays accumulator in hex. *offline/cmtool*

.BITSL <n> Accumulator <<= <n>. *offline/cmtool*

.BITSR <n> Accumulator >>= <n>. *offline/cmtool*

.BITSTO <n> Accumulator = <n>. *offline/cmtool*

.BITSUB <n> Accumulator -= <n>. *offline/cmtool*

.BITXOR <n> Accumulator ^= <n>. *offline/cmtool*

.BLOWMEUP Explode player's vehicle. *internal*

.BRAKEPOWER <value> Sets initial brake power to <value>. *all*

.CARRIERBOUNDS Toggle display of carrier landing deck bounds. *internal*

.CCIP+ Turns on HUD CCIP (Continuously Computed Impact Point) indicator. *offline*

.CCIP- Turns off CCIP indicator. *offline*

.CHECKSIX Plays current "check six" sound file. *all*

.CHOOSETAB

.CLEARMO/.CLRMO Clears all drone missions. *offline,cmtool*

.CLEARMO_PLAYER Clears the special player drone mission. *offline*

.CLOUDS 1|0 <alt> [thickness] [transition] [type] [layerId] Turns display of clouds on/off and allows the appearance of the clouds to change. Alt, thickness and transition set characteristics of the cloud layer. Type is actually the material number, i.e. 1 = cloud1m, 2 = cloud2m, etc... LayerId is just an identifier used if you want to modify the specific cloud layer later. *Offline, cmtool*

.CLUTTERDISTANCE <distance> Sets the distance at which clutter stops being computed. *all*

.CLUTTERLODDISTANCE <distance> Sets the distance at which clutter changes between high LOD and low LOD.

.CLUTTERSCALEFACTOR <factor> Sets the multiplier for increasing size of clutter as distance from the viewpoint increases. *internal or Edittools*

.CLUTTERSCALECAP <range> Sets the distance where clutter size reaches the scale factor size. Clutter stops growing at this distance. *internal or Edittools*

.CLUTTERMULT <value> Sets the number of clutter objects per clutter map location. *internal or Edittools*

.CMAPPID Append prefix "CM" to appid, allows multiple logins by cmtool generals. *cmtool*

.CMEYE Enters CMEYE Mode. *all*

.CMLIST When in CMEYE mode, lists vehicles being tracked. *all*

.CONV <distance> Sets convergence for primary weapons (guns). *all*

.CONVCANNON <distance> Sets convergence for secondary weapons (cannons). *all*

.COUNTRY 1|2|3|4 Sets player's country. *all*

.CPU Displays CPU type selected for V3D optimizations

.DATAIT <alt> Sets altitude for flight data mode. *Internal,flightdata*

.DATE <month> <day> <year> Sets offline date to <month>,<day>,<year>. *offline*

.DATWEIGHT <weight> Sets weight for flight data mode. *internal*

.DELAY <milliseconds> Sets salvoed bomb release rate to <milliseconds>(50-1000). *all*

.DESTROYALL <field> Destroys all ground objects associated with field. *offline,cmtool*

.DESTROYOB <id> Destroys ground object <id>. *offline,cmtool*

.DETAILDENSITY

.DIRMO <directory> Lists .mbl files (drone mission files) in <directory>. *offline,cmtool*

.DISPLAYFILE <path> [<title>] [<left> <top> <bottom> <right>] [<dotfilepath>][1|0]

.DISPLAYFILEX "<path>" [<title>] [<left> <top> <bottom> <right>] [<dotfilepath>][1|0]

.DISPLAYIMAGE

.DISPLAYLBOX

.DLSITEURL

.DLTERRAINSITEURL

.DMGRADIUS <radius> Sets the radius used to compute the size of the Hit Map. *Internal or EditTools*

.DOTFILE <path> Executes dot commands in file <path>. *offline,cmtool*

.DOTFILEEXIT <path> [ALL | LIVED | DIED | EXITED | LANDED | CRASHED | DITCHED | KILLED | BAILED | BADCHUTE | EXPLODED | GSEXCEEDED | HITOBJECT | HITAPLANE]
Sets up a dotfile <path> to be executed upon player exiting flight, based on the condition. If no condition given, clears all exit dotfiles. *offline*

.DRAWOPT Toggles optimized drawing/sounds for cmtool. When on, most sound/3d rendering calls will be disabled, freeing frame rate/cpu time for AI logic. *cmtool,internal*

.DTF_PERFRAME [<path>] Sets dotfile <path> to be run each frame (once per game loop). If <path> not given, clears any previously set. *offline/cmtool*

.DTF_TEXT "<path>" "<text>" (Quotes required) Sets up a host-text triggered dot file <path>. Dot file will be executed on the receipt of radio message <text> from the host. *cmtool*

.DTF_TEXTCLEAR ["<path>"] ["<text>"] Clears host-text triggered dot file <path>, host <text>. If <text> is blank, clears all host-text triggered dot files <path>. If <path> is blank, clears all triggers of <text>. If both <path> and <text> are blank, clears all host-text triggered dot files. *cmtool*

.DTF_TEXTLIST Lists all active host-text triggered dot files. *cmtool*

.DTF_TIMED <path> <time interval> [<repeat count>] Sets up a dotfile at <path> to be run every <time interval> seconds. <repeat count> can be blank (= one time), -1 = run indefinitely, or the number of times to repeat. *offline,cmtool*

.DTF_TIMEDCLEAR <path> Removes timed dotfile <path>. If <path> is blank, removes all timed dotfiles. *offline,cmtool*

.DTF_TIMEDLIST/DTF_LIST Lists all active timed dotfiles, and the per-frame dotfile if set. *offline,cmtool*

.E Exit flight. *all*

.EASYPLIMIT

.ECHO <text> Displays <text> in message buffer. In cmtool builds, also broadcasts message on currently selected radio channel. *all*

.ECHOFILE [<path>] Redirects all **.echo** output to file <path>. If <path> not given, cancels any existing redirection. *all*

.EDITGROUND Start ground-object layout editor. *internal,edittools*

.EDITROADS Start road layout editor. *internal,edittools*

.EDITTERRAIN Start terrain texture editor. *internal,edittools*

.EFFECTSLOAD Reloads the effects.wb3 and ef3.wwm files and re-initializes the effects system. *internal*

.ENABLE 1|0 <plane number> Enables/disables selection of <plane number> offline. If <plane number> is -1, enables/disables all plane types. *offline*

.ENVMAP

.EXIT Log-off from host. *all*

.EXPORTROADS [1] Exports the roads.rds file (binary roads.rdb if parameter == 1). *edittools,internal*

.EXPORTTERRAIN <path> Export current terrain to <path>. *internal*

.EXPORTTIME

.EXPOSE

.EZ 0|1|2 Sets flight mode 0=real,1=easy,2=relaxed realism. *all*

.FARSMOOTH <time> Set far-packet smoothing time. *internal*

.FFBOMBRELEASEENABLE 1|0

.FFBOMBRELEASEGAIN 1|0

.FFCANNONENABLE 1|0

.FFCANNONGAIN 1|0

.FFDAMAGEENABLE 1|0

.FFDAMAGEGAIN 1|0

.FFHIGHSPEEDBUFFETENABLE 1|0

.FFHIGHSPEEDBUFFETGAIN 1|0

.FFHITBUFFETENABLE 1|0

.FFHITBUFFETGAIN 1|0

.FFHUD

.FFINPUT <pitch> <roll>

.FFMACHINEGUNENABLE 1|0

.FFMACHINEGUNGAIN 1|0

.FFROCKETRELEASEENABLE 1|0

.FFROCKETRELEASEGAIN 1|0

.FFSTALLBUFFETENABLE 1|0

.FFSTALLBUFFETGAIN 1|0

.FFTRIMFORCEENABLE 1|0

.FFTRIMFORCEGAIN 1|0

.FIELD <n> Move to Field <n>. *all*

.FLAKYTIMEOUT <time> Set time threshold for flakey connection status. *internal*

.FLDOVERRIDE

.FLEXPOR Exports flight data to fl.bin. *internal*

.FLIGHTDATA Toggles mode for display of flight data. *internal,flightdata*

.FLIGHTSTART <x> <y> <alt> <heading> Set start of flight at x,y, alt and heading. *offline*
.FLITECOM <command> Sends wingmen command <command> on channel 114. *offline*
.FLUSHPLANES Removes all graphical vehicle data from memory. *all*
.FLY Start flight. *all*
.FOGDISABLE Turns off fog. *internal*
.FOGENABLE Turns on fog. *internal*
.FOGFAR Sets the distance at which fogging starts. *internal*
.FOGNEAR Sets the distance at which objects are completely fogged. *internal*
.FONTSIZE
.FORCEDENSITY
.FORCEFEEDBACK
.FORCEMODE REAL|ARCADE|RELAXED Force next flight to be in <mode>. *offline*
.FOV <degree> Set FOV to <degree>, 35-116. *all*
.FRAMEMODE 1|2|1701|1703 Various ways of displaying per frame information. *all*
.FUEL <percent> Set fuel load to <percent> (1-100). *all*
.GACCEPT <handle>|ALL Accept <handle> or ALL as gunner. *online*
.GAMELOOP <n> Executes internal gameloop <n> times. *offline*
.GAMMA
.GETLOADOUT [<plane type>] Get loadout data from mods dir for all | <planetype>. *internal*
.GOLDCOUNTRYCOLOR
.GREENCOUNTRYCOLOR
.GREJECT <handle>|ALL Reject <handle> or ALL as gunner.
.GRNDCOLLIDES Display collision bounds on ground objects. *internal*
.GRNDLABELS Toggle display of ground object id/type icons. *all*
.GROUNDDETAIL
.GUIDES+ Enables HUD waypoint guides. *offline*
.GUIDES- Disables HUD waypoint guides. *offline*
.GUN <handle> TAIL | NOSE | RIGHT | LEFT | TOP | BOTTOM Request gunner position on <handle>. *online*
.GUNNERSAFE Sets all player's gunners on safe. *offline*
.GUNSIGHTOFFSET <x> <y> <z> Offset gunsight by x,y,z (ft). Must be issued prior to loading plane data. *all*
.HBTRANSPARENCYCOLOR
.HEADING <degrees> Sets compass heading indicator to <degrees>. *all*
.HEARALLEXPL Toggles on/off hearing all explosions/detonations in arena or just those in range of player. *online*
.HITMAP [id] Toggle display of hitmap bounds. If id is omitted, then all damage ids are toggled. If the id is set, then only that damage id is displayed. *internal*
.HITSOFF Disable sending hits to host. *internal*
.HITSON Enable sending hits to host. *internal*
.HL <handle> Hilites player <handle>'s icon. *online*
.HPCURVE [<plane type>] Get HPCurve data from mods dir for all | <planetype>. *internal*
.HUD 1|0 Enable/disable in-flight HUD. *all*
.HURTDRONE <damagepart> Inflict <damagepart> on current chase drone See Damage Parts. *cmttool*
.HURTME <damagepart> Inflict <damagepart> on player's plane See Damage Parts below. *all*
.ICON_NAMES Sets offline icons to names only. *Offline*
.ICON_NORMAL Sets offline icons to normal. *offline*
.ICON_OFF Hides offline icons. *offline*
.ICON_PLANES Sets offline icons to planes only. *Offline*
.ICON_RANGE Sets offline icons to range only. *offline*
.IDLELOOP <numloops> Execute internal idle loop <numloops> times. *offline*
.IGNORE <handle> Ignore radio messages from <handle>. *online*
.IMPORTASCI Load plane parameters from mods dir for all planes. *internal*
.INFOMO <mission> Prints information about <mission>. *offline,cmttool*
.INTADD <id> <n> <id> += <n>. *offline,cmttool*
.INTAND <id> <n> <id> &= <n>. *offline,cmttool*
.INTDIV <id> <n> <id> /= <n>. *offline,cmttool*
.INTMUL <id> <n> <id> *= <n>. *offline,cmttool*

.INTNAND <id> <n> <id> &= ~<n>. *offline,cmtool*

.INTOR <id> <n> <id> |= <n>. *offline,cmtool*

.INTPER <id> <n0> <n1> <id> = <n0>/<n1> * 100. *offline,cmtool*

.INTSL <id> <n> <id> >> <n>. *offline,cmtool*

.INTSR <id> <n> <id> << <n>. *offline,cmtool*

.INTSTO <id> <n> Assigns <n> to global named variable <id>. <id> created if not already existing. *offline,cmtool*

.INTSUB <id> <n> <id> -= <n>. *offline,cmtool*

.INTRIGGER <id> <min> <max> <dotfile path> Sets up a one-time trigger which executes commands in <dotfile path> when global named variable <id> is within <min>,<max> inclusive. <id> must exist prior to setting of trigger. *offline,cmtool*

.INTRIGGER_CLEAR [<id>] Clears trigger on <id>. If <id> not given, clears all triggers. *offline,cmtool*

.INTRIGGER_LIST Lists all current triggers. *offline,cmtool*

.INTRES [<id>] displays int val of <id>. If <id> not given, displays all global named variables. *offline,cmtool*

.INTXOR <id> <n> <id> ^= <n>. *offline,cmtool*

.JOYCENTER_ROLL

.JOYCENTER_YAW

.JOYCONTROL_BRAKES

.JOYCONTROL_PITCH

.JOYCONTROL_ROLL

.JOYCONTROL_THROTTLE

.JOYCONTROL_YAW

.JOYDAMPEN_BRAKES

.JOYDAMPEN_PITCH

.JOYDAMPEN_ROLL

.JOYDAMPEN_THROTTLE

.JOYDAMPEN_YAW

.JOYDEADBAND_BRAKES

.JOYDEADBAND_PITCH

.JOYDEADBAND_ROLL

.JOYDEADBAND_THROTTLE

.JOYDEADBAND_YAW

.JOYINVERT_BRAKES

.JOYINVERT_PITCH

.JOYINVERT_ROLL

.JOYINVERT_THROTTLE

.JOYINVERT_YAW

.JOYSHAPE_BRAKES

.JOYSHAPE_PITCH

.JOYSHAPE_ROLL

.JOYSHAPE_THROTTLE

.JOYSHAPE_YAW

.JUMP OBS | COCKPIT | TAIL | NOSE | RIGHT | LEFT | TOP | BOTTOM Jump to gunners position. *all*

.KB

.KEYHELP

.KILLGRUNTS Kill all net-launched and player-launched grunts. *internal*

.KILLMO/.OFFLINEMO <mission> Takes <mission> off of run que. *offline,cmtool*

.KILLTABS

.LANDCOLOR <r g b> Sets the lighting value used when rendering the terrain. Used for Time-of-day lighting effects. *internal*

.LANDERHUD <field> <runway> <turnpoint distance> <turnpoint radius> <glideslope> <approach speed> <descent speed> <approach gear state> <approach flaps position> <descent flaps position> Sets up a HUD-prompted landing at <field>,<runway>. *offline*

.LEAN <side> <height> Set cockpit lean. *internal*

.LEFTCLICKTOTUNERADIO

.LIGHTCOLOR <r g b> Sets the lighting value used when rendering objects. Used for Time-of-day lighting effects. *all*

.LINDABLAIR Toggles ability to pan completely to full 6 view. *all*

.LISTEN <handle> Restore receipt of radio messages from <handle>. *online*

.LISTMO Lists info on loaded drone missions. *offline,cmtool*

.LOADASCII <plane type> Load plane parameters from mods dir for <planetype>. *internal*

.LOADGAMESTATE <path> Loads ground objects'/fields' "state" (color,damage), offline stats to <path>. *offline*

.LOADMO/.LDMO <mission> Loads drone mission file <mission> and puts it on the run que (equivalent to .loadmo_online). *offline,cmtool*

.LOADMODIR/.LDMODIR <directory> Loads *.mbl (drone mission files) in <directory>, placing them on the run que. *offline,cmtool*

.LOADMODIR_OFFLINE/.LDMODIR_OFFLINE <directory> Loads *.mbl (drone mission files) in <directory>, but does not put them on the run que. *offline,cmtool*

.LOADMO_OFFLINE/.LDMO_OFFLINE <mission> Loads drone mission, but does not place it on the run que. *offline,cmtool*

.LOADMO_PLAYER/.LDMO_PLAYER <mission> Loads a drone mission specially to include the player as part of the mission. *offline*

.LOADOUT <n> Set loadout to <n>. *all*

.LODRENDER

.LOGIN Brings up login dialog. *offline*

.LOSTTIMEOUT <time> Set time threshold for lost connection status. *Internal*

.MACROCHAR <c> Changes global named variables expansion character (default '@') to <c>. *offline,cmtool*

.MAKEDIR <path> Creates directory <path>. *all*

.MAKELBOX

.MASTerview Slaves all attached players' views. *online*

.MERGEROADCLUTTER Modifies the clutter map to keep clutter off of the roads. AS3 only. *internal*

.MIPBIAS <value> Modifies the value of a given computed mip level. For example, if the MIPBIAS value is set to -2 and the graphics card calculates that the mip level should be 4, then the mip level rendered will actually be 2 (4 - 2). *all*

.MOUNTMO <mission> <slot> For offline carrier practice, sets player to start flight mounted on drone at <slot> of drone mission <mission>. Drone must be assigned a mountable vehicle (KAGA,CV6). *offline*

.MULTIPASS 0|1 Sets multipass terrain rendering off/on. *all*

.NAV <heading> <speed> Have drone at <slot> assume <heading> and <speed> *online*

.NEARFARPLANE <near> <far> Sets the values for the near and far planes. Changes take affect only after using SETFOV dot command. *internal*

.NOFLYCFG Disables reading fly2.cfg before flight. *offline*

.NUMPADFREE

.NUMPADROTATE

.OBJDOTFILE_DEAD <id> <path> Executes the dot commands in file <path> when ground object <id> is destroyed. *offline,cmtool*

.OBJHARDNESS <id> <hardness> Sets ground object <id> hardness to <hardness>. *offline*

.OCEANMOD <frame> sets the "watermode 0" style ocean to update every "frame" number of frames.

.OFFAIRATTACK Sets offline game to air attack. *offline*

.OFFARENAICONS <planerangelim> <enemyidlim> <friendlyidlim>

ENEMY_PLANES|ENEMY_NAMES Sets up icon ranges and modes for offline play. *offline*

.OFFBOMBERAMBUSH Sets offline game to bomber ambush. *offline*

.OFFBOMBERMODEL <plane> Sets plane type for bomber ambush game. *offline*

.OFFBOMBERS <number of bombers> Sets number of bombers in bomber ambush game(1-8). *offline*

.OFFCMTOOL Sets offline game to a drone mission. *offline*

.OFFFREEFLIGHT Sets offline game to free flight. *offline*

.OFFGAMELEVEL 1|2|3 Sets difficulty level of offline games. *offline*

.OFFGROUNDVEHICLES Sets offline game to ground attack. *offline*

.OFFLANDING

.OFFLINEBLACKOUTS [1|0] Sets offline blackouts on/off. *offline*

.OFFLINEINVULNERABLE [1|0] Sets offline invulnerable on/off. *offline*

.OFFLINESETCOCKPIT 1|0 Turns on/off cockpit. *offline*

.OFFLINESTRUCTLIMITS [1|0] Sets offline structural limits on/off. *offline*

.OFFLINEUNLIMITEDAMMO [1|0] Sets offline unlimited ammo on/off. *offline*

.OFFRPS [SHOW [ALL|SELECTED]] [ADD <plane>] [NME <ndrones>] Controls plane types and numbers flown by opposing drones in the Air Attack offline game. *offline*

.OFFSCORE 1|0 Enables/disables display of offline score. *offline*

.OFFSCOREINIT Resets offline scores. *offline*

.OFFTARGETDRONES Sets offline game to target drones. *offline*

.OFFTIME <hour> <minute> Sets offline time to <hour>,<minute>. *offline*

.OFFWINGMEN <number of wingmen> Sets number of wingmen (if online, in conjunction with host setting). *offline (online)*

.OTTO <otto cmd> Issue <otto cmd> to online gunners (see OTTO COMMANDS) below. *all*

.PARSEDTERRAIN <path> Use ascii file <path> for terrain. *internal*

.PCLOUDS 1|0 <minalt> [maxalt] [width] [height] [depth] [density] Turns display of puffy clouds on/off and allows the appearance of the clouds to change. Density (int percent) controls how many clouds are created. *Offline, cmtool*

.PCLOUDVARS <clumps> <particles per clump> <particle scale>. *Offline, cmtool*

.PINGTEST Measures round-trip message timing through host. *online*

.PITCHLADDER 1|0 Enable or disable display of pitch ladder. *all*

.PKREADLEN Display packet-to-buffer lengths. *internal*

.PLANE <planenumber> Select vehicle <planenumber>. *all*

.PLANESKIN

.PLAYSOUNDFILE

.POS <x> <y> <z> Set position to x,y,z. *internal*

.PRELOAD <planenumber> | <planetype> Preloads plane data. *all*

.PRINTTRAJ Record ballistic/penetration data. *internal*

.PRIVATEARENA <arena code> Displays private list of arenas in online dialog. *all*

.PROMPT

.PURPLECOUNTRYCOLOR

.RADIO <number> <channel> | <handle> Tunes radio <number> to <channel> or player <handle>. *all*

.RADIOLINES <n> Sets <n> radio lines displayed. *all*

.RADIOWIDTH <n> Sets width of radio lines displayed to <n> characters

.RANDOM_WIND <duration> <period> <xMin> <xMax> <yMin> <yMax> <zMin> <zMax> Sets up random wind. *offline,cmtool*

.REDCOUNTRYCOLOR

.RELOADOCEAN Reloads the “watermode 0” style water data.

.RELOADTERRSHADERS Reloads the terrain vertex and pixel shaders.

.RELOAD_TERRAIN Force reload of terrain. *Internal*

.REMOVECLOUDS <layerId> Removes previously created cloud layer. If “layerId” is not specified, will remove all cloud layers.

.REMOVETAB

.RENDER Toggles whether rendering is performed or not. *all*

.RENDERFORCEPOINTS Toggles rendering of vehicle force points. *internal*

.RESPECTSELECTABLE/SELECTABLEON Honors .ENABLE. *offline*

.RESTORE_FIELD <field> Restores all ground objects at field <field>. *offline*

.RESTORE_FIELDS Restores all ground objects at all fields. *offline*

.RETURN [<path>] Causes an immediate return (stops executing commands) from within a dotfile, if one is being executed. If <path> is given, executes dotfile <path>. *offline/cmtool*

.RETURNSNAP Panned snap views return to forward on key release. *All*

.ROSTER/ROS Displays a list of players in the arena. *online*

.ROT <pitch> <roll> <heading> Set orientation to pitch,roll,heading. *internal*

.RUDDERDIFFBRAKE

.RUDDERREPEATKEY

.RUNWAY <n> Select runway <n>. *all*

.SALVO <n> Sets salvo to <n>. *all*

.SAVEGAMESTATE <path> Saves ground objects’/fields’ “state” (color,damage), offline stats to <path>. *offline*

.SAVEMAP Saves terrain as .raw file. *internal*

.SCRIPT <path> Submits file <path> to the script parser (if,else,switch). See **Script Specification** below.

offline,cmtool

.SELECTABLEOFF Ignores **.ENABLE**. *offline*

.SENDDIR <dir> Send all files in dir across selected radio channel. *online*

.SENDFILE <path> Send file <path> across selected radio channel. *online*

.SENDFILE+ <path> Same as **.SENDFILE**, with addition of displaying the file on recipient's FEs. *online*

.SETAPPID

.SETAUTOLOGIN

.SETCOCKPIT

.SETFIELD <field> <country> Sets all ground objects associated with <field> to <country>. *offline,cmtool*

.SETGAMETYPE

.SETHOST

.SETJOYBUTTON

.SETJOYHAT

.SETKEY

.SETPASS

.SETSAVEPASS

.SETUSER

.SETWAYPOINT <x> <y> <z> <speed> Enables HUD waypoint indicator, setting waypoint to coords x,y,z with speed <speed> (mph). *offline*

.SHADOWS 0|1|2|3 Sets shadow level, 0=off,1=vehicles,2=ground objects,3=vehicles and ground objects. *all*

.SHANGHAI <handle> Force player <handle> to attach as observer. *online*

.SHIFT <gear> In ground vehicle, set gear to <gear>. *all*

.SHOWGROUNDMENUS

.SHOWHELPBAR

.SHOWRADIOBUTTONS

.SHOWSTICKSET

.SKINFILE

.SLEWFEET <ft> Move <ft> per frame in slew mode. *all*

.SLEWRADS <radians> Rotate <radians> per frame in slew mode. *all*

.SLIP 1|0 Enable/disable HUD slip indicator. *all*

.SMOKE <interval> Toggles airshow smoke, with <interval> between puffs. *all*

.SPEED <climb speed> Sets climb speed for auto-trim climb. *all*

.STARTINGALT <alt> Starts offline flight at <alt> (ft). *offline*

.STARTMO/.ONLINEMO <mission> Puts <mission> on run que. *offline,cmtool*

.STARTSOUND

.STICKYSNAP Panned snap views "stick". *all*

.STOPSOUND

.STORM 0|1|2 Sets type of storm that is active (0-off,1-rain,2-snow). *Internal*

.STRSTO <id> "<text>" Assigns "<text>" (quotes required) to named global variable <id>. <id> created if not already existing. Integer value of <id> is the length of the text. *offline,cmtool*

.SWITCHKEYSET

.TAKEOFFHUD <field> <runway> <speed> <flaps position> 1|0 Sets up a HUD-prompted takeoff at field, runway, takeoff speed, flaps position, wep state. *offline*

.TERRAIN <terrain> Set current terrain to <terrain>. *offline*

.TERRAINSCALE <scale> Set terrain scale. *internal*

.TERRAINTEXTQUALITY 64|128|256 Sets the texture density of the terrain textures. Terrain must be reloaded after setting this value. WB3 only. *all*

.TERRAIN_LOAD <terrain> Force load of <terrain>. *offline*

.TEXSIZE 0|1|2|3 Sets the maximum texture size (0-2048,1-512,2-256,3-128). Only affects textures loaded after the value is set. *all*

.TIDLELOOP <milliseconds> Execute internal idle loop for <milliseconds> duration. *offline*

.TOGGLEKEYSET

.TOGLLEROADCLUTTER Toggles whether or not road clutter is drawn. Also, when enabled, causes road path to be drawn on the map display. *internal*

.TOGLLEROADS Toggles on/off road rendering. *all*

.TRANSPARENCYCOLOR

.TRIMREPEATKEY

.TURB <value> Does some kind of modification of the sky colors. *all*

.UNATTACH Clear attach. *Offline*

.UNLOADMO <mission> UnLoads <mission>. *offline,cmtool*

.VARLEVEL <2000-100000> Sets the target number of terrain triangles you want to draw in a given frame. Higher number for higher detail, lower number for lower detail. *all*

.VARSAVE <path> Saves all named global variables to file <path>. *offline,cmtool*

.VARREAD <path> Reads named global variables from file <path>. *offline,cmtool*

.VARFREE [<id>] Deletes named global variable <id>. If <id> not given, deletes all named global variables. *offline,cmtool*

.VARGETDISPLAY <id> <path> [“<window title>”] [<x>] [<y>] [<width>] [<height>] Modal dialog which displays file <path> (like DISPLAYFILE) and assigns to global variable <id> 1 if OK was pressed, 0 if CANCEL was pressed. *offline*

.VARGETLBOX <id> <path> [“<default item>”] [“<window title>”] [<width>] [<x>] [<y>] Modal dialog which displays a list of choices from file <path> (each line = 1 choice) and assigns to global variable <id> 0 if cancel was pressed or 1..n representing item chosen. *offline*

.VARGETINT <id> “<prompt message>” [“<window title>”] [<bitmap>] [<x>] [<y>] [<width>] [<height>] Modal dialog which prompts user for a number using “<prompt message>” and assigns it to global variable <id>. *offline*

.VARGETSTRING <id> “<prompt message>” [“<window title>”] [<bitmap>] [<x>] [<y>] [<width>] [<height>] Modal dialog which prompts user for text using “<prompt message>” and assigns it to global variable <id>. *offline*

.VARGETYESNO <id> “<prompt message>” [“<window title>”] [<bitmap>] [<x>] [<y>] [<width>] [<height>] Modal dialog which prompts user for a yes/no response using “<prompt message>” and assigns 1=yes or 0=no to global variable <id>. *Offline*

.VARHANDLEKEY <dotfile> Specifies a dotfile to run on single-key press. Access to the key pressed within dotfile is through global named variables :KEYPRESS =string char of key, :KEYPRESS_ASC =asc code of key, :KEYPRESS_MODS =modflags of key. Setting global named variable \$KEYHANDLED\$ to 1 within dotfile disables further processing of key press. *offline*

.VIDEOPRESET

.VIRTUALMODE 0|1|2 Sets virtual cockpit movement mode. 0=Instant snaps, 1=panned snaps, 2=glimited panned snaps. *all*

.VOL

.VOLBOMB

.VOLBOMBAY

.VOLCANNON

.VOLCHECKSIX

.VOLDAMAGE

.VOLDEATH

.VOLEJECT

.VOLENGINE1

.VOLENGINE2

.VOLENGINE3

.VOLENGINE4

.VOLEXP1

.VOLEXP2

.VOLEXP3

.VOLFLAK1

.VOLFLAK2

.VOLFLAK3

.VOLFLAPS

.VOLGEAR

.VOLGEARTOUCH

.VOLGEXCEDED

.VOLGROUNDROLL

.VOLGUN

.VOLHIT

.VOLHIT2

.VOLHIT3
.VOLIGNITION
.VOLMASTERVOL
.VOLORDNANCE
.VOLOVERSPD
.VOLROCKET
.VOLSCRAPE
.VOLSCREECH
.VOLSECEXP
.VOLSTALL
.VOLTURRET
.VOLWIND
.WATCHMEM Monitor memory allocation. *internal*
.WATERLEVEL <level> Sets water level to <level>. *internal*
.WAVESOFF Turns off waves for boat-type vehicles. *offline,cmtool*
.WAVESON <height> <len> <speed> Turns on waves, setting the height (ft), length (ft) and speed (ft/sec). Waves effect all boat-type vehicles. *offline,cmtool*
.WAYPOINT_QUES [CLEAR | ALL | HEADING | ALT | SPEED | DIST | PROMPT | WARN] Turns on/off specific HUD waypoint ques. *offline*
.WAYPOINT/.WP <n> Selects waypoint <n> as current waypoint for player's flight. *offline*
.WAYPOINT+/.WP+ Selects next waypoint for player's flight. *offline*
.WAYPOINT++/.WP++ Selects last waypoint for player's flight. *offline*
.WAYPOINT-/.WP- Selects previous waypoint for player's flight. *offline*
.WAYPOINT--/.WP-- Selects first waypoint for player's flight. *offline*
.WBUFFER
.WEAPONHIT_OBJECT <object id> <weapon type> <distance> <weapon id> <velocity> sends WPWEAPONHITOBJECT message to host *internal*
.WEAPONWEIGHT <weight> Add <weight> to weapon weight calc. *internal*
.WIRE Set wireframe mode. *internal*
.YESFLYCFG Enables reading fly2.cfg before flight. *offline*
.YESNODLG

Host dot commands

```

{3,"LOG",0,1,0,dcCMD_LOGON},
{3,"TES",0,1,dcPM_GOD,dcCMD_TEST},
{3,"ROS",1,1,0,dcCMD_ROSTER},
{3,"HAN",0,2,0,dcCMD_HANDLE},
{3,"MOV",0,2,0,dcCMD_MOVE},
{3,"COU",0,2,0,dcCMD_COUNTRY},
{3,"PLA",0,1,0,dcCMD_PLANE},
{3,"FUE",0,2,0,dcCMD_FUEL},
{3,"BOM",0,2,0,dcCMD_BOMBS},
{4,"SSET",1,3,dcPM_CM,dcCMD_SYSTEMSET},
{4,"HELP",0,1,0,dcCMD_HELP},
{4,"SHOW",0,1,0,dcCMD_SHOW},
{3,"ZMD",0,1,0,dcCMD_ZMODEM},
{3,"EJE",1,2,dcPM_CM,dcCMD_EJECT},
{4,"LOCK",0,2,dcPM_CM,dcCMD_LOCK},
{3,"FIE",0,1,0,dcCMD_FIELD},
{4,"LOAD",0,1,dcPM_CM,dcCMD_LOAD_FILES},
{4,"SAVE",0,1,dcPM_CM,dcCMD_SAVE_FILES},
{4,"ENAB",0,2,dcPM_CM,dcCMD_ENABLE},
{3,"DAM",0,1,dcPM_CM,dcCMD_DAMAGE},
{3,"ROC",0,2,0,dcCMD_ROCKETS},
{3,"SND",0,2,dcPM_CM,dcCMD_SOUND},
{3,"SCO",0,1,0,dcCMD_SCORE},
{3,"SQU",0,2,0,dcCMD_SQUADRON},
{3,"INV",0,2,0,dcCMD_INVITE},

```

```

{3,"WIN",0,2,0,dcCMD_WING},
{3,"WWI",0,1,0,dcCMD_WWING},
{3,"JSQ",0,1,0,dcCMD_JSQUAD},
{3,"JWI",0,1,0,dcCMD_JWING},
{3,"DSQ",0,1,0,dcCMD_DECSQDN},
{3,"DWI",0,1,0,dcCMD_DECWING},
{3,"NAM",0,2,0,dcCMD_NAMESQDN},
{3,"SLO",0,2,0,dcCMD_SLOGAN},
{6,"REMOVE",0,2,0,dcCMD_REMOVE},
{7,"DISBAND",0,1,0,dcCMD_DISBAND},
{4,"WITH",0,1,0,dcCMD_WITHDRAW},
{5,"CLEAR",0,1,0,dcCMD_CLEARSCORE},
{3,"TIME",1,3,dcPM_CM,dcCMD_TIME},
{3,"RADIO",0,2,0,dcCMD_RADIO},
{2,"LG",0,1,dcPM_CM|dcPM_CO|dcPM_CL,dcCMD_LOG},
{3,"ROOM",0,1,0,dcCMD_ROOM},
{3,"INFL",0,1,dcPM_CM|dcPM_CO,dcCMD_INFLIGHT}, // inflight
{3,"READY",0,1,dcPM_CM|dcPM_CO,dcCMD_READY},
{3,"BDA",0,1,dcPM_CM,dcCMD_BDA},
{4,"SETF",0,3,dcPM_CM,dcCMD_SETFIELD}, // set field
{4,"RANK",0,1,0,dcCMD_RANK}, // Display Rank
{4,"MACH",1,1,dcPM_GOD,dcCMD_MACHINES}, // show client CPU types logged in
{3,"ORD",0,1,0,dcCMD_ORD}, // Set Ordance load
{4,"CONV",0,2,0,dcCMD_CONV}, // Set Gun Convergence client
{6,"FILTER",1,2,dcPM_GOD,dcCMD_PROFANE}, // Set filter on/off
{6,"BWIDTH",1,1,0,dcCMD_BANDWIDTH},
{6,"GCLEAR",0,1,0,dcCMD_GCLEAR},
{7,"LISTINV",0,1,0,dcCMD_GLISTINV},
{7,"LISTATT",1,1,0,dcCMD_GLISTATT},
{7,"BUFRANK",0,1,0,dcCMD_RANKBUFF},
{6,"STATUS",1,2,0,dcCMD_STATUS},
{4,"EASY",0,2,0,dcCMD_EASY},
{4,"BGSET",1,1,dcPM_GOD,dcCMD_TALK},
{6,"EXPORT",0,1,dcPM_CM,dcCMD_EXPORT},
{5,"DELID",0,1,0,dcCMD_DELETEID},
{5,"TRACE",1,4,0,dcCMD_TRACE},
{3,"DET",1,2,dcPM_GOD,dcCMD_DETONATE},
{13,"RESTORE_FIELD",1,2,dcPM_CM,dcCMD_RESTOREFIELD}, // restore field
{4,"RESALL",1,1,dcPM_CM,dcCMD_CLEAR_ALL_DEATHS},
{11,"RESUPPLYINT",1,2,dcPM_CM,dcCMD_RESUPPLYINTERVAL}, //Sets resupply interval
{11,"RESUPPLYAMT",1,2,dcPM_CM,dcCMD_RESUPPLYAMOUNT}, //Sets resupply
amount
{8,"RESUPPLY",1,4,dcPM_CM,dcCMD_RESUPPLY}, // resupply
{3,"RES",1,2,dcPM_CM,dcCMD_CLEAR_DEATHS},
{5,"PLOAD",1,1,dcPM_CM,dcCMD_LOAD_PLANES},
{3,"PSQ",0,3,0,dcCMD_PSQUAD}, // squad permissions
{6,"SETCFO",0,4,dcPM_GOD,dcCMD_SETCFO},
{5,"MODESET",1,2,dcPM_CM | dcPM_CL,dcCMD_SETMODE},
{5,"PSAVE",1,1,dcPM_CM,dcCMD_SAVE_PLANES},
{5,"ALOAD",1,1,dcPM_CM,dcCMD_LOAD_AIRFIELDS},
{5,"ASAVE",1,1,dcPM_CM,dcCMD_SAVE_AIRFIELDS},
{5,"SQSCO",0,1,0,dcCMD_SQDNSCORE}, // squad scores
{5,"MEDAL",0,1,0,dcCMD_MEDAL_MILESTONES}, // medal milestones
{5,"SQRANK",0,1,0,dcCMD_SQUAD_RANK}, // fighter squad ranks
{5,"TTALK",1,2,dcPM_CM,dcCMD_TTALK},
{4,"TMOV",1,3,dcPM_CM,dcCMD_TMOV},
{5,"TPLANE",1,3,dcPM_CM,dcCMD_TPLANE},
{5,"FLDALT",1,3,dcPM_CM,dcCMD_FLDALT},

```

```

{4,"TCOU",1,3,dcPM_CM,dcCMD_TCCOUNTRY},
{4,"TEXIT",1,2,dcPM_CM,dcCMD_TEXIT},
{6,"BSQRANK",0,1,0,dcCMD_SQUAD_RANKBUFF}, // bomber squad ranks
{6,"BMEDAL",0,1,0,dcCMD_BMEDAL_MILESTONES}, // bomber medal milestones
{4,"PAGE",1,1,0,dcCMD_PAGE_TRAINER}, // page an online trainer
{3,"HLS",1,1,0,dcCMD_HL_SQUAD}, // highlight an entire squad
{8,"SHUTDOWN",0,2,dcPM_GOD,dcCMD_SHUTDOWN}, // kill the arena
{6,"REBOOT",0,2,dcPM_GOD,dcCMD_REBOOT}, // reboot the arena
{6,"ENDTOD",0,2,dcPM_GOD,dcCMD_ENDTOD}, // end TOD, reboot arena
{5,"UPERM",0,3,dcPM_GOD,dcCMD_SETPERM}, // set user permissions
{4,"BLOW",0,3,dcPM_GOD,dcCMD_BLOW}, // Destroy objects given field
{8,"GAMEDATE",0,2,dcPM_CM,dcCMD_GAMEDATE}, // set date
{7,"WEATHER",0,1,dcPM_CM,dcCMD_GAMEWEATHER}, // set clouds
{8,"AIATTACH",0,4,0,dcCMD_AIATTACH}, // attach to AI host param id
{6,"AILIST",0,1,0,dcCMD_AILIST}, // list AI hosts info
{8,"AIATTCHX",0,4,0,dcCMD_AIATTACHX}, // attach to AI host param Fd
{3,"NAV",1,3,0,dcCMD_AINAV}, // navigate attached drone
{4,"AUTO",1,1,0,dcCMD_AIAUTO}, // auto navigate attchddrone
{6,"CVENAB",0,4,dcPM_CM,dcCMD_CVENABLE}, // enable/disable planes on cv
{8,"CMISLAVE",0,1,dcPM_CM,dcCMD_CMEYESLAVE}, // slave to cm's cmeye mode
{7,"CMIFREE",0,1,dcPM_CM,dcCMD_CMEYEFREE}, // free cmeye slave
{7,"TONNAGE",0,4,dcPM_CM,dcCMD_TONNAGEONTARGET}, // Tonnage on Target
{7,"TERRDIR",0,2,dcPM_GOD,dcCMD_TERRAIN}, // terrain
{7,"PCOUNTS",0,2,0,dcCMD_PCOUNTS}, // Gets plane counts

```

Damage Parts

```

ENG1
ENG2
ENG3
ENG4
HYDENG1
HYDENG2
HYDENG3
HYDENG4
ELEV
HSTAB
RUDDER
VSTAB
LAILARON
RAILARON
LEFTFUEL
RIGHTFUEL
CENTFUEL
REARFUSE
CENTFUSE
PILOTARM
PILOT
TAILGUN
NOSEGUN
LEFTGUN
RIGHTGUN
TOPGUN
BOTTOMGUN
LEFTWING
RIGHTWING
RIGHTGEAR
LEFTGEAR
FLAPS

```

Otto Commands

- .OTTO PARAMS** Displays the current settings for the otto. If online, does not show settings which player cannot override.
- .OTTO [param] [value>]** Allows the player to override an otto setting. If online, only those parameters permitted by the arena OTTO_OVERRIDES setting can be changed.

When playing offline, the parameters set will apply to both the autogunners in the player's plane and those in the AI-controlled planes.

- .OTTO STATUS** Displays the current status of each ottogunner
- .OTTO SHUTUP** Stops otto chatter
- .OTTO TALK** Resumes otto chatter

The .otto parameter command applies only to the otto on your FE. When you are online, the host's "otto_overrides" setting determines which parameters you control, and which are set by the host.

To display current settings:
".otto params"

To set accuracy level:
".otto accuracy [level]" OR
".otto ac [level]"
Where [level] is from 1 - 10. Example ".otto ac 10" sets accuracy to highest level.

To set the range:
".otto range D[range]" OR
".otto rg D[range]"
Where [range] is from 1-12 ... don't forget to put the "D". Example: ".otto rg D10" sets range at which otto will commence firing to D10.

To set the target refresh time:
".otto retarget_time [time]" OR
".otto rt [time]"
Where [time] is from 0-10 seconds. Example: ".otto rt 3.5" has otto look for new targets every 3.5 seconds.

To set the base burst period:
".otto burston_base [period]" OR
".otto b+ [period]"
Where [period] is from 0.5 - 4.0 seconds. Example: ".otto b+ 2.2" has otto burst for at most 2.2 seconds before pausing.

To set the "max" burst period:
".otto burston_max [period]" OR
".otto b++ [period]"
Where [period] is from 0.5 - 4.0 seconds. Example: ".otto b++ 3.0" has otto burst for at most an additional 3.0 seconds when it is hitting the target.

To set the pause in between bursts:
".otto burstoff_base [period]" OR

".otto b- [period]"

Where [period] is from 0.5 - 4.0 seconds. Example: ".otto b- 1.0" has otto pause for at most 1 second in between bursts.

Script Specification:

Warbirds script specification

Version 1.4

07/30/03

Use:

.script <scriptpath>

Or embedded in a "dot command file" as follows:

... <= 0 or more "non-scripted" dot commands

%% <= indicates start of script

<script grammar>

%% <= indicates end of script

...

Conditionals:

if ()

{

<dotcommands>

}

else

{

<dotcommands>

}

switch ()

{

case <arg>

<dotcommands

break

case <arg>

<dotcommands>

break

}

Functions:

FIELD(<int>) =>field #<int> entity

GROUNDOBJECT("<id>") =>ground object "id" entity

COUNTRY(<entity>) =>country of <entity>

DESTROYED(<entity>) =>true if <entity> destroyed

MBLLOADED("<path>") =>true if mbl "path" is loaded

MBLRUNNING("<path>") =>true if mbl "path" is running

RAND(<int>,<int>) =>random int from n0 ... n1 inclusive

RCL() =>value of global bit accumulator

RCL_AND(<int>) =>value of global bit accumulator & <n>

FIELDSFRIENDLY(<int>,<int>) =>true if fields x,y are same country

INTRCL("<id>") =>value of global named integer variable "id"

// lex/yacc grammar

%token SCRIPTIF_TOK

%token SCRIPTELSE_TOK

%token SCRIPTCOUNTRY_TOK

%token SCRIPTFIELD_TOK

```

%token SCRIPTGROUNDOBJECT_TOK
%token SCRIPTSWITCH_TOK
%token SCRIPTCASE_TOK
%token SCRIPTMBLLOADED_TOK
%token SCRIPTDESTROYED_TOK
%token SCRIPTBREAK_TOK
%token SCRIPTRANDINT_TOK
%token SCRIPTDOTBIT_TOK
%token SCRIPTDOTBITAND_TOK
%token SCRIPTDTRUNNING_TOK
%token SCRIPTFIELDSFRIENDLY_TOK
%token SCRIPTUIINT_TOK
%token SCRIPTUISTRING_TOK

// lex definitions
integer    -?[0-9]+
real       -?[0-9]+|-?[0-9]+\.[0-9]+|-?\.[0-9]+
identifier [a-zA-Z][0-9a-zA-Z_]*
ws         [\t]+
qstring    \"[^\n]*[\"\\n]
comment    /*.**/
uivar      @[^\n]*[@\n]

if         { return SCRIPTIF_TOK; }
else      { return SCRIPTELSE_TOK; }
switch    { return SCRIPTSWITCH_TOK; }
case      { return SCRIPTCASE_TOK; }
break     { return SCRIPTBREAK_TOK; }
COUNTRY   { return SCRIPTCOUNTRY_TOK; }
FIELD     { return SCRIPTFIELD_TOK; }
GROUNDOBJECT { return SCRIPTGROUNDOBJECT_TOK; }
MBLLOADED { return SCRIPTMBLLOADED_TOK; }
DESTROYED { return SCRIPTDESTROYED_TOK; }
RAND      { return SCRIPTRANDINT_TOK; }
RCL       { return SCRIPTDOTBIT_TOK; }
RCL_AND   { return SCRIPTDOTBITAND_TOK; }
MBLRUNNING { return SCRIPTMBLRUNNING_TOK; }
SCRLOADED { return SCRIPTDTRUNNING_TOK; }
FIELDSFRIENDLY { return SCRIPTFIELDSFRIENDLY_TOK; }
INTRCL    { return SCRIPTUIINT_TOK; }
STRRCL    { return SCRIPTUISTRING_TOK; }

#include    { BEGIN(incl); }
<incl>[\t]* { /* eat white space */ }
<incl>[^\t\n]+ { _HandleInclude(); }
^\.        { BEGIN(cmd); }
<cmd>.*[\n\r] {
    yylineno++;
    strncpy(yylval.dotcmd,yytext,511);
    BEGIN(INITIAL);
    return DOTCOMMAND;
}
^#         { BEGIN(linecomment); }
<linecomment>.*[\n\r] { BEGIN(INITIAL); }

[=;{}<>,()] { return yytext[0]; }

{qstring} {

```

```

        int l;
        l=yyleng-2;
        if (l>sizeof(yylval.string)-1)
        {
            yyerror("String too large");
            return STRING;
        }
        memset(yylval.string,0,sizeof(yylval.string));
        memcpy(yylval.string,yytext+1,l);
        return STRING;
    }

{uiivar} {
    int uiVarType;
    int intRes;
    char *stringRes;
    char id[33];
    int l;
    l=yyleng-2;
    if (l>32)
    {
        yyerror("global named variable id > 32");
        return INT;
    }
    memset(id,0,sizeof(id));
    memcpy(id,yytext+1,l);
    uiVarType = uiVarsGet(id,
                          &intRes,
                          &stringRes);
    switch(uiVarType)
    {
        case uiVARSTYPE_INT:
            yylval.integer=intRes;
            return INT;

        case uiVARSTYPE_STRING:
            memset(yylval.string,0,sizeof(yylval.string));
            strncpy(yylval.string,stringRes,sizeof(yylval.string));
            return STRING;

        case uiVARSTYPE_NULL:
            yyerror("Undefined global named variable");
            yylval.integer=-1;
            return INT;

    }
}

{integer} {
    yylval.integer = atoi(yytext);
    return(INT);
}

{real} {
    yylval.real = atof(yytext);
    return(REAL);
}

```

```

// yacc grammar
scriptfile : script_statements

script_statements : script_statement
| script_statements script_statement

script_statement : ifelse_statement
| switch_statement
| DOTCOMMAND

switch_statement : SCRIPTSWITCH_TOK '(' script_result ')' '{' case_statements '}'

script_result : country_result
| script_int

country_result : SCRIPTCOUNTRY_TOK '(' game_entity ')'

case_statements : case_statement
| case_statements case_statement

case_statement : SCRIPTCASE_TOK INT dot_commands
| SCRIPTCASE_TOK INT dot_commands SCRIPTBREAK_TOK
| SCRIPTCASE_TOK INT

ifelse_statement : if_statement
| if_statement else_statement

if_statement : SCRIPTIF_TOK '(' script_condition ')' '{' dot_commands '}'
| SCRIPTIF_TOK '(' script_condition ')' '{' '}'

else_statement : SCRIPTELSE_TOK '{' dot_commands '}'
| SCRIPTELSE_TOK '{' '}'

script_condition : country_test
| destroyed_test
| SCRIPTMBLLOADED_TOK '(' STRING ')'
| SCRIPTMBLRUNNING_TOK '(' STRING ')'
| SCRIPTDTFRUNNING_TOK '(' STRING ')'
| SCRIPTFIELDSFRIENDLY_TOK '(' INT ',' INT ')'
| script_int '=' INT
| script_int '<' INT
| script_int '>' INT
| script_int '>=' INT
| script_int '<=' INT
| script_int
| SCRIPTUISTRING_TOK '(' STRING ')' '=' STRING
| STRING '=' STRING

destroyed_test : SCRIPTDESTROYED_TOK '(' game_entity ')'

country_test : SCRIPTCOUNTRY_TOK '(' game_entity ')' '=' script_int

game_entity : field_entity
| ground_object_entity

field_entity : SCRIPTFIELD_TOK '(' script_int ')'

ground_object_entity : SCRIPTGROUNDOBJECT_TOK '(' STRING ')'

```

dot_commands : dot_command
| dot_commands dot_command

dot_command : DOTCOMMAND

script_int : INT
| SCRIPTRANDINT_TOK '(' INT ',' INT ')'
| SCRIPTDOTBIT_TOK '(' ')'
| SCRIPTDOTBITAND_TOK '(' INT ')'
| SCRIPTUIINT_TOK '(' STRING ')'

Related Dot Commands

1. Return dot command

.RETURN [<path>]

Immediately exits from dotfile. If path is given, executes dot commands in path. Is valid from within the parsed script files/sections

2. Dot commands operating on global bitwise accumulator:

.BITADD <n> Accumulator += <n>.

.BITAND <n> Accumulator &= <n>.

.BITNAND <n> Accumulator &= ~<n>.

.BITOR <n> Accumulator |= <n>.

.BITRES Displays accumulator in decimal.

.BITRESHEX Displays accumulator in hex.

.BITSL <n> Accumulator <<= <n>.

.BITSR <n> Accumulator >>= <n>.

.BITSTO <n> Accumulator = <n>.

.BITSUB <n> Accumulator -= <n>.

.BITXOR <n> Accumulator ^= <n>.

3. Dot commands creating/operating on global named variables:

.INTSTO <id> <n> <id> = <n> (<id> created if not existing)

.INTADD <id> <n> <id> += <n>

.INTSUB <id> <n> <id> -= <n>

.INTMUL <id> <n> <id> *= <n>

.INTDIV <id> <n> <id> /= <n>

.INTAND <id> <n> <id> &= <n>

.INTNAND <id> <n> <id> &= ~<n>

.INTOR <id> <n> <id> |= <n>

.INTXOR <id> <n> <id> ^= <n>

.INTRES [<id>] Displays value of <id>,

if <id> is blank, displays

all named global integers variables

.STRSTO <id> "<string text>" Creates a global named string variable <id>.

The numeric value is set to the length of the string

.INTPER <id> <n0> <n1> id <= n0 / n1 * 100

.VARSAVE <path> saves all global named variables to file <path>

.VARREAD <path> read global named variables from file <path>

.VARFREE <id> delete global named variable <id>

4. Using @<id>@ to expand global integer named variables inline

Global integer named variables can be expanded inline in dot commands via

@<id>@

The expand character '@' can be changed via

.macrochar <c>

5. Named host global variables (CMTOOL only)

Access for host global variables is in the form HOST:<variableid>

Currently defined:

HOST:PLAYERCOUNT returns playercount
HOST:PC<fff>.<ppp>.<c> returns the plane count for field <fff>,
plane type <ppp>, county <c>

examples:

.echo the player count is @HOST:PLAYERCOUNT@

the player count is 123

.echo the plane count for f1,p1,c1 is @HOST:PC001.001.1@

the plane count for f1,p1,c1 is 3

6. Named global variables for accessing offline internal data

The following internal data are available as read-only named variables, valid for scripts running offline. The read-only named variables are case insensitive.

Offline player data

:PLYR:COUNTRY - country
:PLYR:PLANETYPE - planetype (1-255)
:PLYR:PLANETYPESTR - pplabel for plane type (string)
:PLYR:FIELD - current field
:PLYR:ENTRYPOINT - current entry point
:PLYR:INFLIGHT - 1=currently flying
:PLYR:MODE - PLAYERMODE_:LINE_TOWER 3
 PLAYERMODE_:LINE_FLIGHT 4
 PLAYERMODE_OFFLINE_CMEYE 5
:PLYR:EASYMODE - fmREALMODE 0
 fmEZMODE 1
 fmRELAXEDMODE 2
:PLYR:UNLIMITEDAMMO - 1=unlimitedammo enabled
:PLYR:INVULNERABLE - 1=invulnerable enabled
:PLYR:BLACKOUTS - 1=blackouts enabled
:PLYR:STRUCTLIMITS - 1=struct limits enabled
:PLYR:BAILEDOUT - 1=player has bailed out
:PLYR:INCHUTE - 1=player in parachute
:PLYR:DEATHCAM - 1=player in deathcam sequence
:PLYR:STATION - OBSERVER 0
 TAILGUN 1
 NOSEGUN 2
 LEFTGUN 3
 RIGHTGUN 4
 TOPGUN 5
 BOTTOMGUN 6
 BOMBADIER 7
 PILOT 8
 CHASE 9
:PLYR:BLACKOUTSTATE - NOBLACKOUT 0
 MAXREDOUT 1
 MAXBLACKOUT 2
 MAXGREYOUT 3
 STARTREDOUT 4
 CLEARBLACKOUT 5
:PLYR:PILOTGS
:PLYR:CHUTESTATE - CHUTESTATE_CHUTECLOSED 0
 CHUTESTATE_CHUTEOPEN 1
 CHUTESTATE_CHUTEFAILED 2
 CHUTESTATE_CHUTESTREAMED 3
:PLYR:INPLANEVIEW - 0 = external view
:PLYR:VIEWDIR - CPVIEW_AHEAD 1
 CPVIEW_AHEAD_UP 2
 CPVIEW_AHEADRIGHT 3
 CPVIEW_AHEADRIGHT_UP 4
 CPVIEW_RIGHT 5

```

CPVIEW_RIGHT_UP      6
CPVIEW_BACKRIGHT     7
CPVIEW_BACKRIGHT_UP  8
CPVIEW_BACK          9
CPVIEW_BACK_UP      10
CPVIEW_BACKLEFT     11
CPVIEW_BACKLEFT_UP  12
CPVIEW_LEFT         13
CPVIEW_LEFT_UP      14
CPVIEW_AHEADLEFT    15
CPVIEW_AHEADLEFT_UP 16
CPVIEW_STRAIGHTUP   17
:PLYR:VIRTUALCOCKPITMODE - VCMODE_INSTANTSNAPS      0
                        VCMODE_PANNEDSNAPS          1
                        VCMODE_GLIMITED_PANNEDSNAPS 2
:PLYR:STICKYSNAP     - 1 = snaps stick

```

Offline stats:

```

.offscoreinit        this clears the offline stats
:STAT:WEAPSFIRE      - total weapons launched
:STAT:BULLETSFIRED   - total projectiles launched
:STAT:BOMBSDROPPED   - qed
:STAT:ROCKETSLAUNCHED - qed
:STAT:WEAPSHIT       - total weapons hitting
:STAT:HITPERC        - total weapons hitting / total weapons launched
:STAT:PLYRHITS       - total hits on player
:STAT:AIRHITS        - qed
:STAT:GROUNDHITS     - qed
:STAT:SEAHITS        - qed
:STAT:AIRKILLS       - qed
:STAT:GROUNDKILLS    - qed
:STAT:ACKKILLS       - qed
:STAT:GVKILLS        - qed
:STAT:SEAKILLS       - qed

```

Offline Cockpit Gauge Data:

```

:GAUGE:ENGPERR1
:GAUGE:ENGPERR2
:GAUGE:ENGPERR3
:GAUGE:ENGPERR4
:GAUGE:ENGSTATE1
:GAUGE:ENGSTATE2
:GAUGE:ENGSTATE3
:GAUGE:ENGSTATE4
:GAUGE:ALTFEET
:GAUGE:MPH
:GAUGE:FUELPER
:GAUGE:HOSTSTATE
:GAUGE:GEARSTATE1
:GAUGE:GEARSTATE2
:GAUGE:GEARSTATE3
:GAUGE:FLAPDEG
:GAUGE:BOMBDOORSOPEN
:GAUGE:BREAKSTATE
:GAUGE:AUTOPILOTSTATE
:GAUGE:STALLSTATE
:GAUGE:HEADING
:GAUGE:RATEOFCLIMB

```

```

:GAUGE:SLIPTENTHDEG
:GAUGE:DIVEBREAKON
:GAUGE:CURRENTGLOAD
:GAUGE:ENGINETEMP
:GAUGE:FLAPPOSITION
:GAUGE:ELEVTRIMPOS
:GAUGE:RUDDERTRIMPOS
:GAUGE:AILERONTRIMPOS
:GAUGE:PITCHIND
:GAUGE:ROLLIND
:GAUGE:YAWIND
:GAUGE:AOAINDICATOR
:GAUGE:ENGINEREVERSING1
:GAUGE:ENGINEREVERSING2
:GAUGE:ENGINEREVERSING3
:GAUGE:ENGINEREVERSING4
:GAUGE:ENGPERROUND1
:GAUGE:ENGPERROUND2
:GAUGE:ENGPERROUND3
:GAUGE:ENGPERROUND4
:GAUGE:PROPRPM1
:GAUGE:PROPRPM2
:GAUGE:PROPRPM3
:GAUGE:PROPRPM4

```

Offline Cockpit Weapon Data:

```

:WEAP:GUNROUNDS
:WEAP:WEAPONSELECTED
:WEAP:WEAPONROUNDS
:WEAP:SALVOCNT
:WEAP:BOMBDROPINTERVALMS
:WEAP:SIGHTMAG
:WEAP:WEAPONSARRAY
:WEAP:SIGHTDEVIATION

```

Random integer

```

:RAND:[<n0>[:<n1>]] - gens random integer in range <n0>-<n1>. If <n1> not
                    given, range 0-<n0>. If <n0> not given, range 0-65536.
                    <n0>,<n1>can be +-.

```

Millisecond clock

```

:CLOCK - returns current millisecond clock (glGlobalClock)
:CLOCKDIF - return current millisecond clock dif (glGlobalClockDif)

```

Key Presses

```

:KEYPRESSED - string value of last key pressed
:KEYPRESSED_ASC - ascii code int value of last key pressed
:KEYPRESSED_MOD - modifiers bit flags MODS_CMD = 1,
                    MODS_ALT = 2, // "option" key on the Mac
                    MODS_CTRL = 4,
                    MODS_SHIFT = 8,
                    MODS_CAPS = 16,
                    MODS_RIGHTSHIFT = 32,
                    MODS_RIGHTALT = 48,
                    MODS_RIGHTCTRL = 64

```

Examples

```

Example : testscr.dft
.echo testscr.dft
# embed script
%%
# embedded script for if..else, switch/case parsing
# the '.' in dot commands must be at column 1 ;)
.echo running embedded script1
# test if mbl file is loaded
# and load another one if so
if (MBLLOADED("cm\m3.mbl"))
{
.echo cm\m3.mbl is loaded
}
else
{
.echo cm\m3.mbl is not loaded
}
# test if field 1 is country 2
if (COUNTRY(FIELD(1))=2)
{
.echo the first if
.echo field 1 country is 1
}
else
{
.echo else from first if
.echo field 1 country is not 2
}
# try testing a random field
if (COUNTRY(FIELD(RAND(1,24)))=2)
{
.echo the second if
.echo field whatever is country is 2
}
# now for a test switch
switch (COUNTRY(FIELD(12)))
{
case 1
.echo case 1
.echo case 1 line2
case 2
.echo case 2
.echo case 2 line2
case 3
.echo case 3
break
case 4
.echo case 4
.echo case 4 line2
case 4
.echo again case 4
.echo because no break
}
# try out a random switch
switch(RAND(1,4))
{
case 4
.echo random 4
}

```

```

break

case 2
.echo random 2
break

case 1
.echo random 1
break

case 3
.echo random 3
break
}
# test a ground object's country
if (COUNTRY(GROUNDOBJECT("F04G201"))=2)
{
.echo ground object F04G201 is 2
}
else
{
.echo ground object F04G201 is not 2
}
# test if a ground object is destroyed
if (DESTROYED(GROUNDOBJECT("F04G201")))
{
.echo ground object F04G201 is destroyed
}
else
{
.echo ground object F04G201 is not destroyed
}
.echo done with embedded script
%%
.echo done with first
.echo starting second
%%
.echo clearing bit accum
.bitsto 0
# check for a loaded file
# and store result shifted 2 bits
if (MBLLOADED("drone01.mbl"))
{
.bitor 1
# shift it left 2 bits (=4)
.bitsl 2
}
# test country of field 1 & 2
# and keep results in bits 1-2
if (COUNTRY(FIELD(1))=1)
{
.bitor 1
}
if (COUNTRY(FIELD(2))=1)
{
.bitor 2
}
# show result in hex for debug

```

```
.bitshex
# bits 1,2 contain our field country tests,
# and bit 3 is our test for whether drone01 is loaded
if (RCL_AND(1))
{
.echo field 1 is red
}
else
{
.echo field 1 is not red
}
if (RCL_AND(2))
{
.echo field 2 is red
}
else
{
.echo field 2 is not red
}
if (RCL_AND(4))
{
.echo drone1.mbl is loaded
}
else
{
.echo drone1.mbl is not loaded
}
# now use all bits in switch
switch (RCL())
{
case 1
.echo only field 1 is red
break
case 5
.echo drone1 is loaded and
.echo only field 1 is red
break

case 2
.echo only field 2 is red
break
case 6
.echo drone1 is loaded and
.echo only field 2 is red
break

case 3
.echo fields 1 and 2 are red
break
case 7
.echo drone1 is loaded and
.echo fields 1 and 2 are red
break

case 4
.echo drone1 is loaded and
.echo neither fields 1 or 2 are red
break
```

```

case 0
.echo drone1 is not loaded and
.echo neither fields 1 or 2 are red
break
}
%%
.echo done with second
.echo third sample
%%
# use .intsto to declare a new int var
.intsto myvar 0
# Add 1 if fields are friendly
if (FIELDSFRIENDLY(2,1))
{
# do nothing
}
else
{
.intadd myvar 1
}
if (COUNTRY(FIELD(2))=2)
{
.intmul myvar 2
}
else
{
.intmul myvar 3
}
# display myvar using inline expansion
.echo @myvar@
# retrieve var and use
switch (INTRCL("myvar"))
{
case 0
.echo fields 2 and 1 are friendly
break

case 1
.echo this case should never happen
break

case 2
.echo fields 2 and 1 are nmes and field 2 is green
break

case 3
.echo fields 2 and 1 are nmes and field 2 is not green
break
}
%%
.echo end of third sample
# sample 4 using .return
.intsto myvar 1
.echo myvar is @myvar@
%%
if (INTRCL("myvar")=1)
{
.echo myvar is @myvar@
}

```

```

.return myvar1.dtf
}
if (INTRCL("myvar")=3)
{
.echo myvar is 3
.return
.echo shouldn't see this
}
.echo shouldn't see this if myvar is 3 or 1
%%
.echo end of fourth sample
# sample - 5
# Using expand character '@' for
# global named variables to to simplify grammar
.echo start of fifth sample
%%
.intsto my-score 1000
if (@my-score@=1000)
{
.intsto casevar 1
.echo you have a wonderful score!
}
else
{
.intsto casevar 2
.echo you are not so hot, buddy
}
switch(@casevar@)
{
case 1
.echo you are wonderful
break

case 2
.echo you are terrible
break
}
.varfree my-score
.varfree casevar
# string compares
.strsto player-name "SPINDZ"
if (@player-name@="SPINDZ")
{
.echo he's SPINDZ, all right!
}
# strcl can test for non-existent
# vars by comparing empty strings
if (STRRCL("noexist")="")
{
.strsto noexist "Now this exists"
.echo @noexist@
}
.varfree player-name
.varfree noexist
# gen random number
.intsto myrand @RAND:@
.echo Random number generated between 0,65536: @myrand@
.intsto myrand @RAND:5@

```

```
.echo Random number generated between 0,5: @myrand@
.intsto myrand @RAND:5:10@
.echo Random number generated between 5,10: @myrand@
.intsto myrand @RAND:-5:-10@
.echo Random number generated between -5,-10: @myrand@
.varfree myrand
%%
.echo end of fifth sample
```

```
*****
```

```
sample 6 - offline campaign system
```

```
*****
```

```
*****
```

```
offln\Simple Campaign.off
```

```
*****
```

```
.strsto cm-main-dir "offln\campaign\simple"
```

```
.dotfile offln\campaign\simple\initsession.dtf
```

```
*****
```

```
offln\campaign\simple\initsession.dtf
```

```
*****
```

```
# Inits the campaign session
```

```
#-----
```

```
# make sure cm-player is set to something
```

```
# function STRRCL allows comparing "empty" strings
```

```
%%
```

```
if (STRRCL("cm-player")="")
```

```
{
```

```
.strsto cm-player "PLAYER"
```

```
# use modal input dialog to get
```

```
.vargetstring "Enter A Player Id" cm-player Simple
```

```
}
```

```
%%
```

```
.strsto cm-main-dir "offln\campaign\simple"
```

```
# set the default campaign level to 0
```

```
.terrain terr001
```

```
.strsto cm-title "A Simple Campaign"
```

```
.strsto cm-window-title "Simple"
```

```
.intsto cm-level 0
```

```
# get current campaign vars
```

```
.varread @cm-main-dir@\@cm-player@\cm.var
```

```
# get saved ground objects
```

```
.loadgamestate @cm-main-dir@\@cm-player.gme
```

```
# see if we're at level 0, if so zero-out stats
```

```
# and then set level to 1
```

```
%%
```

```
if (@cm-level@=0)
```

```
{
```

```
.dotfile @cm-main-dir@\zero.dtf
```

```
.dotfile @cm-main-dir@\shockandawe\initlevel.dtf
```

```
}
```

```
%%
```

```
# set up echo file
```

```
.echofile echo.txt
```

```
# show current stats
```

```
.dotfile @cm-main-dir@\showstats.dtf
```

```
.dotfile @cm-main-dir@\@cm-sub-dir@\display.dtf
```

```
.echofile
```

```

# display
.intsto cm-result 1
.vargetdisplay echo.txt cm-result @cm-window-title@ 100 100 750 600
%%
if (@cm-result@=1)
{
.return @cm-main-dir@\fly.dtf
}
else
{
.echo campaign session cancelled
}
%%
*****
offln\campaign\simple\exitflight.dtf
*****
# Exit flight stuff
#-----
# accumulate stats
.dotfile @cm-main-dir@\accumstats.dtf
# call campaign's exit flight dotfile
# setup echo file
.echofile echo.txt
.dotfile @cm-main-dir@\@cm-sub-dir@\exitflight.dtf
# save the game
.savegamestate @cm-main-dir@\@cm-player@.gme
# show stats
.dotfile @cm-main-dir@\showstats.dtf
.dotfile @cm-main-dir@\@cm-sub-dir@\display.dtf
# reset echo file
.echofile
# display it
.intsto cm-result 1
.vargetdisplay cm-result echo.txt "@cm-window-title@" 100 100 750 600
%%
if (@cm-result@=1)
{
.return @cm-main-dir@\fly.dtf
}
else
{
.echo campaign session cancelled
}
%%
*****
offln\campaign\simple\killed.dtf
*****
.intadd cm-deaths 1
.return @cm-main-dir@\exitflight.dtf
*****
offln\campaign\simple\landed.dtf
*****
# Exit flight - landed
.intadd cm-landed 1
.return @cm-main-dir@\exitflight.dtf
*****
offln\campaign\simple\accumstats.dtf
*****

```

```

# Accumulate stats after each flight
.intadd cm-sorties 1

.intper cm-hitperc @:STAT:HITS@ @:STAT:WEAPSFIRE@

# calculate score
.intper cm-score @cm-sorties@ @cm-deaths@
.intmul cm-score @cm-landings@
.intadd cm-score @cm-hitperc@
.intmul cm-score @:STAT:KILLS@
# use settings for multiplier
.intsto atemp 3
.intsub atemp @:PLYR:INVULNERABLE@
.intsub atemp @:PLYR:UNLIMITEDAMMO@
.intadd atemp @:PLYR:STRUCTLIMITS@
.intadd atemp @:PLYR:BLACKOUT@
%%
switch (@:PLYR:EASYMODE@)
{
case 0
.intadd atemp 2
break

case 2
.intadd atemp 1
break
}
%%
.intmul cm-score @atemp@
.varfree atemp
*****
offln\campaign\simple\showstats.dtf
*****
# Set rank title
.dotfile @cm-main-dir@\setranktitle.dtf
# display
.echo -----
.echo @cm-title@
.echo -----
.echo Player: @cm-player@
.echo Current mission: @cm-mission@
.echo Current rank: @cm-rank-title@
.echo -----
.echo *****Current stats*****
.echo -----
.echo
.echo Sorties: @cm-sorties@
.echo Landings: @cm-landings@
.echo Deaths: @cm-deaths@
.echo
.echo Weapons launched: @:STAT:WEAPSFIRE@ bullets=@:STAT:BULLETSFIRED@
bombs=@:STAT:BOMBSDROPPED@ rockets=@:STAT:ROCKETSLAUNCHED@
.echo Targets hit: @:STAT:HITS@ air=@:STAT:AIRHITS@ ground=@:STAT:GROUNDHITS@
sea=@:STAT:SEAHITS@
.echo Targets killed: @:STAT:KILLS@ air=@:STAT:AIRKILLS@ ground=@:STAT:GROUNDKILLS@
sea=@:STAT:SEAKILLS@
.echo Hit percentage: @cm-hitperc@%
.echo Hits received: @:STAT:PLYRHITS@

```

```

.echo
.echo Missions succesfully completed: @cm-misscpl@
.intsto scoremod 0
.echo
.echo Score modifiers
.echo -----
%%
if (@:PLYR:INVULNERABLE@=1)
{
.echo Invulnerable to weapon damage -1
.intsub scoremod 1
}
if (@:PLYR:UNLIMITEDAMMO@=1)
{
.echo Unlimited ammo: -1
.intsub scoremod 1
}
if (@:PLYR:BLACKOUTS@=1)
{
.echo Blackouts enabled: +1
.intadd scoremod 1
}
if (@:PLYR:STRUCTLIMITS@=1)
{
.echo Struct limits enabled: +1
.intadd scoremod 1
}
if (@:PLYR:EASYMODE@=0)
{
.echo Real mode flight: +2
.intadd scoremod 2
}
if (@:PLYR:EASYMODE@=1)
{
.echo Easy mode flight: -1
.intsub scoremod 1
}
if (@:PLYR:EASYMODE@=2)
{
.echo Relaxed mode flight: +1
.intadd scoremod 1
}
%%
.echo -----
.echo Net score modifier: @<%+d>scoremod@
.varfree scoremod
.echo Current score: @cm-score@
*****
offln\campaign\simple\zero.dtf
*****

.offscoreinit
.intsto cm-hitperc 0
.intsto cm-sorties 0
.intsto cm-landings 0
.intsto cm-deaths 0
.intsto cm-misscpl 0
.intsto cm-rank 0
.intsto cm-score 0

```

```

*****
offln\campaign\simple\shockandawe\initlevel.dtf
*****
.intsto cm-level 1
.intsto ms-sorties 0
.intsto ms-goal @:STAT:GROUNDKILLS@
.intadd ms-goal 4
.strsto cm-sub-dir "shockandawe"
.strsto cm-mission "Shock And Awe"
*****
offln\campaign\simple\shockandawe\fly.dtf
*****

.intrigger_clear
.intrigger :STAT:GROUNDKILLS @ms-goal@ @ms-goal@ @cm-main-dir@\shockandawe\trigger.dtf
.offcmtool
.clearmo
.loadmo @cm-main-dir@\shockandawe\p1.mbl
.waypoint_ques prompt
.fly
*****
offln\campaign\simple\shockandawe\trigger.dtf
*****
.echo Good work ... you've shocked and awed those bastiges!
.e
*****
offln\campaign\simple\shockandawe\exitflight.dtf
*****
.intadd ms-sorties 1
%%
if (@:STAT:GROUNDKILLS@>=@ms-goal@)
{
.intadd cm-misscmpl 1
.intadd cm-rank 1
.echo Congrats ... you've destroyed 4 or more ground targets,
.echo and have earned yourself a promotion!
.return @cm-main-dir@\backporch\initlevel.dtf
}
if (@ms-sorties@>=8)
{
.echo You've flown enough sorties to
.echo go to the next mission.
.echo However a promotion is not in store
.return @cm-main-dir@\backporch\initlevel.dtf
}
%%
*****
offln\campaign\simple\shockandawe\display.dtf
*****
.echo
.echo Your mission is to destroy at least
.echo at least four ground targets

*****
offln\campaign\simple\backporch\initlevel.dtf
*****

```

```

.intsto cm-level 2
.strsto cm-mission "Back Porch"
.strsto cm-sub-dir "backporch"
.intsto ms-sorties 0
.intsto ms-goal @:STAT:AIKILLS@
.intadd ms-goal 2
*****
offln\campaign\simple\backporch\fly.dtf
*****
.intrigger_clear
.intrigger :STAT:AIKILLS @ms-goal@ @ms-goal@ @cm-main-dir@\backporch\trigger.dtf
.offcmtool
.clearmo
.loadmo @cm-main-dir@\backporch\p1.mbl
.loadmo @cm-main-dir@\backporch\d1.mbl
.fly
*****
offln\campaign\simple\backporch\trigger.dtf
*****
.echo Good shooting, ace!
.e
*****
offln\campaign\simple\backporch\display.dtf
*****
.echo Your mission is to shoot down at least 2
.echo enemy aircraft
*****
offln\campaign\simple\backporch\exitflight.dtf
*****
.intadd ms-sorties 1
%%
if (@:STAT:AIKILLS@>=@ms-goal@)
{
.echo Congrats ... you've destroyed 2 or more air targets.
.echo You have completed both missions of the campaign
.echo and have been promoted. You can continue again
.echo with the first mission if you wish.
.intadd cm-misscmpl 1
.intadd cm-rank 1
.return @cm-main-dir@\shockandawe\initlevel.dtf
}
if (@ms-sorties@>=8)
{
.echo You've flown enough sorties to
.echo end the campaign.
.echo However a promotion is not in store. You can
.echo continue again with the first mission if you wish.
.return @cm-main-dir@\shockandawe\initlevel.dtf
}
%%

```