Guide to Using
Campaign Manager for Windows
version 1.15

By New Fangled Software

Contents

Welcome!

Welcome to Campaign Manager, the computer solution to all the hassles of managing a rich, varied and unique campaign. By creating this program I hope to relieve some of the burden that goes along with customizing a campaign. Campaign Manager was designed to allow Game Masters to easily create, modify and keep track of all the information he or she uses. It is also designed to allow players to create characters specifically for a given campaign.

Most GMs start out working within the rules for a given role-playing system laid out clearly and concisely in the rule books. But it usually doesn't take very long for the GM to start modifying those rules in one way or another. If nothing else a GM will want to create his own monsters, races, magical items, spells, etc. The paperwork for all this creativity can be mind boggling. Simply keeping track of all the rules in all the official supplements can be a chore. And what about all those articles from gaming magazines?

But now that you have Campaign Manager you have a program designed for storage and retrieval of all that information. But CampMan is not just a database. Within this program you can specify how all your objects interrelate. In addition, by storing all this information in CampMan you simultaneously create a unique character creation system designed specifically for your campaign. You can let your players use

this program to build characters which follow the rules you've laid out, use the objects you've created, and reference only the supplements you've added.

Included Topics:

CampMan's Purpose

Campaign Manager is intended to be one solution to that endless of GM quests, to find an easy, simple way to create and manage a unique, personalized campaign. CampMan is not intended to teach you how to play role-playing games. It is assumed that you already know how to play one of the many role-playing games on the market. The game(s) you know how to play will probably serve as your guide to what kind of campaign you wish to create with this program.

Although I have tried to make CampMan easy to use, that is not to say that CampMan is not a powerful program which will take practice to understand and use properly. It was purposely designed to be generic enough to adapt to most games on the market and to allow you to come up with your own, unique gaming system. It is the very fact that CampMan is generic that makes it so powerful.

next topic:

Who needs this program?

Who needs this program?

Roleplayers

Anyone who is sick and tired of searching through a half-dozen rule books to create their characters. Campaign Manager lets you keep all that information in one place.

Game Masters

Every GM who has a bookshelf full of rule books, magazine articles and handwritten notes they have to take to all their games. Every GM who has to constantly look over his/her players' shoulders while their creating characters to remind them of the rules. Campaign Manager not only lets you store and retrieve all the rules, objects and paperwork but also give's you a custom character creation program for your players.

next topic:

This is shareware

This is shareware

This version of Campaign Manager has been released as shareware. If I had chosen a commercial route, it might very well have been several more years until this program was released. It also would probably have cost more for you to buy the program.

By releasing CampMan as shareware, I grant you the right to use the program for an evaluation period of 30 days. If you find the program to your liking and wish to continue using it, you must pay a $20 registration fee.

Why you should register

As a self-employed and independent programmer I have to make a living. It is your registration fee which pays my bills. If a program I create does not bring in any money, I must abandon its development in favor of those that do.

This program is nowhere near the final version I hope to create. There are many features I plan to include in future versions and I have already thought of ways to improve the current features. By showing your support, you encourage me to go ahead with those plans. So if you like the current version and wish to see better, improved versions it is in your best interests to register.

next topic:

Limitations

Limitations

Campaign manager has certain limitations a would-be user should keep in mind.

- Although characteristics have been designed to accept any value, attributes are designed to accept only positive, non-zero values. Since there exist easy ways around this limitation, current plans do not include changing this aspect of the program.
- Although this program can print notes, gm notes, creatures, races, and characters, it can not distribute the information over multiple pages. This can cause the information to "scroll off page" if printed in too large a font. But the fact that you can print in either portrait or landscape mode in fonts as small as 8 pts should go a long way in alleviating this problem. (You may be surprised at how much information can fit on a page when printed in landscape with an 8 pt font)
- CampMan does not currently support printers which aren't BitBlt compatible (i.e. plotters or text only printers)
- This program does not have WYSIWYG capabilities. That is, you can't view a note/character on screen the way it will print. This will not be too much of a problem for most situations. However, there is one situation where this is a problem... tables of information in notes. For more information on this problem and circumventing it as much as possible see Printing and The Problem With Tables Printing.

next topic:

How to use this manual

How to use this manual

This manual is distributed as a Windows help file. The capability of having on-line access to this material as well as the hyper-textual nature of the medium enhances the usefulness of the manual.

The first part, "Introduction and Getting Started", contains chapters on Installation and Starting the Program.

The second part, "Tutorials", has two tutorials on how to use Campaign Manager. If you are player, you should go through the first tutorial Building A Character.

The first tutorial guides you through the creation of a character for a campaign (tutorial.cmp) specifically designed for the tutorial. Once you understand how to build a character for this sample campaign, you should have little trouble building characters for the specific campaign(s) your GM will create. GMs may also find this tutorial useful in learning exactly how the objects they create relate to building characters.

The second tutorial is for GMs. Building A Campaign guides you through the creation (from scratch) of the campaign used in the previous tutorial.

The last part of this manual, "Reference", is devoted to chapters dealing with all the different aspects of the program. Unlike the previous two parts, this part is not intended to be read from beginning to end, but to be explored when your understanding of a particular aspect of the program needs to be refreshed or deepened.

To use this manual while running the program, simply choose a topic from the "Help" menu or press the F1 key. The "Reference" and "Menus" topics bring up different parts of the manual depending on which window you are using.

What's New

Filter Dialog

The filter used to select what objects to list in the main window has been moved to a dialog box. Also, a default filter can now be saved that will be used whenever CampMan is launched.

In addition, while both the Filter dialog and Note Search dialog are open, it is possible to "link" object filtering to either include only those notes listed or to exclude all the notes listed in the Note Search Dialog's found list.

For more information see "The Filter Dialog"

Append Button in Note Search Dialog

A new button has been added to the Note Search dialog that will search for notes which match the given criteria and append them to the current list of found notes.

For more information see The Note Search Dialog

Copy/Paste Object Names in lists

It is now possible to use "Ctrl-C" to copy the selected object name in the main window's objects list to the Clipboard. Similarly, it is possible to use "Ctrl-V" to insert the name in a race/character's attributes or items list.

For more information see the side notes in The Main Window and Characters

Print Cost Breakdown for Races and Characters

You can now print a listing of the individual costs which contribute to a race/character's total cost.

For more information see Printing

VGASYSI.FON Replacement VGA System Font

CampMan uses the Windows "system" font which is "sans-serif." This causes the letters "I" and "l" to be indistinguishable from one another. I have therefore included a replacement system font.

VGASYSI.FON is exactly the same file as VGASYS.FON except that "serifs" have been added to capital i's.

For more information see Replacing the VGA system font

Installation

The following table lists the files that come with the program. To install the program, create a directory to hold the Campaign Manager program files and use the following table to determine what files to copy where:

File	Copy To	Description
campman.exe	CampMan directory	program executable
campman.hlp	CampMan directory	program help file
tutorial.cmp	CampMan directory	tutorial campaign
vwbas20.dll	CampMan directory	runtime library
vwvm20.dll	CampMan directory	runtime library
vwfloat.dll †	CampMan directory	floating-point emulator
commdlg.dll ‡	windows\system directory	common resource library
winhelp.exe ‡	windows directory	Windows 3.1 help utility
winhelp.hlp ‡	windows directory	winhelp.exe help file
vgasysi.fon	CampMan directory	replacement VGA system font

† the floating-point emulator is only needed if your computer does not have an 80x87 math coprocessor.

‡ these files are only needed if you are running the program under Windows 3.0

subtopics:

Creating a CampMan Icon

The following steps are used to create an icon in the Program Manager that comes with windows. If you use a different program launcher, you will have to consult its documentation.

1. Activate a group window in which to place the new icon.
2. Select "New" from the "File" menu.
3. Click on Program Item and then press OK.
4. Type **Campaign Manager** in the description field.
5. Type **campman.exe** into the command line field.
6. Enter the full path to the program directory you created into the working directory field.
7. When finished, press the OK button and the icon will be created.

Replacing the VGA system font

To use the replacement VGA system font, your system must currently use the VGASYS.FON system font that comes with Windows 3.1 and you must edit the SYSTEM.INI file as follows:

1. Using the File Manager, copy SYSTEM.INI to SYSTEM.BAK
2. Launch the Notepad utility that comes with Windows.
3. Select "Open" from the "File" menu.
4. Type **system.ini** in the filename field and press OK.
5. Search for the line "fonts.fon=vgasys.fon"
6. Replace the text "vgasys.fon" with the path/filename to the new font (i.e. "c:\campman\vgasysi.fon")
7. Select "Save" from the "File" menu.
8. Exit and restart Windows.

The system font should now look exactly as it did before, except that capital i's now have serifs to distinguish them from l's. If Windows won't start are there are any other problems, simply recopy your backup file, SYSTEM.BAK, to SYSTEM.INI

Starting the Program

To launch the program, double click on the icon you made during Installation. After the log on screens you will be prompted for your name. If you want to start the program in Player Access mode simply type in your name and press OK. If you wish to start the program in GM Access mode, type in "GM" and press OK. You will then be asked to verify you identity by entering the GM's password. For more information see Player vs. GM Access.

Building A Character

This tutorial guides you through the creation of a sample character for the tutorial campaign provided with Campaign Manager for Windows. When you are done with the tutorial, you will have learned all you need to know to build characters for the campaign(s) your GM will create.

This tutorial assumes you are familiar with the Windows operating environment. In particular you should know how to move/size/close a window, enter/modify text, use a menu and click on a button or list item. If you are not familiar with Windows you should first go through the *Getting Started with Microsoft Windows* manual and the Windows tutorial.

Included Topics:

Creating a New Character

If you haven't done so already, start the program in Player Access mode. For more information, see Starting the Program.

The first window you will see after the startup dialogs is the main window. It presents you with a list of object names and types for the loaded campaign. At the moment, you will see only "Human" listed in the objects list. This is because we have not loaded a campaign into the program and you are seeing the default, empty campaign.

To load a campaign you must choose "Load" from the "Campaign" menu. This brings up a dialog box used to select the file containing the campaign you wish to load. Among the files listed should be "tutorial.cmp". Double-click on that filename. After a moment to load the file, you will see a new list of objects.

To create a character you need to select "Create Character" from the "Objects" menu. This will bring up a window titled "Character -- new" with the default statistics for a new character. Looking at the contents you will see that "Human" has been entered as the character's race and you have been entered as the character's player. All we need to do now is to give this character a name. Enter "Delgath" into the name field and select "Save" from the "Character" menu.

In the next section, we will see how to get information about the objects in the campaign and how to keep some notes about your character.

Next Topic:

Object Descriptions

Object Descriptions

When you saved your character you may have noticed that the main window was updated with the new name and object type. Try clicking on "Human" in the objects list. When you select a name, the types list is updated to show you all the different objects with that name. Click on "Race" in the types list and then select "Open" from the "Objects" menu. This will bring up a window to let you read the note attached to the race "Human".

Normally, the description of an object is kept in an attached note with the same name. This lets you find out about an object without digging through a book or asking the GM. You can also create and edit notes which accompany your characters. Use them to keep records of your characters' backgrounds, histories, and/or any other tidbits you might want to remember.

Let's create a note to accompany Delgath. We first need to have Delgath's character manager open. Click on "Delgath" in the object list and select "Open" from the "Objects" menu. Now select "Create Notes" from the "Character" menu. The note created will be titled "Delgath".

Put the following in the text window:

> "Delgath is a smart, tough but poor young man who makes his money by relieving it from the more fortunate in as inconspicuous a fashion as possible. He has not been on good terms with the local "guild of professionals" since his refusal to join a year ago. Overconfident, like many his age, he tends to be a flamboyant showoff although he abstains from drugs and alcohol. Only 5'6", he is sensitive about his height which may account for his fondness for high places. Del has fair skin, dark hair and dark eyes, weighs 110 lbs and is right handed."

After you're done editing the text you can "Save" the note and close the window. Let's go back to the character manager and see how characteristics work.

Next Topic:

Playing with Characteristics

Playing with Characteristics

You may have noticed the button in the top right corner with the text "cost: 0". In many campaigns you "buy" characteristics and abilities with experience points. This button shows the last calculated cost for the character. The cost is calculated whenever you open or save the character, choose "Recalculate" from the "Cost" menu, or press this button.

Going back to Delgath, we can see that he currently doesn't cost any points. But then, we haven't made any changes to his characteristics or attributes. First we'll assign some new values to his characteristics and get more familiar with the character manager's capabilities.

Each character manager has a box just underneath the race field to contain all the characteristics. Some characteristics may be assigned their value. These have entry fields next to their names. Other characteristics are calculated. Their values are updated when the cost is calculated. Finally there are random characteristics. These characteristics are generated when the character is created and then stay the same until the GM changes them.

OK, let's go back to Del's character manager and see this in action. As examples of random characteristics you should look at the Mana and Exp characteristics. Exp represents the experience awarded to you by the GM and always starts out at 0. Mana is the result of rolling a 4-sided die and subtracting 1. We will need a mana score that is not 0 in a later section of this tutorial. If Del has 0 mana, you may wish to delete and recreate him now.

To see how assigned and calculated characteristics work enter 8 into the ST field then press the "cost:" button. You'll see that "basic dmg" and "ch cost" have been updated and the button now reads "cost: -20".

Now try entering 15 into the ST field and 12 in the IT field but **don't** recalculate the cost. Then choose "Undo" from the "Character" menu. This causes the fields to return to their previous settings. Any time "Undo" is chosen, **all** fields are reset to the value they had just **after** the last calculation occurred.

Go ahead and enter the rest of Del's characteristics as follows: ST 8, IT 12, AGL 15. When you're done Del's cost should be 60. Save the changes you've made and we'll proceed to give him some attributes and possessions.

Next Topic:

Modifying Attributes

Modifying Attributes

A character isn't just a name with some characteristics attached. There are always going to be specific abilities, attributes and possessions which uniquely define the character. The character manager uses two lists to display a character's attributes and possessions. These lists work almost identically to each other to keep track of your modifications.

At the moment, Delgath's attribute list shows "Average Wealth" and the possessions list is empty. Choose "Add" from the "Attribute" menu. This places an entry field at the top of the attributes list. Type in "Poor" and press Enter. Poor will be added to Del's attributes and Average Wealth will disappear.

Average Wealth disappeared because Poor is an indication of wealth and the campaign has been designed so that a character can't have both attributes. In fact, the two attributes share the same note, titled "Wealth". Now try adding the "Sneak" attribute on your own.

Notice that the Sneak is followed by a 1 in the attributes list. This is the level that's been assigned to it. Unless the attribute's level is calculated, a level of one is automatically assigned to any attribute which you enter without a following number. Poor does not show a 1 because it is an "owned" attribute. You can't own Poor at a level higher than 1 so there is no reason to display a level for it. Try adding "Sneak 3" (don't forget the space). You'll see that Sneak has now been assigned a level of 3.

There is another way you could have changed Sneak's level. Try selecting Sneak and then typing 15 followed by Enter. An entry field popped up when you hit 1 didn't it! Whenever you start typing and one of the lists boxes has the input focus an entry field will pop up to accept your modifications. If you enter only a number, the number is interpreted as a new level for the currently selected entry.

Now try adding "Pick Pockets". A dialog box will tell you Pick Pockets is restricted to Delgath. That is because the campaign requires you to have "Thief's Training" before you can have "Pick Pockets". So add "Thief's Training 2" and then add "Pick Pockets". Since Pick Pockets is a "calculated" attribute, a level is calculated for Delgath and displayed. This level will be recalculated every time you calculate Delgath's cost.

Lets give Delgath some fighting abilities. Add the two attributes "Short Sword 18" and "Long Sword 16". This is fine, but if you look in the objects list you will see that there is an attribute called "Short Sword [fromLongSword]" which would probably be cheaper for Delgath. Try entering "Short Sword [fromLongSword]". The new attribute replaced "Short Sword" and took on its level. This is exactly the same way "Poor" replaced "Average Wealth" and for the same reasons.

Finish entering Delgath's attributes by adding "Blather 13" and "Horseriding 14". Save your changes and look at the cost button. It should read "cost: 66 1/2" if everything is correct. Now lets give Del some possessions.

Next Topic:

Adding Possessions

Adding Possessions

Adding and modifying possessions is exactly the same as adding and modifying attributes. You just use a different list. Try adding Del's possessions as follows: "Small Knife", "Short Sword", "Clothes: low class", "Shoes: pair", "Leather Jacket", "Ring: silver", "Horse [riding]", and "G.S. 11".

After you're done adding possessions, save the changes. Something went wrong, didn't it? A message box popped up telling you that "Poor" was restricted to Delgath. This is because Del has too much money (G.S.). But before we correct this problem, lets explore what exactly is going on.

First look at Delgath's attributes. You will see that Poor has been removed and Average Wealth is now there. Average Wealth was added because Poor was removed and characters must have a wealth attribute in this campaign. If we change Delgath's money to "G.S. 100" and then press the cost button, we get a similar message indicating that Average Wealth is now restricted to Delgath.

But Average Wealth has not been removed from Del's attributes. This is because the default wealth attribute for Humans is Average Wealth. If Delgath had been an Elf, the Poor attribute would never have disappeared, since Poor is the default wealth attribute for Elves. Every time Delgath's cost is recalculated, this message will appear until we change either G.S. to a legal value, or change Del's wealth attribute to an appropriate one.

Change Del's money to "G.S. 10" and add "Poor" to his attributes again and save the changes. Delgath is now a complete character! But don't close the program yet. We still should explore how races relate to characters. We also have to save our changes to the file on disk.

Next Topic:

A Character's Race

A Character's Race

In the previous sections, we never changed Delgath's race from the default race Human. Although races may be unimportant to some campaigns, in many campaigns (including tutorial.cmp) a character's race affects the cost of characteristics and what attributes the character may have.

To explore how a character's race may affect its manager's actions, let's change Delgath's race to "Elf" and press the cost button. You will see that Delgath's cost has dropped by 6 points. This is due to the fact that Del's characteristics (particularly AGL) cost less. However, this cost savings is offset by the fact that Delgath now has the "Extra Mana" attribute and also the race's cost has been added in.

If Delgath had a Mana score of 0 before you tried to make him an Elf you received a message stating that he doesn't have the characteristics to be an Elf. This is because elves are required to have a "base mana score" of 1 or better. The base mana score is the value shown in the Mana field when the character has no levels of the "Extra Mana" attribute.

You'll note that Delgath now has a mana score 2 greater than he did before. This corresponds to the two levels in the Extra Mana attribute he was given when his race changed. The Extra Mana attribute is required for all Elves, but of course you can change its level.

Do I hear you saying: "Wait a minute! Mana is a *random* characteristic. It shouldn't change until the GM alters it." Well... I sort of lied. You see, the Mana characteristic displayed on the screen is actually a calculated characteristic that gets its level by adding a character's Extra Mana to "baseMana", a random characteristic that is hidden from view. In effect, Mana is a random characteristic that takes into account the "bonus levels" from a character's Extra Mana.

To see more clearly the differences between a Human character and an Elf character you can look at the default settings for each race. Choose "Def Characteristics" from the "Character" menu. Then choose "Def Attributes".

Choosing "Def Characteristics" sets a character's characteristic values to their default, racial values. Similarly, choosing "Def Attributes" removes all attributes but those required and sets them to their default, racial levels. If you now press the cost button you will see that a new Elf character initially costs 24 points, a bit more than the 0 points used by a new Human character.

Close Delgath without saving the changes we've made. Press the Abandon button when the confirmation dialog pops up. We will now complete the tutorial by saving our changes to disk.

Next Topic:

Saving changes to the campaign

Saving changes to the campaign

We're done building the sample character. Now its time to save our changes to disk. The previous times we selected "Save" in the character manager, we were saving our changes **to memory**. The file on disk still contains the original campaign without our modifications.

Before I go on to tell you how to save your changes to disk, let me remind you that Delgath didn't exist when you started this tutorial. It would be unfair of you to save him to disk. Other people might get confused if they go through this tutorial and find Del already in the objects list. This can be particularly frustrating when they discover they can't access anything but Delgath's notes because they are not the GM or Delgath's player! So, if you just finished the tutorial, you should either delete or rename the sample character before proceeding to save the changes to disk. To delete Delgath from memory, activate the main window, select "Delgath" and choose "Delete" from the "Objects" menu.

To save the characters you build to the disk, as part of the campaign you loaded, activate the main window and choose "Save" from the "Campaign" menu. Make sure you've saved the character to memory first. The best way to make sure of this is never to save the campaign while other object managers are open.

You've now learned the basics of using CampMan to build characters. If you are a GM you should go on to the next tutorial Building A Campaign. Thanks for trying out the program and I hope you enjoy using it.

Building A Campaign

This chapter guides you through the creation of the tutorial campaign provided with Campaign Manager for Windows. Since the tutorial campaign was also used in the first tutorial, you may find it insightful to read through Building A Character before reading this chapter.

This tutorial assumes you are familiar with the Windows operating environment. In particular you should know how to move/size/close a window, enter/modify text, use a menu and click on a button or list item. If you are not familiar with Windows you should first go through the *Getting Started with Microsoft Windows* manual and the Windows tutorial.

Included Topics:

Creating a New Campaign

If you haven't done so already, start the program in GM Access mode. For more information, see Starting the Program.

The first window you will see after the startup dialogs is the main window, the campaign manager, after which the program is named. It presents you with a list of object names and types in the open campaign as well as a special list for the characteristics.

To create a new, empty campaign you must choose "New" from the "Campaign" menu. This will empty all the lists except for the object names list, which will contain one name. "Human" is the name of the default race, the only object in an empty campaign.

Before we go any further, we should save the new campaign to a file. Choose "Save As..." from the "Campaign" menu. This brings up a dialog box to let you specify what file you want to use. Type "practice.cmp" into the file name entry field and press OK. From now on, we will periodically save our changes to disk so that we won't loose all our work if we make a mistake or the computer crashes.

Next Topic:

The Simplest Objects, Items

The Simplest Objects, Items

Although items are probably the last objects you will create for your own campaign, they serve as a good object to look at first. Items are the simplest objects you can create. They aren't much more than a name to which you can attach descriptions.

To create a new object, you use the "Create" sub-menu located on the "Objects" menu from the main window's menu bar. Select the object type you want to create from this sub-menu, in this case "Item". A new item will be created and its item manager opened.

The item manager has two entry fields. One, labeled "name", is simply used to give the object a name. The other is a multi-line entry field labeled "attached notes". This field is used to display/modify the text part of the item's attached note. An object's attached note can be used to hold the object's description. It is important that you have a description for each object you create, since the description is the only thing your players will be able to see in Player Access mode.

More on notes later. For now, just enter "G.S." into the name field and enter "Gold Sovereign (G.S.) The basic unit of monetary wealth." as the description. Then choose "Save" from the "Item" menu. Notice when you saved the item, the item manager's title changed to reflect the new name. The main window's objects list was updated as well.

Once an object has been saved, you can reopen its manager using the main window. First select the object's name and type from the appropriate lists and then choose "Open" from the "Objects" menu. Go ahead and practice creating/saving objects by creating the rest of the items needed by the tutorial campaign: "Clothes: low class", "Horse [riding]", "Leather Jacket", "Ring: silver", "Shoes: pair", "Short Sword", "Small Knife". Feel free to use whatever descriptions you wish.

Next Topic:

Working with Notes

Working with Notes

Keeping good notes, as any experienced GM will tell you, is very important. Even more important is keeping those notes closely associated with their subject matter. Campaign Manager helps you do this by letting you create both unattached and attached notes.

subtopics:

Next Topic:

Before Adding Characteristics

Two Unattached Notes

Unattached notes are created in the normal fashion for creating objects, by selecting "Note" from the "Create" sub-menu. This creates a new note and opens its manager which has three entry fields. The use for the title and text entry fields should be obvious.

The topics field is used to give a list of subjects that the note touches on. Each subject should be separated from the others by a comma. The topics field is important in helping search for notes on related subjects. For more information see The Note Search Dialog.

The tutorial campaign has two unattached notes -- "Cost of Basic Characteristics" and "Cost of Abilities". Lets create two notes with those titles. Enter "Characteristics, Cost" and "Ability, Cost" into the appropriate topics fields. Then enter the following text for the first note:

> "There are three basic characteristics:
>
> Strength (ST), Intelligence (IT), and Agility (AGL)
>
> The cost for any of these characteristics is based on the difference between the assigned value and the default value. (i.e. difference between a character's ST and its race's ST)

difference	cost	difference	cost
< -7	-80	0	0
-7	-70	1	10
-6	-60	2	20
-5	-50	3	30
-4	-40	4	45
-3	-30	5	60
-2	-20	6	80
-1	-10	7	100

> 20 additional cost each point difference after 7"

And enter the following text for the second note:

> "The cost for any given level for an ability is calculated from the difference between the levels you have with the ability and a base characteristic/attribute defined differently for each attribute.
>
> Each ability may also have a modifier used to shift the cost to indicate more expensive/cheaper abilities.

difference	cost
< 0	0
0	1/2
1	1
2	2
3	4
4	8
5	12
6	16

> 4 additional cost each point difference after 6"

When you're done, save both notes and close their managers. We will refer to these two tables later when creating characteristics and attributes.

You may have noticed when you created the above notes that "GM Note" is also listed in the "Create"

sub-menu. GM notes are exactly like other notes except that they can't be viewed in Player Access mode. Thus you can create both public notes (Notes) and private notes (GM Notes). The tutorial campaign does not have any GM notes.

next subtopic:

Using Attached Notes

Using Attached Notes

Unattached notes are useful, but in the real world it would get kind of messy if all your notes were kept on loose sheets without any order. Of course, that's one reason for the topics field, to let you group your notes. But you usually keep notes about characters on the same sheet as the character ... or on attached sheets. So Campaign Manager also has attached notes.

Attached notes are note objects that are associated with a specific non-note object. There are several ways to create an attached note, but perhaps the most direct is to choose "Create Notes" from the first menu on an object manager's menu bar (i.e. the "Item" menu for item managers). This creates a new note attached to the edited object.

Attached notes usually have a title exactly the same as the name of the attached object. But there is nothing to prevent you from changing the title of any note. Attached notes are not listed in the main window's object list, to open them you must first open the attached object. Then you can choose "Open Notes" from the object manager's first menu.

For example, reopen the "G.S." item and select "Open Notes" from the "Item" menu. A note manager will come up with the note attached to G.S. You can see that the attached note's title is also "G.S." and that the text is exactly the same text we placed in the item manager's attached notes field. However, the topics field is empty.

The item manager is unique among object managers in that it has this "attached notes field" which can be used to modify its attached notes. All other object managers must have their attached notes specifically created/opened.

next subtopic:

Regarding Attached GM Notes

Regarding Attached GM Notes

I mentioned above that attached notes are only attached to non-note objects. But there may be times when you want to create private "side notes" about some public note. These are called attached GM notes.

Like unattached GM notes, attached GM notes can't be viewed in Player Access mode. But, GM notes can only be attached to note objects. To create an attached GM note, simply choose "Create GM Notes" from the "Note" menu of a note manager.

Next Topic:

Before Adding Characteristics

Before Adding Characteristics

So far we've created objects which don't cost a character any experience to own. Before we do, there are a few considerations to be made. Choose "Campaign" from the "Configuration" menu. This brings up a dialog box used to set particular aspects of how the creature, race, and character managers look and behave.

Right now we're interested in the lower left section titled "Race costs added ...". This section has two check boxes to indicate which racial costs should be added to characters. It is important to decide which costs will be added to characters before we actually start creating those costs. Although there is nothing to prevent you from changing these settings later, if you change these settings in a campaign which was designed for different ones, character costs may become artificially inflated or deflated.

The tutorial campaign is designed to add racial costs for both characteristics and attributes to characters. Make sure both of these check boxes are checked and press OK.

Next Topic:

Adding Characteristics

Adding Characteristics

It's time to add the characteristics to the tutorial campaign. Before continuing, you might want to save the campaign's changes thus far. To do so, simply choose "Save" from the "Campaign" menu. Then create a new characteristic.

In addition to the name field, characteristic managers have a combo box and an equation entry field. The combo box is used to set the characteristic's type. If you change the type, you will see that the title of the equation entry field changes to reflect what the entered equation is intended to calculate.

Assigned characteristics have their level assigned by the character's creator. They also have an associated cost for that level (otherwise everyone would have 18's). The cost for an assigned characteristic is determined by its cost equation.

Enter "ST" as the new characteristic's name, make sure that "assigned" is selected in the combo box, and save the changes. Then choose "Edit" from the "Equation" menu. This brings up an Equation Editor which will allow us to derive a cost equation for the strength characteristic.

subtopics:

Next Topic:

Modifying the Human Race

Deriving Strength's Cost

Since strength's cost is based on the difference between a character's strength and its race's strength lets start by creating an equation that calculates this difference. Enter "ST - @race:ST" into the multi-line entry field, then press the "Evaluate" button.

Several things will occur when you press Evaluate. First, the list box titled variables will be updated to include the strings "ST{0}" and "@race:ST{0}". Also the value entry field will now show a 0.

"ST" and "@race:ST" are called variables. Variables act as place holders for numbers which will be inserted when the equation is evaluated. If this equation were evaluated for a character, the character's strength level would replace the "ST" variable and the character race's strength level would replace the "@race:ST" variable. For a complete discussion on the uses of variables in CampMan see Variables.

The equation editor allows you to set test values for each variable in an equation and see what the equation's result will be with those settings. The current test value assigned to a variable is displayed in the brackets after its name in the variables list. As you can see, ST and @race:ST are both currently assigned a value of 0. The equation's result (zero minus zero) is displayed in the value field.

To change a variable's test value, select the variable's name in the variables list. When a variable is selected, the value field changes to reflect the currently assigned test value. Type in a new value and then click in some other entry field. When the value field looses focus, it converts the entered text to a number and assigns the result to be the selected variable's new test value. You can continue doing this until you've changed all the test values you wish, then press the Evaluate button again to see the new result. If no variable is selected, the value field displays the result of evaluating the edited equation with the test values.

Looking at the cost table for basic characteristics, we can see that eventually the cost will increase in increments of 20 points. To simulate this in the equation, enter "20 * (ST - @race:ST - 2)" as the equation's text and press Evaluate. When you pressed evaluate, the equation changed. Whenever you press evaluate, the equation editor tries to convert the entered text into an equation. The result will be an equation which is equivalent to the entered equation, but possibly has been rearranged or simplified to conserve memory.

If there is a problem with the text you entered that prevents the equation from properly being interpreted, a description of the problem is displayed in the value field. This is called a compilation error. For a complete discussion of these errors see Compilation Errors.

Lets keep the test value for @race:ST set to 0, and set the test value for ST to various values as we refine our equation. If we evaluate the equation for various test values between 7 and -8 we can see that the equation at the moment always evaluates to a value that is too low for differences less than 5. To correct the problem let's try using the max() internal function. The max() function answers the largest value among its arguments (separated by commas). For more information see Functions.

Modify the equation so that it reads:

"max(20 * (ST - (2 + @race:ST)), 15 * (ST - @race:ST - 1))"

and evaluate the equation. Further testing shows that this only partially solves our problem. The equation still evaluates to a value which is too low for differences less than 3. So we shall modify the equation once more:

"max(15*(ST - (1 + @race:ST)), 20*(ST - (2 + @race:ST)), 10*(ST - @race:ST))"

Complete testing of this equation reveals that we have finally created an equation that properly evaluates to the cost of the strength characteristic. Press the "Done" button. The equation dialog will close and our new equation will be inserted into the cost field for the ST characteristic. Now save the changes to the ST characteristic.

For a full explanation about equations and the equation dialog see Equations.

next subtopic:

Defining Functions

Defining Functions

There are two other basic characteristics that are part of the tutorial campaign. To create correct cost equations for them we could just copy the cost equation we used for strength, replacing the variables to reflect the new characteristics. But it would be easier if we could define a function that takes the difference as a single value and returns the appropriate cost. Then we could simply use the one function ... three times.

To define a function we first must choose "Functions..." from the "Campaign" menu on the main window's menu bar. This opens a dialog box used to manage the functions defined in the current campaign. It looks and behaves very similar to an equation editor.

To define our function we will use the same equation that we used for strength's cost, but we will replace each "ST - @race:ST" with "arg1". Enter

"max(-80, 10 * arg1, 15 * (arg1 - 1), 20 * (arg1 - 2))"

into the multi-line entry field and press the Evaluate button. Just like the equation editor, the function editor updates a list of variables (called arguments this time) and the value field. If you test this equation, you will see that as arg1 changes for -8 to 7, the answered value correctly changes from -80 to 100. Note that we've added a
-80 to the beginning of the arguments to the max() equation. This prevents the equation form returning values below -80.

Press the Save button and enter "BasicCharCost" when you are prompted for the function's name. Now close the function dialog and reopen the equation dialog for strength's cost equation.

Replace the old equation's text with "BasicCharCost(ST - @race:ST)" and press evaluate. Go ahead and convince yourself that this equation works properly and then press Done. Finally, save the changes to the ST characteristic.

Now we can easily add the two other characteristics to our campaign. Create two characteristics with the names "IT" and "AGL". Make sure both are of type "assigned". Finally, enter "BasicCharCost(IT - @race:IT)" into the cost field for the IT characteristic and enter "BasicCharCost(AGL - @race:AGL)" into the cost field for the AGL characteristic. Now save the changes for both characteristics and we'll add the remaining calculated and random characteristics to our campaign.

For a full explanation of functions and the function editor see Functions.

next subtopic:

Two Calculated Characteristics

Two Calculated Characteristics

In addition to the three basic characteristics, the tutorial campaign also has two calculated characteristics. The "ch cost" characteristic is supposed to have a level equal to the cost of the three assigned characteristics. To add this, simply create a characteristic named "ch cost", select "calculated" in the type combo box and enter the following equation into the level field...

"BasicCharCost(ST - @race:ST) + BasicCharCost(IT - @race:IT) + BasicCharCost(AGL - @race:AGL)"

The other calculated characteristic is called "basic dmg" and is supposed to display a dice expression which represents the character's damage with his bare fists/feet. This is a fifth of the character's strength in six-sided dice plus the remainder expressed as a modifier. To add this to our campaign we should first create a function that calculates the remainder of a division. Simply open the function dialog, enter the equation "arg1 - arg2 * floor(arg1 / arg2)" and save it under the name "remain".

Now we can create a calculated characteristic named "basic dmg" with a level equation that looks like this:

"floor(ST / 5) d6 + remain(ST, 5) - 1"

Save your changes and then we'll add the remaining two random characteristics.

next subtopic:

Two Random Characteristics

Two Random Characteristics

The tutorial campaign has two random characteristics: "Mana" and "Exp". Random characteristics generate an initial level for characters when they are created and stay at that initial level until changed by you in GM Access mode.

A character's Mana is initially determined with a roll of a 4-sided die and subtracting one from the result. We indicate this with the dice expression "1d4 - 1" placed in the characteristic's "init level" equation field. Unlike other equations, when the init level equation is evaluated any dice encountered are rolled (replaced with random numbers between 1 and the number of sides).

The Exp characteristic is used to represent the experience you've awarded a character. Of course, this is not a "randomly generated" characteristic, but only the GM is supposed to change it. So you should create a random characteristic with an initial level of "0".

Don't forget to save your changes. Also, be fair to the players, create attached notes describing each of the campaign's characteristics.

When you want to change a character's random characteristics you must check the "Edit Random (gm)" check box in the Program Configuration dialog. When this feature is enabled, character managers display random characteristics in entry fields to allow you to change and save their settings.

Next Topic:

Modifying the Human Race

Modifying the Human Race

Now that we have the campaign's characteristics squared away, we should set the basic characteristic values for the Human race. Open Human's race manager. You will see that all the assigned characteristics have been set to 0. We'd like to set ST, IQ, and AGL to 10. Just enter 10 into the three entry fields next to the characteristic names and save your changes. Then close the race manager.

The characteristics in the race manager you just closed were listed in a single column and may have been out of order. When a new characteristic is saved, it is placed at the end of the characteristic list. But it is possible to reorder the list of characteristics.

The order characteristics are listed in is controlled by the "characteristics" list in the main window. This list box is specially designed to allow its contents to be reordered. Simply click and hold the middle mouse button on the name you wish to move. Then drag the indicator to a new position and release the mouse button.

You may also want to change the number of rows/columns the characteristics are listed in. This is controlled by the "Characteristics" section of the Campaign Configuration dialog. Simply enter the number of rows/columns you wish to use. The dimension not specified is free to change as needed to list all the characteristics.

Take a moment here to save the campaign. We don't want to loose all our work so far.

Next Topic:

Adding Attributes

Adding Attributes

The tutorial campaign contains quite a few attributes. Attributes are kind of like characteristics, but not all character's are intended to own any single attribute. Attributes can be used to represent a character's advantages, disadvantages, abilities, skills, super powers, etc.

Some of the attributes to be added are what we'll call "abilities". Each is used to represent the level of skill a character has with some learned ability. They all will share a common way to calculate their costs defined in the "Cost of Abilities" note we created earlier.

Because the process for determining an ability's cost is common to all abilities, we will want to define a function, "AbilityCost", which simulates this process. By following a procedure similar to the one we used in deriving the BasicCharCost function, we find that a function defined as follows fits our needs:

"max((arg1 = 0) / 2, arg1, 4 * (arg1 - 2))"

Simply open the function dialog and define the above equation as the AbilityCost function.

Create a new attribute. When the attribute manager opens, you will see that attributes come in three flavors: assigned, calculated and owned. All attributes have a cost equation used to define the cost for a character to own the attribute.

Every attribute also has a second equation. For calculated attributes, the second equation calculates the attribute's level for a given character. For assigned attributes, this second equation is used to define the maximum level that may be assigned to the attribute for a given character. A character may only have a level of 1 or 0 with any owned attribute so this second equation is only used to determine if a character may own an owned attribute.

In all cases, if this second equation answers 0 for a given character, the attribute is considered restricted to that character. If the maximum level for an assigned attribute is -1 for a given character, the attribute's level is considered to have no upper limit for the that character.

Lets name the new attribute "Horseriding". This attribute will be assigned, and will indicate a character's level of skill at riding horses. The cost will be based on the character's agility as follows: "AbilityCost(Horseriding - AGL)". The maximum level is indicated by "-@character". This second equation answer 0 for races and -1 for characters. We restrict the attribute from races because the cost equation does not properly take into account the possibility that a character's race also has this ability.

Save Horseriding and then choose "Copy" from the "Attribute" menu. This creates a new attribute with properties copied from Horseriding. This greatly simplifies the work required to create similar attributes. Name the new attribute "Sneak" and change the cost equation to "AbilityCost(1 + Sneak - AGL)". The additional 1 makes this ability's levels more expensive than the average ability.

After saving the changes copy Sneak and name the new attribute "Blather". This attribute will be based on intelligence not agility. Change the cost equation to "AbilityCost(Blather - IT - 1)". The subtraction of 1 here indicates that the ability's levels will be less expensive than the average ability.

Copy Blather and name the new attribute "Thief's Training". This attribute is not considered an ability and does not use the AbilityCost function. Rather it is used to indicate general knowledge which will affect the level of another attribute we will create. Change the cost to "5 * Thief's Training". This will give the attribute a cost of 5 for each level assigned.

After saving Thief's Training, create a new characteristic and name it "Pick Pockets". This will represent the common thief ability to pick pockets undetected. It is a calculated attribute which will be restricted to all character's that do not have the Thief's Training attribute. Its cost will be "2" for all characters, simply enter a 2 in the cost field. Then enter

"boolean(Thief's Training) * trunc(Thief's Training + (AGL + IT) / 2)"

into the level field. This equation answers 0 if the character does not own Thief's Training. If the character owns Thief's Training, the equation answers the sum of the character's Thief's Training level

with the average of the character's agility level and intelligence level truncated to an integer.

Don't forget to create attached notes for each of these attributes so that players can find out exactly what they represent and how they interrelate.

Next Topic:

Adding Attribute Lists

Adding Attribute Lists

An Attribute List is a group of attributes, only one of which may be owned by a character at any given time. Such an object has several uses, some of which are demonstrated in the tutorial campaign. The first example of an attribute list is the group of wealth attributes.

subtopics:

Next Topic:

Creating an Alternate Race

The Wealth Attributes

We would like to create an attribute which reflects a character's poverty or affluence. We could create a single attribute called "Wealth" with several numbered levels, but it would be nicer if we could name these levels: "Poor", "Average Wealth", "Rich", and "Filthy Rich". In fact we can do something similar by creating an attribute list with four attributes.

Create an attribute list and enter "Average Wealth" when prompted for the name of the first member. This is required, since an attribute list can't exist without at least one member attribute. Attribute lists do not have a name per say. If and when a name is needed to identify an attribute list, the title of the attached note is used or one of the member names is used if no note is attached.

The attribute list manager looks and acts like an attribute manager except for the name field, which has been replaced by a member combo box. The current name displayed in the member field indicates which attribute's properties are being displayed/modified in the other fields. At the moment, the "Average Wealth" attribute is being edited.

Create an attached note for the attribute list titled "Wealth", with topics "Wealth, Money", and the following text:

> "The Wealth attributes indicate how rich or poor your character is monetarily. The following table indicates the cost & limitations of each wealth attribute.
>
> | attribute | cost | maximum wealth |
> |---|---|---|
> | Poor | -10 | 10 gold sovereigns |
> | Average Wealth | 0 | 50 gold sovereigns |
> | Rich | 15 | 200 gold sovereigns |
> | Filthy Rich | 25 | no upper limit" |

This note will be used by all member attributes. When a character tries to open any of the member attributes while in Player Access mode, this note will be opened.

Make sure that Average Wealth is "owned" and enter "G.S. <= 50" into the level equation field. This equation will either answer 0 or 1 depending on the amount of money the character has. The cost of this attribute will be 0, but since we are allowing races to own this attribute, we must add/subtract any different wealth costs associated with the character's race. So enter

"10 * @race:Poor + -15 * @race:Rich + -25 * @race:Filthy Rich"

into the cost field for Average Wealth. Save the changes to Average Wealth and then choose "New" from the "Attribute" menu. Enter "Poor" as the new attribute's name. Enter "G.S. <= 10" as the new level equation and

"-10 * not(@race:Poor) + -15 * @race:Rich + -25 * @race:Filthy Rich"

as the new cost equation. The not() function in the above equation prevents a character from getting an extra -10 cost if its race is already poor. Save the changes and add a new member named "Rich". This time enter "G.S. <= 200" in the level field and enter

"10 * @race:Poor + 15 * not(@race:Rich) + -25 * @race:Filthy Rich"

as Rich's cost equation. Finally, add "Filthy Rich" to the attribute list. Put "1" in the level field and

"10 * @race:Poor + -15 * @race:Rich + -25 * not(@race:Filthy Rich)"

in the cost field. Make sure all these attributes are "owned" and save your changes. Now whenever a user adds one of these attributes to his character, any other wealth attribute the character already owns will be removed before the new one is added.

Wouldn't it be nice if we could force every character to own one of these attributes? Then there would be a system in place which forces a character with a huge amount of money to pay the experience points

necessary to raise his level of wealth. There is a way to do this. We simply need to add the "Average Wealth" attribute to the default race (in this case "Human"),

Open Human's race manager and choose "Add" from the "Attributes" menu. Then enter "Average Wealth" into the entry field that pops up in the attributes list. Press return to accept your entry, then save the changes to the Human race. Now any player that removes the current wealth attribute from his character will cause Average Wealth to be added to his character.

Take a moment here to save the campaign. We don't want to loose all our work so far.

next subtopic:

Two Weapon Abilities

Two Weapon Abilities

We would like to create two abilities, "Short Sword" and "Long Sword" which indicate a character's level of skill at wielding those weapons. Both abilities will have their costs based on the character's agility. But we want to take into account the possibility that a character's skill at wielding one of these weapons might be helped (made cheaper in cost) by his skill at wielding the other weapon. We will do this by creating two attribute lists.

Create an attribute list with "Short Sword" as the first attribute. Make sure the attribute is "assigned" and enter "-@character" into the max level field and "AbilityCost(Short Sword - AGL)" into the cost field. Save your changes.

Similarly, create a second attribute list with "Long Sword" as the first attribute. Make sure it is "assigned", enter "-@character" into the max level field and enter "AbilityCost(Long Sword - AGL)" into the cost field. Save your changes.

We have two attributes based on a character's agility. We will now add one more member attribute to each list which will reflect the possibility that one of the abilities is actually based on the other. Add a member to the list containing "Short Sword" called "Short Sword [fromLongSword]". Similarly, add a member to the list containing "Long Sword" called "Long Sword [fromShortSword]".

Since it is illogical for a character to own both of these attributes at one time, we will restrict each of these attributes to character's which own the other attribute. Also, we will restrict a character from owning "Short Sword [fromLongSword]" at a level lower than "Long Sword" and similarly for "Long Sword [fromShortSword]" at a level lower than "Short Sword". Enter

"-not(Short Sword [fromLongSword] | (@list:Long Sword > @list:Short Sword)"

into the max level field for "Long Sword [fromShortSword]". This equation answers -1 if the character does not own "Short Sword [fromLongSword]" and his level with "Short Sword" is less than his level with "Long Sword [fromShortSword]" or his level with "Long Sword" (whichever of the two attributes he owns). Otherwise it answers 0. Now enter

"-not(Long Sword [fromShortSword] | (@list:Short Sword > @list:Long Sword)"

into the max level field for "Short Sword [fromLongSword]".

Finally we can enter "AbilityCost(Long Sword [fromShortSword] - Short Sword)" and "AbilityCost(Short Sword [fromLongSword] - Long Sword)" into the appropriate cost fields. Save the changes for both attribute lists.

We now have four attributes in two attribute lists which can be used to represent any of the following:

- A character's skill at wielding the short sword
- A character's skill at wielding the long sword
- A character's skill at wielding the short sword
  (based upon the character's skill at wielding the long sword)
- A character's skill at wielding the long sword
  (based upon the character's skill at wielding the short sword)

Next Topic:

Creating an Alternate Race

Creating an Alternate Race

Now that we have completed most of the tutorial campaign, it might be nice to give players a choice of races for their characters. Lets create a race called "Elf". Elves will be somewhat weaker than Humans, but more intelligent and much more agile. Also, the average Elf will be poorer than the average Human.

Create a new race and name it "Elf". Enter 9 in the ST field, 11 in the IT field and 13 into the AGL field. Then add "Poor" to its attributes and save the changes. Note that the cost field now reads "cost: 20". This is because a Human character with these characteristics and attributes would cost 20.

Elves are often considered somewhat magical in nature, and it would be nice if we could somehow indicate that the average Elf has more Mana than the average Human without changing Mana into an assigned characteristic. We can do this, but first we must create an attribute and use its level when calculating the Mana characteristic.

Create a new attribute called "Extra Mana". Make sure it is "assigned" and then enter "2 * (Extra Mana - @race:Extra Mana)" into the cost field. Now we need to address the fact that the current Mana characteristic is random. We will circumvent this by renaming the random characteristic as "baseMana" and then creating a new, calculated Mana characteristic that adds a character's baseMana and Extra Mana levels together.

First open the current Mana characteristic manager, enter "baseMana" into its name field and save it. Answer yes when asked about renaming the characteristic. Then create a new *calculated* characteristic with "baseMana + Extra Mana" as its level equation and name it "Mana".

Save all your changes, close and reopen Elf's manager so that the characteristic changes can be reflected in its display/entry fields and then add "Extra Mana 2" to its attributes list. Now all Elf characters will automatically get 2 extra mana points at the cost of 4 experience points.

We'd like to restrict characters with a base mana of 0 from becoming elves. This can be done by assigning 1 to the Elf's baseMana level. Races use their random characteristic levels as "minimum requirements". To assign a character to a new race, each of its random characteristics must meet or exceed the level assigned to the new race's random characteristics. For this reason, Race managers always display random characteristics in entry fields to allow you to assign and save their values.

Finally, we would like to "hide" the baseMana characteristic since it is somewhat annoying and rather extraneous information. We can do this by selecting "baseMana" in the main window's "characteristics" list and choosing "Hide" from the "Characteristics" menu. The "baseMana" characteristic will be listed in gray text and will no longer be displayed in creature, race or character managers.

Save the changes to Elf and we're done.

Believe it or not, we are done creating the tutorial campaign! Save everything to disk. To make sure everything was entered correctly and to convince yourself this campaign works, you might want to go through the Building A Character tutorial using this campaign instead of the one supplied with the program.

I realize you probably still have quite a few questions about how everything works. That is what the Reference part of this manual is for, to answer your questions. But, whether or not you think so, you probably could come up with a decent campaign with just a bit of logic and using the tutorial campaign as a guide.

Go ahead and contact me on Compuserve, if there are any real tough problems you run into. But try to use the reference for the basics, OK? Thank you for trying out the program, and I hope you have a fun time using it. Good Luck <g>.

Command Line Parameters

Although you can easily access all of the program features without the use of command line parameters, you may get tired of typing the necessary information into all the startup dialogs. You may use any of these parameters on the command line to modify the startup behavior of CampMan.

Command line format:

campman.exe [[d:] [path] *filename*] [/P[name] | /G] [/A]

Parameter		**Action**
d: *path* *filename*	drive, path and Campaign file	The program tries to load the given campaign file after the startup dialogs.
/P access /P*name*	Startup with player	The program enters the given name for you and disables the ability to enter GM as the player name.
/G access	Startup with GM	The program is placed in GM access mode and the player name dialog is bypassed.
/A	Auto OK	Automatically presses the OK button in the startup dialogs. In most cases the dialog will simply not appear.

See the documentation on Program Manager (or your replacement) on how to add these command line parameters to CampMan's icon.

EXAMPLE: to automatically load the "tutorial" campaign use

campman.exe tutorial.cmp

EXAMPLE: to disable the ability to enter "GM" as the player name and thereby disable the ability to enter GM Access mode use

campman.exe tutorial.cmp /p

EXAMPLE: to automatically enter "John" as the player bypassing the player name dialog use

campman.exe tutorial.cmp /pJohn /a

EXAMPLE: to startup in GM Access mode bypassing the player name dialog use

campman.exe tutorial.cmp /g

EXAMPLE: if the current program password is empty you can startup in GM Access mode, bypassing the player name dialog and the password dialog, by using

campman.exe tutorial.cmp /g /a

Player vs. GM Access

GM Access mode gives full access to all the features the program has to offer. With GM Access you can create/modify any object, configure the program and change password settings.

Player Access mode is intended to be used to create/modify characters without the ability to create/modify other objects. With Player Access you may load a campaign from disk, create/modify your characters and their attached notes and save those changes to the campaign. You may also read, but not change, notes that are not attached to your characters. You may not view/modify other objects or GM Notes although all objects other than GM Notes will be listed in the main window.

Related Topics:

Passwords

Passwords

A password is usually required to start the program in GM Access mode. This password is called the *program password*. The program password is stored in the WIN.INI file using a coded format to make it hard to read. The program password is intended to be a nuisance to unauthorized access, not a foolproof deterrent. As long as the campaign(s) you create are only used on your computer, a program password should be sufficient security. If you plan to distribute your campaign to other users (i.e. your players) but don't want them to have full access to your file, use a *campaign password*.

Each campaign may also be assigned a campaign password. A campaign password is stored in the file along with the campaign. To successfully load a campaign while CampMan is running in GM Access mode, you must first correctly enter the campaign's password if it has one. Since this password is stored in binary form as part of the campaign's file, it is next to impossible to determine the password from the file itself unless you know the file structure.

USE CAMPAIGN PASSWORDS Sparingly and keep good records of the passwords you set because there is nothing I can do to help you if you forget a campaign's password. Without a campaign's password, you might as well scrap that file and build a new one.

Equations

Equations are the "rules" that CampMan follows. They specify costs, calculated levels and maximum levels for races and characters. Equations consist of one or more operands separated by operators. An operand may be a number, a variable or a function.

Equations are entered as text into appropriate entry fields (i.e. the cost field for characteristics) but are not compiled until the object is saved. After the equation is compiled, it may not look the same since the compiler simplifies certain expressions to save memory. But rest assured that the compiled equation evaluates to the same value as the original expression.

To check that an equation will compile without errors simply click in the equation's entry field and choose "Check" from the "Equation" menu. If you are not sure if a field is supposed to contain an equation, click in the field and then look at the "Equation" menu. If all the items in the menu are disabled (grayed), then the field is not used for an equation.

Equations can be a difficult aspect of the program to learn. You may want to spend some time playing with the Equation Editor to get a feel for how operators, variables and functions work.

Related Topics:

- Operators
- Variables
- Functions
- The Equation Editor
- Compilation Errors
- Evaluation Errors

Operators

Operators are the actions to be taken with the numbers in an equation. For instance, + is the operator that adds two numbers together.

The following operators are currently supported in CampMan:

Operator	Syntax	Name /Action
( )	**(** *expression* **)**	surrounding parenthesis /evaluates expression first
d	*dice* **d** *sides*	dice separator /creates a dice expression
- (unary)	**-** *expression*	negation /evaluate *0 - expression*
~ (unary)	**~** *expression*	boolean negation /evaluate *1 - boolean(expression)*
^	*base* **^** *exponent*	power /raise *base* to *exponent*'s power
*	*op1* * *op2*	multiplication /multiply *op1* by *op2*
/	*op1* / *op2*	division /divide *op1* by *op2*
+	*op1* **+** *op2*	addition /add *op1* and *op2*
-	*op1* **-** *op2*	subtraction /subtract *op2* from *op1*
=	*op1* **=** *op2*	equal to /evaluate *~(op1 - op2)*
<	*op1* **<** *op2*	greater than /evaluate *boolean(max(op2 - op1, 0))*
<=	*op1* **<=** *op2*	greater or equal to /evaluate *(op1 < op2)* | *(op1 = op2)*
&	*op1* **&** *op2*	boolean and /evaluate of *boolean(op1 * op2)*
~&	*op1* **~&** *op2*	boolean nand /evaluate of *~(op1 * op2)*
|	*op1* **|** *op2*	boolean or /evaluate of *boolean(op1 + op2)*
~|	*op1* **~|** *op2*	boolean nor /evaluate of ~(*op1 + op2)*

Given any two operators separated be one or more lines in the above table, the one which occurs in the

highest position is evaluated first.
If two operators are not separated by a line, then the operator that occurs first in the equation is evaluated first.

EXAMPLE: In equation (a) the / operator is evaluated before the + operator.
In equation (b) the expression surrounded by parentheses is evaluated before the / operator.

(a) **1 + 2 / 3** evaluates to 1

(b) **(1 + 2) / 3** evaluates to 1

EXAMPLE: In equation (c) the * operator is evaluated first followed by the - operator and finally the + operator.
In equation (d) the expression surrounded by parentheses is evaluated first, thus the * operator is evaluated first followed by the + operator and finally the - operator.

(c) **1 - 2 + 3 * 4** evaluates to 11

(d) **1 - (2 + 3 * 4)** evaluates to -13

subtopic:

The Dice Operator and Dice Expressions

Related Topics:

Variables
Functions

The Dice Operator and Dice Expressions

Campaign Manager has the ability to create dice expressions using the "dice" operator. Such expressions may be used in any equation or function.

In any equation other than a random characteristic's level equation, a dice expression will simply propagate through the calculation. An accidental use of a dice expression in a cost equation will cause the cost to be answered with some kind of dice expression included. On the other hand, it is possible to use this feature to create calculated characteristics and attributes that report their levels as dice expressions.

When CampMan calculates the level of a random characteristic for a character, it "rolls" all dice expressions, replacing each dice expression it comes across with an appropriate, randomly generated number.

It can be important that you make the dice operator clearly separate from its arguments since "d" may also be used as part of a variable name.

EXAMPLE: The following expression will be interpreted as a single variable

STd6

The following expression will be interpreted as "Strength" number of 6-sided dice

(ST)d6

Variables

Variables act as place holders for numbers. Replacement values are inserted for each variable before an equation is evaluated. The variable's name indicates what value to insert.

A variable with a race's name will be replaced by a 1 if the evaluating object is the race itself or is a character of that race. A variable with a characteristic's name will be replaced with the character's or race's level for that characteristic. Similarly, if the variable has an attribute's name, it will be replaced with the character's or race's level for that attribute. Finally, if the variable has the same name as an item it will be replaced with the number of such items owned by the character. Otherwise, the variable will be replaced with 0.

It is important to note that if a characteristic, attribute or item has the same name as a race, its level can't be accessed by a variable since the variable will always be replaced with either a 1 or 0. Attribute levels similarly override item levels, and characteristic levels override both attribute and item levels.

EXAMPLE: The following equation gives the average between a character's or race's Strength and Agility.

(Strength + Agility) / 2

EXAMPLE: The following equation answers 3 for the race Elf and for characters which are of the Elf race and answers 0 for all other races and characters.

3 * Elf

The following procedure may be used to determine what value a variable will be replaced with:

Variable Replacement Procedure

subtopics:

The @race and @character variables
The @race: modifier
The @list: modifier

Related Topics:

Operators
Functions

The @race and @character variables

The two variables **@race** and **@character** are replaced with either 1 or 0 depending on whether the equation is being evaluated for a character or race.

EXAMPLE: The following equation evaluates to half a character's Strength or half a race's Agility.

(@character * Strength + @race * Agility) / 2

The @race: modifier

The **@race:** modifier is used to access either a character race's levels or the default race's levels. Any variable preceded by **@race:** is treated as if the equation were being evaluated for the appropriate race rather than the object for which it is actually being evaluated.

EXAMPLE: For a character, this equation calculates the average between the character's Strength and the character race's Agility.
For a race, this equation calculates the average between the race's Strength and the default race's Agility.

(Strength + @race:Agility) / 2

EXAMPLE: If Elf is not the default race, the following equation answers 3 for characters which are of the Elf race, but answers 0 for all races and all other characters.
If Elf is the default race, the following equation answers 0 for characters which are not of the default race and answers 3 for all races and all characters of the default race.

3 * @race:Elf

The @list: modifier

The **@list:** modifier is used to access the level a character or race has with any member of an attribute list.

EXAMPLE: If Long Sword and Long Sword [fromShortSword] are two members of the same attribute list:
For a character with neither attribute, both these equations answer -Agility.
For a character with the Long Sword attribute, both these equations answer Long Sword - Agility.
For a character with the Long Sword [fromShortSword] attribute, both these equations answer Long Sword [fromShortSword] - Agility.

@list:Long Sword - Agility

@list:Long Sword [fromShortSword] - Agility

Variable Replacement Procedure

1) If the variable name is "@race", replace the variable with 1 if the equation is being evaluated for a race. Otherwise replace the variable with a 0.
2) If the variable name is "@character", replace the variable with 1 if the equation is being evaluated for a character. Otherwise replace the variable with a 0.
3) If the variable name begins with "@race:" and the equation is being evaluated for a character, replace the variable with the value it would have been replaced with if the variable name did not begin with "@race:" and the equation was being evaluated for the character's race.
4) If the variable name begins with "@race:" and the equation is being evaluated for a race, replace the variable with the value it would have been replaced with if the variable name did not begin with "@race:" and the equation was being evaluated for the default race.
5) If the variable name begins with "@list:" and the rest the name is the same as an attribute list member's name, replace the variable with the race or character's level for any member of that attribute list. Otherwise replace the variable with a 0.
6) If the variable name is the same as a race name: If the equation is being evaluated for that race or a character of that race, replace the variable with a 1. Otherwise replace the variable with a 0.
7) If the variable name is the same as a characteristic name: Replace the variable with the race or character's level with that characteristic.
8) If the variable name is the same as an attribute name: Replace the variable with the race or character's level with that attribute.
9) If the variable name is the same as an item name: If the equation is being evaluated for a character, replace the variable with the number of those items possessed by the character. Otherwise replace the variable with a 0.
10) If the variable name is not a race name, characteristic name, attribute name, or item name, replace the variable with a 0.

Functions

Without functions, many equations would become long and complicated while others might not be possible at all. Functions, like operators, act upon numbers in a pre-defined way and return a number. But unlike operators, which act on the operands around them, functions act on the numbers in their argument list.

EXAMPLE: The following equation answers either the character's Strength or the character race's Strength plus one (whichever is greater) and divides it in half.

```
max(Strength, @race:Strength + 1) / 2
```

subtopics:

Related Topics:

Internal Functions

Campaign Manager has a number of internally defined functions. Most of these functions could not be realized any other way. The following is the current list of internal functions:

Function	# args	Returned Value
boolean()	1	0 if its argument is equal to 0, otherwise 1
ceil()	1	least integer greater than or equal to its argument
floor()	1	the greatest integer less than or equal to its argument
max()	# > 0	the greatest valued argument
min()	# > 0	the least valued argument
not()	1	1 if its argument is equal to 0, otherwise 0
sqr()	1	the value of its argument squared
sqrt()	1	the value whose square is equal to its argument
trunc()	1	the integer portion of its argument

User Defined Functions

In addition to internal functions you may also have user-defined functions at your disposal. User-defined functions are stored as part of the campaign file. As their name implies, user-defined functions are functions which you (the GM) define. You will probably want to define functions for often used sequences of operations. (i.e. calculating a characteristic's cost)

A user-defined function is basically a special equation referenced by a name. It may use all the operators, numbers, variables and functions that other equations may use. However, variables are handled differently from standard equations.

Standard equations (those not used to define a function) are evaluated for races and characters. As a result, standard equations use variables to represent values in races and characters. But functions are independent of other objects and only act on the numbers given in their argument list. So variables are used in functions to represent their arguments.

When a function is compiled, each variable is renamed as arg1, arg2, arg3, etc. The trailing number indicates the position in the argument list from which the variable gets its value.

EXAMPLE: If equation (a) is compiled as a function, it becomes equation (b).

(a) **(a + b - arg2) / 2** (b) **(arg1 + arg3 - arg2) / 2**

If equation (b) is then given the name EqB() and evaluated as follows:

EqB(1, Strength, Agility / 3)

the result is the value of (1 + Agility / 3 - Strength) / 2

Functions may be defined using the Function Editor. See that topic for more information.

The Equation Editor

The Equation Editor is a dialog box used to test different equations you wish to use. You do not need to use the Equation Editor to enter text into an equation field. However, an equation field simply takes the text you enter and tries to compile it into an equation when the object is saved. The Equation Editor allows you to enter text, compile it into an equation (if possible), and test that equation by giving test values to its variables.

To bring up the Equation Editor for a given equation, click in the equation's entry field and select "Edit" from the "Equation" menu. When the dialog box opens, its title will indicate what equation you're editing. This is important, because when you press "Done", the text representing your new equation will replace the contents of that entry field.

The Equation editor has a large, multi-line entry window at its top. This holds the text of the edited equation. To compile an equation press the "Evaluate" button. Any changes you've made to the entered equation will be compiled. If a compilation error occurs, a description of the problem will be displayed in the value field. If the equation compiles successfully, it will be evaluated and the results will be shown in the value field.

Whenever the edited equation is successfully compiled, the field titled "variables" will be updated to list all the variables from the current equation. The variable names are followed by their assigned values (initially 0) contained in brackets. To change a variable's assigned value, select it in the list box and enter the new value into the value field. Click in any other field (or press return) and the variable's value is updated.

In order to remove the selection from the list box completely, you must press the evaluate button. In this way the value field serves a dual purpose. When a variable is selected, it displays the variable's value and lets it be modified. When no variable is selected, the field displays either the compilation error or the result of evaluating the edited equation.

When you are done editing the equation, you may press either "Done" or "Cancel" to close the Equation Editor. If you press Cancel, the dialog box is closed. If you press Done, the dialog box is closed and the text for the new equation is inserted into the edited equations entry field. The Done button is disabled unless the edited equation compiles without error.

Closing the dialog box with the system menu is the same as pressing Cancel.

Related Topics:

Compilation Errors
Evaluation Errors

The Function Editor

User-defined functions are managed with the Function Editor. It allows you to create, delete and rename user-defined functions. It can also be used to explore the internal functions.

To open the Function Editor, choose "Functions" from the "Campaign" menu on the main window's menu bar. You will see a dialog box very similar to an Equation Editor. In fact, the Function Editor is a modified version of the Equation Editor. You may wish to read that section first if you are unfamiliar with it.

Just like in the Equation Editor, the multi-line entry window is used to hold the edited function's text. The arguments field lists the arguments accepted by the edited function, and the value pane is used to display the function's compilation/evaluation errors, the function's evaluation, or the selected argument's value. The value pane may also be used to change an argument's assigned value.

Unlike the Equation Editor, the dialog box has a "fn:" combo box used to select the function name. Also two big buttons have been renamed to reflect their new functions. When you are done editing a function's definition, you press the "Save" button to save the changed definition. Note that the Save button does not close the dialog box. When you wish to close the dialog box, press the "Close" button.

The fn: combo box includes a list of all the currently defined function names. To view/modify a function which is already defined simply select its name. You may then modify, compile, and test your changes. Don't forget to press Save when you are through making changes. It is important to realize that a function's definition does not change until the Save button is pressed.

If you select an internal function, a description of the function will be displayed along with an appropriate number of arguments and a value. Although you can't redefine or overwrite an internal function, you can give it test values and see the results. This lets you experiment with the internal functions until you are more comfortable with how they work.

Selecting "New" from the combo box's pop-up menu will clear the definition entry field and display an empty function name. Whenever the function name is empty, you will be prompted for a new name when the Save button is pressed. You may not overwrite other defined functions.

Selecting "Delete" from the combo box's pop-up menu will remove the selected function from the list of defined functions, clear the definition entry field and display an empty function name. You may not delete functions which are used by other defined functions or equations used elsewhere in the campaign.

Selecting "Rename" from the combo box's pop-up menu will prompt you for a new name. The selected function will be renamed to the new name and the new name will be displayed in the combo box. The name change is also reflected in every other function or equation that uses the renamed function.

For more information on functions see Functions.

Related Topics:

Compilation Errors
Evaluation Errors

Compilation Errors

When Campaign Manager can't compile an equation due to some problem with the entered text, it answers a compilation error. Each error reflects a different problem that can arise while compiling an equation. Objects can't be saved unless their equations can be compiled.

The following list explains all the possible compilation errors:

Empty Equation This error occurs when an empty string is compiled or when a pair of empty parentheses is detected in the entered text.

EXAMPLE: 1 + max(2, Strength - ( ) / 2)

Mismatched Parentheses This error occurs when parentheses have not been paired incorrectly.

EXAMPLE: 1 + max(2, (Strength - 1 / 2)

Missing Operator This error occurs when two operands are not separated by an operator.

EXAMPLE: 1 + max(2, (Strength - 1) 2)

Missing Term This error occurs when two operators are not separated by an operand or when an operator is at the beginning or end of the entered text.

EXAMPLE: 1 + max(2, (Strength -) / 2)

Unknown Function This error occurs when a function name has been used that is not defined.

EXAMPLE: 1 + mx(2, (Strength - 1) / 2)

Related Topics:

Evaluation Errors

Evaluation Errors

When an equation is evaluated (i.e. to calculate a cost or level) an evaluation error may occur. Evaluation errors are reported by a message box. The message box's title indicates the type of error. The text reflects the equation that caused the problem and the object for which the equation was being evaluated.

An evaluation error always indicates an equation which has not properly dealt with all the conditions under which it might be evaluated. For instance, dividing by Strength, when Strength might be 0. It should always be kept in mind that a variable could be set to 0 or a negative number.

The following list explains all the possible evaluation errors:

Divide by zero At some point while the equation was being evaluated, an attempt was made to divide by 0.

Imaginary Number At some point while the equation was being evaluated, an attempt was made to take the root of a negative number.

 # Sides < 0 At some point while the equation was being evaluated, a dice expression was formed that had a negative number of sides.

Circular Reference At some point, a function/ equation was called that uses itself as part of its definition (either directly or by using second function/ equation that calls the first).

Related Topics:

Compilation Errors

The Main Window

The main window is where all the objects in a campaign are listed, accessed, created, destroyed, etc. In essence, the main window is **the** campaign manager.

Objects are listed in two list boxes labeled "name" and "type". The first lists all the names used by at least one object in the loaded campaign. The second lists all the types of objects which have the name selected in the first list. The objects included in these lists may be narrowed by using the Filter Dialog.

The "Create" sub-menu located on the "Objects" menu is used to create new objects. Choose the type of object to be created form this sub-menu. While the program is running in Player Access mode, this sub-menu is replaced by a single item "Create Character".

In addition to the two object lists, while running in GM Access mode two other panes are available. The first is a "characteristics" list box, used to manipulate the order in which characteristics are displayed. To move a characteristic name, click and hold the middle mouse button on the name and drag the indicator to a new position.

The final field, available only in GM Access mode, is not initially visible when the program starts up. Simply resize or maximize the main window to make it visible. This multi-line entry field, labeled "attached notes", works similarly to the notes field in item managers. When an object is selected in the object lists, the text for its attached note is displayed here and can be modified directly. However, to add an attached GM note or to change the title/topics of the attached note, it will be necessary to actually open the attached note's manager.

Related Topics:

The Filter Dialog

The Filter Dialog

The Filter dialog is used to narrow the number of objects that are included in the main window's object lists by specifying which object types to list. A more advanced feature allows further narrowing using the Note Search Dialog as an additional filter. To open the Filter dialog simply select "Filter..." from the main window's "Objects" menu.

The Filter dialog consists of two sets of check boxes and several push buttons. The upper section, titled "Included Types", is used to select which object types to include in the main window's object lists. Each checked box indicates an object type to include.

The "all" and "none" buttons are used to quickly check/uncheck all the different types. Any changes you make in the filter do not take effect until you press the "OK" button. If the OK button is disabled, then there have been no changes made.

You can save a default set of object types by using the "Save as Default" button. When this button is pressed, a list of all the currently checked object types is saved. This list will be used as the initial filter whenever CampMan is started.

The bottom section, titled "Link to Note Search", is an advanced feature that allows you to use the Note Search dialog as an additional filter. You may limit the filter to include only notes and objects attached to notes which are listed in the Note Search dialog's found list. Or you may exclude those objects instead. The settings in this section only have an effect only when both the Filter dialog and the Note Search dialog are open.

EXAMPLE: To list all objects which do not have an attached note...

i: If the Note Search dialog is currently open, close it.

ii: Open both the Note Search dialog and the Filter dialog.

iii: Press "Start Search" in the Note Search dialog.

iv: Press "all" in the Filter dialog
and check the "exclude found list" check box.

v: Press the "OK" button in the Filter dialog.

Objects

The following sections describe in detail the specific properties of each CampMan object type and how their object managers work.

Included Topics:

Attributes

Attributes represent those abilities and/or descriptive qualities that only some of the characters/races have (i.e. absolute direction, magical ability, tracking skills, etc). Attributes help "round out" a character and make him unique.

Attributes come in three types: assigned, calculated and owned. Assigned attributes are assigned their level and have an associated maximum level they may be assigned. Calculated attributes calculate their level (using levels from other attributes, characteristics, etc). Owned attributes may only have a level of 1 or 0, that is they are either "owned" or "not owned". All attributes have a calculated cost. You may select the attribute's type using the combo box in it's manager.

Each attribute uses two equations. The bottom entry field is labeled "cost" and accepts the cost equation. The entry field just above the cost field is for the equation which calculates the level, the maximum level, or the restriction. This field is labeled to remind you what the equation is supposed to calculate.

When an attribute's level/max level/restriction equation evaluates to 0 for a given race or character, CampMan assumes that the attribute is restricted to that race/character and refuses to add it to (or removes it from) the object's attributes.

When an assigned attribute's max level equation evaluates to -1 for a given race or character, CampMan assumes the attribute has no maximum level for that object.

For more on equations see Equations.

Unlike characteristics, CampMan assumes attributes have a positive level. If your original concept was for a disadvantage with strictly negative levels, you should simply label it a disadvantage and use positive levels to indicate the severity of the disadvantage. If your original concept was an attribute with both positive and negative levels, you should use an attribute list with two members: an advantage and a disadvantage. See Attribute Lists for more information.

Attribute Lists

An attribute list is a group of attributes, only one of which may be owned by a race/character at any give time.

An attribute list will often be used to create the equivalent of a single attribute which uses names for its levels rather than numbers. Another use for an attribute list is to group conflicting attributes together so that no two of them can be owned at the same time.

Attribute list members are treated as separate attributes when an equation is being evaluated. Thus a variable with the name of one member will not be replaced with the level of another member unless it is preceded with the @list: modifier. For more on equations and variables see Equations.

Although each member of an attribute list appears to the program as if it is a separate attribute, they are all actually part of one object. Whenever you open an attribute list's member, you in fact open the attribute list itself. Thus all attributes in an attribute list share a common object manager and share a common attached note.

When an attribute list is first created, you are prompted for a new member name since attribute lists can not exist without at least one member. An attribute list manager looks and acts exactly like an attribute manager except for the name field which has been replaced with a member combo box.

The name selected in the member combo box indicates which attribute's properties are being displayed/modified by the type, level, and cost fields. Each of these properties may be changed for each member of the list.

To create a new member for the list, choose "New" from the "Attribute" menu. To delete the selected member, choose "Delete". To rename the selected member choose "Rename".

When an attribute list needs to be identified by a name, it uses the title of its attached note, or the name of one of its members if there is no attached note.

For more on attributes see Attributes.

Characteristics

Characteristics represent those abilities and/or descriptive qualities which are a part of every character (i.e. strength, intelligence, agility, hit points, etc).

There are three types of characteristics: assigned, calculated and random. You may select the characteristic's type by using the "type" combo box in its manager.

Assigned characteristics are assigned a level and have a cost associated with that level.

Calculated characteristics do not have a cost associated with them since their level is calculated (using levels from other characteristics, attributes, etc).

Random characteristics are similar to calculated characteristics in that they have no associated cost and their level is calculated. When a random characteristic's level equation is evaluated, all dice expressions are replaced with appropriately generated random numbers.

A random characteristic's level is calculated only for characters and is calculated only once (i.e. when the character is created). A race's level for a random characteristic is assigned and is used as the minimum level required by characters to be assigned as members of that race. For more information see A Character's Race.

Random characteristic levels for characters may be changed during GM Access mode by checking the "Edit Random (gm)" box in the Program Configuration dialog.

Each characteristic uses one equation to calculate its cost/level for races and characters. This equation is entered/displayed in the bottom entry field in each Characteristic Manager. This field is labeled "cost", "level" or "init level" to remind you what the equation is used to calculate.

For more on equations see Equations.

subtopics:

Characteristic Entry Sections

The characteristics displayed in creature, race, and character managers are grouped into an input section surrounded by a box. There are two ways to affect how this section will look.

The size of the characteristics box can be specified using the campaign configuration dialog box. Simply enter the number of rows/columns of characteristics to display. Characteristics are listed within columns first. (i.e. first four in the first column, second four in second column, etc.)

The order in which characteristics are listed can be specified by using the characteristics list in the main window. This list is a special list that can be reordered. To move a characteristic, simply click and hold the middle mouse button on the characteristic to be moved. Then drag the indicator box to a new position and release the button.

Hidden Characteristics

Characteristics may be hidden. Hidden characteristics are not shown and may not be edited by the user while the program is running in Player Access mode. When a character's race changes, any hidden assigned characteristics are assigned new values equal to those of the new race.

A characteristic may be either hidden or shown by selecting it in the main window's characteristics list and choosing "Hide" or "Show" from the "Characteristics" menu.

Hidden characteristics may be made visible during GM Access mode by checking the "View Hidden (gm)" box in the Program Configuration dialog.

Characters

Characters represent the people (played by the GM or by the roleplayers) in a campaign.

The character manager is probably the most complicated window used in Campaign Manager. It not only has name, player and race entry fields but also has a cost button, a characteristics section with multiple entry/display fields and two list boxes used to display/modify a character's attributes and possessions.

The player entry field is important, since this not only indicates the character's author, but also restricts others from editing the character in Player Access mode. When CampMan is in Player Access mode, only those characters whose player name is the same as the entered user name can be edited.

A character's cost is calculated whenever the character is opened, the character is saved, "Recalculate" is chosen from the "Character" menu, or the cost button is pushed. If the campaign is configured to add race costs to the character cost, they will be added to the cost of characteristics and/or attributes. The total will be shown in the cost button.

In the characteristics section, each assigned characteristic has an entry field and each calculated characteristic shows its last calculated level. To recalculate the calculated characteristics, you must recalculate the character's cost. To change all the characteristics to the their default, racial values choose "Def Characteristics" from the "Character" menu.

The attributes section lists all the characters attributes along with their levels. To add an attribute choose "Add" from the "Attributes" menu, enter the attribute name followed by a space and the desired level, then press return. To change the level for an attribute, select the attribute, type in the new level, and press enter. Finally, to remove an attribute, select it and choose "Remove" from the "Attributes" menu.

If a character's race has attributes, they are automatically added to the character's attributes if they are not already listed. Although you can change a racial attribute's level, or switch to another member of its attribute list, you may never completely remove a racial attribute from a character. Deleting a racial attribute just changes that attributes level back to its racial default. You may change a character's attributes to the racial default list by choosing "Def Attributes" from the "Character" menu.

The possessions section works much like the attributes section, but is used to enter items and the number owned. To add a possession choose "Add" from the "Possessions" menu, enter the item name followed by a space and the desired number, then press return. To change the number of an item owned, select the item, type in the new number, and press enter. Finally, to remove a possession, select it and choose "Remove" from the "Possessions" menu.

Related Topic:

A Character's Race

A Character's Race

The race entry field is used to indicate/change the character's race. The character's race will probably have an affect on its cost and/or the attributes it will possess. It is important to note that a new race entered into the race field is not actually assigned to the character until the cost is recalculated. It is best to always recalculate the character just after changing the race field, so that you won't get confused as to what race is actually assigned to the character.

When a character's cost is recalculated and a new entry has been placed in the race field, CampMan checks that there actually is a race of that name and that the character is allowed to become a member of that race. To become a member of any given race, a character must have levels in each of its random characteristics which meet or exceed the levels assigned to that race's random characteristics.

But once a character has been assigned as a member of a given race, no more checking is done. Once a character is an Elf, it will legally remain an Elf until its race is changed to something else. To change the character's race back to Elf again would require that the character meet the minimum requirements again.

This checking does not occur while running in GM Access mode. Thus a GM may reassign a character to any race and "make it stick".

Creatures

Creatures are used to keep track of monsters and animals within your campaign.

Creatures, like characters and races, have characteristics and attributes. But there are no restrictions as to how these attributes/characteristics are assigned. It is assumed there is a good reason for any discrepancies that arise.

In many ways, the creature manager is just a form you fill which keeps you from adding non-existent attributes or two attributes from an attribute list. For creatures, all characteristics and attributes are treated as assigned. Also, there are no restrictions on what attributes you may add or what levels you may assign.

Adding attributes in a creature manager is just like adding attributes in a race or character manager. Simply choose "Add" from the "Attributes" and enter the attribute you wish to add, followed by a space and the level, then press return. You may change an attribute's level by selecting it, typing the new level and pressing return. Finally you may remove an attribute by selecting it and choosing "Remove" from the "Attributes" menu.

Note: Creatures may not be part of a character's possessions and may not be used as a character's race. If you wish to create a domesticated animal, simply create both a creature and an item with the same name. If you wish to keep track of both a monster and a race with the same name but different attributes, you must create both a creature and a race.

Items

Items are the objects which people may possess in your campaign.

Items are the simplest object supported by Campaign Manager. Each has a name and that's it! To make the Item Manager a little more useful it also has a notes pane. The notes pane contains the text from the item's attached note. This way, you can enter/modify the item's description without the need to open the attached note. However, if you want to add an attached GM note or change the attached note's title/topics, you will have to open the attached note's manager.

Notes & GM Notes

Notes are exactly what their name implies, text which you have entered to describe or explain some topic of interest. Although many of the notes you create will be attached to another object -- used to describe that object -- others will stand on their own, containing explanations of special rules or histories or other interesting tidbits about your campaign.

The Note Manager has three entry fields, one for the title, one for a list of topics and one for the actual text which comprises the note. The title, of course, should describe the primary subject that the note addresses. The title is what is shown in the main window's objects list if the note is unattached.

The topics field is used to hold a comma separated list of topics on which the note applies or discusses. Topics are used to help categorize a note and how it relates to other notes or objects. Although the topics are not displayed in the main window, good topics lists help immensely when you use the Note Search dialog.

GM notes are special notes which only the GM is supposed to read. GM notes may not be read in Player Access mode. In fact, GM notes are not even listed in Player Access mode.

subtopics:

Attached Notes
Attached GM Notes

Related Topic:

The Note Search Dialog

Attached Notes

Although a note or GM note may be created separately from other objects, it is very useful to keep object descriptions/explanations closely associated with those objects. To this end, Campaign Manager utilizes *attached notes*.

Attached notes are actually very important, since they are the best way to let a player know what he needs to know about an object. In Player Access mode, the user can only view notes and the characters he creates.

When the user tries to open an object other than a note or character in Player Access mode, the attached note is opened. As long as a good description of every object is kept in an attached note, a player will never be left wondering exactly what an object represents.

In GM Access mode, an attached note is not accessed through the main window, but through the first menu on an object manager's menu bar (i.e. the "Characteristic" menu for Characteristic Managers). The "Create Notes" menu item is used to create a note attached to an object. The "Open Notes" menu item is used thereafter to open the object's attached notes. In this way, descriptions may be kept of all objects without cluttering up the objects list with a "Note" type under every object name.

Attached GM Notes

GM notes may be attached only to notes and notes may only have GM notes attached to them. In this way, a special addendum or "side note" may be added to a regular note which can only be read by the GM. To attach a GM note to an object other than a note, you first must create an attached note for the object and then create an attached GM note for the attached note.

The Note Search Dialog

The Note Search dialog can be used to help find a note for which you have forgotten the title, or all the notes which pertain to a given subject, or all the notes which use a word or phrase within their text. For example, a player might use the Note Search dialog to find all the characteristic descriptions by searching on the topic "Characteristic". After a list of notes has been found, they can be opened directly from the Note Search dialog.

To open the Note Search dialog you must choose "Search Notes..." from the "Notes" menu on the main window's menu bar. The Notes Search dialog uses three entry fields, one or two three-state check boxes and two radio buttons to specify search criteria. It has four push buttons used to indicate what action you wish to take, and it has a list box used to display the set of notes found in the last search.

The Start Search button is used to find all notes which match the search criteria. When the Note Search dialog is first opened, the fields are set to search for all notes and GM notes. To limit the search to only notes/GM notes then you must clear/ check the "GM Notes" check box. Similarly, to limit the search to only attached/ unattached notes you must check/clear the "Attached" check box.

If you know a portion of the note's title you wish to find, you may enter it into the title field. Similarly, if you know a word or phrase which is contained in the body of the note, you may enter it into the text field. Finally, to search for notes which pertain to a particular subject(s), you may enter a comma separated list into the topics field. If "some" is checked in the "match topics" field, then all notes with at least one of the topics listed will be found. If "all" is checked, then only those notes which have all the topics listed will be found.

After a search is made, the titles of all the notes matching the search criteria are displayed in the "found" list box. If no notes were found that matched the given criteria, "... none found ..." is displayed. To open one of the notes displayed in the found list, simply double click on its title.

The Append button is used to add more notes to the current list of found notes. Whenever Append is pressed, the group of notes which match the new search criteria do not replace the current list, but are instead added to the current list.

The Narrow Search button is used to narrow the number of notes displayed in the found list. By specifying new search criteria and then pressing Narrow Search, you find notes which meet both the last search criteria and the new search criteria.

The Last Search button is used to retrieve the last search criteria which found at least one note, along with the found list of notes.

The following procedure may be used to determine if a given note matches the specified search criteria:

Note Search Matching Procedure

Note Search Matching Procedure

1) If the "Append" button was pressed and the note is currently listed in the found list, then the note matches the search criteria regardless of the following conditions.
2) If the "Narrow Search" button was pressed and the note is not listed in the found list, then the note does not match the search criteria.
3) If the note is a GM note and the "GM Notes" check box is clear (or does not exist), then the note does not match the search criteria.
4) If the note is not a GM note and the "GM Notes" check box is checked, then the note does not match the search criteria.
5) If the note is attached to another object and the "Attached" check box is clear, then the note does not match the search criteria.
6) If the note is not attached to another object and the "Attached" check box is checked, then the note does not match the search criteria.
7) If the title field is not empty and the note does not include the entered text as part of its title, the note does not match the search criteria.
8) If the topics field is not empty and the note includes none of the entered topics, then the note does not match the search criteria.
9) If the topics field is not empty, the "all" radio button is on and the note does not include all the entered topics, then the note does not match the search criteria.
10) If the text field is not empty and the note's text does not include the entered text, then the note does not match the search criteria.
11) If none of the above conditions are true, the note matches the search criteria.

Races

Races not only represent the different races a character may be but also indicate common default values for characteristics and attributes which are owned by every member of that race.

Other than the name entry field, race managers have two major input sections and a cost button. The first section is used to enter characteristics. Each assigned and random characteristic has a corresponding entry field. Each calculated characteristic displays its last calculated value. To recalculate the race's characteristics, you must recalculate the race's cost by pressing the cost button.

The cost button displays the sum of all characteristic and attribute costs for the race when the costs were last calculated. These costs are obtained by evaluating the cost equations for attributes and assigned characteristics. A race's cost may or may not effect the cost of a character depending on how the campaign is configured.

The last input section is for attributes. This list box shows all the race's attributes and the level assigned/calculated for each. To add an attribute choose "Add" from the "Attributes" menu, enter the attribute name followed by a space and the desired level, then press return. To change the level for an attribute, select the attribute, type in the new level, and press enter. Finally, to remove an attribute, select it and choose "Remove" from the "Attributes" menu.

Related Topic:

A Character's Race

Printing

This version of Campaign Manager includes some rudimentary printing capabilities for Notes, GM Notes, Creatures, Races, and Characters. This version also introduces the capability of printing a "cost breakdown" for a race or character.

To print any of the above object types you must open its manager and select "Print" from the first menu on the menu bar (i.e. the "Note" menu for note managers). To print a cost breakdown you must select "Breakdown" from the "Cost" menu.

In either case, this will bring up a "Printing Dialog" which allows you to select the printer, the font and the font point size you wish to use. You may also invoke the currently selected printer's "Setup Dialog" which will let you set the more advanced/specific features of the printer.

CampMan always prints the information based on what is displayed by the object manager, not necessarily the information saved to memory/disk.

At this time, CampMan can't print multiple pages. This can cause the printed information to "scroll off the page". Should this happen, changing to a smaller size font or switching the printer to "landscape" mode (with the setup dialog) might alleviate the problem.

The following two sections discuss the specific aspects of printing the different object types.

Included Topics:

Printing Notes & GM Notes

Notes and GM Notes are printed with their title centered at the top of the page followed by their list of topics and the text wrapped between the page margins. GM Notes have "(GM NOTES)" displayed under their titles.

subtopic:

The Problem With Tables

The Problem With Tables

Since CampMan doesn't fully support WYSIWYG displays of a note's text, it can be difficult to format a table of information that will both display nicely and print nicely. Here are some hints:

- **Do Not Use Tabs**
  use only spaces to format a table
- **Use Solid Table Lines**
  when separating titles/rows of a table use a solid dashed line rather than separate lines for each column. These separation lines might print too long or short but at least they won't be "misaligned"

The "Text" can be used to switch the text displayed between the System font and an approximation of the last used printer font. Although this may help in arranging a table to print properly, I inexplicably have had more success with the System font.

Printing Creatures, Races & Characters

For this discussion I will refer only to printing Characters. Creatures and Races print the same with appropriately omitted sections of information.

CampMan prints a Character's name and player centered at the top of the page. It then prints the characteristics, formatted like the displayed characteristics, in the upper left corner. Next to the characteristics, the character's race is printed. The character's attributes and possessions are printed in separate sections fitted below and around the other information. The character's total cost is displayed in a rounded rectangle in the upper right corner of the printout.

Before a character is printed, CampMan recalculates its cost/levels to insure that it is printing a "legal" character. If any problems arise it will inform you and probably refuse to print. In the special case that an attribute was found to be illegal and subsequently removed, you will be given the option to continue printing anyway. You probably want to abort printing since the character is no longer the one you originally intended to print.

Printing Cost Breakdowns

The cost breakdown is a listing of each assigned characteristic and each attribute along with the cost it contributes to the total cost of a race or character. If the campaign is configured to add racial costs to a character, the total cost of racial characteristics and/or racial attributes will be listed as a separate "race cost" for characters. If any hidden, assigned characteristics contribute to the cost, they will be totaled and listed as a "hidden cost" in the characteristics subsection.

The printout consists of up to three subsections; one each for characteristic costs, race costs, and attribute costs. In addition to the separate costs for each characteristic and attribute, those two subsections also have a "subtotal" of the costs listed. Centered at the top of the page is the name of the race/character, the text "Cost Breakdown", and a copy of the total cost.

Before a cost breakdown is printed, CampMan recalculates the cost/levels to insure that it is printing a "legal" race/character and that all the costs are current. If any problems arise CampMan will inform you and probably refuse to print. In the special case that an attribute was found to be illegal and subsequently removed, you will be given the option to continue printing anyway. You probably want to abort printing since the race/character is no longer the one you originally intended to print.

The Dice Roller

The Dice Roller is a special dialog box that lets you roll sets of dice and see all the results or evaluate a dice expression and see the results. You open a Dice Roller by selecting "Dice Roller..." from the "Campaign" menu in the main window.

To roll a set of dice, all with the same number of sides, and see the results make sure the "# dice" button is checked and enter a number of dice and sides in the two entry fields. You may use the direction keys to increase/decrease these values by 1 or you may use the "PgUp" and "PgDn" keys to increase/decrease these values by 10. Pressing a number key while the "# dice" field has the input focus will enter the given number into the field. While the "sides" field has the input focus some number keys will enter a commonly used dice (i.e. 1 for a 10 sided die, 2 for 20, 4 for 4, etc). The "sides" field also has a pop-up menu that can be used to enter commonly used dice.

Once you're satisfied with the field values, press the Roll Dice button. If "add dice" is checked, the dice are added together and the total is shown in the bottom field. If "add dice" is clear, the results of each die roll will be listed separately (you may need to resize the dice roller to see all the results).

You may also use the Dice Roller to evaluate a dice expression. To do this, make sure the "exp" button is checked and enter an equation into the exp field. This is exactly like entering an equation into the Equation Editor and the results are exactly the same except that dice expressions are replaced with appropriate random numbers when the equation is evaluated. When you press Roll Dice, the equation will be compiled and evaluated (variables will be replaced with 0's). The results are shown in the bottom field.

EXAMPLE: to display the results of rolling 3 six-sided dice and adding 3 enter the following expression into the exp field

3d6 + 3

EXAMPLE: to display the results of rolling two 10 sided dice and interpreting them as a percentile enter the following

1d10 + 10 * (1d10 - 1)

EXAMPLE: if you define a function called maxSum3of4() as follows

max(arg1 + arg2 + arg3, arg1 + arg2 + arg4, arg1 + arg3 + arg4, arg2 + arg3 + arg4)

then the following expression will answer the sum of the three largest numbers among four six-sided dice

maxSum3of4(1d6, 1d6, 1d6, 1d6)

For a full discussion of equations and functions see Equations and Functions.

Program Configuration

To change how the program behaves or to set/change the program password you use the Program Configuration dialog. To open this dialog, choose "Program" from the "Configuration" menu on the main window's menu bar.

The Program Configuration dialog has four check boxes and four buttons. Each check box is used to activate/deactivate a program feature.

Dialog Emulation

When Dialog Emulation is active, the object managers respond to navigation keys like dialog boxes. This means that the input focus will shift when the tab/direction keys are used and the default button (indicated by a thick border) will be pressed when the enter key is used.

If Dialog Emulation is active, the Ctrl key must be held down to put a tab or carriage return in a multi-line entry field. Also, the menu bar may not respond to the Alt key unless one of the entry fields has the input focus. If this should occur, simply press the Tab key once and try again.

Application Minimize

If Application Minimize is active when the main window is minimized, all CampMan windows are hidden from view until the main window is restored. This gives you a quick way to "minimize" all the CampMan windows if you need to clear the screen for something else without closing CampMan.

However, this means you can't minimize the main window to make room for other object managers.

View Hidden (gm)

View Hidden is used to display/modify hidden characteristics while running in GM Access mode. If View Hidden is not active, hidden characteristics may not be viewed in GM Access mode.

This feature has no effect on Player Access mode.

Edit Random (gm)

Edit Random is used to modify a player's random characteristics while running in GM Access mode. If View Random is not active, random characteristics may not be modified in GM Access mode.

This feature has no effect on Player Access mode.

Keep Backup File

If Keep Backup File is active, whenever a campaign is saved to a file which already exists, that file is renamed to have the ".BAK" extension instead of deleted. Whether this feature is active or inactive, any backup file already in existence is deleted.

Save Prog Configuration at Exit

If Save Prog Configuration at Exit is active, the current settings of this and the other three features will be saved when you exit the program. The settings will then be restored, the next time you run CampMan.

If you press the Cancel button, any changes to the above four features will be ignored. Pressing the OK button activates any checked features and deactivates any unchecked features. Pressing the Save button not only activates/deactivates the features, but also saves the settings to the WIN.INI file.

To set/change the program password, press the "Prog Password..." button. This brings up a dialog box used to change the program password. To successfully change the program password, you must enter the old password correctly and enter the new password twice for verification. If you should press the Cancel button, the dialog will close and the program password will remain unchanged. Pressing the OK button while the other fields are correctly filled, immediately and irrevocably changes the program password from the old password to the new password.

Campaign Configuration

To change how the campaign works or to set/change the campaign password you use the Campaign Configuration dialog. To open this dialog, choose "Campaign" from the "Configuration" menu on the main window's menu bar.

The Campaign Configuration dialog has four sections and three buttons.

Characteristics

> This section is used to control the appearance of the characteristics box in creature, race and character managers. It specifies either a number of columns or a number of rows to use when listing characteristics in those managers.

Attributes & Possessions

> These two sections are use to control the placement of the attributes and possessions lists in creature, race and character managers.

Race costs added

> This section is used to determine which costs (if any) from a race should be added to a character of that race. You should be careful when either or both of these costs are added to characters to make sure that all cost equations take into account the character's race. Improper use of these settings can artificially inflate the costs of a character.

If you press the Cancel button, any changes to the above four features will be ignored. Pressing the OK button activates any changes made. These settings are saved to the file with the campaign. When a campaign is loaded, these settings are reset to the values they had when that campaign was last saved.

To set/change the campaign password, press the "Camp Password..." button. This brings up a dialog box used to change the campaign password. To successfully change the campaign password, you must enter the old password correctly and enter the new password twice for verification. If you should press the Cancel button, the dialog will close and the program password will remain unchanged. Pressing the OK button while the other fields are correctly filled, immediately and irrevocably changes the campaign password from the old password to the new password. This password will be saved to the file when the campaign is saved.

USE CAMPAIGN PASSWORDS Sparingly and keep good records of the passwords you set because there is nothing I can do to help you if you forget a campaign's password. Without a campaign's password, you might as well scrap that file and build a new one.

Menus

This section describes the menus for the main windows and each object manager. Some menu choices may not appear in Player Access mode.

Included:

Main Window Menus (GM Access)

- Campaign
  - New — Create a new campaign
  - Load — Load a campaign from disk
  - Save — Save the edited campaign
  - Save As... — Save the edited campaign to a new file
  - Functions... — Open the Function Editor
  - Exit Program — Exit CampMan
- Objects
  - Open — Open the selected object
  - Copy — Copy the selected object
  - Delete — Delete the selected object
  - Filter... — Open the Filter dialog
  - Create
    - Item — Create a new Item
    - Note — Create a new Note
    - GM Note — Create a new GM Note
    - Characteristic — Create a new Characteristic
    - Attribute — Create a new Attribute
    - Attribute List — Create a new Attribute List
    - Creature — Create a new Creature
    - Race — Create a new Race
    - Character — Create a new Character
- Characteristics
  - Hide — Hide the selected characteristics
  - Show — Show the selected characteristics
- Notes
  - Save — Save any modifications of the notes field
  - Find Text... — Open a Find Text dialog for the notes field
  - Search Notes... — Open the Search Notes dialog
- Configuration
  - Campaign — Open the Campaign Configuration dialog
  - Program — Open the Program Configuration dialog
- Help
  - Contents — Open help and display Contents topic
  - Welcome! — Open help and display Welcome! topic
  - Tutorial — Open help and display the appropriate tutorial
  - Reference — Open help and display The Main Window topic
  - Menus — Open help and display this topic
  - Using Help — Open help's help file
  - About... — Open a dialog box describing the software

Main Window Menus (Player Access)

- Campaign
  - Load — Load a campaign from disk
  - Save — Save the edited campaign
  - Exit Program — Exit CampMan
- Objects
  - Open — Open the selected object
  - Delete — Delete the selected object

	Filter...	Open the Filter dialog
	Create Character	Create a new Character
Configuration		
	Dialog Emulation	Toggle on/off keyboard emulation as for dialogs
	App Minimize	Toggle on/off application minimization
Help		
	Contents	Open help and display Contents topic
	Welcome!	Open help and display Welcome! topic
	Tutorial	Open help and display the appropriate tutorial
	Reference	Open help and display The Main Window topic
	Menus	Open help and display this topic
	Using Help	Open help's help file
	About...	Open a dialog box describing the software

Item Manager Menus

Item		
	Undo	Restore the original contents of all input fields
	Save	Save any changes to the edited item
	Delete	Delete the edited item
	Create Notes	Create and attach a note to the edited item
	Open Notes	Open the notes attached to the edited item
Notes		
	Find Text...	Open a Find Text dialog for the notes field
	Search Notes...	Open the Search Notes dialog
Help		
	Contents	Open help and display Contents topic
	Welcome!	Open help and display Welcome! topic
	Tutorial	Open help and display the appropriate tutorial
	Reference	Open help and display Items topic
	Menus	Open help and display this topic
	Using Help	Open help's help file
	About...	Open a dialog box describing the software

Note Manager Menus

Note		
	Undo	Restore the original contents of all input fields
	Save	Save any changes to the edited note
	Copy	Create a new note with the edited note's properties
	Delete	Delete the edited note
	Print	Print the currently entered title, topics and text
	Create GM Notes	Create and attach a GM note to the edited note
	Open GM Notes	Open the GM notes attached to the edited note
	Search Notes...	Open the Search Notes dialog
Text		
	Find Text...	Open a Find Text dialog for the text field
	Printer Font	Display the text in the (Window's approx) printer font
	System Font	Display the text in the System font
Help		
	Contents	Open help and display Contents topic

Welcome!	Open help and display Welcome! topic
Tutorial	Open help and display the appropriate tutorial
Reference	Open help and display Notes/GM Notes topic
Menus	Open help and display this topic
Using Help	Open help's help file
About...	Open a dialog box describing the software

Characteristic Manager Menus

Characteristic

Undo	Restore the original contents of all input fields
Save	Save any changes to the edited characteristic
Copy	Create a new object with the edited object's properties
Delete	Delete the edited characteristic
Create Notes	Create and attach a note to the edited characteristic
Open Notes	Open the notes attached to the edited characteristic

Equation

Check	Check if cost equation compiles without error
Edit	Open an equation editor for the cost equation

Help

Contents	Open help and display Contents topic
Welcome!	Open help and display Welcome! topic
Tutorial	Open help and display the appropriate tutorial
Reference	Open help and display Characteristics topic
Menus	Open help and display this topic
Using Help	Open help's help file
About...	Open a dialog box describing the software

Attribute Manager Menus

Attribute

Undo	Restore the original contents of all input fields
Save	Save any changes to the edited attribute
Copy	Create a new object with the edited object's properties
Delete	Delete the edited attribute
Create Notes	Create and attach a note to the edited attribute
Open Notes	Open the notes attached to the edited attribute

Equation

Check	Check if the selected equation compiles without error
Edit	Open an equation editor for the selected equation

Help

Contents	Open help and display Contents topic
Welcome!	Open help and display Welcome! topic
Tutorial	Open help and display the appropriate tutorial
Reference	Open help and display Attributes topic
Menus	Open help and display this topic
Using Help	Open help's help file
About...	Open a dialog box describing the software

Attribute List Manager Menus

Attribute

Undo	Restore the original contents of all input fields
Save	Save any changes to the selected member
Delete	Delete the edited member
New	Create a new member
Rename	Rename the currently selected member
Create Notes	Create and attach a note to the edited attribute list
Open Notes	Open the notes attached to the edited attribute list

Equation

Check	Check if the selected equation compiles without error
Edit	Open an equation editor for the selected equation

Help

Contents	Open help and display Contents topic
Welcome!	Open help and display Welcome! topic
Tutorial	Open help and display the appropriate tutorial
Reference	Open help and display Attribute Lists topic
Menus	Open help and display this topic
Using Help	Open help's help file
About...	Open a dialog box describing the software

Creature Manager Menus

Creature

Undo	Restore the original contents of all input fields
Save	Save any changes to the edited creature
Copy	Create a new creature with the edited creature's properties
Delete	Delete the edited creature
Print	Print the currently entered settings for the creature
Create Notes	Create and attach a note to the edited creature
Open Notes	Open the notes attached to the edited creature

Attributes

Add	Add a new attribute to the edited creature
Change	Change the assigned level of the selected attribute
Remove	Remove the selected attribute from the edited creature

Help

Contents	Open help and display Contents topic
Welcome!	Open help and display Welcome! topic
Tutorial	Open help and display the appropriate tutorial
Reference	Open help and display Creatures topic
Menus	Open help and display this topic
Using Help	Open help's help file
About...	Open a dialog box describing the software

Race Manager Menus

Race

Undo	Remove any entered changes since the last recalculation
Save	Save any changes to the edited race
Copy	Create a new race with the edited race's properties
Delete	Delete the edited race
Print	Recalculate and print the current settings for the race
Make Default	Make the edited race the campaign's default race

- Create Notes — Create and attach a note to the edited race
- Open Notes — Open the notes attached to the edited race

Attributes
- Add — Add a new attribute to the edited race
- Change — Change the assigned level of the selected race
- Remove — Remove the selected attribute from the edited race

Cost
- Recalculate — Recalculate the cost for the edited race
- Breakdown — Recalculate and print a cost breakdown for the race

Help
- Contents — Open help and display Contents topic
- Welcome! — Open help and display Welcome! topic
- Tutorial — Open help and display the appropriate tutorial
- Reference — Open help and display Races topic
- Menus — Open help and display this topic
- Using Help — Open help's help file
- About... — Open a dialog box describing the software

Character Manager Menus

Character
- Undo — Remove any entered changes since the last recalculation
- Save — Save any changes to the edited character
- Copy — Create a new object with the edited object's properties
- Delete — Delete the edited character
- Print — Recalculate and print the character's current settings
- Def Characteristics — Set the characteristic entry fields to the racial values
- Def Attributes — Set the character's attributes to the its race's attributes
- Create Notes — Create and attach a note to the edited character
- Open Notes — Open the notes attached to the edited character

Attributes
- Add — Add a new attribute to the edited character
- Change — Change the assigned level of the selected character
- Remove — Remove the selected attribute from the edited character

Possessions
- Add — Add a new item to the edited character
- Change — Change the number of the selected item owned
- Remove — Remove the selected item from the edited character

Cost
- Recalculate — Recalculate the cost for the edited character
- Breakdown — Recalculate and print a cost breakdown for the character

Help
- Contents — Open help and display Contents topic
- Welcome! — Open help and display Welcome! topic
- Tutorial — Open help and display the appropriate tutorial
- Reference — Open help and display Characters topic
- Menus — Open help and display this topic
- Using Help — Open help's help file
- About... — Open a dialog box describing the software

License Agreement/Disclaimer

You are hereby given the right to use this software (Campaign Manager for Windows ver 1.15) for 30 days on a trial basis. After 30 days you must pay a one-time registration fee of $20 U.S. to continue using the software.

New Fangled Software and Doug D'Angelo expressly disclaim any warranties for supplying accurate or functional materials relating to or including this software, The user assumes all risk as to quality and performance of the software. In no event will the liability of New Fangled Software or Doug D'Angelo exceed the price paid to register the software regardless of the form of the claim. Use or registration of this software indicates your understanding that you are using this product as-is, without any warranties of any kind.

If you register this software, you are responsible for insuring that the registered software will be used only by a single user*. Further, you warrant that you will not knowingly distribute the registration number given to you.

*The players/gm associated with the registered user may use this software on his/her machine but must register if they wish to use it on their own

Registration Form

CAMPAIGN MANAGER FOR WINDOWS ver 1.15

REGISTRATION FORM

To obtain a registration number, print and fill out this form. Mail it along with your registration fee to:

New Fangled Software
3671 Rocky Creek Court
San Jose, CA. 95148 USA

To ensure that the name you wish to register under can be used, enter your name in the Registration Dialog, leave the reg number blank and press OK. A message box will inform you if the name can be used for registration purposes.

This registration does not grant you the right to use Campaign Manager as part of a commercial business or enterprise. Should you wish to use CampMan as either a testing or a production tool as part of a business, please contact New Fangled Software directly at the above address to discuss licensing terms.

Please type or print neatly:

Name: _____

Street Address: _____

City, State: _____

Zip Code: _____

Phone Number (optional): _____

Compuserve ID (optional): _____

please fill out the amount paid:

Registration Fee	$20	_____
Mail you copy of 1.2 when released? (optional)	$5	_____
Tax (CA residents only..... 8.25% for Santa Clara County 7.25% for all other Ca residents)		_____
Foreign postage (outside U.S.)	$2	_____
	total:	_____

Your signature indicates that you have read and understand the included License Agreement/Disclaimer. In addition, your signature warrants that the program registered will be used by only a single user and that you will not knowingly distribute the registration code given to you. No registration forms without a signature will be processed.

your signature: _____

Thank you for registering, Douglas D'Angelo

Product Support

You may use electronic mail to send questions, problems and/or bug reports for Campaign Manager for Windows. Send your mail to:

Doug D'Angelo

Compuserve ID: 72611,1263

Registered users may get answers to their questions by calling (408) 223 - 2461 from Monday through Friday between the hours of 10 A.M. and 6 P.M. Pacific Standard Time. Have your registration number handy for possible verification.

Glossary of Terms

attribute list manager
attribute manager
character manager
characteristic manager
creature manager
default race
input focus
item manager
middle mouse button
note manager
object manager
pop-up menu
race manager
three-state check boxes

input focus

The window pane/field that responds to keyboard typing is the window pane/field with the "input focus".

Entry fields indicate they have the input focus by displaying a flashing cursor at the current insertion point. Lists show they have the input focus by displaying a dotted-gray rectangle about the current selection. Buttons show they have the input focus by displaying a dotted-gray rectangle about their label.

middle mouse button

The third mouse button on three-button mice. If your mouse does not have three buttons, hold down the shift key while using the right mouse button.

three-state check boxes

This is a special kind of check box that can be checked, unchecked or "grayed". Campaign Manager generally uses the grayed state to indicate a "don't care" or "both" condition.

attribute list manager

the object manager for attribute lists

attribute manager

the object manager for attributes

character manager

the object manager for characters

characteristic manager

the object manager for characteristics

creature manager

the object manager for creatures

default race

This is the race upon which all other races and characters are based. Any changes to the default race have an effect on all races and through them on all characters.

item manager

the object manager for items

note manager

the object manager for notes and GM notes

object manager

Each type of object in CampMan has a different manager. The object's manager is the window you use to enter/modify the objects properties.

pop-up menu

A pop-up menu is a special menu that is accessed by clicking the right mouse button within a window. The pop-up menu usually contains only specific functions used with that window.
Many of the entry fields in Campaign Manager have pop-up menus for the user's convenience.

race manager

the object manager for races

To register this program,
fill out and send in the

$ K + Registration Form

You can use the File Manager to create directories and copy files. See the Windows Manual.

Command line parameters
can change the startup
behavior of the program.
See Command Line Parameters.

Once a variable has been selected, the only way to remove the selection completely is to press the evaluate button.

variables and functions are replaced by numbers before an equation is evaluated, so they may be used as if they were numbers

The value field may display an error rather than a number.
See Compilation Errors and Evaluation Errors for more information.

The save button is disabled until the entered definition compiles without error when the evaluate button is pressed.

functions may have empty argument lists

one unary operator may proceed any operand without causing this error

When an object name is selected and the list of object names has the input focus, pressing Ctrl-C copies the selected name to the Windows Clipboard.

The "Attribute Lists" object type is a special case. If "Attributes" is checked but "Attribute Lists" is unchecked, then each member of an attribute list will be shown. If "Attribute Lists" is checked, then only the attribute list's name is shown.
The "all" button does not check the "Attribute Lists" check box.

Since owned attributes always have a level of 1 when they are listed, Race and Character managers do not indicate a level for them in the attributes list.

Whenever you start typing on the keyboard while the attributes list has the input focus, an entry field will pop up to accept your input.
In addition, you may simply press "Ctrl-V" and the entry field will pop up with the text currently in the Windows Clipboard already entered.
Finally, when you press "Ctrl-C" the selected attribute's name followed by its assigned level is copied to the Windows Clipboard.

Whenever you start typing on the keyboard while the attributes list has the input focus, an entry field will pop up to accept your input.
In addition, you may simply press "Ctrl-V" and the entry field will pop up with the text currently in the Windows Clipboard already entered.
Finally, when you press "Ctrl-C" the selected attribute's name followed by its assigned level is copied to the Windows Clipboard.

While running in Player Access mode, GM notes may not be searched for and the "GM Notes" check box will not be present.

Titles for GM notes are preceded by "(gm)" in the list box

Whenever you start typing on the keyboard while the attributes list has the input focus, an entry field will pop up to accept your input.
In addition, you may simply press "Ctrl-V" and the entry field will pop up with the text currently in the Windows Clipboard already entered.
Finally, when you press "Ctrl-C" the selected attribute's name followed by its assigned level is copied to the Windows Clipboard.

This is a full fledged modeless dialog box.
You can switch the input focus with the Tab and direction keys
and pressing the Return key will roll the dice.
This is true whether the "emulate dialogs" feature is on or off.

This dialog can be accessed through the startup dialogs.
Start the program,
press the "View License..." button
and then press "Register" button.