CCCheat

CCCheat is a Windows or Linux application that connects to your PS3 via CCAPI. The application allows you to scan your PS3 memory to find in-game values that can be modified. Essentially allowing you to modify or cheat in your game. Have fun with it and don’t use it online. It won’t work.

This documentation was created for version 1.20 of CCCheat.

CCCheat is open source so modify it.

Contents

[Connecting To the System 2](#_Toc400720211)

[Scanning For Valid Memory Ranges 2](#_Toc400720212)

[Performing a Code Search 3](#_Toc400720213)

[Choosing a Code from the Results 5](#_Toc400720214)

[The Code Table 5](#_Toc400720215)

[Viewing Values Around a Code 7](#_Toc400720216)

[Trainer Creator 7](#_Toc400720217)

[In Closing 9](#_Toc400720218)

Connecting To the System

The first step to using CCCheat is to connect to you PS3. This is done by selecting the correct CCAPI version that is being run on the PS3, and also filling in the IP address assigned to your PS3.

* IP Address – This is where you put in the IP address of your PS3.
* CCAPI Version – This is the version of CCAPI running on the system. If you do not know the correct version, just try both. One is bound to work, but if it doesn’t, make sure you are actually running CCAPI on your system.

After the information is entered, press ‘Connect.’ The light on the Connect button should change from clear to yellow when connecting, and then to green when connection with the PS3 is established.

Scanning For Valid Memory Ranges

When trying to find a code for a particular game, you should first find the general memory range that it is located. By default, CCCheat shows a range of 0x00000000 to 0xFFFFFFFF. You can perform a code search over this range, but it would take quite a long time to complete. Instead, you should scan for ranges to search over.

* Find Ranges – This will search the PS3 for valid memory ranges. Once finished, the range list will be updated with sets of ranges to search for codes. This should only be used inside a game, and generally only needs to be done once since the ranges can be saved afterwards.
* Save Ranges – This saves the list of ranges into a file for future use.
* Load Ranges – This will load a range of memory regions to scan from a file.
* Clear Ranges – This clears the list of ranges back to the default 0x00000000 – 0xFFFFFFFF.
* Delete Range – This deletes the currently selected ranges.
* Add Range – You can manually add a range of memory to the list. This is useful if you know where a particular code is stored and want to narrow the amount of memory the searcher has to scan. The two boxes require the hexadecimal start and end value for the memory range.

After connecting, the next step would be to either perform a ‘Find Ranges’ or ‘Load Ranges.’ Once the ranges are found, make sure to save them so you do not have to find them again later on.

Performing a Code Search

A code search will scan memory ranges selected in the Range section. The search will find all memory locations that match your search criteria. Searching is done in several rounds. Generally, you will search for a value, have that value change in the game, perform a search on the new value, and repeat until the number of memory locations is narrowed down. Once the list is narrow enough, you can add the codes to the code table to find out if a successful code was found.

Setting Up a Search

To perform a search, first a memory region needs to be selected from the ranges list. Multiple ranges can be scanned at once.

* Search Type – This dropdown allows you to select the how the code searcher will function.
  + Exact Value – This searches for a value in the game that you can see. For example, you may know the amount of ammo your weapon has, or the number of HP your character has. When searching for a code, this search type will match all codes that are equal to this value.
  + Less Than Value – This performs a search for a value that is less than the inputted value.
  + Greater Than Value - This performs a search for a value that is greater than the inputted value.
  + Not Equal To Value - This performs a search for a value that is greater or less than the inputted value.
  + Fuzzy – This search type allows you to scan for codes that that have an unknown value. For example, your character may have a health meter. These type of searches can also work with values that your know, but may be represented a different way in the game.
  + Pointer – This search type will search for codes that change often. You might find a code that works fine, but next time the game is started the code no longer works. This will scan to find a common base for the changing code and follow it so that the code works all the time.
* Value – This is the value that the searcher will find when scanning. This can be a base 10 integer or a decimal number depending on the Value Type. If the Hex box is checked, the value will transform into a base 16 number. Values are not available when doing a Fuzzy Search Type. This is because we do not know the value we are searching.
* Value Type – Values can be stored in memory in several different capacities. This drop down allows you to select how ‘big’ the value you are searching is.
  + 1 Byte – This will store a value that is 8 bits in size. If the value you are searching for is between -127 to 127 or 0 to 255, then a 1 byte scan will work. If you anticipate that the value can change to something higher than 255, use 2 or 4 byte types.
  + 2 Byte - This will store a value that is 16 bits in size. If the value you are searching for is between -32767 to 32767 or 0 to 65535, then a 2 byte scan will work. If you anticipate that the value can change to something higher than 65535, use the 4 byte type.
  + 4 Byte - This will store a value that is 32 bits in size. This is generally the most common type of value you will search. If you are searching for values that can be large, such as money. You will use this value type.
  + Float - This will store a value that is also 32 bits in size, but interpreted as a decimal value. With Float, your value can be fractional numbers. This is also a commonly used type. Use this type when searching for percentages, meter bars, and money (sometimes.)

Running a Search

Once all the search parameters are filled in, start the search by pressing the “New Search” button. This will start the search. The progress the search can be seen on the progress bar at the bottom of the window. Once a search is complete, you can see the results under the Results section on the upper right side of the screen. If the number of the results are high, you should change the value in game (lose health or money.) After that, input the new value in the Search section and press “Next Scan.” When this scan completes, you should have a fewer number of results. Repeat these steps until you are confident that one of the results is the code that you are trying to find.

Resetting a Search

If you want to start a new search in the middle of a current search process or you find that the number of results become 0. You can reset the search process by clicking the “Reset Scan” button. This will clear all results so that a new search can be started.

Choosing a Code from the Results

After a search has been completed, the Results section will display the number of results found as well as the results themselves. The list of results has a limit of 200 items. If you have more than 200 results, you will only be shown the first 200. If you want to add a code to the code table from the Results section, simply highlight the result(s) and press the “Add Result” button. If you know a result from this section is not the code that you are looking for, you can highlight that code and press the “Delete Result” button to remove it from the list. Finally, if your in-game value changes and you want to see if the value from your result list has changed as well, press the “Refresh” button. This will re-read the values from all the items on the list and update them. This can help to avoid another search round.

Once codes are added to the code table from the results section, they can be modified.

The Code Table

The code table will list all the codes that you have found for the game. Codes can be manually added to this list or added from the Results section. Once a code is added, it can be given a description, a new value, a type (1, 2, 4 bytes, or Float) and a freeze status. Codes can also be saved and loaded from a file for later use. Finally, a trainer can be created from the found codes and distributed.

Adding a Code from the Results Section

After you have found a code from scanning and added it from the Results section, you will see that code on the code table. Initially the code table will show that the code has no description, but you can modify this as well as the other parameters of the code.

* Description – Double click the description cell of a code to edit the code’s name or description. (Click outside the cell to apply this description)
* Address – Double click the address cell to change the code’s address. This should not be done for codes added from the Result section or your code will no longer work.
* Value – Double click the value cell to change the in-game value of the code. Once this is applied, the value will be pushed to the game’s memory one time. You should see the change in the game if the code is correct.
* Type – Double click this cell for a dropdown that will modify the capacity of the value. This is something that is not done very often.
* Frozen – Press this checkbox to enable freezing the code. This will continuously modify the in-game value while the CCCheat application is connected. For example, if you want your HP in a game to remain at 9999, you would input 9999 in the value cell, and make sure the frozen checkbox is checked.

Manually Adding a Code

By pressing the “New Code” button, you can add your own codes to the code table. There are two types of codes you can add:

* Address – This is the mode common type of code. It is just a memory location of the value in the PS3’s memory. If you know the address of a certain value that you want to modify, chose this type.
* Pointer – This type consist of a memory location that ‘points’ to one or more locations that eventually contain the value to modify. Pointers are used in cases where the address that contains your value constantly changes. In general, you will not add a pointer manually, but instead add one by completing a search.

When an address is added manually, you will have to edit the description for the code and also the address. The default address for a manually added code is 0x00000000. Make sure to change this to your desired address. Once added, the code can be modified just as described in the previous section.

Editing the Code Table

Besides adding new codes, any selected code can be deleted by pressing the “Delete Code” button. This simply removes the code from the code table.

To save codes to a file for later use, press the “Save Codes” button. This will ask you for a filename to save the codes.

Press the “Load Codes” button to load a previously saved code file. When a file is loaded, all codes will first be cleared from the code table. Also, values that were frozen will have to be re-frozen. The values of your codes are also saved to the code file, so after loading the file, the values will be modified in-game to what is stored in the saved code file.

Creating a Trainer

When one or more codes are in the code table, you can press the “Create Trainer” button to create a whole new application that will apply the codes that you’ve found. You can distribute this trainer. See the Trainer section for details.

Viewing Values Around a Code

Often you can find other codes for a game near the same address of a particular code that you’ve added to the code table. You can view all values around a code and potentially find other codes without having to run another search process. For example, you found your character’s HP code, but you want to modify the characters MAX HP. You can sometimes the MAX HP code very close to your HP code.

To view the values around a code, right click the address cell of a code and select “View Nearby.” This will open a new window contain a list of addresses and values around the location of your code. The green highlighted item on the list is where your code is stored in memory. The value should correspond to the value of your code in the code table. You can browse this list for other values that correspond to in-game values that you are interested in. For example, if your HP code is located at address 0x00000400 and it is a 4-byte type code, you might see that your MAX HP value is located at 0x00000404 in the value viewer.

If you find a value that you are interested in, and want to add that code to your code table, simply select the value and click on the “Add Code” button. This will add the new code to your code table. Make sure you modify the description of this new code.

You can also change the way that the values are interpreted by click on the Type drop box. You can select 1, 2, 4 byte and Float. If your code you are searching around is an integer value, but you want to find codes that have decimal values nearby, you can simply change the Type to Float.

1, 2, and 4 byte values are only integer values, but can be positive or negative. If the value you browsing has an in-game negative value, check the “Signed” button to display values that are potentially negative.

Trainer Creator

Trainers can be created from the codes in the code table. When creating a trainer, you want to make sure that the code from your code table will work reliably. The best way to check this is make sure your code works after quitting the game completely. Reload the game, and if your code still works, there is a pretty good chance it will work all the time.

To create a trainer, click on the “Create Trainer” button. This will take you to a new window where you can enter some information about the trainer you are about to create.

* Trainer Name – This is what you want the title of your trainer to be. This will be the title of the window.
* Author – This is your name.
* Game Title – This is the name of the game you are training.
* Game Region – This is the region (USA, EU, JP, etc….) You could also put in the product ID (SLUS, SLES, SLPS, etc.…)
* Trainer Information – This is just any other information you want to include with your trainer. It would be a good idea to include what version of the game you trained. For example, codes for version 1.01 of a game will probably not work in version 1.00.
* Codes – This is the actual trainer codes that the user will be able to enable/disable inside the trainer.

Once all the information is entered, you will need to add your codes from your code table to the trainer. This is done by pressing the “New” button in the Codes section of the window.

After hitting the “New” button, a new window will appear asking for some more information about the code. This consist of the following

* Code Name – This is what you want the code to be called or described in the trainer.
* Conditional – This checkbox allows you to assign a code from the code table to be a trigger that will enable/disable other codes. For example, if you want your character’s health to be set to 1000 only when it drops lower than 100, then your health code would be a trigger. You could also use joker codes as triggers. This will allow you to activate a certain code whenever key combinations are pressed on the controller. The condition code is completely optional. Without one, the codes set below will activate whenever they are enabled in the trainer.
* IF – This set of 3 inputs consist of a dropdown to select a trigger code, an operation that will evaluate the trigger, and a value that will cause the trigger. In our previous example, we set the trigger code to Health, then the operation to Less Than (<), and the value to 100. This will cause the codes we set below to trigger when Health < 100.
* Set Codes – These dropdown/value combos allow you to select a code to enable, and the value that you want the code to set. From our previous example again, we would select Health in the dropdown, and set the value to 1000.

Additional codes can be added to this by selecting the “Add” button, while you can also delete a selected code by pressing the “Delete” button. Once all codes are set to your liking, press the “OK” button.

You will now be back to your trainer information window with a new code in the Codes section. This code can be edited by selecting the code and clicking the ‘Edit’ button.

This whole process can be repeated to add additional codes to your trainer. Once you are satisfied, you can press the “Create” button to export a new executable trainer. This trainer will be created in the same directory where CCCheat resides. The name of our trainer executable will be a combination of the trainer name and game title. You can run your trainer now and make sure it works that way that you want it.

In Closing

Thank you again for trying out CCCheat. I hope that it works the way that you want. I will continue to work on the program to fix bugs and implement new features. Remember this is an open source project so feel free take a look at the code and modify it in any way that you want.

<https://github.com/primetime00/CCCheat>

primetime00