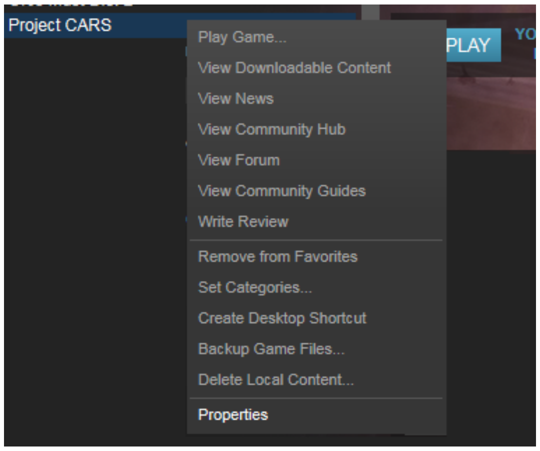
Quick and short instructions for running and using Project CARS dedicated servers on PC. Last updated 27.2.2015.

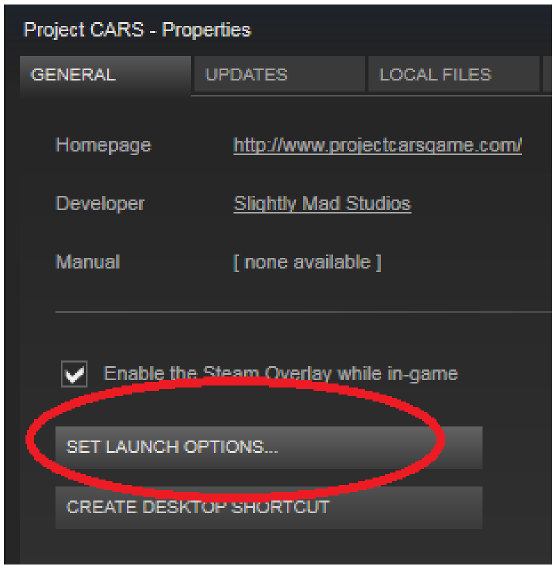
Using Dedicated Servers from the game

In current build the game does not yet have the GUI updated for Dedicated Servers so to use them you will have to utilize the command-line.

To change the arguments passed to the game from Steam, go to your Library, right click Project CARS and choose Properties.



Choose SET LAUNCH OPTIONS on the GENERAL tab:



And then enter the command-line options in the box that appears.

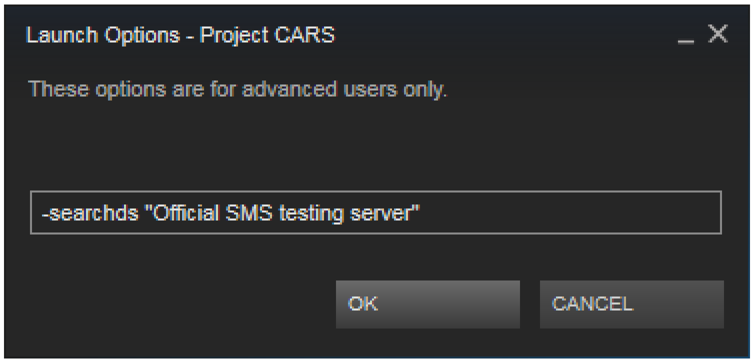
You can't use the built-in Steam's server browser to access Project CARS dedicated servers.

Creating multiplayer session hosted on a DS

Creating multiplayer sessions that are hosted on a Dedicated Server is not very different from creating peer-to-peer session. In fact, right now it's exactly the same, and the type of the session created when you click the "Create" button is can be controlled only from the command-line. There are two command-line options you can pass to the game:

- -autods: In this mode the game will enumerate all dedicated servers that are currently available - the servers must be public, not password-protected and not used by the game. It first looks for servers available on your LAN, then the Internet, and automatically pick a server with the best latency if there are any public servers available.
- -searchds "SERVER NAME": Searches for available dedicated server matching SERVER NAME. If there are multiple servers matching that name (as substring) the server with the best latency will be used. So for example '-searchds "test"' would match servers named "SMS test server" or "my testing server". The server can be password-protected, in which case the game will ask you for the password before creating a session on the server.

So for example to create a session on one of our public dedicated servers you would use this command-line argument:


Launch Options - Project CARS

These options are for advanced users only.

```
-searchds "Official SMS testing server"
```

Note that we can't guarantee that this server will be always available. And even if it is, there might not be enough official servers running to satisfy everyone attempting to host a game on them - the Create attempt will fail if all servers matching this name are already in use.

Proper GUI to create servers hosted on dedicated servers will be added later.

Joining multiplayer session hosted on a DS

Searching for and joining multiplayer sessions hosted on dedicated server is exactly the same as joining peer-to-peer sessions. You can use the browser, or join via multiplayer invitations or friend lists. The browser has not yet been updated to show which sessions are hosted on a server and which are peer-to-peer, but by default the session's name will be the name of the server when hosted, not the name of the user who created the session.

Dedicated Server and password-protected sessions

The password that can be configured in the Create options is not used when creating sessions hosted on a dedicated server. Instead the password is controlled by the administrator of the dedicated server, and must be entered when creating a new session on the server or when joining the session. There is just one password for the server, the same password is required for Create and Join. This is unlikely to change in the future because it would lead to too many confusing options, and we'd like to have empty servers appear in the Browser as well, where joining them would automatically create a session with default settings.

Naming sessions hosted on Dedicated Server

By default new sessions created on a dedicated servers use the server's name as the session's name. This makes it easier to tell which sessions are peer-to-peer (named after the user who created them) and which ones are hosted on a server (named after the server). However the user creating the session can override the name in both cases, only when the name is left empty in the options will the default be used.

Servers hosted by SMS

Generally we will not be hosting sufficient amount of servers for Project CARS to let anyone play on them. During WMD pre-release period we will have at least some servers up most of the time hosted on our servers. This is mostly so we can test them and analyze logs if anything wrong happens, so we can’t guarantee anything about their availability and uptime.

The public servers will be always called “Official SMS testing server” with the server’s number attached (or nothing attached when we run only one server). We might also have “Internal SMS testing server” available, those will be password-protected and intended for internal testing.

Server browser

Until in-game server browser is available, you can view the full list of running servers at http://cars-stats-steam.wmdportal.com/index.php/servers

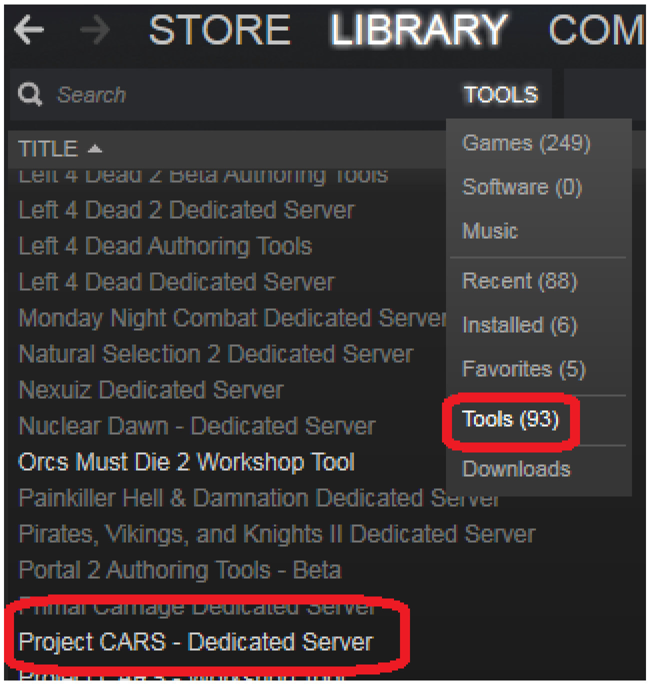
Hosting Dedicated Servers

The dedicated server is available on Steam to all users, not just owners of the game. So you can download and run the server anywhere. Windows and Linux versions are available, Steam will download the appropriate build for your current platform automatically.

Only command-line version of the server is available for now, with simple text file used to configure it. The executable is called DedicatedServerCmd.exe (on Windows) or DedicatedServerCmd (on Linux).

Download and update DS via Steam Client

You can download the server from the Steam’s library. Go to your Library, then switch it from Games to Tools, search for Project CARS - Dedicated Server and install it. Since the server is a command-line application, while you can run it from the library it won’t show any nice configuration window. So first I recommend configuring it.


STORE
LIBRARY
COM
Search
TOOLS
TITLE
Left 4 Dead 2 Beta Authoring Tools
Left 4 Dead 2 Dedicated Server
Left 4 Dead Authoring Tools
Left 4 Dead Dedicated Server
Monday Night Combat Dedicated Server
Natural Selection 2 Dedicated Server
Nexuiz Dedicated Server
Nuclear Dawn - Dedicated Server
Orcs Must Die 2 Workshop Tool
Painkiller Hell & Damnation Dedicated Server
Pirates, Vikings, and Knights II Dedicated Server
Portal 2 Authoring Tools - Beta
Primal Carnage Dedicated Server
Project CARS - Dedicated Server
Games (249)
Software (0)
Music
Recent (88)
Installed (6)
Favorites (5)
Tools (93)
Downloads

Download and update DS via SteamCMD

You don’t have to have Steam Client installed to be able to download and run the server. The alternative to full-blown Steam Client is the SteamCMD utility. Full SteamCMD documentation and download links are available here: https://developer.valvesoftware.com/wiki/SteamCMD

The installation via SteamCMD will differ slightly depending on your platform and the exact version of the OS. Right now SteamCMD is 32-bit only, so on 64-bit Linux platforms you might have to perform extra steps. For example when I was installing it on our 64-bit Ubuntu-based testing server, the steps were:

- Install 32-bit gcc support: “apt-get install lib32gcc1”
- Create Steam user: “adduser steam”. You will also probably want to edit /etc/sudoers to let select user switch to this account, and remove password from this user to prevent direct remote login: “passwd -l steam”.
- Switch to the Steam account: “sudo su - steam”
- And install SteamCMD:
  - cd
  - mkdir steamcmd
  - cd steamcmd
  - wget http://media.steampowered.com/installer/steamcmd_linux.tar.gz
  - tar -xvzf steamcmd_linux.tar.gz

The to install the dedicated server via SteamCMD (or to update it, that’s done the same way), while running as the steam user:

- cd
- cd steamcmd
- ./steamcmd.sh
- login anonymous
- force_install_dir ./pcars
- app_update 332670 validate
- quit

Or you can just run it all on with a single command

- ./steamcmd.sh +login anonymous +force_install_dir ./pcars +app_update 332670 validate +quit

Of course the "force_install_dir" directory is entirely up to you.

Note that on 64-bit Linux you might have to copy some *.so files from steamcmd/linux32 directory to steamcmd/pcars/lib32 if you have troubles starting it. The server is also 32-bit and Steam does not bundle all necessary libs by default with the server package.

DS configuration

After you install the server, before running it, it's strongly recommended to customize the server's config. The configuration is read from file "server.cfg" from the current working directory when the server starts. By default there is no such file, but there is a subdirectory config_sample with some examples. It's strongly recommended to start with those. So usually you would start by copying config_sample/server.cfg to ./server.cfg and then edit it.

There is also more advanced config sample available in config_sample/server_with_lists.cfg, config_sample/blacklist.cfg and config_sample/whitelist.cfg. Use those files if you want to use blacklists or whitelists on your server. Don't forget to rename server_with_lists.cfg to server.cfg. Also make sure you edit the whitelist in that config - if there is any whitelist set, **only** users in that list will be able to connect to the server, and the sample config has just my ID in the list.

The options in the sample will usually match the default options baked into the executable. So if you don't want to customize some of them, feel free to completely remove them from the config. The only exception are the sample blacklist and whitelist, those are empty by default but set to non-empty lists to provide examples.

The config's format is loosely based on JSON, but it's a bit more free-form and allows you to omit commas between individual options, and it's not required to have the whole config enclosed in top-level { } block.

The syntax to set an option's value is:

OPTION_NAME : *OPTION_VALUE*

The option names are case sensitive.

Strings (so all option names for example) need to be double-quoted, values can be strings, integers or booleans ("true" or "false", without any quotes). In some cases values can also be lists or objects. The syntax for a list option value is:

OPTION_NAME : [*OPTION_VALUE_1*, *OPTION_VALUE_2…*]

and the syntax for an object option value is:

OPTION_NAME : { *KEY_1* : *VALUE_1*, *KEY_2* : *VALUE_2…* }

Some of the useful options you will want to change are:

- "//", "#" or "rem": This options will be ignored, you can use them to write comments into their values.
- "name": The server's name. This will appear in server browser (when implemented) and will be also the default name of sessions hosted on the server.
- "password": The password required to create sessions on the server as well as to join the sessions. Password set in Create options is ignored when using a dedicated server.
- "blackList": List of Steam IDs to never let onto the server. The value can be either a list or an object. In both cases only the values are used, which allows you to use the map syntax to write any comments in the "keys" of the map - so for example the user's name and reason for banning.The values can be either Steam IDs of the users to bad, or file names (if they are strings). The files referenced from the blacklist should contain just a plain list or object in similar format, with more Steam IDs. Steam IDs and file names can be mixed in any order. This for example allows you to reference two files with blacklists from the main config - one file with users banned from this specific server, and another file with globally well-known bad people banned from a set of servers (admins can then share this list on a public web site for anyone, for example).

There are more options available, all of the supported ones are included in the sample configs, including short documentation.

Running DS in background on Linux (using screen)

There are many ways to run dedicated server in Linux environment. You could write an init.d script and then use the usual server/daemon commands to control it. We don't provide such scripts at the moment, as they tend to slightly differ from OS to OS.

One easy way to run a server via SSH without having to stay remotely logged in forever is "screen". On Debian/Ubuntu-based OS you would use "apt-get install screen" to install it.

To run the server for the first time:

- SSH to the server
- sudo su - steam
- script /dev/null

- screen (then press enter to confirm the initial screen, read through it if you want to)
- The following commands will be run inside the screen
    - cd
    - cd steamcmd/pcars
    - ./DedicatedServerCmd 2>&1 | tee server.log
- Press ctrl-a followed by d to detach from the screen.
- Press ctrl-d to leave the screen, then few more times to leave the script/sudo/logout.

To reattach to the server later:

- SSH to the server
- sudo su - steam
- You can review server.log in /home/steam/steamcmd/pcars ; follow to reattach to the server
- script /dev/null
- screen -d -r

Note that the “script /dev/null” command is a semi-ugly trick to make screen work in sudo sessions.