

The Freeciv Server User's Guide

Overview

The Freeciv server is the core of the Freeciv system. It maintains the complete states of all game variables (map, units, cities, research, etc.), updates them based on player client requests, and sends updates to the clients. For multiplayer games it is the central arbiter of the game's progress. For solitaire play, it implements the Artificial Intelligence (AI) races.

Whether for solitaire or multiplayer games, the server operator runs the server in a window. The operator may set game options, including loading entirely new sets of game rules, or load prepackaged scenarios. Players use client programs to connect to the server via the network (even solitaire games, run on one computer, use the network software to connect the client and server programs). Once all players have connected, the server operator starts the game, and players then proceed with play.

In the course of a typical game, server operators will monitor progress, using mechanisms inside the game ("chat") or outside the game (E-mail) to communicate with players. The operator may also play, running a client program in another process. As the game progresses, the operator may use server commands to adjust game options, save the game for later restart, or terminate the game.

The Freeciv Metaserver is a server on the Internet which provides a coordination point for Freeciv multiplayer games. If enabled by the operator by command line option `--meta`, the server informs the Metaserver of the status and progress of the game. This one-way information flow (server to Metaserver) should, in time, allow players to locate servers and games to play on.

This document reflects the Freeciv server program for the 1.8.0 build. The general concepts involved will also help in using Freeciv 1.7.2 and earlier versions.

Usage

On a Unix system, you will need to start the server in a window (e.g., xterm, virtual console), since commands must be entered during play. Run the server with the command `civserver` followed by your desired command line options (described later). (If you do not have Freeciv installed with `"make install"`, use the `"ser"` script provided in the distribution.) The server will initialize and prompt for input.

At this point, you will wish to set game options, minor ones (timeout times), medium ones (map size), and major ones (rules for Civilization I or II). For the game options, see [Command Line Options](#) and [Files and Environment](#) below. If you have a collection of option settings which you use often, you may enter them into a flat ASCII file and read them into the server quickly with the `-r` option. You should coordinate your option choices with the players to reduce confusion. You may also load entire scenarios with the `-f` command line option.

While you are entering commands, the server will have announced the game to the Freeciv Metaserver (if you enabled it), and will have enabled client network connections. Player client programs will connect to the server and wait; you can use the `"l"` command to show the connected players. Once all settings have been made, and all desired players have connected, you start the game with the `"s"` command. Once the game is started, each player will be asked to select a race. After these selections, the server will assign AI races, generate the map, send all initial data to the clients, and the game is on.

In Civilization I and II, each race's turn proceeds in sequence: first the human player moves, and presses Next Turn; then the Romans move, then the Greeks, and so on through a standard list of AI races, ending with the Barbarians, after which the human player gets control again. In Freeciv multiplayer, all of the players enter orders simultaneously. The server processes all orders from clients in the order in which they are received. Some things, however, only update at the end of a "turn": Cities grow and eat food, research happens, revolutions complete, units heal damage, and the calendar turns to a new year. That end-of-turn update takes place when all human players have indicated Next Turn (but see the Timeout option setting). If a player takes several minutes to finish moves, all players wait until that last one completes before the update takes place. If there are AI races, they make their moves at the start of the turn update.

Once the victory conditions have been met (see README in the distribution), the game ends. But you may suspend the game at any time, and resume it another day if you wish. The "save" command will write out all game state into a file as of the instant of the command (even with players entering orders). Then you may quit the server. When you wish to resume (and have coordinated the restart of the game with the players), run the server again with the -f option to read in the save file. That will reset the internal data to reflect the state in the file, and leave the game paused. Once the player clients reconnect, an "s" command will get the game moving again. In addition, if the server or the system fails, you may be able to fall back to the list automatic save done every few turns (set by option setting saveturns); its files are the same.

Command Line Options

The following options are accepted on the command line of the server. They may not be combined; that is, "civserver -fp savegame.sav 555" will not work, instead you would need to enter "civserver -f savegame.sav -p 555". Most options have a short form (single hyphen and single letter) and a long form (double hyphen and a complete word); their effects are identical.

-f filename or --file filename

Loads a saved game into the server before initialization, instead of starting a new game. This is for reloading saved games, or for loading scenarios. The distribution comes with three such scenarios, typically stored in /usr/local/share/freeciv:

earth-160x90.sav

the real Earth at 160x90 size

earth-80x50.sav

the real Earth at 80x50 size

europa.sav

the real Europe at 200x100 size

-h or --help

Prints out a description of the command line options and exits.

-l filename or --log filename

Defines a log file to be produced during processing. By default, "fatal" and "normal" messages are printed to standard output. With a log file, such messages go to the log instead. Use the -d option to set how much is logged.

-g filename or --gamelog filename

Defines a log file to be produced during processing. This is a game progress log, of game-related events recording the activities of the players.

-m or --meta

Directs the server to communicate with the Freeciv Metaserver.

-p number or --port number

Specifies the TCP port number to which clients will connect; players must know this number to be able to connect if they are not to use the default of 5555 decimal. You may need to use this if 5555 is not available for your use on your system.

-r filename or **--read filename**

Specifies a file of server commands which the server will automatically read and process, as if you had typed them in at the server's prompt. The distribution has an example that sets options to be similar to Civilization I, which the option `-r /usr/local/share/freeciv/civ1.serv` would read.

-s name or **--server name**

Specifies the name to be given to the Freeciv Metaserver as the name of this server. This is distinct from the description line set by the "meta" command; both are displayed in the Metaserver's output.

-d number or **--debug number**

Sets the amount of debugging information to be logged in the file named by the `-l` option. The number should be 0 for fatal errors only, 1 for fatal and normal messages, or 2 for fatal, normal, and debugging messages.

-v or **--version**

Causes the server to display its version number and exit.

Commands

You may enter commands into the server at any time, either before or during the running of a game.

Command names are case-sensitive, as are "filename" and "setting"; argument "playername" generally is not. In all cases where a "playername" is expected as a command argument, it is the name of the ruler of a race in the same; e.g., you probably want to refer to "caesar" rather than "romans."

comment

This is intended for script files. If the first non blank character on a command line is the "#"-sign, the remainder of the line is ignored.

remove playername

Removes the named player from the game. The race and all its works are completely erased.

save filename

Saves the state of the game into the specified file in the working directory. Only allowed once the game is running.

meta comment

Sets the rest of the line to be the server's description which is sent to the Freeciv metaserver. You may want to use this instead of the default description of "(default)"; for instance "daily restart."

h or **help**

Both of these display a list of the available commands.

l

Lists the players in the game.

ai race

If the named race is run by a human player, it is changed to be controlled by an AI, of the skill set by the last easy/normal/hard command. If it is controlled by an AI, it is changed to be controlled by a human player (even though the player might not be connected).

create playername

Creates a new player in the game, run by an AI. This is only allowed before you start the game with the "s" command. At that time, if the player name is that of the default leader of some race, and that race was chosen by any human player, the AI will be given that race; otherwise it will get an available race at random.

crash

This command immediately crashes the server. It makes no backup file. Its purpose, in addition to quitting, is to cause a core file to be produced for debugging.

log string

Writes to the log file (opened by the command line `-f` option) the rest of the line.

easy/normal/hard [playername]

These commands set the skill levels of AI players to the specific state; the names of the levels are relative (see distribution file README.AI for a note on the AI's). If the playername is specified then the skill of that one race is set; otherwise it sets the skill of AI's created in the future.

q

Quit game. Does not make a backup.

c playername

Cuts connection to a player. The race continues to exist, and yearly updates will update its status, but it will enter no orders. If timeout is not set, this will effectively block the game until a client reconnects or an AI is assigned to play the race.

show

Shows all current option settings

explain

Shows a list of the settings.

explain setting

Provides built-in help for an option setting.

set setting value

Sets an option setting to the specified value.

score

Displays the current score.

s

Starts the game.

Option Settings

These are the options that may be set with the "set" command, shown with the "show" command, and explained with the "explain" command. Many, but not all, are automatically sent to client programs; some are private to the server. Most options have for their values simple integers.

The detailed descriptions of the options are best found via the server's own "explain" command. Repeating those descriptions here would only require maintaining them in two places.

The following options may only be set if the map has not been generated. They are used in the map generation process, establish the general habitability of the map: xsize, ysize, generator, landmass, mountains, rivers, forests, swamps, deserts, seed, grass.

The first three are of interest to most users. The xsize and ysize options define the size of the map, which should be based on the number of players and the planned length of the game. The default map size of 80x50 is big enough for a fairly quick two player game, but will be a frustratingly fast game for more than three people. More than three should try 80x80; for five or more you should consider 100x100.

The "generator" option alters the map generation process. The default generator can produce tiny islands from which it is impossible to make a viable nation. Map generator 2, on 80x50 maps, makes seven islands of equal size; generator 3 will also create some small islands.

The following options may be set if the map has not been generated, or if the map is already generated but is a scenario's predefined map and the game has not actually started. They also relate to map and game creation, on a different level: randseed, specials, huts, minplayers, maxplayers, aifill, settlers, explorer, gold, techlevel, techs, units, buildings, researchspeed, techpenalty, diplcost, conquercost, freecost, railprod, railfood, railtrade, foodbox, aqueductloss, unhappyysize, cityfactor, razechange, civstyle.

The options `minplayers`, `maxplayers`, and `aifill` will be useful to most players. The options `techs`, `units`, and `buildings` are for selecting variant rulesets. The rest are mainly of interest to scenario developers.

The following may be set even after the game has started: `dipchance`, `spacerace`, `civilwarsize`, `endyear`, `timeout`, `saveturns`, `scorelog`, `gamelog`.

A note about the `timeout` option is necessary. Recall the description above of the end-of-turn updates that take place when all players have entered Next Turn. If there are just a few players, in close communication, leaving it in this mode is reasonable. If there are more than just a few, or if there will likely be breaks where one leaves for a minute or so and you don't want to wait, you can set the `timeout` to some interval such as 60 seconds. In that case, 60 seconds after one update, the next will take place even if a player has not entered Next Turn. Later in the game this can be annoying, and larger timeouts will be necessary. In general, the more players you have, the longer a timeout you will need, but be aware that going above 300 seconds will bother players.

You may wish to make adjustments in various settings (`gold`, `techlevel`, `researchspeed`, etc.) to make the game easier; if you are inexperienced, and playing with inexperienced people, probably no one will object. But this is not a good way to learn how to play; starting with uncommon advantages makes it more difficult for you to learn how to copy with the common settings. For the most part, the game settings are there for those who wish to make an entire "scenario", an alternative world with its own environment and consistency, such as a Fall of Rome scenario, or a Medieval Europe scenario.

Files and Environment

The Freeciv server accepts these environment variables:

`FREECIV_CAPS`

A string containing a list of "capabilities" provided by the server. The compiled-in default should be correct for most purposes, but if you are familiar with the capability facility in the source you may use it to enforce some constraints between clients and server.

`FREECIV_DATADIR`

A string containing the name of the directory in which the Freeciv data files are stored. If not specified, the default `/usr/local/share/freeciv` is used.

The Freeciv server requires the following files in the Freeciv data directory, which is `/usr/local/share/freeciv` by default:

```
default/buildings.ruleset
default/techs.ruleset
default/units.ruleset
```

These are the default rule sets used for the game. Alternate sets of rules can be installed with the option settings `"techs"`, `"units"`, and `"buildings"`, each of which provides a name to replace the directory name `"default"`.