City of Heroes / City of Villains
Technical Reference Guide

(formerly the *CoH/CoV Keybind, Macro & Emote Guide*)

V2.76 – June 2019 (Issue 14+/Post-Live)

Compiled by
The Legendary Death-Defying
Shenanigunner
@Shenanigunner or **shenanigunner [at] dgath [dotcom]**
Now always found at **www.Shenanigunner.com**

Version Notes

- This is a major update to a guide I wrote over three or four years, back in the Live era. I've renamed it since it now covers so much ground regarding the "hidden" parts of the game.
- There are many, many, many references out on the web, as docs, forum posts, wiki entries and the like. I don't know of any that are as comprehensive, organized and thoroughly checked out as this one… but feel free to let me know of any competitors so I can challenge them to an Arena fight.
- I took the information in here almost entirely from the game itself, long before most guides and lists were widely available. The various "command dump" commands were very useful in finding and listing things. When I found new information in other lists, I followed up and verified it myself. Very few of these lists had author or creator names attached, or I would have gladly credited them. As it is, most of the game information is drawn from the same resources I used; I don't think there's one word in here that is someone else's unique creation.
- I simply never thought this material would be useful again, or that I'd spend several days happily updating and checking it. It's been a pleasure!
- This has been something of a lonely effort, since I never spent much time in the various online communities. It would mean a lot if every user who finds this guide helpful could drop me a note, send along a contribution or correction or game story, and most of all pass the word… repost and cross-list this Guide all to hell out there in the new Cityverse.
- Or you can throw some Inf at me, @Shenanigunner, on Excelsior. I wouldn't mind.
- While I retain rights to the specific effort used to create this guide, all of its contents are released (or just left in) the public domain and may be copied anywhere by anyone… but credit and a link to the whole thing would be really appreciated.
- **UPDATE: See the website for the new GABB updated bindfile set! …and a whole lot more!**

TOC

SECTION 0 – INTRODUCTION

0.0 WE'RE BACK!

Literally ten years later, our Cities and our community are back. Welcome to this never-expected update of what I hope the new community will find useful. **All material new to this update is highlighted in red.**

And I've renamed the guide to what it's become: a complete technical reference guide to the game's background features and options.

0.1 What are Keybinds & Macros?

Keybinds and macros are ways to remap the keys, mouse buttons and game commands into a control configuration that better suits a given player's play style and preferences. Instead of being locked into a fixed set of control keys and commands, as some older games do, or providing a simple reassignment feature as most newer games do, sophisticated games like City of Heroes/City of Villians permit you to remap, change and combine the game commands and controls in an almost unlimited fashion.

For example, instead of simply letting you change your "run" command from the default R key to another, keybinds allow you to bundle two or more commands onto one key, so that you initiate running and go into Super Speed at the same time. Another example is the very useful "engage" keybind, which targets the nearest foe and locks you onto him in "follow" mode. For a melee player (scrapper or tanker) in the middle of a multi-foe fight, being able to whack one key and lock onto a foe for focused attacks can change your play style and success rate.

Or, on the fun and silly side, you can combine chat strings with actions – a local string "C'mere, you ugly SOB!" with a taunt, or "Roast in hell!" combined with a AoE (multi-foe area of effect, that is) Burn or Scorch power. Or "Let's get 'em!" combined with a suitable emote (character animation), to tell your teammates it's time to get down to business. (There are several of those predefined in the QuickChat emote menu.)

A macro is exactly the same as a keybind, except that the string of commands is bound to a power-tray button and has to be activated by a click or an associated power-activation command (by default, the associated keyboard-top number or alt-number). Generally, you should use keybinds for commands and command sequences you need to activate quickly and often, while macros can be used for actions for which you'll have time to find and click a power button.

Some players might be happy with a few reassignments from the default command keys. Others might want to do the crazy thing and create a completely custom mapping of everything. However, all players can benefit from a few keybinds that make key powers speedier and easier to use in the heat of battle.

0.2 Why do I need a Keybind & Macro Guide?

Creating effective keybinds (or binds, for short) and macros takes some knowledge and experience, which the basic game manuals don't really cover. At a minimum, you need to know the basic syntax for writing a bind or macro, and have a list of all the console or "slash" commands (so called because they begin with a slash that identifies them as commands when you type them into the chat window).

Since the developers of CoH don't provide a comprehensive guide, and are a little loose about providing consistent information about the list of available slash commands with each update, it's fallen to the player community to keep track of the commands and teach each other how to use them. A guide of some sort is essential to help you master this complicated and flexible set of commands.

0.3 There doesn't seem to be any shortage of Bind & Macro Guides. Why this one?

Any time documentation is written by a community of users, you're going to see some common limitations. Those who are real hotdogs with the tools may not be very good at writing about them; those who can write well may not know enough about the process to get all the details right; even those who can do both might not have time to gather all the details or keep things updated.

I set out to write this guide because, as a player new to CoH (as was everyone at one time) and a player new to multiplayer online games, I couldn't find a good, complete, up-to-date guide that was written in language a non-MMOG maven could understand. The guides that were any good in the information

department seemed to be written in poorly-translated Martian, assuming far too much previous knowledge on the part of the reader. And the guides that were incomplete, sloppily compiled and out of date were even more frustrating for a newcomer, because it was hard to identify what was right and what was useless.

So here's this guide, added to the pile, and I hope an improvement and of value to both new and existing players. My aim was to combine many years of experience writing software and programming manuals (most for novice and nonexpert users) with my fascination of City of Heroes and all the reliable information I could lay hands on. It's my aim to keep it updated, both with new info gleaned from the forums and other users, and from reader feedback.

0.4 How This Guide is Organized

The organization of this guide is simple. Section 0 is the Introduction, which you're reading. Section 1 is Basics, like terminology and syntax, which I urge all users to read carefully, so that they can follow the terser language in the following sections. Section 2 explains how to write and use Keybinds. Section 3 covers Macros (mostly, in how they differ from Keybinds). Appendix A lists all the known console or slash commands, with their individual syntax and notes on how to use them. Appendix C lists the names of all the bindable keys. Appendix D lists the known window and menu names. And Appendix E lists all the known "emote" commands, with notes where appropriate.

0.5 Updates

For many reasons, this will probably be the final update of this guide. The last update was around Issue 8 in late 2006 and not very much grew out of date; even with massive changes to the game the command and emote set hasn't changed as much as in prior Issues. I am also nearing my end of involvement with the game, at least to the point where I am inspired to maintain this guide. But you never know…

Updates and corrections, especially to the command and emote lists, are encouraged. Comments on everything are welcomed. And pestering when I let the guide fall out of currency is solicited – I have a tendency to move on and not maintain efforts like this, especially when there's no feedback.

And then came 2019…

Email to **shenanigunner [at] dgath [dotcom]**, or to whatever maintenance email address is listed on the web site at **http://www.dgath.com/coh/**, is the best way. You can also send in-game mail to @Shenanigunner, or tap me whenever I'm online. My global chat handle is **@Shenanigunner**.

0.6 Acknowledgements

Only the general presentation of this guide, along with much direct verification of the commands, is solely mine. All of the information came from other sources – mainly, the game itself, its user manual, and the Prima game guide.

The original list of slash commands was provided by Xocyll, in the Usenet forum **alt.games.coh**, copied from the Binds forum on the official CoH web site. Xocyll has also posted a number of discoveries of his own on the Usenet group, which are included here, and has provided continual feedback on the guide. Neil Cerruti provided some useful info and feedback as well for the I6/I7 material.

A lot of the basics came from other guides and helpful people in the forums. As nearly all of it traces back to information from the game developers, and because I didn't keep track of who told me what, all I can do is offer a general, generous and heartfelt thanks to everyone who helped increase my understanding of how this is done. I make no claim at all that I could have done it without all that help.

Thanks to Korbian on Titan Network for some very useful pointers to command and emote information.

0.7 City of Heroes vs. City of Villains

As far as I know, both games are identical in their use of binds and macros, and almost all commands are interchangeable between the games. There are a very few commands that are peculiar to one or the other

(mostly, to CoV alone) and a number of commands that have different names but appear to be synonyms – you can "lackey" someone in CoH if you wish, and "sidekick" them in CoV.

0.8 Things To Come – Future Plans

This guide will likely never be "complete" since there are always hidden or unknown commands or tweaks, and changes with every Issue and running fix by the Developers. Some of the things in the ongoing agenda include:

- Thorough testing of commands I was not able to test – mostly, those to do with groups, supergroups and the global chat feature. Anyone who does a lot of teaming or SGing, and is messing with the new global chat stuff, is invited to test and correct the commands listed here, and pass along nifty things they find.
- Adding more "cool bind" info, now included in Appendix W.
- Adding more detail on groups of slash commands and how to put them to good use.
- More details, corrections and data for things like variable and window names and hidden commands and command features. Send 'em along!
- A word about complex bind sets: I have intended to bring in much info about pet, healer, defender and controller binds, but each is a topic for its own guide. There are in particular some very good Mastermind pet bind guides out there. Eventually I'll collate, edit and present the material… but not today. In the meantime, check the HEROICA! site for collections of cool bind sets for these archetypes.
- I've done some of the above in this 2019 update. More to come if the efforts seem appreciated.

SECTION 1 – BASICS

1.1 Overview

I'm going to put all the special terminology in this section – so if you run into an unfamiliar or cryptic term in the later sections, it's either because you didn't read this one or because I slipped and forgot to include it. (Let me know, in that case.) I'm also going to put most of the general, basic information that applies to both keybinds and macros in here, with a few repeats of key items in other sections.

Read this section – it will help you get going much faster and with fewer problems than if you just jump to the how-to sections!

1.2 Terminology

Keybind – A string of game commands "bound" to a single key or mouse button, which will be executed when that key or button is pressed. Also called just a "**bind**."

Macro – A string of game commands assigned to a Powers tray button, which will be executed when that Power button is clicked or activated via a keypress.

Syntax – The precise rules by which a keybind or macro string is constructed. If a string is constructed wrong – has faulty syntax, that is – it probably won't work, or at least won't do what you want it to do.

Toggle – To turn a power on or off, whichever state it isn't in, with a single command. Most shields and buffs are toggle powers, which you activate with a click of the button and then deactivate with a click of the same button. There are ways to force toggles to the on and off states, no matter what state they are in to begin with.

Window – Any of the individual dialogs, menus, and separate windows that are part of the user interface.

1.3 Entering Keybinds & Macros

Keybinds and macros are entered from within the game, by typing strings into the chat window's entry line. The current chat channel selected does not matter; you're going to override the chat function and direct the command to the "console" (the game's command input window) by typing a foreslash (/) as the first character. You're not going to forget that slash. Trust me, after the first time you send a bind string out to the entire zone because you forgot the slash, you're not going to forget the slash.

There are some good rules for entering binds and macros. The first is to park your character in a safe place, so you won't have to deal with unexpected foes while you're tinkering. Inside a tram station or store is a good place. Face your character to the wall, a universal multiplayer game announcement that you're busy with some internal task and don't want to be interrupted. If you're going to be at it a while, you might type the command "`/hide`" into the console to start. This will make you invisible to everyone else in the chat and search windows, so they won't bother you. (Remember to "`/unhide`" when you're done!) Finally, select a safe chat channel, so that if you do screw up, no one will be privy to your bobble. Using the Team channel while you are not in a team is good – if you accidentally send chat message, either nothing will happen or you'll just get a warning that you're not on a team.

You can also enter keybinds by editing a text file and then loading it, but that's an advanced step we'll cover separately. For now, the easiest way to start entering binds and macros is directly, in the game.

I strongly recommend that you start with a clean, new set of default keybinds (by going to the Controls menu and resetting everything to Default), and then slowly entering your new binds and testing them. You should also save your keybinds to a local text file every time you are about to make a new set of changes, so that you can quickly reload a working set if you mess up something and need to re-Default things.

1.4 Basic Syntax

The basic syntax for a keybind, which is typed into the chat window's message-entry window, is:

```
/bind key command_string
```

This will "bind" the specified command string to the specified key. You can bind commands to almost all of the keys on the keyboard, with some limitations. Once a valid command string is successfully bound to a key, any prior assignment to that key is erased and pressing that key will execute the command string.

The basic syntax for a macro, typed in exactly the same way, is:

```
/macro macro_name command_string
```

This will "bind" the specified command string to a power-tray button with the identifying name specified. Macro names can be one to three letters or numbers, and some punctuation. (Actually, there is no limit to the length of a macro name, but only three characters will fit on a macro button.) Macros can, confusingly, be given identical names, which is not recommended. Once a valid command string is successfully bound to a macro button, activating that button will execute the command string.

The slash at the beginning of those commands is very important: if you don't include it, you'll simply send the string out to whatever chat channel you have selected, provoking much humor and wrath from whoever sees it. (Sending a bind string out into a zone-wide Broadcast is one of the top not-quite-a-newbie tricks. You are allowed to avoid it. See the suggested rules in 1.3.)

1.5 Variables

Binds and macros are a lot more useful if you can insert variables, such as player or foe names, your own name, level and archetype, etc. City of Heroes includes such variables, which may be inserted into any command string in place of fixed text. It is the dollar sign ($) first character that identifies the label as a variable, which is why you can't use a dollar sign in most macro and bind text strings.

`$archetype`	Your player's archetype – Blaster, Tanker, etc.
`$battlecry`	The string you've entered in your ID as your battle cry. Limited to 32 characters.
`$level`	Your player's level – 2, 10, 35, etc.
`$name`	Your player's name – Shenanigunner, Wolf Moon, etc.
`$origin`	Your player's origin – Natural, Magic, Science, etc.
`$target`	The name of your currently selected target, which can be a foe, another player, or an object.

It's been suggested that `$battlecry` could be used as a universal variable string, since (unlike the others) it can be set by the user. It has no effect on any aspect of gameplay otherwise.

1.6 Useful References

There are several useful references for creating binds and macros. Two are included here: Appendix A lists all the currently known slash commands, and Appendix E lists all the currently known emote codes. You'll likely wear out a few copies of both in your gaming time.

More current lists, and many tips and tricks, can be found on CoH-related web sites and in the official CoH forum devoted to binds. Look these resources up for help, ideas, and information I haven't included here.

Perhaps the most useful reference you can have is a copy of the complete default keybinds, which I haven't included here because it's bulky, but easy to get. And that's one of the key (heh, heh) secrets here: Very few keys in CoH are "hard coded" and unchangeable. Nearly all keyboard and mouse commands are "bound" in a changeable manner. You could erase or eliminate nearly every game command from the keyboard (not that that would be very useful, but it also means you can completely, totally rearrange and remap how the commands are used.) Out of the box, the game simply has a default set of binds that move your character, open and close windows, activate powers, etc. Looking at this default list can be very informative.

To get your very own copy of the default key binds, perform the following steps:

- In the game, go to **Menu | Options | Controls** and select "**Reset to Defaults**." This is recommended if you've done any inexpert tinkering with binds; otherwise, skip this step. If you do, the file you generate will include any changes you've made.
- In the chat window, type:

```
/bind_save_file c:\defaultbinds.txt
```

You can substitute any path and filename you like. Open the file and you'll find a complete list of the default binds and command strings. (When you get more experience, you can edit this file directly, making as many changes as you like and then load it into the game to make all the bind changes at once.)

It will be assumed that you have this file, printed out for reference, as a companion to this guide.

1.7 Editing Keybind Files

Once you start messing with binds, you'll probably want to move on to making wholesale edits rather than laboriously typing in strings in the game. It's pretty simple; you can even do it while you're in the game, subject to some cautionary notes.

First, save your current keybinds as just described above. In the chat window, type:

```
/bind_save_file c:\defaultbinds.txt
```

It's probably best to use the name of the character, so that each file you save and edit is distinct from the others.

Now switch to the Windows desktop and open this file in your favorite editor. Wait, before you do that, save a backup copy of the file, so you can load your "last good state" if you screw up the file.

Edit away. When you're ready to try the commands, switch back to the game and in the chat window, type:

```
/bind_load_file c:\defaultbinds.txt
```

Test away.

Two notes: You should park your character in a very safe place, like the inside of a store or tram terminal, before switching away to the editor. You don't want to come back and wake up dead. You might also want to `/hide` while you're working.

Also, you may find that switching in and out of the game messes up your mouse control. In this case, go to Control Panel, open your mouse applet, and be sure that "Disable acceleration in games" is unchecked. If it's unchecked, check it. One of those should keep the annoying problem of your mouse going to one-tenth control speed from happening.

1.8 Extending Keybind Files

One neat thing about the way the keybinds work is that you can selectively overwrite them in the game. That is, if you enter a new bind in the console, it is added to the set, or overwrites only that specific bind. To go further, you can load a keybind file with only selected entries, and those will become part of the total set, and only overwrite any specific existing binds.

This is useful when you want to, for example, load in a bunch of emote binds tailored to a specific alt. If you maintain a bindfile with emotes bound to the Shift-Fxx keys, you can edit it to suit and then load it to overwrite only those bind keys.

There are a few guidelines to do this effectively:

- If you're adding a keybind set to an existing bind set, check to make sure it won't overwrite any existing binds you want.
- The new binds will not become part of your standard keybind file for that alt automatically. If, for example, you want to add a Mastermind pet control bind set to an alt, you have to load the specialized binds, and then
    - SAVE the updated bind set to the alt's bindsave file (by default, in this guide, Alt-F5 and set to a unique filename (MyScrapper-SAVE.txt, for example).

- SAVE your prior bindload file for archival purposes, then copy the newly saved version to the load-file name (default, bound to Alt-F6 and loading MyScrapper-LOAD.txt, for example).

That will provide a reverse path should you want to undo the addition, or use a load file for a new alt without that specific of emote, combat, heal, pet or whatever binds included.

See Appendix W for an example of a serial extension/overwrite/rollover bind.

SECTION 2 – KEYBINDS

2.1 Keybind Overview

To recap things you should have read above:

- A keybind binds one or more slash commands to a single key. When that key is pressed, the command string will be executed.
- You enter keybinds by typing them into the chat entry window, prefaced by a foreslash (/), in the form:

```
/bind keyname command_string
```

- The command string should normally be enclosed in one set of double quotes, although they can be omitted for single-word commands.
- Any binds you enter will overwrite any existing bind on that key.
- You can erase a keybind, either one you've entered or a default one, and make the key "dead" in the game, by using the "nop" (no operation) keyword:

```
/bind keyname nop
```

- Finally, you can retrieve the current bind for any key using:

```
/showbind keyname
```

2.2 Key Names

Nearly every key on the keyboard can be used for binds, but, like magical spells, you have to know each key's "true name" – which might not be obvious. For example, to bind something to the equals key, you can't use "=" – it won't work. You have to use "`equals`" instead. Many keys have similarly odd, but sensible once you understand them, names.

The list of allowable key names can be found in Appendix D.

All, or nearly all bindable keys can be combined with the Shift, Alt and Ctrl keys to allow additional combinations. That is, `K`, `SHIFT+K`, `CTRL+K` and `ALT+K` represent four different binds. Also, while left-right shift key names such as LSHIFT and RCTRL are allowed, my experience is that all three in each group are synonyms. That is, `ALT`, `RALT` and `LALT` can all be used but always represent both Alt keys. It is possible some keyboard drivers might interpret them separately; you can only experiment to see.

The alphabetic keys are case-insensitive in bindfiles; binding to R and r is exactly the same. However, `Shift+[alphakey]` is a different bind.

2.3 Basic Command Usage & Command Modifiers

In some cases, all that needs to be done to use a slash command in a keybind is to type the name of the command:

```
/bind F "follow"
```

Note that the command string is in quotes; although you can sometimes get away without the quotes, you should make it a practice to always use them, even when the command is a single keyword, as here. This command, which mimics the default bind for the F key, will cause your character to follow the selected target. However, the following example:

```
/bind A "left"
```

won't do quite what you think (what the default bind for the A key does). Since hardware and operating system key repeats are disabled within City of Heroes (actually, they are discarded everywhere except in the chat text entry window), pressing A with this bind will cause your character to move the default amount in

a strafe-left manner. And stop. Since what you probably want is for the character to keep strafing left as long as you hold the key, you have to add a modifier:

```
/bind A "+left"
```

It's that + that makes the key repeat the action as long as it's held down.

Now suppose you want to toggle on a power or state – like autorun (R in the default key mapping). If you use

```
/bind R "autorun"
```

what you'll get is a status response: you'll see "autorun 0" in the chat window, since the above command is treated as an inquiry into the state of the autorun command. If you try:

```
/bind R "+autorun"
```

you'll get autorun as long as the key is held down... or the same as holding down the W key, not very useful. To make autorun toggle on and off the way the default is mapped, you have to use:

```
/bind R "++autorun"
```

...and there's the trick. **The ++ tells the game that it's a toggle command**: each press will toggle the state of that power on or off. If you were to be silly and use:

```
/bind Q "++turn_left"
```

what you would get is your character spinning in left circles when you pressed Q, until you pressed Q again to stop it. Silly, but again not very useful.

Commands that toggle can usually also accept a numeric toggle code. For instance:

```
/bind R "autorun 1"
```

would force autorun on, no matter how many times it was pressed. You could then bind another key:

```
/bind V "autorun 0"
```

to turn autorun off unambiguously. This isn't a very useful example, since toggling autorun on and off with one key is quite enough for most players, but there are many situations where you want a firm "on" command and a firm "off" command, with no possibility of, say, dropping your shields during a battle, or turning off Hover or Fly in a sticky situation.

We'll go into more detail about toggles later.

Note also in all these examples that there is no slash in the bind string, except at the beginning. A slash in put in front of a slash command only when it is being executed directly, by itself, from the console line (which is rare except for user interface commands). It's that first slash that tells the game that what follows is a console command of some sort.

2.4 Command Separators

The real power of binds and macros isn't in binding a single command to a key or macro button: it's in the ability to string multiple commands together in that bind. There are some limitations in how you can combine actions – mainly, you cannot easily combine two attack powers into one bind or macro – but generally any reasonable combination of actions can be made. If there is a limit to the length of a bind command string, it's long enough that it will rarely be a problem.

Here is perhaps the single most useful custom bind for melee types:

```
/bind G "target_enemy_near$$follow"
```

This extremely useful bind causes your character to target the nearest foe and follow (lock onto) them. By binding it to my G key, I have the option of tapping F to follow a selected foe (useful when I want to home in on a boss surrounded by minions who might be closer to me), or G to just pounce on the closest foe. In the middle of a fight, surrounded by foes, it is a huge timesaver (and occasionally a butt-saver) to be able to whack G and retarget the nearest foe, rather than the one the game selects (who might be out of point-blank range).

The trick here is the "$$" characters, which act as a separator between commands. If you were to simply type a list of commands separated by spaces, the console would be unable to parse the line and while it might do something, it's not likely to be what you wanted. So each command needs to be separated from the next by a "$$" pair, with **no** spaces around it.

You can string multiple commands together using the $$ separator, but there are often limitations on which commands will work in certain cases and sequences. In particular, if a power has an activation time, it will block all subsequent powers in a bind string. This seems to have changed in the post-Live servers; I have old bindfiles that would activate three shield powers with a single key, but now will only activate one. A weak workaround for this is to press the bind key more than once; each time it is pressed, the next unactivated power will be activated. More on this later. You will probably have to experiment with each new combination to find one that works the way you want it to. Here's a simple mod to the above bind that can be helpful in a team situation:

```
/bind G "target_enemy_near$$g I've got the $target!$$follow"
```

This bind will target the nearest enemy, announce in the Team channel "I've got the Bone Daddy!" (or whichever foe was targeted, by name), and then follow him. Since the chat text is only in the Team channel and simply won't show up when you're not teamed, it won't bother non-team players.

And... **ahem**. A word about that. It's an annoying newbie trick to put a chat message on your power activations; no one you're not teamed with cares that you've activated Fly, hurled a Zapp, turned on your Plasma Shield, etc. Most newbies who discover the joys of chat-binding do it.. once. And get howled out of the zone, most likely. Don't be a clueless jerk; don't bind chat messages to your powers except very selectively in the Team channel, when the message will be helpful – every single time! – to your mates.

A useful variant of this example bind, although it's hard to make it fully automatic, is:

```
/bind H "g I'm assisting $target!$$follow"
```

If you click on a teammate you wish to assist (for example, a tanker pounding on a boss) and then hit H, you will announce to your teammates, "I'm assisting Shenanigunner!" and follow that mate as he moves from target to target. There are some limitations on this bind, but it might be helpful to some players. The `/assist` command might be useful in a bind like this, too.

2.5 Toggles and Forced Toggles

One of the problems with keybinds is that most are, by default, a toggle – the bind will simply turn the power to whichever state it's not in. Sometimes, as with the autorun key, that's exactly what you want. Other times, you want an absolute, guaranteed "power on" or "power off," even if you hit the key by mistake.

Easy enough. There are several "power activation" commands that operate in different ways, and it's esy to select the one you want.

You can toggle a power by specifying its name (preferred) or which tray slot it resides in:

```
/bind P "powexec_slot 3"
```

```
/bind P "powexec_name Fire Shield"
```

Assuming Fire Shield was in slot 3 of the main tray, these binds would work exactly the same – pressing P would toggle Fire Shield on and off. (I can think of some uses for the slot-number method, but in general, you should stay with the power-name method.)

But if you want Fire Shield to go on, and on only, when you whack a specific key, so that you never inadvertently drop the shield during a battle, you would use:

```
/bind P "powexec_toggleon Fire Shield"
```

Which would always force Fire Shield on, even if it was already on. (That is, if the power is on, the command would have no effect.) You could turn the power off by clicking its tray button, or by adding a forced off bind:

```
/bind O "powexec_toggleoff Fire Shield"
```

If you want to bind one key to activate multiple powers, it is (now) essential to use `_toggleon` as the command, so that serial keypresses don't toggle powers on and off unpredictably.

Because few powers have activation delays when turned off, most bind strings combining a series of `_toggleoff` commands will work with a single key press.

SECTION 3: MACROS

3.1 Macro Overview

If you've read this far, macros are simple: they are exactly like keybinds in every way, except that they are bound to a Power tray button instead of a keyboard key. The only difference is that the basic syntax is:

```
/macro AST "g I'm assisting $target!$$follow"
```

which will create a button labeled AST in the first open power tray slot. Clicking this button, or activating it with an associated keypress, will be exactly the same as pressing H in the above keybind example.

Macros are the primary reason you have 90 power tray slots. Besides being able to create a couple of alternate power configurations, you can create any number of macro trays – one for soloing, one for team work, one to primarily control or defend, one for melee or ranged attack work, etc.

The only other useful thing to say about macros, except for what's already been said under the previous section, is that there is a second macro creation command, `/macroslot`. This command assigns the newly created macro to a specific slot instead of the first available one:

```
/macroslot 5 AST "g I'm assisting $target!$$follow"
```

...would put the new macro in slot 6 of the primary tray. (Yes, the numbering for this command is zero-based, so 0-9 correspond to slots 1-10.) I'm not sure this has any great usefulness, as you can create a macro and then drag it anywhere you like. But there you go.

Macros may be named with any combination of letters and numbers from one character to... many, I'm not sure what the limit is. However, more than three characters will not fit on the button, so you would be wise to keep your macro names to three characters or less.

It is possible that a macro called by a bound key can permit some action combinations that binds alone do not. Experiment if you run into bind limitations. (This applies to using pet emotes right now, for example.)

3.2 Macros Using Tray Rollover

An advanced trick that can be used for both binds and macros is to use "tray rollover" – swapping power trays to change the macro or bind action each time.

The basic process is this:

- Create two macros that do the desired pair of actions and end by swapping trays:

```
/macro A1 "emote drumdance$$gototray2"
/macro A2 "emote victory$$gototray1"
```

- Put the first macro in tray 1, slot 1. Put the second macro in tray 2, slot 1.
- Now create a bind that fires the power in slot 1 of the primary tray:

```
/bind Q "powexec_slot 1"
```

- Now, each time Q is pressed, it will execute the macro in slot 1 of the primary tray, then swap trays. The alt will alternately drumdance and victory-wave. (Obviously, you can use more elaborate and useful commands here.)

This can be taken to very complex and extreme levels, swapping among all available trays, but requires very careful management of tray contents and other power positioning. You don't want to swap in a primary tray that lacks a combat power or rearranges them, not just as you engage that Archvillain.

A more sophisticated version that avoids messing with the primary tray would use the following tray control commands:

```
/macro A1 "emote drumdance$$gototraystray 3 8"
/macro A2 "emote victory$$gototraystray 3 9"
```

And the bind would be changed to:

```
/bind Q "powexectray 3 1"
```

to fire the power in tray 3, slot 1 and then swap tray 3 between trays 8 and 9. This can be extended over all of the upper trays – 3 through 9 – and still leave the primary and secondary trays in place for normal combat and interaction.

Or, even primary trays can be swapped quickly for changing combat situations. All I can say is, test the hell out of your setup before you tackle that Archvillain. Review all of the tray and power-execute slash commands for further ideas.

For additional "rollover" functions, see Rollover Binds in Appendix W.

3.3 Macros Using Tray Icons

A very cool option for advanced macro users is to be able to use a power-tray icon for macros instead of the limited, gray+letters icon. Everything about using this command is the same as above – that is, actually creating macros is the same – but with one additional argument, you can assign any power-tray icon used in the game to your new macro.

The command is:

```
macro_image TEXTUREFILE NAME COMMAND_STRING
```

`NAME` is required, as for other macro commands, but will only show up on hover or when the Info panel is opened. Names longer than two or three characters can thus be used for clarity.

`COMMAND_STRING` is the same as for any bind or macro definition: all the commands you want execute when this macro is called.

`TEXTUREFILE` is where it gets interesting. This must be a string that points to an existing power tray icon bitmap or texture within the game's PIGG files, and is composed of `POWERSETNAME_POWERNAME`, where `POWERSETNAME` is the power set name – duh – all run together as one word, and `POWERNAME` is the power within that set, again all run together. The underscore is probably optional but should be used for clarity. The most important thing is that no spaces are allowed in this texture-name string.

All power sets and power names are listed in unpacked PIGG files. A complete listing has been extracted and uploaded to the website; it's as useful for figuring out naming anomalies as it is for finding any one power set or power name. You can guess many names with reasonable accuracy; some examples:

```
SuperStrength_Rage
DualPistols_SwapAmmo
Flight_Fly
```

You can use any powerset and power icon, regardless of your alt's power sets. Go grab the list for complete possibilities.

*Thanks to Yuro on the Titan forums and **many** thanks to Kala in the game!*

Appendix A: Slash Commands Reference

- Keywords in *italics* represent values to be specified.
- Elements in [brackets] are optional. If an element is not in brackets, it is required.
- Numbers in {braces} are required:
  - Numbers separated by vertical bars {0|1} represent the valid selections.
  - Numbers separated by a dash {1-4} or {0.1-2.0} indicate the range of acceptable values.
  - Some commands that require a numeric value will return the current state if entered without a number; others will return an error message.
- Commands that use an underscore (_) to separate words can also be entered without the underscore, for example, `/window_hide` and `/windowhide` are equivalent. The underscore versions are generally used here. All commands are also case-insensitive; UPPERCASE and CamelCase words are only for convenience.

I have listed all known synonyms and, in this update, combined all of them into single listings.

A.1 Slash Command Listing

This has been a massive update and rewrite of this section, combining many formerly separate listings and adding all known new ones. None of the slash commands have been tested on the I14+/Post-Live servers (yet). Many commands may have changed; many older commands may be obsolete. All testing, verification and info welcome from the community!

Much of the global chat and Supergroup stuff remains untested from the last, long-ago update.

Anything highlighted in blue is something I have not yet verified or which I have found to be buggy – so use it cautiously and be sure to tell me anything useful you find out about it.

Anything highlighted in green is obsolete or reported to be so.

Anything highlighted in red is new for 2019, I14+/Post-Live (and also may be incomplete or buggy).

Slash Command	Description
/ac *message_string* **/arena *message_string***	Send message on the Arena chat channel.
/ah **/auctionhouse** **/blackmarket** **/wentworths**	Open Auction (Wentworth's/Black Market) window. *You no longer have to go to a specific place for this.*
/afk *message_string*	Marks the player as Away From Keyboard. If no string is specified, a little balloon with "AFK" in it appears over your character's head. Otherwise, the string is displayed there. Note that an auto power like Hasten can interrupt an AFK status. Note also that this command is how to put a text bubble up while you're typing a chat message… see Appendix W for details.
/ai *string* **/arenainvite *name***	Invite player to join arena event.

Slash Command	Description
/altinvite *name*	Invite your alt character by name to your current Supergroup. (Note: you must have invite privileges for this to work.)
/alttray {1-9}	Activate the power in the specified slot of the current secondary tray.
/alt2tray {1-9}	Activate the power in the specified slot of the current tertiary tray.
/alttraysticky	Toggle the secondary trays in and out of visibility. This command cycles between the base tray, base+secondary tray, base+secondary+tertiary tray, and base tray again.
/architect	Activate the mission search menu.
/assist	Set your current target to the selected ally's target.
/assist_name *name*	Set your current target to the named ally's target.
/autoperf {0|1}	Automatically change world detail for performance. (Function unclear.)
/autoreply	Start a chat reply including the character of the bind key used to summon it. Works only for keybinds and not as a slash command.
/autorun {0|1}	Toggle autorun on and off. Usually bound to the R key with ++
/b *message_string* **/broadcast *message_string*** **/y *message_string*** **/yell *message_string***	Send message to entire zone.
/backward	Move backwards. Usually bound to the S key with +
/beginchat *message_string*	Starts chat-entry mode with given string. See also `startchat`.
/bind *key commandstring*	Binds a key to a command string. See the rest of this guide for details.
/bind_load	Reads a list of keybinds from `keybinds.txt` in the default CoH directory.
/bind_load_file *filespec*	Reads a list of keybinds from a specified file location and name. As of Issue 12 or so, this command started echoing the file load to the status window.
/bind_load_file_silent *filespec*	Reads a list of keybinds from a specified file location and name. Functions like the old command, without an echo to the screen. Probably best for loading rolling bind sets.
/bind_save	Saves all keybinds to `keybinds.txt` in the default CoH directory.

Slash Command	Description
/bind_save_file *filespec*	Saves all keybinds to specified file location and name. As of Issue 12 or so, this command started echoing the file load to the status window.
/bind_save_file_silent *filespec*	Saves all keybinds to specified file location and name. Functions like the old command, without an echo to the screen.
/bloomscale {2|4}	Sets bloom blur size. Valid values 2 or 4 only.
/bloomweight *n*	Sets bloom scale. Valid values 0.0 – 2.0.
/buffs {0|1}	Toggle display of member buffs in the team list.
/build_save	Save current character build to BUILD.TXT file.
/build_save_file *filespec*	Save current character build to file designated by *filespec*.
/bug *subject_string*	Report a bug to the developers. Specify a concise subject; you will be given a window to enter additional text. System not used in Post-Live hosting.
/buy_coh	Opens the buy City of Heroes window. Note that this announces you are playing on a trial copy of the game even when your copy is registered. Obviously obsolete.
/c *message_string* **/coalition *message_string***	Send message to the coalition chat channel. You must be a member of a supergroup that is in a coalition with another group for this function to work.
/camdist {0-120}	Sets the distance in feet that the third person camera pulls back behind the player. 0 equals first-person view; the upper limit was changed from very large (in I5 and previous) and 65 (in I6) to about 120 in I7. Larger values will not take the view past 120. Note that the mousewheel adjustment is limited to 80 feet, and if you set a larger distance with this command any touch of the mousewheel will zoom to an 80 foot view.
/camdistadjust	Adjusts the camera distance relative to the current camera distance. Reads mousewheel for input and allows a range of 0-80 feet. *Probably not useful in console commands, as it appears to be permanently bound to the mouse wheel.*
/camreset	Resets the camera to a few feet behind the player, looking forward. Bound to the PAGEDOWN key by default.
/camrotate	Camrotate (bound to PAGEUP by default) allows controlled camera rotation around the player. The bound key must be pressed while the view is rotated with the mouse. This command should be bound to a suitable key, and not invoked through the console.
/camturn	Turns the camera to match player facing direction. Similar to `camreset` except that camera distance is not reset. See also `playerturn`.

Slash Command	Description
/canlook {0|1}	Toggles "mouselook," which permits the character to look around using the mouse instead of moving the in-game pointer.
/costume_change {0-9} **/cc {0-9}**	Change costume instantly. The upper number depends on the number of active costume slots. In the Live era, there were up to four, three of which had to be earned by in-game actions. This has been expanded to six free slots plus four earned ones. Note that this is another of the anomalous zero-based commands! See also `cc_emote` for a more elaborate option.
/cc_emote {0-9} ***emotestring*** **/cce {0-9}** ***emotestring*** Complex to use. See Appendix E.2 for the list of emote names and usage.	Combines an emote and a costume change. The upper number depends on the number of active costume slots. In the Live era, there were up to four, three of which had to be earned by in-game actions. This has been expanded to six free slots plus four earned ones. Note that this is another of the anomalous zero-based commands! The string is one of a dozen or so special costume-change emotes. Only these emotes can be used with this command, and they cannot be used as regular emotes. See `/costume_change` for a simpler option.
/cgshaderpath ***pathspec***	Set parent directory for /shaders/cgfx. If relative, path assumes .EXE file directory as root.
/chan_create ***channel***	Create a new chat channel.
/chan_desc ***channel string***	Set chat channel's description to ***`string`***.
/chan_invite ***channel***	Invite player or chat handle to a chat channel.
/chan_invite_deny ***channel name_string***	Deny/refuse chat channel invite for named player on named channel.
/chan_invite_gf ***channel***	Invite your entire global friends list to a chat channel.
/chan_invite_sg ***channel rank***	Invite your entire supergroup to a chat channel. Only leaders may use this command. You can invite members by rank: 0 – Invite all supergroup members. 1 – Invite captains and leaders only. 2 – Invite leaders only. See also `ginvite_sg`
/chan_invite_team ***channel***	Invite your entire team to a chat channel.
/chan_join ***channel***	Join an existing chat channel.
/chan_leave ***channel***	Leave a chat channel.
/chan_members ***channel***	List all members of channel.

Slash Command	Description
/chan_mode *channel options*	Changes default access rights for new user who joins the channel. **Valid Options**: **-join** kicks user from channel **+send/-send** gives/removes user ability to send messages to channel **+operator/-operator** gives/removes operator status from another user in the channel
/chan_motd *channel string*	Set the channel's Message Of The Day, which is sent to everyone that joins the channel.
/chan_send *channel string*	Send message to chat channel. You must be in the channel and have Send privileges. (Synonym: send)
/chan_timeout *channel duration*	Allows channel moderators to set number of days before inactive users are removed from the channel. A valid channel name for which the user is a moderator is required. The number of days can be set from 0 (no removal limit) to 30 days.
/chan_user_mode *channel name options*	Sets user permissions for specified user on channel. You must have operator status to set permissions. **Valid Options**: **-join** kicks user from channel **+send/-send** gives/removes user ability to send messages to channel **+operator/-operator** gives/removes operator status fromanother user in the channel
/change_handle *name*	Change your global user name, if allowed. There are limits on how often a global handle can be changed (e.g., it may be a one-time change for some users), so use this with caution.
/chat	Toggles the chat window. (Synonyms: `toggle chat`, `window_toggle chat`)
/chat_beta {0|1}	Permit participation in the Chat Server Beta Testing. (Way, way obsolete.)
/chat_cycle	Cycles through the default chat channels.
See Appendix G for complete information on using chat save and load functions.	
/chat_load	Reads a saved chat configuration (tabs, channels, names) from the CHAT.TXT file in the default installation folder.
/chat_load_file *filename*	Reads a saved chat configuration (tabs, channels, names) from the specified file name in the default installation folder, or, from the file on another path if it is specified.
/chatoptions {0-4}	Toggles the option menu for the specified chat window.

Slash Command	Description
/chat_save	Saves the current chat configuration (tabs, channels, names) to the CHAT.TXT file in the default installation folder.
/chat_save_file *filename*	Saves the current chat configuration (tabs, channels, names) to the specified file name in the default installation folder, or, to the file on another path if it is specified.
/chat_set *channel*	Sets the channel to the given string. Works only for global channels, not defaults.
/clearAttributeView	Clear the attribute target. Function uncertain.
/clearchat	Clears all chat buffers – equivalent to executing "Clear History" in each chat tab.
/clear_tray	Clear all power trays (excludes macros).
/clear_petnames	Clears all names of all your named pets.
/clearRewardChoice	Choose "no reward" in the current reward choice list.
/clicktomove {0|1} **/ctm {0|1}**	Enable and disable click-to-move. When enabled, clicking on any non-clickable point with create a pretty crystalline cursor, and your character will move to it. Maximum move range is about 60 yards. Useful for zooming around missions and such. See also: `ctminvert`, `ctmtoggle`
/cmdlist	Displays all console commands available in the system chat window. (Useful for finding updates and changes to this list – turn on chat logging first to save to a text file, or use copychat to copy all text to the clipboard.)
/coalition_cancel	Cancel coalition with selected supergroup.
/coalition_invite *player_name* **/ci *player_name***	Invites the named player to join a coalition. The player must be the leader of a supergroup for the function to work. (Synonym: `ci`)
/coalition_mintalkrank	Set the minimum rank of a supergroup who your supergroup can hear. (Values unknown.)
/coalition_nosend	Stop your supergroup from sending coalition chat to an ally supergroup.
/coalition_sg_mintalkrank	Set the minimum rank of a your supergroup who can talk on the coalition chat. (Values unknown.)
/comment *string*	Sets your group-search string to *`string`*. See also `getcomment`. Not functional?

Slash Command	Description
/compatible_cursors {0|1}	Shows the **status** of selection of standard Windows cursors instead of graphical cursors. (The Windows cursors are not as flexible and don't change color but may work better on some systems.) This command cannot be used to set the option, which must be set on the command line at game startup.
/contextmenu *menu_num*	Activate a context menu slot. (Function unclear.)
/copychat *tab_name*	Copy the entire chat history from specified chat tab into the clipboard. Useful for saving extended game info passed on by other players, or abuse.
/copydebuginfo	Gathers debug info, prints it and copies it into the clipboard. (Functionality unclear.)
/cov *int-arg*	Function unknown. Probably obsolete.
/ctm_invert {1|0}	Functionally identical to `clicktomove/ctm` except that the enable value is reversed (0=enabled).
/ctm_toggle	Toggles click-to-move status. No argument allowed. **Known bug**: Displayed enable/disable message can get inverted if used with other CTM change commands. Also seems to hard-override other CTM settings.
/cursorcache {0|1}	Enable cursor cache for smoother cursor changes.
/custom_window	Create a custom window. (Complex to use; requires writing window definition file.)
/custom_window_toggle	Open or close a custom window.
/demorecord *filename*	Begin recording a demo with specified filename.
/demorecord_auto	Begin recording a demo with generated filename.
/demostop	Stop demo record/play.
/demote *name*	Demote supergroup member one rank.
/dialog_answer *string*	Answer dialog with button matching provided text. (Function unclear; assume it is for use with dialogs that have other than Yes/No options.)
/dialog_no	Answer OK, No, or Cancel to current dialog.
/dialog_yes	Answer OK, Yes, or Accept to current dialog.
/disable2D {0|1}	Disables 2D sprite drawing. (Main, and perhaps only effect seems to be to turn all UI elements on and off.)
/dofweight {0.0 – 2.0}	Sets DOF scale. See also `usedof`.
/down	Move down (if flying). Bound to the X key by default.

Slash Command	Description
/e3screenshot {0 - ?}	Enables "Special E3 2004 screenshot mode." (Values and function unknown. Probably obsolete.)
/e *emotestring* **/em *emotestring*** **/emote *emotestring*** **/me *emotestring***	Causes player to display an emote animation or emote string. **Emote codes can be found in Appendix E.** Any string that does not match a valid emote code will be displayed in a visible thought bubble for a few seconds.
/emaildelete *message_number*	Delete message [message number].
/emailheaders	Request email headers. (Function unclear.)
/emailread *message_number*	Request message [message number].
/emailsend *name subject message_body [influence]* **/emailsendattachment *name subject influence attachment location message_body*** Thanks to Hopewarden & "Mikhail" for details on this one!	Send email message to another player, with or without attachments, which can include Influence, Inspirations, Enhancements, Salvage and Recipes. The first command only sends email, despite some help notes to indicate that Influence can be added to the transmission. (The influence keyword there is probably bogus.) It is recommended that you just use the Email UI for this. The second command can be used to send just a message, a message with Influence, or, with care, almost any item in player inventory. • `name` is a valid player name, preferably @global • `subject` is the message subject line – use quotes • `influence` is a numeric amount of Inf to send – **use caution, it can't be retrieved** • `attachment` is an attachment type, of which around 17 are possibly defined: 2 – Inspirations 10 – Enhancements (does not work) 11 – Salvage 12 – Recipe • `location` is the item's location in the corresponding inventory window, counting from 1. *Practice sending to yourself before you send an ultra-rare item to some random stranger!* • `message_body` is the body of the message, and will include anything in or after that position, in or out of quotes.
/estrange *name* **/unfriend *name***	Remove player from friend list.
/f *message_string*	Talk to Friends channel.
/face	Turn player to face selected target.
/findmember	Search for player. Appears identical to `search`.

Slash Command	Description
/first {0|1}	Toggles between first and third person camera. (Inverse of `third`.)
/fl **/friendlist**	Display friend list in chat window.
/follow	Toggle follow mode. Very important for targeting.
/forward	Move forward.
/forward_mouse	Move forward; enable autorun after 2 seconds.
/friend *name*	Add player to friend list.
/fsaa {0|2|4}	Sets the amount of full screen antialiasing. Other values can be set but their impact is unclear. *Note: FSAA has more impact on framerate than nearly any other graphics setting! Even with modern video cards, high FSAA rates can bog down framerate enormously—technical implementation in the engine may be poor/outdated.*
/fullrelight	Disable cap on number of relit vertices per frame. (Function unclear.)
/fullscreen {0|1}	Effect not entirely clear. Sets video mode to fullscreen. If set to 0, the game will start in windowed mode next time; when set to 1, game will start in fullscreen mode. Cannot be changed during gameplay; you have to make this setting and then restart to change the view. See also `maximize`.
/g message_string **/group *message_string*** **/team *message_string***	Send message to group channel.
/gamereturn **/windowcloseextra**	Reset UI by leaving fullscreen mode, closing dialogs and closing all secondary (nonessential) windows.
/getarenastats	Get your arena combat statistics.
/getallarenastats	Get your arena combat statistics, more comprehensive display.
/getratedarenastats	Get your arena combat statistics for rated matches.
/getcomment	Get your group-search string. (See also `comment`.) Function not clear.
/getglobalname *charname*	Get player's global name from character name. Using command without name returns an "unknown command" error.)
/getglobalsilent *charname*	Get player's global name from character name without reporting results to chat window. Appears to work but if no display is generated, not sure of purpose…

Slash Command	Description
/getlocalinvite *globalname* **/getlocalleagueinvite *globalname***	Invite current character name from global player name. (Assuming both commands are identical, here.)
/getlocalname *globalname*	Get currently active character name from global player name.
/getpos **/loc**	Displays current XY coordinates and altitude – in the System channel. A useful bind for mapping, and evaluating jump height and teleport increments.
/gfriend *name*	Add a player to your global friends list.
/gfriends	Display all members of your global friends list.
/gfriend_player *name*	Add player to global friends list via their player name. (Functionality uncertain.)
/gignore *name*	Ignore user on global chat.
/gignoring	Display list of ignored users on global chat.
/ginvite *player_name*	Invites the named player to join a global chat.
/ginvite_sg *channel rank*	Invite your entire supergroup to a global chat channel. Only leaders may use this command. You can invite members by rank: 0 – Invite all supergroup members. 1 – Invite captains and leaders only. 2 – Invite leaders only. (See also `chan_invite_sg`.)
/gmotd	Recall the global message of the day, as displayed at first login.
/goto_tray {1-9}	Set the main tray to the specified tray number.
/goto_tray_alt {1-9}	Set the secondary tray to the specified tray number.
/goto_tray_alt2 {1-9}	Set the tertiary tray to the specified tray number.
/goto_trays_tray {1-3} {1-9}	Set the specified tray (1-3) to the desired tray number (0-9).
/graphfps {1|2|4|8}	Graph current framerate. 1 – Large FPS graph 2 – GPU Info? 3 – FPS and GPU info 4 – SLI info?
/guide **/helpchat** **/h** **/hc**	Selects the global Help channel.
/gunfriend *name*	Remove a player from your global friends list. (Via global name?)
/gunfriend_player *name*	Remove player from global friends list. (Functionality unknown.)

Slash Command	Description
/gunignore *name*	Un-ignore user on global chat.
/help **/helpwindow**	Open Help window.
/hideall **/unhideall**	Hide or unhide your name from other users in all of the "who's on" lists. Complete player invisibility.
/hide **/unhide** **/ghide** **/gunhide** **/hidefriends** **/hidegchannels** **/hidegfriends** **/hideinvite** **/hidesearch** **/hidesg** **/hidetell** **/ghide** **/unhide…**	These commands used to have individual effect, but no longer do. The entire player-hide system is in a questionable state. It may be best not to use any slash commands for hiding purposes, but bring up the Hide menu instead. Each of the first four commands brings up the same Hide window with 7 click on/off buttons for each area of hiding your current login from other players. This is the recommended method (for now) and only the basic `hide` command needs to be remembered. The other `hide`… commands appear to hide the user on the specified channel (Server Friends, Global Friends, Global Channels, Invites, Email, Searches, your Supergroup and Tells), but also unhide them on all others. The parallel `unhide`… commands work similarly. It is recommended that you use `/hide` to bring up the Hide window for Hide settings, unless you use `hideall/unhideall` as a global invisibility command. Do not confuse any player-hide command with the anomalous `windowhide`, although this may have been a midstream change in the game.
/hideprimarychat	Toggle primary chat window text messages. You can reduce the chat window to just the chat entry line with this command. You should be sure another chat window holds whatever channels you are participating in.
/i *name* **/invite *name***	Invite player to join team.
/ignore *name*	Ignore user.
/ignorespammer *name*	Ignore user as spammer. Automatically reports name as spammer; not sure this is useful in the Post_Live era.
/ignorelist	Displays a list of ignored users.
/info	Displays the information on a selected item, same as right-clicking and selecting Info from the pop-up menu.
/info_self	Displays your own information, the same as others see when they "info" you.

Slash Command	Description
/info_self_tab {0-6}	Displays your own information, the same as others see when they "info" you, opening the window to the named info tab. Tabs are referenced by number: 0, 1 – Description. 2 – Powers. 3 – Badges. 4 – Alighnment. 5 – PvP. 6 – Arena.
/info_tab {0-6}	Displays the information on a selected item, same as right-clicking and selecting Info from the pop-up menu, opening the window to the named info tab. Tabs are referenced by number: 0, 1 – Description. 2 – Powers. 3 – Badges. 4 – Alighnment. 5 – PvP. 6 – Arena.
/insp_combine *inspname1 inspname2*	Combines three of the first named Inspirations to create one of the second name and next power level. You must put quotes around multi-word Inspiration names, e.g. "break free" or "catch a breath".
/insp_delete *inspname*	Delete named Inspiration. Might be a useful bind in combat to clear out, for example, stamina Insps on a high-stamina build.
/inspexec_name *inspname*	Activate an Inspiration by name.
/inspexec_pet_name *inspname petname*	Activate a named Inspiration on a pet by pet name.
/inspexec_pet_target *inspname*	Activate a named Inspiration on the targeted pet.
/inspexec_slot *column* **/inspirationslot *column***	Activate an inspiration slot in the first row of the specified column.
/inspexec_tray *row column*	Activate an inspiration slot in the specified row and column.
/keybind_reset **/unbindall**	Resets all keybinds to default. See also `unbind`. **Use with caution!**
/k *name* **/kick *name***	Kick player from team.
/kiosk	If you're within range of an Info kiosk, this will pop up the "home" info page as if you'd clicked on it. (Obsolete.)
/l message_string **/local *message_string***	Send message to anyone in your immediate area, about a 250 foot radius.

Slash Command	Description
/leaveteam	Quit your current team.
/left	Strafe left. Bound to A key by default.
/levelingpact *playername*	Invite named player to join a Leveling Pact.
/lfg [0|1]	Toggle LFG (looking for group) status. See also `lfgset`. (Functionality uncertain.)
/lfg_event_response	Accept or reject invitation to join event. (0 and 1 assumed values; functionality uncertain.)
/lfg_remove_from_queue	Remove self or team from LFG queue. (Functionality uncertain.)
/lfg_request_event_list	Get LFG system event list. (Functionality uncertain.)
/lfgset {0|1}	Set LFG (looking for group) status. See also `lfg`. (Functionality uncertain.)
/lightmaplodscale	Set lightmap LOD scale. (Obsolete.)
/link_channel *channelname*	Activates context menu for named channel.
/link_info	Provides info window for named channel. (Functionality uncertain.)
/link_interact *playername*	Activates context menu for named player interactions.
/link_interact_global *arg arg*	Activates context menu for global player name. (Functionality uncertain.)
/localtime	Displays (your computer's) local time.
/lodbias {0.0-2.0}	Multiplier for LOD (Loss of Detail) distances for entities. The default is 1.0. Setting this to 0.5 will cause detail switches to happen at half the distance; 2.0 will cause switches to happen at twice the default distance. Lower values improve performance; higher ones increase your character's vision. Appears to be obsolete; see also `DOFweight`.
/logchat	Toggle chat logging. Chat logs appear by date in the \logs folder under the main CoH folder. Chat can also be extracted using `copychat`.

Slash Command	Description
/lookdown **/lookup**	Moves look angle down or up. Normally, this command and `lookup` are used with the + and ++ modifiers to permit controlled up and down looking. Works in conjunction with `lookup` to control free look capability. (If both `lookdown` and `lookup` are set to 1, or both are set to 0, you will have free look capability. Setting one or the other to 1 will force the view to a straight up or straight down view, persistent against changes. There must be some use for these settings, but I can't figure it out. I think it's a slightly buggy side effect.)
/lp *messagestring*	Sends message to Leveling Pact channel.
/macro *macroname command_string*	Add a macro to first empty slot. See the rest of this guide for details.
/macro_image *texturefile macroname command_string*	Adds a macro to the first empty slot and uses the specified texture file for the icon. The texture file must be an existing power-tray icon from within the game's PIGG files. There are hundreds, if not thousands, callable by using a combination of powerset and powernames. See Section 3.3 for details.
/macroslot *slotnum macroname command_string* Thanks to hooliganj for details on this one!	Add a macro to the specified slot of any tray. This command will overwrite any command or macro already in that slot. The value for `slotnum` is complicated in that it can be 0-89, with 0 being the first slot in the primary tray, 10 being the first slot in the second tray, and so on up to 89 being the last slot in tray 9. Calculate carefully. Note that the numbering here is zero-based.
/makeleader *name* **/ml *name***	Designated new team leader. Can be used only by current leader.
/mailview	Sets view to use on the Mail tab. (Arguments and functionality unknown.)
/manage	Open the Enhancement management window. (This appears to be the only menu/window name that does not work in the other window-control commands.)
/map	Toggles the map window. (Synonym: `toggle map`, `window_toggle map`)
/maxcolortrackerverts	Maximum number of world object vertices to relight per frame. (Functionality uncertain.)

Slash Command	Description
/maxfps {1-?}	Set the maximum FPS (frames per second) rate. This seems to be capped at 30 but now appears to accept any value. Normally you will want this maximized (at 30), but it may be useful in some circumstances to enter a slower rate. Very slow rates (under 5) are NOT recommended but can be fun to play with in safe areas.
/maximize {0, 1}	Effect is unclear. Compare with `fullscreen`.
/maxinactivefps {1-?}	Set the maximum FPS (frames per second) rate while the game is not in the foreground. Reducing this value will lessen the impact on other programs brought forward during gameplay. The rate should be high enough for you to be able to keep track of what is happening – 5-8 fps is recommended.
/maxmenufps {1-?}	Set the maximum FPS (frames per second) rate while the game is in a full-screen menu.
/maxrtframes	Set how many frames forward to allow buffering. (Arguments and functionality unclear.)
/menu	Opens the main menu. (Synonyms: toggle menu; window_toggle menu)
/missionmake	Activate the My Arcs menu of Mission Search.
/missionsearch	Open the Mission Search window.
/mmentry	Choose between making and starting a mission maker story arc.
/monitorattribute *string* **/stopmonitorattribute *string***	Adds a display line to the Attribute Monitor. The first command adds, or toggles any specified line; the second command removes the specified line. See Appendix A.3 for a list of known arguments and more detailed usage information.
/mouse_invert {0|1}	When active, inverts the mouse Y axis (pitch) for mouselook.
/mouse_look *num*	Command key for mouselook. (Function unclear.)
/mouse_speed {0-6}	Mouse speed scale factor for mouse look. 1.0 is default; values over 3 make control erratic in most cases.
/mousepitchmode {0|1|2}	Set mouse pitch mode: 0 – Free look. 1 – Return to center after release. 2 – Always centered. (Confusing, not recommended.) (Obsolete.)
/myhandle	Display your global chat handle.
/mypurchases	Show list of purchases you have access to. (Functionality uncertain.)
/nameCaptain *name_string*	Renames the 'Captain' supergroup rank.
/nameCommander *name_string*	Renames the 'Commander' supergroup rank.

Slash Command	Description
/nameEnforcer *name_string*	Renames the 'Enforcer' supergroup rank.
/nameFlunky *name_string*	Renames the 'Flunky' supergroup rank.
/nameLeader *name_string*	Renames the 'Leader' supergroup rank.
/nameLieutenant *name_string*	Renames the 'Lieutenant' supergroup rank.
/nameMember *name_string*	Renames the 'Member' supergroup rank.
/nameOverlord *name_string*	Renames the 'Overlord' supergroup rank.
/nameRingleader *name_string*	Renames the 'Ringleader' supergroup rank.
/nameTaskmaster *name_string*	Renames the 'Taskmaster' supergroup rank.
/nav	Toggles the navigation window. (Synonyms: `toggle nav; window_toggle nav`)
/neterrorcorrection {0|1|2}	Adjusts network error correction limits. (Details unknown.)
/netgraph {0|1|2}	Displays network connection information. Option 1 is low-profile, Option 2 is higher-profile; not sure of other differences.
/newspaper	Open mission newspaper. (Obsolete.)
/next_tray	Go to next primary tray.
/next_tray_alt	Go to next secondary tray.
/next_tray_alt2	Go to next tertiary tray.
/next_trays_tray {1-3}	Go to the specified tray's next tray.
/nojumprepeat {0|1}	Disable jump auto-repeat. This means you'll jump only once, no matter how long the key is held down; another jump will require another keypress. Can be useful for better control indoors or with really bouncy alts.
nop	Not really a command, but a null placeholder used to cancel a bind. If you enter `/bind x nop`, for example, any bind on X will be deleted. Useful for clearing out default binds you don't want.
/noparticles {0|1}	Turn off particle graphics for better performance.
/norenderthread	See also `renderthread`. Function unknown. Use not recommended.
/noreport {0|1}	Do not default to error reporting window on crash. This may suppress the Windows error reporting screen after a crash; confirmation and other purpose unknown.
/nosunflare {0|1}	Disables sun flare (for performance debugging). Removes and restores flare/glare from sunlight (and moonlight?)
/noversioncheck	Disable mapserver version check. Probably very useful in this era of rogue servers.
See Appendix G for complete information on using option set, save and load functions.	

Slash Command	Description
/option_list	Lists option names.
/option_load	Reads option configuration from the file options.txt in the default installation folder.
/option_load_file *filename*	Reads option configuration from the specified filename in the default installation folder, or, if specified, in a different location.
/option_save	Saves window configuration to the file options.txt in the default installation folder.
/option_save_file *filename*	Saves option configuration to the specified filename in the default installation folder, or, if specified, in a different location.
/option_set	Sets an option.
/option_toggle	Toggles an option.
/petcom *stance*	Set selected pet to specified action/stance. **For this group of commands, the valid pet stances are:** **aggressive** – attack any nearby foe without orders. **defensive** – respond to attack by any foe without orders. **passive** – do nothing without orders. **And the valid pet actions are:** **attack** – attack currently selected target. **dismiss** – dismisses pet gracefully. **follow** – follow me. **goto** – go to the selected spot. **stay** – stay at the selected spot.
/petcom_all *stance*	Set all pets to specified action/stance.
/petcom_name *petname stance*	Set named pet to specified action/stance.
/petcom_pow *powname stance*	Set the stance for all pets cast by the named power.
/petition *subjectstring*	Add user petition (stuck, cheated, etc.) to the database. This is more for immediate help from a game master than `/bug`. Give a GM time to get the petition and help you – it can take a few minutes or more. System not used in Post-Live hosting.
/pet_select *petnum*	Select pet by number, starting with 0.
/pet_select_name *petname*	Select named pet.
/petoptions	Displays pet window options menu only if the Pet window is displayed. Use `/show pet` to open the Pet window.
/petrename *petname*	Renames selected pet.
/petrename_name *oldname newname*	Renames named pet.

Slash Command	Description

/petsay *string*	Have selected pet say or emote *string*.
	This and the following three targeted commands use a very specific subset of string format to work correctly. Everything in string will be "spoken" by the chosen pet, including most control characters. To have a pet perform an emote, it must be enclosed in angle brackets: `<em Wave>` or `<emote Bow>`. To combine an emote and a chat-bubble string, just run them together: `<em Bow>At Your Service!` Multiple emotes (in brackets) and text can be strung together.
	There is also a protocol for synchronizing "smash" emotes by the player and "react" emotes by the pet, so that the pet will cower as the player attacks them. Briefly, it's
	`<em batreact>$$em batsmash`
	`<em slapreact>$$em smack`
	and other such emote pairs.
	NOTE: pet emotes *do not work in binds* due to a current bug in the way strings are processed. To get around this limitation, write pet emote commands to macros and call the macro with a bind. See section E.2 for details.

/petsay_all *string*	Have all pets say or emote *string*.
/petsay_name *petname string*	Have named pet say or emote *string*.
/petsay_pow *powname string*	Have all pets cast by specified power say or emote *string*.
/playernote *playername*	Opens note window for specified global player name.
/playernotelocal *playername*	Opens note window for specified current player name.
/playerturn	Turns player to match camera angle. Does not change camera distance. See also `camturn`.
/popmenu *menuname*	Pops up custom menu at the current mouse location. Very complex; see ParagonWiki for developer-level info: `https://paragonwiki.com/wiki/Popmenu_(Slash_Command)`
/powers	Toggles the power inventory window.
/powexec_abort	Cancels the auto-attack power and the queued power.
/powexec_altslot {1-10}	Executes the given power slot from the secondary tray.
/powexec_alt2slot {1-10}	Executes the given power slot from the tertiary tray.
/powexec_auto *power_name*	Sets the named power to automatically execute or activate each time it has recharged. Useful for 'booster' powers like Hasten. If no power is named, the current autoexec power assignment will be cancelled. Only a single power may be set to auto-exec at any one time. Appears to persist between sessions.
/powexec_location *target power_name*	Executes the named power as directed, without a reference click. Complex to use: see section A.3 for details.

Slash Command	Description
/powexec_name *power_name*	Executes a power with the given name.
/powexec_server_slot	Executes the specified power slot from the server-controlled tray. (Functionality and arguments unknown.)
/powexec_slot {1-10}	Executes the specified power slot from the current tray.
/powexec_toggleoff *power_name*	Toggles a given power off. If it's already off, does nothing.
/powexec_toggleon *power_name*	Toggles a given power on. If it's already on, does nothing.
/powexec_tray *slot tray*	Executes a power in the given slot and tray.
/powexec_unqueue	Cancels the queued power. Bound to the Z key by default.
/prev_tray	Go to previous primary tray.
/prev_tray_alt	Go to previous secondary tray.
/prev_tray_alt2	Go to previous tertiary tray.
/prev_trays_tray {1-3}	Go to the specified tray's previous tray.
/priorityboost	Set game process priority to Above Normal (from Normal) when running in the foreground. (Functionality uncertain.)
/profiler_record *filename*	Record client profiler information to specified filename. Purpose of this file and function unclear – appears to be for debugging and tech support use.
/profiler_stop	Stop recording client profiler information.
/promote *name*	Promote supergroup member one rank. See also `demote`.
/quickchat	Pops up the quickchat (emotes+chat bubble) menu.
/quit	Quits game to the desktop. **(10 second abort.)**
/quittocharacterselect	Quits game to the character selector. **(5 second abort.)**
/quittologin	Quits game to login screen. **(15 second abort.)**
/r *message_string* **/reply *message_string***	Reply to last *received* private message. This is differentiated from replying to the last *sent* private message, which can be replied to using the `tell_last` command.
/raid_invite	Invites selected player's supergroup to join an instant raid.
/release	Activate medicom unit for emergency medical transport. (Equivalent to clicking "Go to Hospital" button when defeated.)
/release_pets	Deactivate all current pets. (They fall dead, instead of leaving as with the menu "dismiss" command.)

Slash Command	Description
/reloadgfx	Reload all graphics textures. Useful when something has messed up your screen display. **Warning: scrambles display for at least a few seconds – do not use in combat.**
/renderscale {0.1-1.0}	Changes the scale at which the world is rendered, relative to your screen size. Permits you to keep your screen size the same as desktop, or sufficiently high for well-rendered UI elements, while lowering the effective resolution for performance. This command affects both X and Y scaling simultaneously; see also `renderscalex` and `renderscaley`. Not effective unless `userenderscale` is set to 1. (Setting this value to 0, or cycling `userenderscale` from 1 to 0, will reset renderscaling to the default of 1.0.
/renderscalefilter	Changes the method of filtering used in renderscaling. Value range and function unknown.
/renderscalex {0.1-1.0}	Changes only the X scaling of the world rendering. See `renderscale`.
/renderscaley {0.1-1.0}	Changes only the Y scaling of the world rendering. See `renderscale`.
/rendersize *xsize ysize*	Changes the size at which the world is rendered. Sizes are specified in x and y screen values and may be "normal" or odd values. Specifying non-multiple values will result in nonlinear x or y scaling.
/renderthread {0-?}	Function unknown. See also `norenderthread`.
/req *message_string* **/request *message_string*** **/sell *message_string*** **/auction *string***	Send a message to the Request channel.
/requestexitmission {1|*n*}	Leave mission map once completed. Equivalent to clicking "Mission Completed" text in Nav window. The "1" is required; "0" does nothing. Other values may have other effects – testing is required. Does not set "auto-exit" if called before end of mission.
/respec	Goes to the Respec screen if you have a "free respec" available. **Warning**: you should only use this command with your character in a safe place – you can be attacked while in this mode.
/right	Strafe right. Bound to the D key by default.
/s message_string **/say *message_string***	Sends the given text on the **current** chat channel.
/screen *x_dimension y_dimension*	Sets X and Y screen dimensions. Should be constrained to standard screen dimensions supported by your video card (640x480, 1024x768, 1280x1024, 1600x1200, etc.)

Slash Command	Description
/screenshot	Save a JPEG (.jpg) format screenshot in the \screenshots directory under the default CoH directory.
/screenshottga	Save a Targa (.tga) format screenshot in the \screenshots directory under the default CoH directory.
/screenshottitle *filename*	Save a JPEG (.jpg) format screenshot in the \screenshots directory under the default CoH directory, using the specified filename.
/screenshotui {0|1}	Enables or disables the user interface elements for screenshots. If set to 1, the UI will be visible in screenshots; if set to 0, the UI will not be included in screenshots.
/sea **/search**	Displays a searchable list of online player alts with their name, archetype, level, zone and looking for group status.
/see_everything {0|1}	Toggle hidden elements such as volume boxes, lights and spawn points while in the SuperGroup base editor.
/selectbuild {0-*n*}	Select the currently active build for your alt. The number of builds available varies with alt level and slots earned. There is a 60-second delay between build changes.
/send *channel message_string*	Send message to the named chat channel. You must be a member of the channel and have send privileges.
/servertime	Displays the current official (game server) time.
/set_title *badgename*	Set badge title. (Must be one you have, of course.) (Bug: clears currently selected badge title no matter what string is used.)
/sg *message_string* **/supergroup *message_string***	Send message to super group channel.
/sgcreate	Start a supergroup. (Obsolete.)
/sgenterpasscode	Open Supergroup base entry passcode dialog.
/sgi *name* **/sginvite *name***	Invite player to join supergroup.
/sgk *name* **/sgkick *name***	Kick player from supergroup.
/sgkickyes *name*	Kick player from supergroup, without confirmation.
/sgleave	Leave your current supergroup.
/sgmode	Toggle supergroup mode.
/sgmodeset {0|1}	Set supergroup mode.
/sgraidinvite	Invite selected player's supergroup to join raid.
/sgraidwindow *daybits hour*	Set your supergroup's raid window. (Values unknown.)
/sgsetcostume	Sets supergroup costume parameters.
/sgsetdemotetimeout	Sets supergroup demote timeout.

Slash Command	Description
/sgsetdescription *string*	Sets supergroup description.
/sgsetmotd *message_string*	Sets supergroup MOTD.
/sgsetmotto *message_string*	Sets supergroup motto.
/sgstats	Display supergroup info in chat window.
/shaderdetail {0|1|2}	Controls shader detail level.
/shadowvol {0|1}	Controls whether or not shadow volumes are drawn. Behaves very oddly when set to 1 on some systems.
/show *window_name*	Forces the given window to be shown. (Synonym: `window_show`) Has no opposite, although `hide` is sometimes incorrectly cited.
/showbind *keyname*	Returns current bind string for specified key.
/showfps {0-3}	Show current framerate and other information as a small boxed number on top right edge. 0 – off. 1 – show FPS. 2 – show FPS and camera POS/PYR 3 – show FPS and camera POS/PYR, large font
/shownewtray	Opens a tear-away Tray window. May be repeated to open multiple trays. As with the + button on the main tray, it will open trays beginning with the last one you had open.
/showpetnames	Lists names of all named pets.
/showtime {0|1}	Show the **in-game time of day** on screen. This is a 24-hour decimal clock (each 'hour' has 100 minutes) that counts from 00.00 to 24.00, with game noon at 12.00 and midnight at 24.00. The factor is apparently scalable by the system and currently sits at 48… meaning each game day is 30 real-world minutes long. Very useful when waiting for night to hunt certain foes! Note that asking any civilian NPCs with names that begin with E or F will also get you this in-game time.
/sidekick *name* **/ex** **/exemplar** **/lackey** **/lk** **/sk** **/rsk**	Invite player to be your sidekick. Many obsolete synonyms!
/sidekick_accept	Accept an invitation to be a sidekick.
/sidekick_decline	Decline an invitation to be a sidekick.

Slash Command	Description
/slashchat	Starts chat-entry mode and copies whatever key is pressed into the chat buffer. Used by default with "/" but could be used with other keys – to what purpose is not clear.
/sliclear	Clear each FBO to help SLI/CF (0 to disable). For SLI systems only; functionality uncertain.
/sliFBOs	Number of SLI/CF framebuffers to allocate (1 to disable). For SLI systems only; functionality uncertain.
/slilimit	Limit number of SLI/CF frames to allow in parallel (0 to disable limiter). For SLI systems only; functionality uncertain.
/speed_turn {1-359}	Set the number of degrees for each increment of rotate left/right. Used by `turn_left` and `turn_right`.
/ss {0|1}	Controls whether or not simple shadows are drawn.
/startchat	Starts chat-entry mode.
/stopinactivedisplay	Stops rendering when the game is not the foreground application.
/stopmonitorattribute	Removes attribute from Attribute Monitor.
/stuck	Tries to shift your character to the nearest unstuck position; for use when you get stuck between objects or in map flaws. (If it doesn't work, try sending a `/petition` and waiting a bit to see if a GameMaster will help you.)
/supporthardwarelights	Enable support for AlienFX/LightFX case lights. (May only function on next game start.)
/suppressCloseFx [0|1]	Hides all character effects when the camera is closer than the `SuppressCloseFxDist` setting. Useful when close camera viewpoint is obscured by powers effects, etc.
/suppressCloseFxDist *feet*	Within this camera distance, character effects will be suppressed. The practical limit is the maximum viewpoint camera distance, about 120 feet.
/sync **/synch**	Try to resynchronize character/client with game server. Use when character cannot be moved, becomes invisible to teammates, etc.
/tabglobalnext	Cycle forward through all chat tabs in all windows. Will open the corresponding chat window if necessary.
/tabglobalprev	Cycle backwards through all chat tabs in all windows. Will open the corresponding chat window if necessary.
/tabnext {0-4}	Cycle forward through all chat tabs in indicated chat window (0-4).
/tabprev {0-4}	Cycle backward through all chat tabs in indicated chat window (0-4).

Slash Command	Description
/tabselect *tabname*	Select the given chat tab. Will open the corresponding chat window if necessary.
/tabtoggle	Make the previously active chat tab the new active tab. Used to flip between two tabs.
/target	Toggles the target window. (Synonyms: `toggle target`, `window_toggle target`)
For more information on custom targeting, see Appendix T. There's a lot of cool stuff you can do with these commands, especially the nebulous 'base' targeting.	
/target_custom_...	Powerful customizable targeting comand. There are four variants, which conclude with the following suffixes:
...near	Closest target.
...far	Farthest target.
...next	Next target, in near to far order.
...prev	Next target, in far to near order.
Each of these commands can be directed to a specific class of targetable object by one of these keywords:	
enemy	Identical to `target_enemy`.
friend	Identical to `target_friend`.
defeated	Enemy, friend or NPC with zero hit points.
alive	Enemy, friend or NPC with nonzero hit points.
mypet	Any pet spawned by you.
notmypet	Any pet not spawned by you
base	Complex, but basically target all targetable objects including doors, glowies, civilians and hidden items. Both seem to work identically (and a bit erratically).
notbase	
teammate	Any teammate.
notteammate	Any non-teammate player.
/target_enemy_far	Targets the farthest visible enemy.
/target_enemy_near	Targets the nearest enemy.
/target_enemy_next **/toggle_enemy**	Cycles through visible targetable enemies in near to far order.
/target_enemy_prev **/toggle_enemy_prev**	Cycles through visible targetable enemies in far to near order.
/target_friend_far	Targets the farthest friend. A friend is any friendly player or pet, not just teammates.
/target_friend_near	Targets the nearest friend.
/target_friend_next	Cycles through visible targetable friends in near to far order.
/target_friend_prev	Cycles through visible targetable friends in far to near order.

Slash Command	Description
/target_name *string*	Target any entity whose name begins with *string*. Can be useful for finding specific civilians to query about time on server, etc. Find, Follow, Click.
/team_accept	Accepts an invitation to a team.
/team_decline	Declines an invitation to a team.
/team_kick_internal	Kicks a character without warning from team.
/team_quit_internal	Quits team without warning.
/team_select [1-8]	Select team member (by number in team list).
/team_task *int int int*	Select the team task. (Functionality uncertain.)
/p *name, message_string* **/private *name, message_string*** **/t *name, message_string*** **/tell *name, message_string*** **/whisper *name, message_string***	Send a message to only one player. (Grouped out of alpha order since `tell` is the most common synonym.)
/tell_last *message_string* **/tl *message_string***	Reply to the same person to whom you last *sent* a private message. This is different from replying to the last *received* private message using the `reply` command.
/texaniso {0|1|2|4|8|16}	Sets amount of anisotropic filtering. UI permits only those values shown, but other integer values can be entered. Effect of these interim values uncertain.
/texwordeditor *texname*	Edit the text layout for translatable textures. Exact function unknown. **Probably not something for users to mess with.**
/third {0|1}	Toggles between first and third person camera. (Inverse of `first`.)
/toggle *window_name*	Show a window if hidden, hide a window if shown. (Synonym: `window_toggle`.)
/trade *name*	Invite player to trade.
/trade_accept	Accepts an offer to trade. Not validated.
/trade_decline	Declines an offer to trade. Not validated.
/tray	Toggles the tray window. (Synonyms: `toggle tray`, `window_toggle tray`)
It is not clear exactly what these three commands do; function seems erratic and dependent on existing tray state.	
/traysticky {0-2} {0|1}	Sets the sticky-state of the specifed tray (or tray window). 0 is not sticky, any nonzero value is sticky.
/traystickyalt {0-2} {0|1}	Sets the sticky-state of the secondary (or specified?) tray. 0 is not sticky, any nonzero value is sticky.

Slash Command	Description
/traystickyalt2 {0-2} {0|1}	Sets the sticky-state of the tertiary (or specified?) tray. 0 is not sticky, any nonzero value is sticky.
/turnleft	Rotate left a fixed number of degrees (set by `speed_turn`).
/turnright	Rotate right a fixed number of degrees (set by `speed_turn`).
/uiskin	Function unknown.
/unbind *keyname*	Unbinds a user-bound key and restores it to the default bind. To unbind a key without restoring the default, use `/bind <key> "nop"` See also `unbind_all` and `nop`.
/unhide…	See `hide`.
/unignore *name*	Stop ignoring user.
/unlackey **/unlk**	No longer be a lackey. May be obsolete.
/unlevelingpact	Bring up dialog for quitting a leveling pact.
/unmalefactor **/unmal**	No longer be a malefactor. (Obsolete.)
/unselect	Unselects currently selected thing. Bound to ESC by default.
/unsidekick **/unex** **/unexemplar** **/unlackey** **/unlk** **/unrsk** **/unsk**	No longer mentor (or be a sidekick). (Obsolete.)
/up	Jump or fly up. Bound to SPACE by default.
/usebumpmaps {0|1}	Use bumpmaps if available. Default: 1
/usecelshader {0|1}	Turns on cel shading effect (primarly, thin black outlines around all characters and objects). Default: 0 This option seems to override and clash with some other settings, although it may be video card/driver specific. For example, turning on cel shading with water effects set at 4 created some odd wavering effects. (The water effects remain at the base level as well.) So if you get weirdness when using this option, try turning off/down some of the other visual effects. (Some hate this look. I think it freshens the game a lot.)
/usecubemap {0|1}	Use cube map. Function unclear.

Slash Command	Description
/usedof {0|1}	Use Depth of Field (DOF) effects if available. Warning: enabling DOF can seriously impact rendering speed and framerate. See `dofweight`.
/usefp {0|1}	Use floating point render target for HDR effects if available. Function unclear; default seems to be 1.
/usehdr {0|1}	Use HDR lighting effects (bloom, tonemapping) if available. Function unclear; default seems to be 0.
/usehq {0|1}	Use high quality shader variants if available. Function unclear.
/usenewcolorpicker {0|1}	Use updated color picker in game editors.
/useenvfence {0|1}	Use NV fences instead of ARB queries. Function unclear.
/userenderscale {0|1}	Use renderscaling if available; see also `renderscale`, `renderscalex`, `renderscaley`, `rendersize`.
/usewater {0-4}	Use fancy water effects if available. Higher numbers render more detail and advanced effects.
/vis_scale {0.0-16.0}	Controls draw distance. 1.0=default. Set closer to improve performance, further to improve your alt's visual acuity. Higher settingsmay have notable impact on framerate. Limit increased from 2 to 16; has interesting results when combined with cel shading.
/watching	List all channels that you belong to.
See Appendix G for complete information on using window save and load functions.	
/wdw_load	Reads window configuration from the file wdw.txt in the default installation folder.
/wdw_load_file *filename*	Reads window configuration from the specified filename in the default installation folder, or, if specified, in a different location.
/wdw_save	Saves window configuration to the file wdw.txt in the default installation folder.
/wdw_save_file *filename*	Saves window configuration to the specified filename in the default installation folder, or, if specified, in a different location.
/whereami	Tells you mission name, map name and location.
/who *name*	Show info on player. Appears to be identical to `search` except for requiring full or partial name string.
/whoall	List who's on the current map, in the system chat window.

Slash Command	Description
/window_color *R G B T*	Changes the window colors. R-G-B-T should each be replaced with a number from 0-255, where R=Red, G=Green, B=Blue and T=Transparency (for which 255=100% black).
/window_hide *window_name*	Forces the given window to be hidden.
/window_names	Lists all valid window names for use with toggle and scaling commands.
/window_resetall	Resets all window locations, sizes, and visibility to their defaults.
/window_scale *window_name* {0.6-3.0}	Changes the named window to the display scale indicated. Scaling was increased to 3.0 from 2.0 for I14+/Post-Live.
/window_show *window_name*	Forces the given window to be shown. (Synonym: `show`)
/window_toggle *window_name*	Show a window if hidden, hide a window if shown. (Synonym: `toggle`.)
/zoomin {0|1} **/zoomout {0|1}**	Controls the zooming in and out of the view. Usually used with the + and ++ modifiers. As with the `lookup`/`lookdown` pair, this command pair will accept the 0/1 variable: if both are set to 1 or 0, camera zooming is unaffected; if one or the other is set to 1, the zoom will persist towards one extreme. (There might be some useful purpose to this, but it's eluded me. I think it's a slightly buggy side effect.)

A.2 Base Editing Commands

I hesitated at including this information, because it's down into technogamewonkism that may be of little value to any but the most advanced players/base builders. But what the hell.

When you enter base editing mode from within the supergroup base, several things happen. First, you get a god's-eye view and the ability to run right through anything but exterior walls. Second, your command bindset gets changed, with an addendum of about 15 lines that override any prior definition of those keys. In theory, these extensions are removed when you exit editing mode. The takeaway is that there are "fixed" keys for base editing, but they could be changed or extended.

A general warning: base editing is frequently buggy. One bug I ran into to was that exiting base-edit mode did not always remove those appended commands, and it's startling to have a command bound to DEL or F1 suddenly generate a mysterious "bad command" error. If it happens, reload your bindfile or save and edit it, removing the lines at the end beginning with `DELETE "sell"`.

The default base editing keys are as follows:

- LBUTTON click – select item or option. Place selected item.
- LBUTTON double-click – center alt on that spot.
- LBUTTON drag – move selected object. Can be VERY tricky and erratic to move the right object in a crowded or overlapping spot.
- R or RBUTTON click – rotate object 90 degrees.
- TAB – select next object in view ("target next")
- SHIFT TAB – select previous object in view ("target prev")

- CTRL+Y – Redo last Undo; possibly repeat last action under some conditons.
- CTRL-Z – Your endless friend: undo that last misbegotten action.
- F1 – Toggle grid snap for placement (¼ , ½, 1. 2. 4, Off)
  F2 – Toggle angle snap for drag rotation (Off, 1, 3, 5, 10, 15, 30, 45 degrees).
  (Note that Click-rotate is fixed at 90 degrees.)
  F3 – Toggle room clipping on and off for object placement.
  F5 – Toggle object placement attachment (Floor, Wall, Ceiling, Surface).
- Esc – Cancel selection or action. (Same as general 'abort' command in zones?)
- The shift keys operate on mouse-drag as follows, enabling and restricting object motion…
  - Shift: …to vertical (Z axis, up-down).
    Ctrl: …to horizontal (X/Y axes, rank/file).
    Alt: …to rotation on Z axis (vertical axis).
    Ctrl+Alt …to rotation on X axis (crosswise axis).
    Shift+Alt: …to rotation on Y axis (perpendicular axis).

By the way, you can search through the dozens of tabs and hundreds of base items when in "Place Item" mode. It's a bit obscure that the black oval in the Item menu is a search field.

The relevant and mostly useless base editing slash commands (arranged for convenient pagination) are as follows. Most of these commands are not in the cross-reference.

Base Edit Slash Command	**Description** (Default Key)
`/editbase {0-3}`	Enable base editing. Arguments are as follows: 0 – Exit base editing. 1 – Enter base editor. Works from anywhere but will reposition alt in Entrance Room. 2 – Open isometric view of plot, without editing privileges. 3 – Open plan (overhead) view of plot, without editing privileges. **Warning**: some users have reported that calling this command in regular zones results in permanent map breakage – falling through floor even after game restart.
`/angle_snap {0-359}`	Set angle snap to degrees for drag rotation.
`/angle_snap_cycle`	Toggle angle snap for drag-rotation (Off-45 degrees). (F2)
`/attach_cycle`	Toggle object placement attachment (F5)
`/base_redo`	Undo "Undo" and/or repeat action (Ctrl-Y)
`/base_select`	Select base object. (Left-click) Works as slash command at point of cursor.
`/base_undo`	Undo last action. Number of steps saved unknown. (Ctrl-Z)
`/center`	Center editing alt on spot indicated. (Left-doubleclick)
`/grid_snap {0-`*`n`*`}`	Set grid snap for object placement in grid units. Limits unknown; works from small fractions to 50.
`/grid_snap_cycle`	Toggle object placement grid. (F1)

Base Edit Slash Command	Description (Default Key)
/room_clip {0,1}	Set wall clipping on and off.
/room_clip_cycle	Toggle wall clipping on and off. (F3)
/mousedrag	Enable dragging object. (May work in regular zones, but with no effect or possibly disastrous ones.) (Left-drag)
/quit	Exit/cancel selection or action. (Esc)
/rotate {0,1}	Rotate object 90 degrees. 0=CCW, 1=CW. (Right-click, 0)
/see_everything {0,1}	Turn block outlines of all objects on and off. Also shows inherent 'objects' like lighting grids. Works with odd/even arguments beyond 0 and 1.
/select_next	Select next present object in series. (Tab)
/select_last	Select previous present object in series. (Shift-Tab)
/sell	Delete selected object. ("Sell" would put the value back in base funds, which are now moot.) (Del)
/sg_passcode *string*	Set base entry code for non-members. The string will be suffixed by a base-specific number, e.g. COMEIN-1234. Any player with this current code can enter the base. Works in regular zones but is SG privilege-controlled.
/stuck	Return editing alt to Entrance Room. It can happen.
/base_default_sky {0-15}	Sets the "open sky" style for the entire base. To see it, set any square of a room to maximum ceiling height and select "Open Sky" as the texture. Options come from the various zone styles and are: 1 Praetoria 2 Atlas Park 3 Boomtown 4 Grandville 5 Cimerora 6 Night Ward 7 Shadow Shard 8 Storm Palace 9 Dense Fog 10 Rikti Invasion 11 Zombie Apocalypse 12 Praetorian Invasion 13 Lighted Paths 14 Shadowed Paths 15 Starlit Space

A.3 Using the /powexec_location Command

This incredibly cool command was added to the Post-Live I25 release. It allows the execution of a power that typically requires a mouse click from a keybind or macro.

The command takes two arguments: location and power name to execute.

The `location` parameter may be any of several elements:

- To focus the power on yourself or the spot where you're standing, use `me` or `self`.
- To focus the power on a selected target (which may be anything targetable – friend, foe, pet or object – use `target`.
- To execute a power in a specific direction at a specific distance, use a `direction:distance` compound element:
  - `direction` may be any of six keywords: `up`, `down`, `left`, `right`, `forward` or `back`.
  - or, `direction` can be a numeric value in degrees, with `0` directly ahead, `90` right, etc.
  - or, `direction` can be specified as `camera` – the direction the view is currently pointing.
  - `distance` is numerically specified in world-feet. The keyword `max` may be used to specify the maximum range of the power.
- The value for `power` is any valid power name. It does not need to be enclosed in quotes unless you prefer to for clarity.

Some examples:

- `powexec_location me Fire Imps` will summon your Fire Imps at your location. (A cool variation for pets is to use `0:max`, which will cast them a bit further than your normal range and make them come scampering back, which is exactly where you want them in combat.)
- `powexect_location target Tar Patch` will enmesh the targeted enemy in a tar patch.
- `powexec_location up:100 Teleport` will teleport you up 100 feet.
  (This, used with either a numeric value or `max`, could be a great escape power when you're about to be overwhelmed by foes. For teleporters, anyway.)
- `powexec_location 0:50 Recall Friend` will teleport any selected teammate to a spot 50 feet in front of you. (A rolling bind could be used to vary the spot for serial use.)
- `powexec_location camera:max Teleport` will teleport you your maximum teleport distance in the exact direction the view is looking, including elevation. (More or less the default teleport action.)

There are a number of limitations with this command, mostly obvious ones related to the power and target being specified. If the combination won't work as a normal command, it won't work in this way, either. If you don't have the specified power, the command will do nothing. If you have no target selected for a `target` location, the command will do nothing.

More and better examples and suggestions solicited! Think creatively, experiment and give a good evil laugh over all the possibilities this new power method offers.

Thanks to Korbian of Titan Network for providing me with the information from the I25 Release Notes!

Appendix B: Group List of Slash Commands

Slash commands listed by functional group. Refer to the prior section for details of use. Synonyms are separated by commas. Commands may appear in more than one group as appropriate.

This list has been laboriously updated to the Post-Live commands. Someone owes me a Zookeeper badge!

Commands in red have issues; see their entries. Commands in green are obsolete.

Binds & Macros

/bind
/bind_load
/bind_load_file
/bind_load_file_silent
/bind_save
/bind_save_file
/bind_save_file_silent
/keybind_reset, /unbindall
/macro
/macro_image
/macroslot
/showbind
/unbind
nop

Chat & Email

/ac, /arena
/autoreply
/b, /broadcast, /y, /yell
/beginchat
/c , /coalition
/chan_create
/chan_desc
/chan_invite
/chan_invite_deny
/chan_invite_gf
/chan_invite_sg
/chan_invite_team
/chan_join
/chan_leave
/chan_members
/chan_mode
/chan_motd
/chan_send
/chan_timeout
/chan_user_mode
/change_handle
/chat
/chat_cycle
/chat_load
/chat_load_file
/chat_save
/chat_save_file
/chat_set
/chatoptions
/clearchat
/copychat
/emaildelete
/emailheaders
/emailread
/emailsend/, emailsendattachment
/f
/g, /group, /team
/getlocalname
/gfriend
/gfriend_player
/gfriends
/ghide
/gignore
/gignoring
/ginvite
/guide , /helpchat, /h, /hc
/hide
/hideall
/hidefriends
/hidegchannels
/hidegfriends
/hideinvite
/hideprimarychat
/hidesearch
/hidesg
/hidetell
/ignore
/ignorelist
/ignorespammer
/ignorespammer
/l, /local
/link_channel
/link_info
/logchat
/lp
/mailview
/myhandle
/playernote
/playernotelocal
/quickchat
/reply, /r
/req, /request, /sell, /auction
/say, /s
/send
/startchat
/supergroup, /sg
/t, /tell, /private, /whisper
/tabglobalnext
/tabglobalprev
/tabnext
/tabprev
/tabselect
/tabtoggle
/tell_last, /tl
/unhide
/unhideall
/unhidefriends
/unhidegchannels
/unhidegfriends
/unhideinvite
/unhidesearch
/unhidesg
/unhidetell
/unignore
/watching

Movement

/backward
/clicktomove, /ctm
/ctm_invert
/ctm_toggle
/down
/follow
/forward
/forward_mouse
/left

/loc,/getpos
/lookdown
/lookup
/mouse_invert
/mouse_look
/mouse_speed
/nojumprepeat
/playerturn
/right
/target
/target_custom_far,
/target_custom_near,
/target_custom_next,
/target_custom_prev
/target_enemy_far
/target_enemy_prev,
/toggle_enemy_prev
/target_friend_far
/target_friend_near
/target_friend_next
/target_friend_prev
/target_name
/turnleft
/turnright
/up

Character Control

/afk
/cc_emote, /cce
/costume_change, /cc
/emote. /e, /em, /me
/face
/first
/info_self
/info_self_tab
/release
/requestexitmission
/respect
/selectbuild
/set_title
/stuck
/suppressCloseFx
/suppressCloseFxDist
/third
/whereami

Powers Control

/alt2tray
/alttray
/alttraysticky
/clear_tray
/goto_tray
/goto_tray_alt
/goto_tray_alt2
/goto_trays_tray
/insp_combine
/insp_delete
/inspexec_name
/inspexec_pet_name
/inspexec_pet_target
/inspexec_slot,
/inspirationslot
/inspexec_tray
/manage
/next_tray
/next_tray_alt
/next_tray_alt2
/next_trays_tray
/powexec_abort
/powexec_alt2slot
/powexec_altslot
/powexec_auto
/powexec_location
/powexec_name
/powexec_server_slot
/powexec_slot
/powexec_toggleoff
/powexec_toggleon
/powexec_tray
/powexec_unqueue
/prev_tray
/prev_tray_alt
/prev_tray_alt2
/prev_trays_tray
/shownewtray
/tray
/traysticky
/traystickyalt
/traystickyalt2

Viewpoint Control

/camdist
/camdistadjust
/camreset
/camrotate
/camturn
/canlook
/zoomin
/zoomout

Pets

/clear_petnames
/inspexec_pet_name
/inspexec_pet_target
/pet_select
/pet_select_name
/petcom
/petcom_all
/petcom_name
/petcom_pow
/petoptions
/petrename
/petrename_name
/petsay
/petsay_all
/petsay_name
/petsay_pow
/release_pets
/showpetnames

Targeting

/target
/target_custom_far,
/target_custom_near,
/target_custom_next,
/target_custom_prev
/target_enemy_far
/target_enemy_prev,
/toggle_enemy_prev
/target_friend_far
/target_friend_near
/target_friend_next
/target_friend_prev
/target_name

Search & Info

/cmdlist
/comment
/sea, /search
/whereami
/window_names
/who
/whoall
/window_names

Teams & Friends

/assist
/assist_name
/buffs
/fl, /friendlist

/friend
/team, /g, /group
/getlocalinvite
/getlocalleagueinvite
/gunfriend
/gunfriend_player
/gunhide
/gunignore
/invite, /i
/kick, /k
/leaveteam
/levelingpact
/lfg
/lfg
/lfg_event_response
/lfg_remove_from_queue
/lfg_request_event_list
/lfgset
/lfgset
/link_interact
/link_interact_global
/makeleader /ml
/playernote
/playernotelocal
/team_accept
/team_decline
/team_kick_internal
/team_quit_internal
/team_select
/team_task
/trade
/trade_accept
/trade_decline
/unfriend, /estrange
/unlackey, /unlk
/unlevelingpact
/who

Supergroups

/altinvite
/coalition, /c
/coalition_cancel
/coalition_invite, /ci
/coalition_mintalkrank
/coalition_nosend
/coalition_sg_mintalkrank
/demote
/editbase
/findmember
/getcomment
/getglobalname
/getglobalsilent
/ginvite_sg
/lfg_event_response
/lfg_remove_from_queue
/lfg_request_event_list
/nameCaptain
/nameCommander
/nameEnforcer
/nameFlunky
/nameLeader
/nameLieutenant
/nameMember
/nameOverlord
/nameRingleader
/nameTaskmaster
/promote
/raid_invite
/sgenterpasscode
/sgkick, /sgk
/sgkickyes
/sgleave
/sgmode
/sgmodeset
/sg_passcode
/sgraidinvite
/sgraidwindow
/sgsetcostume
/sgsetdemotetimeout
/sgsetdescription
/sgsetmotd
/sgsetmotto
/sgstats
/sidekick
/supergroup, /sg
/sginvite, /sgi

Auctions

/auctionhouse, /ah
/blackmarket,
/wentworths
/mypurchases

UI & Windows

/chat
/chat_save
/clearRewardChoice
/compatible_cursors
/contextmenu
/custom_window
/custom_window_toggle
/dialog_answer
/dialog_no
/dialog_yes
/gamereturn,
/windowcloseextra
/graphfps
/help, /helpwindow
/info
/info_self
/info_self_tab
/info_tab
/map
/maxfps
/maximize
/maxinactivefps
/maxinactivefps
/maxmenufps
/maxmenufps
/menu
/monitorattribute
/nav
/netgraph
/popmenu
/powers
/quit
/quittocharacterselect
/quittologin
/screen
/screenshot
/screenshottga
/screenshottitle
/screenshotui
/show
/showfps
/stopmonitorattribute
/stopmonitorattribute
/tabglobalnext
/tabglobalprev
/tabnext
/tabprev
/tabselect
/tabtoggle
/toggle
/unselect
/usenewcolorpicker
/wdw_load
/wdw_load_file
/wdw_save
/wdw_save_file

/window_color
/window_hide
/window_resetall
/window_scale
/window_show
/window_toggle
/window_names

Arena

/ai /arenainvite
/arena, /ac
/getallarenastats
/getarenastats
/getratedarenastats

Architect

/architect
/editbase
/missionmake
/missionsearch
/mmentry
/see_everything

UI Graphics

/bloomscale
/bloomweight
/cgshaderpath
/cursorcache
/disable2D
/dofweight
/fsaa
/fullrelight
/fullscreen
/lodbias
/maxcolortrackerverts
/maxrtframes
/noparticles
/norenderthread
/nosunflare
/reloadgfx
/renderscale
/renderscalefilter
/renderscalex
/renderscaley
/rendersize
/renderthread
/screen
/screenshot
/screenshottga
/screenshottitle
/screenshotui
/shaderdetail
/shadowvol
/sliFBOs
/ss
/ss
/stopinactivedisplay
/suppressCloseFx
/suppressCloseFxDist
/sync, /synch
/texaniso
/usebumpmaps
/usecelshader
/usecubemap
/usedof
/useenvfence
/usefp
/usehdr
/usehq
/userenderscale
/usewater
/vis_scale

System

/autoperf
/autorun
/bug
/build_save
/build_save_file
/cmdlist
/copydebuginfo
/demorecord
/demorecord_auto
/demostop
/e3screenshot
/editbase
/gmotd
/localtime
/neterrorcorrection
/netgraph
nop
/noreport
/noversioncheck
/option_list
/option_load
/option_load_file
/option_save
/option_save_file
/option_set
/option_toggle
/petition
/priorityboost
/profiler_record
/profiler_stop
/servertime
/showtime
/stopinactivedisplay
/stuck
/supporthardwarelights
/netgraph
/window_names

Unknown

/clearAttributeView
/texwordeditor
/uiskin

Obsolete

/buy_coh
/chat_beta
/cov
/ex, /exemplar/lackey,
/lk, /sk, /rsk
/kiosk
/lightmaplodscale
/mousepitchmode
/newspaper
/sgcreate
/unmalefactor, /unmal
/unsidekick, /unex,
/unexemplar, /unlackey,
/unlk, /unrsk, /unsk

Appendix C: Bindable Key & Mouse Button Names

Unless noted, all keys can be bound with the ALT+, CTRL+ and SHIFT+ modifiers.

Avoid assigning binds to both synonyms of a key; only the last stored will be used and inadvertent overwriting of the first bind will occur.

Many keys, such as the three shift sets and the "lock" keys, will have system actions as well as activating a bind. Use them sparingly if at all.

C.1 Bindable Keyboard Key Names

Key	Notes
`A through Z`	Main keyboard alphabetical keys. These keys are case-insensitive in bind definitions; F and f are the same key. Use SHIFT+ to bind two commands to the same alpha key based on "case."
`1 through 0`	Top numeric keys. Each of the symbols above the numbers is bindable as SHIFT+[number].
`F1 through F12`	Top function keys.
`SPACE`	Space bar.
`COMMA`	
`/` `SLASH`	The second character on each of these keys is bindable as SHIFT+[key]
`\` `BACKSLASH`	
`;` `SEMICOLON`	
`` ` `` `APOSTROPHE`	
`-` `MINUS`	Top function key.
`[` `LBRACKET`	The { and } keys are bindable as SHIFT+[and SHIFT+].
`]` `RBRACKET`	

`[ALT]`
`[LALT]`
`[RALT]`
	The three "shift" key types and their left side/right side codes, which are nine separate options.
	The three base shift codes can only be used in combination with another key: ALT-T, SHIFT-F9. etc.

`[CTRL]`
`[LCTRL]`
`[LCONTROL]`
`[RCTRL]`
`[RCONTROL]`
	The six left/right key variants can be used as a synonym for the base code in combination with any key – that is, CTRL-R and LCTRL-R work identically and with both left and right Ctrl keys.
	The six left and right codes can be used as "tap" keys – for instance, /bind LALT "emote wave" will trigger that emote with a

Key	Notes
[SHIFT] [LSHIFT] [RSHIFT]	tap of only the left Alt key, and the right Alt key can be separately bound in the same way. To completely confuse things, LCTRL and LCONTROL and its right-side twins are locked synonyms… if you set one, the game will write bind lines for both. Only if you delete both with a "" definition will both disappear. I suggest not using them at all. **I believe this is an archaic and potentially confusing option. I recommend against using the left and right shift keys as tap keys, to prevent accidental activation of the wrong command.**
BACKSPACE	
END	
ESC	
ENTER	Main keyboard Enter/Return.
EQUALS	= key. The + key is bindable as SHIFT+=.
HOME	
INSERT	Does not appear to be (re)bindable.
PAGEUP	
PAGEDOWN	
TAB	
SYSRQ	SysReq/PrintScrn key. ALT+SYSRQ not functional.
DELETE	
PAUSE	Pause/Break key. (Does not insert pauses.)
NUMPAD0 – NUMPAD9	The numeric keypad number keys.
NUMPADENTER	The numeric keypad ENTER key.
DECIMAL	The numeric keypad Del/. key.
MULTIPLY	The numeric keypad * (asterisk/multiply) key.
DIVIDE	The numeric keypad / (slash/divide) key.
SUBTRACT	The numeric keypad – (minus) key.
ADD	The numeric keypad + (plus) key.
UP UPARROW	Extended keyboard arrow keys. Although both keys in each set are separately listed in the default keybinds, they are synonyms in all tested situations. Because the default setup writes out both, it may be best to define both in all cases, so that unintended defaults or leftover custom binds aren't overwritten by the other, and vice versa. (Very annoying bug, really.)
DOWN DOWNARROW	
RIGHT RIGHTARROW	
LEFT LEFTARROW	
CAPITAL	Caps Lock key (as tap key) – will toggle Caps Lock as well.

Key	Notes
SCROLL	Scroll Lock key (as tap key) – will toggle Scroll Lock as well.
NUMLOCK	Num Lock key (as tap key) – will toggle Num Lock as well.

C.2 Bindable Mouse Button/Action Names

Although most lists, including the last iteration of this guide, assumed that only basic mouse buttons could be used for binds, recent work with the base editing commands revealed that there's much, much more you can do with the pointing device.

As with keyboard keys, it appears that all of these mouse commands can be combined with the three shift keys for a vast range of control options.

Keep in mind that many mouse actions are inherently controlled by game and UI needs and may not take kindly to being over-bound.

Mouse Button	Notes
LBUTTON **LEFTCLICK**	Left mouse button. NOTE: Left-click is permanently bound to 'select' and using LBUTTON by itself for a bind is not a good idea. NOTE: Ctrl+left-click is permanently bound to selecting the auto power in the trays. Binding to this combination might have slightly erratic action. These two terms are synonyms, with `LEFTCLICK` overwriting `LBUTTON` in the default file order.
RBUTTON **RIGHTCLICK**	Right mouse button. NOTE: Right-click is bound by default to 'canlook' and using `RBUTTON` by itself for a bind is not a good idea. These two terms are synonyms, with `RIGHTCLICK` overwriting `RBUTTON` in the default file order.
MBUTTON	Middle mouse button, or mousewheel click. May not be bindable in all systems.
MOUSECHORD	Combination of the left and right mouse keys. One cool use for this is to bind it to UP, so that as you're running along, steering with the mouse, you can jump over obstacles one-handed.
MOUSEWHEEL	Mouse wheel – does not appear to be rebindable from the default use of camera distance adjustment from 0-80 feet.
BUTTON4 **BUTTON5** **BUTTON6** **BUTTON7** **BUTTON8**	Additional buttons on advanced pointing devices. Not clear how well this feature is implemented, nor what the limits on multi-button gaming mice might be; possibly driver-dependent. No secondary (drag, double-click) options appear to be supported.
LEFTDOUBLECLICK **RIGHTDOUBLECLICK** **MIDDLEDOUBLECLICK**	Double-click of the specified key.
LEFTDRAG **RIGHTDRAG**	Pointer-drag with either key depressed. Note: rightbutton-down is used for free look. A bind to `RIGHTDRAG` may have unpredictable results.

Usage Notes

The mouse button combinations can be used in a variety of modes, since they inherently point as they activate. However, only the left button will simultaneously call a power and activate it. (For example, if you bind Teleport to the left button, you will teleport to any clicked spot. If you bind TP to the right button, it will only call the power; a left-click is still needed to activate it. This difference can be useful: you might want to TP using the fastest method, but call Recall Friend or a pet summoning in two clicks. (Using only shift-key combinations with the right and left buttons is strongly recommended, though.)

Keep in mind that:

- The left button is bound to the required 'select' function. Any bind that's bound to this key, or any modified key (`CTRL+`, `ALT+`, etc.) will have to share operation with this inherent function. That could have unexpected results.
- `CTRL+LBUTTON` is the action to set and unset an auto power in the tray. If you use this bind, it will have overlapping action whether the pointer is on a tray power or in the general screen.
- `RBUTTON-down` and thus `RIGHTDRAG` are bound by default to the camera-look action. If you change this, you'll need a different camera-look key, or you might get erratic results. Might be best to avoid user binds to either of these key names.

C.3 Joystick/Controller Button Names

CoX is not a controller-friendly game. It is designed entirely around a keyboard+mouse control system.

That said, keywords do exist to map binds and macros to joystick and controller buttons. This table lists the key names; *vaya con Statesman* in getting them to work with your controller. It will take a combination of button mapping, game adjustments and controller-driver tuning.

A useful trick is to load a sub-bindfile that maps every one of these keynames to a short Local-channel ID string: /bind JOY1 "l JOY1" and so forth. Then you can park somewhere very remote, load the binds, and map out your controller's keys while entertaining NPCs within range. Doing this in a base is perhaps the best idea. (Doing it so it spams a global channel, as I managed, is a *bad* idea.)

Both "load" and "clear" bindfiles can be found on the *Heroica!* website. (The latter will wipe all the loaded controller binds so they don't clutter up your file.)

Joystick Button Names		
JOY1	JOY9	JOY17
JOY2	JOY10	JOY18
JOY3	JOY11	JOY19
JOY4	JOY12	JOY20
JOY5	JOY13	JOY21
JOY6	JOY14	JOY22
JOY7	JOY15	JOY23
JOY8	JOY16	JOY24
		JOY25
Joypad Key Names		
JOYPAD_UP		
JOYPAD_DOWN		
JOYPAD_LEFT		
JOYPAD_RIGHT		
POV Hat Key Names		
POV1_UP	POV2_UP	POV3_UP
POV1_DOWN	POV2_DOWN	POV3_DOWN
POV1_LEFT	POV2_LEFT	POV3_LEFT
POV1_RIGHT	POV2_RIGHT	POV3_RIGHT
X/Y Joystick Commands		
JOYSTICK1_UP	JOYSTICK2_UP	JOYSTICK3_UP
JOYSTICK1_DOWN	JOYSTICK2_DOWN	JOYSTICK3_DOWN
JOYSTICK1_LEFT	JOYSTICK2_LEFT	JOYSTICK3_LEFT
JOYSTICK1_RIGHT	JOYSTICK2_RIGHT	JOYSTICK3_RIGHT

Appendix D: Window & Menu Names

- All these window names can be used with the window control commands – `toggle`, `windowscale`, etc.
- All windows are scalable from 0.6 to 3.0 using the `windowscale` command.
- Windows with a tick in the **O** column can be opened with a command. Some windows cannot be opened except under certain conditions (e.g, if you're not a Mastermind, you can't open a Pet window).
- Windows with a tick in the **C** column can be closed with a command.
- Windows with names highlighted in green can be used as direct slash commands. For example, the map window can be controlled with window commands or toggled with `/map`.

Post-Live updates on this list and its details actively solicited.

Window Name	O	C	Description
`actions`	x	x	Actions window.
`auction`		x	Auction window.

`/ah`
`/auctionhouse`
`/blackmarket`
`/wentworths`
	x	x	Window name can only be used to close the window.
			To summon (or dismiss) the window, use any of the four slash commands. *May be summoned anywhere!*

`badge`	x	x	Badges window.
`badgemonitor`			Badge award window?
`browser`	x	x	*Unknown.*
`chansearch`	x	x	Channel Search window

`chat, chat0`
`chat1`
`chat2`
`chat3`
`chat4`
	x	x	Chat windows (not tabs).
			`chat` and `chat0` are synonyms.
			Chat windows will be opened even if they have no tabs assigned.

`clue` `clues`	x	x	Clues window.
`combatmonitor`			Combat Monitor display – a line-by-line configurable display window for character and combat attribute numbers. Cannot be summoned or closed directly. See Section D.1 below for detailed usage information.
`combatnumbers`	x	x	Combat Numbers window, opened with the Combat Attributes menu item on the Powers window. Provides the source attributes for the Combat Monitor window.
`compass` `nav`	x	x	Navigation window.
`compose`	x	x	Email Composition window.
`contact` `contacts`	x	x	Contacts window.

Window Name	O	C	Description
`contactdialog`		x	Contact Dialog window.
`contactfinder`	x	x	Contact Finder window. Comes up null if no new contacts are available.
`convertenhancement`		x	Convert Enhancement window.
`costume` `costumeselect`	x	x	Costume window.
`defeat`			Defeated window.
`email`	x	x	Email window.
`enhancements`	x	x	Enhancement tray.
`friend` `friends`	x	x	Friends window.
`health` `status`			Health/Level/Main Menu bar. *(Looked for this one a LONG time!)*
`help`	x	x	Help window. Not the same as `/help` command.
`incarnate`	x	x	Incarnate window.
`info`		x	Object Info window.

`insp`
`insps`
`inspiration`
`inspirations`
	x	x	Inspirations tray.

`league`	x	x	League window. Not the same as `/league` command.
`lfg`	x	x	LFG window. Not the same as `/lfg` command.
`lfgdialog`			LFG Dialogue window.
`loyaltytreeaccess`			*Unknown.*
`lwcui`			*Unknown.*
`mainstoreaccess`			*Unknown.*
`map`	x	x	Map window.
`mission`	x	x	Mission window.
`missionreview`	x	x	Mission Review window.
`missionsummary`		x	Mission Summary window.
`newfeatures`	x	x	New Game Features window.
`options`	x	x	Options dialog.
`paragonrewards`	x	x	Paragon Rewards window.

Window Name	O	C	Description
`pet`	x	x	Pet window. Useful when new MMs lose their window. The Pet Options window can only be opened with `/petoptions`.
`petition`	x	x	GM Petition dialog.
`playernote`	x	x	*Unknown.*

`power`
`powers`
`tray`
	x	x	Powers tray.
			Note inconsistency of `powers` window name and `/powers` command, below!

`powerlist` `/powers`	x	x	Power List window.
`quit`	x	x	Quit dialog.
`razertray`			*Unknown – Razer mouse features?*
`recipe` `recipes`	x	x	Recipe window.
`salvage`	x	x	Salvage window.
`salvageopen`			*Unknown.*
`scriptui`	x	x	*Unknown.*
`search`	x	x	Player Search window.

`sg`
`super`
`supergroup`
	x	x	Supergroup window.

`store`		x	Store dialog.
`target`	x	x	Target window.
`team` `group`	x	x	Team window. `/group` is not a synonym

`tray`
`tray1`
`tray2`
`tray3`
`tray4`
`tray5`
`tray6`
`tray7`
`tray8`
	x	x	Power trays.
			`tray` is a synonym for `power` etc.
			`tray`*`n`* controls one of eight tear-off power trays like those created by the + menu item. There does not seem to be a way to call up an individual number of tray except by using the tray clickers. However, if each tray is set to a specific tray, that instance will be persistent.
			Floating trays may be closed by right-clicking on the tray number. Note also that you can reconfigure the tray layout from this menu!

`vault`	x	x	Vault window. *May be summoned anywhere!*

D.1 Using the Combat Monitor Window & `/monitorattribute` Command

One of the most obscure user-interface options is a tiny window that can be used to list any of a vast number of character/build attribute values, one per line. All of these numbers can be looked up in the Combat Numbers window (and health and endurance are of course shown in the main display bars) but advanced players may find it useful to have a customized window displaying realtime health, defense, combat and other numbers of their choosing.

The **Combat Monitor** window consists of one or more lines with a label and a value, taken from your current status. If there is a limit to the number of lines that can be displayed, I don't know it. Lines have to be added to the window one at a time, using one of two methods.

Window Commands

The most direct way to build an Combat Monitor window is to open the Combat Numbers menu (via slash command `/show combatnumbers`, or the menu item on top of the Powers window). From the list of attributes, right-click on the ones you want displayed.

With one or more attributes in the Combat Monitor window, right-click on that window for commands to remove and re-order the attributes shown, or to close all of them.

The Combat Monitor window has some peculiar limits for control. It cannot be called up or closed using the usual window commands (`show`, `toggle`). It must be opened by adding lines, as above, or closed by either removing all lines or using the right-click, Close All command.

The window can, however, be scaled using the `windowscale` command… but only when it is currently displayed. The window name is `combatmonitor`.

The window-click method of configuration is simple but can be tedious to keep adjusting and re-calling displays. So…

Slash Commands & Binds

Attributes can also be added and removed from the Combat Monitor window by executing a slash command to call them. The basic usage is:

`/monitorattribute` *`attribute_name`*

which will add the named line to the window. Since this is a toggle command; using it with the same argument will add and then remove the relevant line from the window. This may not be an optimal method, as removing all the lines with toggle commands tends to make the last few lines reappear when the alt changes zones or logs back on. It seems far more certain to use this command only to add lines, and use:

`/stopmonitorattribute` *`attribute_name`*

to turn off and remove each window line.

NOTE: Given the clumsy method and long command strings required, and the slow process of using the window method, this is a *really* good place to use a set of binds to open and close selected lines.

The arguments that can be used to specify each line of data are *extensive*. It would take too much space to list them all here, so the alternative is simply to note that **every single line in the Combat Numbers window can be called up and added to the Combat Monitor**. Serious players should review the many panels of the Combat Numbers menu and choose which values will be useful in a small, realtime, heads-up display.

The argument value for for each display value is the **full string name of the equivalent line in the Combat Numbers list**; multi-word names must be fully spelled out. Case does not matter and quotes are not necessary.

A few of the most useful lines that can be added to the Attribute Monitor are as follows:

Argument	Display Line Contents
`Current Hit Points`	Current hit points (green bar)
`Current Endurance`	Current endurance level (blue bar)

Argument	Display Line Contents
`Endurance Consumption`	Current endurance consumption rate
`Recovery Rate`	Endurance recovery rate (blue bar)
`Regeneration Rate`	Hit Points regeneration rate (green bar)
`Experience Debt`	Current experience debt
`Experience to Next Level`	Experience needed to reach next level
`Influence`	Current Influence/Infamy

There are so many more detail values in the Combat Numbers window that the only useful way for a user to make their selections for this display is… ***Go. Look. Find Numberz.***

Appendix E: Emote Codes

These codes can be executed at almost any time using the slash code `/emote`, `/em`, `/e` or (amusingly) `/me`. They can also be selected from the QuickChat menu, which is raised by clicking the small button at the right end of the chat text entry window, or by the slash code `/quickchat`.

To use emotes in a bind or macro, use "`emote` *`emotename`*" as the command string.

The best way to see what each emote does is to find a quiet corner of the map, use camera rotate (default: PAGEDOWN plus the mouse) to spin around so you're looking at your character from the front, and try each one out. Set the chat input to something null like team so mistakes won't get hoots from the other players.

I have combined some codes out of alphabetical order, and under generic headings, for clarity.

A code is "static" if it stays until an interrupt key (such as movement) is pressed. Mouselook can often be used during a static emote without interrupting the emote. Powers on auto can interrupt as well.

Note that many of the QuickChat options are similarly named, but include fixed chat bubbles as well.

If you use any emote string besides one of these valid codes, the string will appear in a thought bubble over your head, visible to others, preceded by your character name. ("Shenanigunner wishes he had a beer.")

See the end of this section for complete information on using costume-change emotes.

See following note on using pet emotes in binds.

Code	Static?	Sound?	Animation/Artifact	Notes & Description
afk **newspaper**	Yes	No	Read newspaper	Good basic "I'm waiting" or AFK emote.
afraid **cower** **fear** **scared**	Yes	No	Cower in fear	You too can be a civilian.
airguitar	Yes	No	Play mad air guitar riff	
alakazam	No	No	Dramatic magician gesture	
alakazamreact	No	No	Turn into a random object (and back)	A major hoot. Just try it.
akimbo **wings**	Yes	No	Stand with hands on hips	See also STANCES.
angry	No	No	Animated anger	Many others.
assumepositionwall	Yes	No	Stand against wall as if to be searched	Looks pretty stupid unless you stand facing a wall or other surface as closely as you can before executing.
atease	Yes	No	Stand at ease	
attack	No	No	One-arm motion forward	
backflip **flip**	No	No	Perform a backflip	
batsmash	Yes	No	Animated lay about you with a baseball bat	
batsmashreact	Yes	No	Animated react to getting hit with a baseball bat	

Code	Static?	Sound?	Animation/Artifact	Notes & Description
bb **boombox** **dropboombox**	Yes	Yes	Character places boombox in front of him/her	The basic socialization, showoff and time waster emote – haul out the boombox and dance. The tune will be randomly selected from those listed below
bbAltitude **bbBeat** **bbCatchMe** **bbDance** **bbDiscoFreak** **bbDogWalk** **bbElectroVibe** **bbHeavyDude** **bbInfoOverload** **bbJumpy** **bbKickIt** **bbLooker** **bbMeaty** **bbMoveOn** **bbNotorious** **bbPeace** **bbQuickie** **bbRaver** **bbShuffle** **bbSpaz** **bbTechnoid** **bbVenus** **bbWahWah** **bbWindItUp** **bbYellow**	Yes	Yes	Boombox + dance	Using these codes will select specific boombox tunes instead of randomly choosing one of them. Avoid newbie zones where as many newbies as possible attempt to set up competing-tune boomboxes. For one thing, it can crash your client. For another, it can crash your brain. See also `drumdance`.
beatchest **tarzan**	No	Yes	Chest-pounding	Audible growl.
biglaugh **laugh2** **laughtoo**	No	No	Hearty laugh or chuckle	
bigwave **overhere**	No	No	Animated big wave	
binoculars	Yes	No	Look through binoculars	
blankfiller	NA	No	NA	Appears to be the emote equivalent of "nop" for slash commands. Does nothing but generates no error either.
bow	No	No	Bow	
bowdown	No	No	Demand person before you bow down	
burp	No	Yes	Burp	Audible. Look, I'm a *rude* Warwolf!

Code	Static?	Sound?	Animation/Artifact	Notes & Description
`buzzoff` `goaway`	No	No	Shooing motion with hand.	
`calculate`	Yes	No	Write and regard foggy runes in front of you	
`camera`	Yes	No	Take pictures (continuously) with old-fashioned Speed Graphic camera.	Cannot be interrupted until first part of animation is completed.
`cameraphone`	No	No	Take out your smartphone and snap a picture.	Cannot be interrupted until first part of animation is completed.
`cardtrick`	Yes	No	Produce a deck of cards, do a trick, toss them in the air	The toss effect repeats every few seconds. See also `juggle`.
`cellphone`	Yes	No	Talk on cel phone	Wow, that's a big phone!
`champion`	No	No	Clasped-hands victory shake	See also `victory`.
`cheer`	Yes	No	Shake-fists encouragement	
`chicken`	No	No	Do the chicken dance	
`clap`	No	Yes	Applaud	Audible over Local distance.
`clipboard`	Yes	No	Write on clipboard	

`[DECIDE]`
`cointoss`
`coin`
`flipcoin`
	Yes	No	Animated coinflip motion; show head or tail coin overhead	Make a choice for the team or group, or yourself. Coin remains until interrupted.
				See also `dice` and `paper`.

`crack`
`knuckle`
`knuckles`
	No	No	Crack knuckles	Loud sound effects!

`crossarms`	Yes	No	Cross arms	
`crouch`	Yes	No	Crouch down, frog style	
`curseyou` `noooo`	No	No	Animated shaking fist at the heavens in dismay	
`dance`	Yes	No	Animated dancing	Several random dances; repeat emote for others. You can also use the dance*n* commands following to select specific dances. See also `drumdance` and `boombox`.
`dance1`	Yes	No	Cha-cha dance	
`dance2`	Yes	No	Rah-rah dance	
`dance3`	Yes	No	Twist dance	
`dance4`	Yes	No	Hands waving in air dance	
`dance5`	Yes	No	Hands in air hop dance	
`dance6`	Yes	No	High-energy twist dance	

Code	Static?	Sound?	Animation/Artifact	Notes & Description
[DECIDE] **dice** **rolldice** **dice7***	No	No	Dice roll motion; show die overhead	Make a choice for the team or group, or yourself. Die fades after a few seconds. `*dice7` is a special emote unlocked by completing the Hess task force; the die always rolls 7. Heroes only, although dice cheating would seem to be more appropriate to Villains… See also `cointoss` and `paper`.
dig	Yes	Yes	Dig hole with shovel	
disagree	No	No	"No" wave with short lecture animation	
dontattack	No	No	Two-hand no wave	
donut **eatdonut**	Yes	No	Eat a donut	
drat	No	No	Thump both fists	Express frustration in a friendly way.
drink	Yes	No	Drink from glass	See also `eat` and `donut`.
drinkenriche	Yes	No	Drink from bottle	
drinktea	Yes	No	Drink cup of tea	See also `teabag`.
drum	Yes	Yes	Pound on huge tribal drum	Loud sound effect.
drumdance	Yes	No	Raindance	See also `bb` and `dance`.
drumlow	Yes	Yes	Pound on small tribal drum	Bongo sound effects.
dustoff	No	No	Brush off hands	
eat **food**	Yes	No	Eat food item	Alternates between burger, hot dog and sandwich.
eatdonut **donut**	Yes	No	Eat donut	
evillaugh **elaugh** **muahahaha** **villainlaugh** **villainouslaugh**	No	No	"Bwah-ha-hah" villain laugh	(How many synonyms are needed for one emote!?)
explain	No	No	Animated "hold it," with short lecture animation	Cannot be interrupted until first part of animation is completed. See also `lecture`.
facepalm **doublefacepalm**	No	Yes	Smack own face with one or two hands	Choose based on just how stupid that noob's move was.

Code	Static?	Sound?	Animation/Artifact	Notes & Description
fancybow **elegantbow**	No	No	Animated elaborate bow	
[EXERCISE] **jumpingjacks** **kata** *-or-* **martialarts** **pushups**	Yes	Yes	Do jumping jacks, martial arts kata, or pushups	See also SPORTS.
explain	Yes	No	Explain your reasons	See also `lecture`.
fishing	Yes	No	Fish with long pole	
flashlight **flashlightdown**	Yes	No	Looking around with large flashlight over shoulder, pointed mid-downwards	Does not appear to project light.
flashlightup	Yes	No	Looking around with large flashlight over shoulder, pointed mid-upwards	Does not appear to project light.
flex **flex1** *–or–* **flexa** **flex2** *–or–* **flexb** **flex3** *–or–* **flexc**	Yes	No	Animated bodybuilder poses	Impress newbies and that cute controller by doing your Arnie impression. Three different poses for your convenience. (`flex` and `flex2` are the same.)
flippingcoin	Yes	No	Flip coin gambler style	Not same as `flip`; does not generate "result."
floatbooks	Yes	No	Float three books in front of you and appear to study them	
[FLY] *These four emotes work only when you are already flying. If you pause, your character will revert to the standard flying pose. There is no emote to return to the standard flying posture. A fly-forward plus emote keybind is recommended for regular use, or a keybind that cycles through the options.*				
flypose1	Yes	No	Fly with fists out front	
flypose2	Yes	No	Fly with one fist out front	Superman pose
flypose3	Yes	No	Fly with hands flat out front	Swan dive pose
flypose4	Yes	No	Fly with fists to sides	Invisible hang glider pose
frustrated	Yes	No	Animated shake both fists	Stays in fist-clenched posture after shake.
getsome **kissit**	No	No	Turn fanny to front, pat it	Ruuuuude. Love it.
grief	Yes	No	Grief on knees	Stays on knees after initial animation. Not to be confused with `propose`, which also leads to grief.
hand **talktohand**	No	No	Hand out in "talk to the hand!" style	Yeah, right, enough outta you.

Code	Static?	Sound?	Animation/Artifact	Notes & Description
`handsup` `surrender`	Yes	No	Hands in the air	Alternate positions: standing and kneeling
`hi` `wave`	No	No	Wave	
`hiss`	No	Yes	Hiss and spit like a cat	See also `sniff`, `roar`, `howl` and `savage`.
`holdtorch`	Yes	No	Hold a tall flaming torch	Not sure if it actually projects any light.
`hottemper`	No	No	Rage until steam comes out of your ears	
`howl`	No	Yes	Howl like a Warwolf	See also `sniff`, `roar`, `hiss` and `savage`.
`hmmm` `plotting`	No	No	Stare into space and rub chin	

`huh`
`shrug`
`what`
	No	No	Shrug	

`inspiration`	No	No	Think and get bright idea	
`invent`	Yes	No	Manipulate a cool luminescent grid thingy	That or it's a new-gen Rubik's Cube. Used whenever a character is interacting with an invention table.
`jackhammer`	Yes	Yes	Use jackhammer	Noisy.

`[JUGGLE]`
`juggle`
`juggleelectricity`
`jugglefire`
`jugglemagic`
	Yes	No	Juggle three balls. The first command conjures three colored balls. The other three add blue electricity, fire and glow effects; magic also adds sparkly auras.	See also `cardtrick`.
				For fun, keep changing among these options.

`kneel`	Yes	No	Kneel down	
`laptop`	Yes	No	Work on laptop that appears on pedestal	Occasionally seem to experience computer trouble. (Is this a backhanded joke at a "boss" key?) See also `type`.
`laugh`	No	No	Hands-on-hips laugh	Why, yes, I *am* Errol Flynn!
`lecture`	No	No	Deliver a lecture	See also `explain`.

Code	Static?	Sound?	Animation/Artifact	Notes & Description
`listenpoliceband`	Yes	Yes	Whip out your way-cool holographic police radio	Used for the police band mission contact. Hero side only. There is also a listenstolenpoliceband emote, but it is blocked.
`lotus` `yoga`	Yes	No	Animated lotus position	Sophisticated resting posture.

`[LOYALTY]`
`heroloyal`
`rogueloyal`
`vigilanteloyal`
`villainloyal`
	Yes	No	Pose on one of four specific medallions, with other animations	All can be used by any alt type

`marriageproposal` `propose`	Yes	No	Down-on-knee proposal	Was originally part of the wedding pack. See also `throw`....
`mixformula`	Yes	Yes	Pour liquid between two flasks	
`no`	No		Animated wave-hands "no"	See also `disagree`.
`nod`	No		Animated nod	
`observedice`	Yes	No	Closely watch dice game on ground	
`opengift`	No	Yes	Open wrapped gift to a spray of confetti	
`pamphlet`	Yes	No	Hand out flyers or pamphlets from your stack	
`panhandle`	Yes		Animated sit with cup, offering as to passersby, occasionally looking it it disappointedly	One way to bug inf off of high-level players.

`[DECIDE]`
`paper`
`rock`
`scissors`
	No	No	Play rock-paper-scissors or Rochambeau; choices appear overhead.	Settle disputes. Animation shows all three icons for five seconds, then your selected one. See also `cointoss` and `dice`.

`peerin`	Yes	No	Animated peering in window with hands cupped around face	Occasional look-around to see who's watching.
`picklock`	Yes	Yes	Kneel and bang on something	
`plot` `scheme`	Yes	No	Hunch and rub hands together as if making evil scheme	
`point`	No	No	Animated one-hand point straight ahead	
`praise`	Yes	No	Animated salaam on knees	

Code	Static?	Sound?	Animation/Artifact	Notes & Description
protest	Yes	No	Shake large protest sign	Appear to be three different signs that come up at random. All are illegible except for a large STOP, NO and red circle/slash over an indistinct outline of something.
protestloyalist	Yes	No	Shake large protest sign	Appear to be different signs that come up at random, All combine "no" with a yellow star.
protestresistance	Yes	No	Shake large protest sign	Appear to be different signs that come up at random. All combine "no" with a blue chevron insignia.
raisehand stop	Yes		Animated raise one hand	
readbook	Yes		Read from book	
research	Yes		Animated refer to book, then examine what's in front of you	Circle of Thorns seen doing this in Hollows and elsewhere.
researchlow	Yes		Animated refer to book, then examine what's in front of you, while squatting down	Circle of Thorns seen doing this in Hollows and elsewhere.
roar	No		Roar like a Warwolf	See also `sniff`, `hiss`, `howl` and `savage`.
[ROBOT] robotpowerup robotpowerdown	Both	No	Power up emulates a robot being powered up, with aura effects, and then ends. Power down makes the alt hang forward, arms loose, and stay in that position.	
rooting wavefist	No		Animated wave fist, hands-to-face shout and clap with sound	Only clapping has sound.
[SALUTE] salute militarysalute praetoriansalute	Both	No	Three saluting options. The first delivers a salute and ends. The second holds a salute until interrupted. The third is a very elaborate Roman-style salute.	
savage	No	Yes	Hop around like an ape	See also `sniff`, `roar`, `howl` and `hiss`. Naming of this emote perhaps just a tad racist?

Code	Static?	Sound?	Animation/Artifact	Notes & Description

`score1`
`score2`
`...`
`score9`
`score10`
	Yes	No	Hold up score card with 1 to 10 on it, Olympics-style	Show your opinion of another player's move. Fun to use with costume contests, etc.
				(What, no score zero?)

`screen` `touchglass`	Yes	No	Reach out and touch surface in front of you as if not sure it's there, or touch wall-screens	Fabulous animation if character is in a bubble or if you pivot viewpoint so that you're looking right into character's face. Fun.
`shucks`	No	No	Animated thump one fist	Aw, it was nothing.
`sit` `ledgesit`	Yes	No	Animated sit down, either on ground or as if on ledge.	Take a load off. Fun to do on benches, trees, etc. Takes some practice in pre-positioning to get it right.
See section E.2 below for advanced sit and ledgesit commands, which are differentiated by alt gender.				
`slap`	No	Yes	Animated forehand slap	With light burst and slap sound. Combine with `slapreact` from other character for more fun.
`slapreact`	No	No	Reaction to being slapped or struck	
`slash` `slashthroat`	No	No	Draw finger across throat.	Stop; Shut up, dude; *or* You're dead, you know.
`sleep`	Yes	No	Fall asleep standing up, with stream of Z's rising	
`smack`	No	Yes	Backhand slap	Great sound effect. See also `slap`.
`smackyou` `threathand`	No	No	Threaten to backhand someone	See `smack`.
`sorry`	No	No	Apology gesture	

`[SPORTS]`
`basketball`
`pool`
`soccer`
	Yes	Both	Standing animations that do fancy basketball dribbles, fancy soccer ball moves, or just stand talking while holding a pool cue	See also EXERCISE.

`spraypaint`	Yes	No	Take out can of spray paint, paint on surface in front of you.	

Code	Static?	Sound?	Animation/Artifact	Notes & Description
[COMMON STANCES] **idle1** **idle2** **batlookout** **[HERO STANCES]** **stancehero1** **stancehero2** **[VILLAIN STANCES]** **stancevillain1** **stancevillain2**	Yes	No	Four different standing poses, all with arms at sides. The two `idle` variants are identical, with arms slightly away, but turn differently. The two `stance`… variants are identical, with arms closer to the body, and turn differently. The first villain pose is a sort of menacing crouch; the second is with arms tightly folded. The `batlookout` code has the alt holding a bat in a menacing posture.	All emotes can be used with either hero or villain alts, but (like the sit emotes) have slightly different results for each type. `idle1` is the standard alt posture. See also `akimbo` and `crossarms`.
talk	Yes	No	Talk as if in conversation	
taunt **taunt2** **tauntb**	No	Yes	Two-hand taunt with "hoooah" sound	Character stays in combat pose after taunt
taunt1 **taunta**	Yes	Yes	One-hand taunt with "aaaaah" sound	Character continues to pound fists with sound effect after taunt
teabag	Yes	No	Dunk teabag in a teacup	See also `drink, eat`. No, not *that* kind of teabag.
text	Yes	No	Take out your smartphone and thumb-chat away.	
thanks **thankyou**	No		Left-hand gesture	See also `yourewelcome`, which is a mirror-image gesture.
thewave	No		Vertical "wave" animation	
throwconfetti **throwrice** **throwrosepetals** **throwsnowflakes***	Yes		Throw confetti, rice or rose petals	Originally only available with the Wedding Pack. *The snowflake emote is blocked and may only be active during a Winter event.
thumbsup **yes**	No		Thumbs-up animation with nod	
trainwhistle	No	Yes	Pull cord on train whistle that magically appears	
type **typing**	Yes		Type as if on keyboard – same as `laptop` but without prop	Great for consoles in missions.
ultimatepower	No	No	Dramatic "change" or "use power" effect with auras	Would be a good costume change emote.

Code	Static?	Sound?	Animation/Artifact	Notes & Description
vendor	Yes	No	Call out like a carnival barker, then make your pitch	
victory	No		Victory arm wave	See also champion.
waiting	Yes	No	Various impatient waiting actions	
walllean	Yes	No	Relaxed lean back against wall	Alternate positions: hands in pockets or arms crossed. Stand as close to wall or object as possible before executing. You can also get into amusing positions if you do it back-to-back with static NPCs – it looks as if your character and the NPC are in a *very* close embrace.
warmhands	Yes	No	Rub hands, shiver and warm hands over fire	
welcome	No		Two-hand welcome	
whistle	No	Yes	One-hand whistle with piercing sound	Audible over Local distance – loud!
winner	No	No	Animated clasped-fist victory wave	See also victory and champion.
wounded	Yes	No	Wobble woozily	Like a weebelo. Remember Weebelos?
yatayata yata	No	No	Animated "talk-talk-talk" with hand	
yourewelcome	No	No	Animated right-hand gesture	See also thanks, which is a mirror-image gesture.

E.1 Using Emotes in /petsay Commands

Because of a bug in string processing (which is slated to be fixed, someday), pet emotes do not work in binds. They work fine in direct entry, and from macros, but not in binds. Various workarounds such as loading the binds from a text file don't work (either/any more).

The *proven* workaround is to write the commands to macros, then call the macros with binds:

```
/macro_slot 80 PD "petsayall <emote DrumDance>
/bind ALT+1 "powexec_tray 1 9"
```

…creates the macro in Tray 9, Slot 1 and calls it from there. (Note slot-first numbering.) You can use both regular macro and macro_image commands and move things around manually, too.

(Hopefully this workaround will have a short useful life…)

E.2 Advanced Sit Emotes

A huge selection of fancy sit emotes was added with Issue 8. They are somewhat complicated to list, because they are different for male/huge and female characters. (Ladies sit differently, guys, in case you've never noticed…) A variety of ledge-sit emotes were added Post-Live.

All are static.

Note: I have not tested these with Huge characters. I assume they are the same as male but if someone wants to test things and report back…

Emote	Male & Huge action	Female action
`ledgesit1`	Hands to sides, sitting straight.	Same.
`ledgesit2`	Slump sideways on one hand, one knee up.	Same.
`ledgesit3`	Sit hunched forward, same as `ledgesit`.	Same (and same).
`ledgesit4`	Hands behind, kick your feet a bit.	Same.
`sitbench1`	Legs out straight, hands straight behind	Same
`sitbench2`	Sideways sprawl with one leg up and one arm along bench back	Same as `sitchair1`
`sitbench3`	Sprawled back, feet flat, arms on bench back	Same as `sitchair1`
`sitbench4`	Same as `sitchair3`	Same as `sitchair1` – elevates over surfaces, though.
`sitchair1`	Straight back, feet flat, hands on knees	Straight back, knees crossed, hands center
`sitchair2`	Leaning forward, feet flat, hands loose in middle	Leaning back, feet flat, hands on thighs
`sitchair3`	Straight back, feet flat, hands on thighs	Same as `sitchair1`
`sitexecutivechair`	Lean back, hands on chair arms, feet flat	Same but legs crossed
`sitstool`	1 foot down, 1 foot on rungs, 1 hand on knee	Feet up on rungs, legs crossed, hands clasped on knee
`sittable1`	Straight back, knees loose, 1 arm on table, other hand to face	Same but knees together
`sittable2`	Same as `sittable1`, but hands loose on table	Same as `sittable1`, but lower table surface

E.3 Costume Changes & The Costume Change Emotes

One of the last features added to the Live version of the game was the ability to fire off an emote as you changed costumes. I am not sure this was ever fully functional in the Live game, but might have been on the last iteration to hit the Test server. It is fully functional now.

In the Live era, you had up to four different costumes, slots for three of which had to be unlocked through game achievements. The Post-Live game gives you six free slots and four that can be earned, so your alts can do the full Ken and Barbie wardrobe thing if you're so inclined.

Changing costumes is simple. Either open the Costume window and click on the one you want, or use:

`/change_costume 1` or `/cc 1`

to select the second costume in your set. Note that this is another of the 'zero based' lists, with your default costume being number 0. This command can be bound to any key or macro.

There is a delay before a costume can be changed again – I believe it was a full minute on Live, and was 30 seconds for a time, but is now 15 seconds. **Important**: doing a faulty call of this command (such as with an invalid emote name, or specifying your current costume slot number) starts the timer and you will have to wait. However, executing this command before the delay elapses does *not* reset the timer.

If you use the slash code, your costume will change instantly with no fuss.

If you use the Costume window, however, you have an interesting option. The small menu at the bottom lets you choose one of over two dozen special emotes that will bridge the costume change, from fairly simple salutes and puffs of smoke to some of the most dazzling effects in the game. Since the window is in your way, it's hard to get the full effect of the emote, but your teammates and passing noobs will be very impressed.

If you want to have more control and actually see your change emote, you can use a slash code, which again can be bound to a key or macro for convenience… or even a rolling macro or bind for variety:

`/cc_emote 1 ccSalute` or `/cce 1 ccHowl`

The emotes used for costume change are special, begin with `'cc'` and can only be used for this purpose; regular emotes can't be specified and the costume ones can't be used on their own. All of them include sound effects.

The choices, which are mostly fairly self-explanatory, can be found in the Costume window menu (where you might look for updates and changes), and are as follows:

cc Emote Code	Effect
`ccBackFlip`	Do a backflip and land in your new costume.
`ccCast`	Make a Dr. Strange/Constantine spell cast to change.
`ccConfettiThrow`	Throw a giant burst of confetti from your pocket and change.
`ccDimensionShift`	Spread into multiple dimensions and reassemble changed.
`ccDrinkFormula`	Drink a flask of formula to effect the change.
`ccEnergyMorph`	Crouch and fire off an energy burst to change.
`ccEvilLaugh`	Evil laugh and burst into flame to change.
`ccFeatherBurst`	Change in a spray of feathers.
`ccFireworks`	Change in a burst of fireworks.
`ccFurBurst`	Change in a spray of fur.
`ccGiftBurst`	Gift falls on you and explodes open to reveal change.
`ccHowl`	Howl and change.

cc Emote Code	Effect
ccIceBlock	Disappear into an ice block and emerge in your new costume.
ccInnerWill	Focused energy in your chest triggers the change.
ccLightMagic	Glowing ground sigil and vertical effects make the change.
ccLightning	Change in a furious burst of lightning.
ccMurderOfCrows	Change by bird … sort of an evil Cinderella thing.
ccNinjaLeap	Leap high into the air and land changed.
ccNuke	Impressive nuclear blast changes your looks.
ccOilStrike	Giant gusher of oil changes you.
ccPressureRelease	Stomp ground to release geyser that changes you.
ccPrestoChango	Change with a dramatic magical gesture.
ccPureEnergy	Change in an energy burst.
ccRainbow	Change in a magical haze at the end of a rainbow.
ccRapidBoil	Go to a bubbling boil of green sewer ick and emerge changed.
ccSalute	Give a full salute and change to your new costume.
ccSmokeBomb	Throw down a smoke bomb and emerge changed.
ccSpin	Spin rapidly and stop in your new costume.
ccStoneBlock	Disappear into a stone block and emerge in your new costume.
ccSuperSerum	Inject super serum, beat chest and emerge changed.
ccVanguardSigil	Vanguard ground sigil and green haze leave you changed.

This list was carefully checked against the current Post-Live server, I25.

Note: The ParagonWiki page on these emotes was well filled out and useful in figuring out this feature. It's not yet common for the new and changed powers to be updated on that original reference, but it was a surprise and a pleasure to find the info there. Kudos to the contributors!

Appendix F: Chat Bubble Color Codes

It is possible to change the appearance of your character's chat bubble in two ways. The simplest is to set the text color and the background color in the Options menu. For some reason, though, this setting only affects some chat bubbles; many will default to black-on-white.

The second way to set chat bubble color – and other characteristics – is to use inline format codes. These codes can be used in manually entered chat strings or as parts of binds. The complete code set is:

<color *ccode*><bgcolor *ccodetransparency*><border *ccode*><scale *factor*><duration *seconds*>

As far as I know, each command can be used separately and in any order.

`color` sets the text color. The value `ccode` can be any standard color name (not sure of the range, but basics like red, yellow, white, blue etc. should all work). You can also use hex codes in the `#rrggbb` format – look up those codes anywhere on the web if you're not familiar with them. This works the same as the text slider in the Options menu.

`bgcolor` sets the chat bubble background color, and works the same as color except that you can add an additional value to control the chat bubble background transparency. If you use only a color code, you get 100% color (that is, no transparency). If you add two digits to the end of the color code, you set the transparency, from 0 to 99%, with zero being fully transparent. This setting does not appear to have full 100-step granularity; there may be as few as 8 steps of transparency. I am not sure if strings like `'yellow50'` will work, but codes like `'#FFFF0050'` will.

`border` sets the color of the bubble border. Identical in operation to `color`.

`scale` sets the text and bubble size. It is supposed to scale from 0.0 to 4.0, with 1.0 being the default size, but it only works 0-2.0 for me. Useful for blowing up important bubbles like "Here!" when you've found more foes or a glowie.

`duration` sets the persistence of the bubble in seconds. Default is about 8 seconds. You can make bubbles like "Here!" more persistent, to give mates time to find you, by setting the value to 15 or so.

To use this method, embed the codes in a chat string, like this simple example:

```
g <color red><bgcolor black>Oh, no, dead again!
```

Note that any spaces between the codes will be added to the chat string.

If you want to make all your chat bubbles a specific style, or have multiple styles for different uses, you need to bind a key to start the chat and load the codes – then you type your message after the codes. A little murky, but it works. For example, the normal Chat key is Enter, so:

```
/bind ENTER "beginchat <color white><bgcolor blue><scale 2><duration 10>"
```

And whenever you press ENTER, you'll be ready to chat in large white-on-blue text with a 10-second persistence. The same thing can be used in general binds:

```
/bind CTRL+T "g <color blue><bgcolor red>Teleporting $target to me!$$powexecname Recall
                                        Friend"
```

…although be warned I have found this usage to get flaky at times.

A final bind you might find useful to experiment with or frequently change the settings is:

```
/bind CTRL+F1 "beginchat /bind ENTER "<color #00000><bgcolor #FFFFFF75>
                <border #FF0000><scale 1.0><duration 10>""
```

This mess will, when you press CTRL+F1, load the chat entry window with "/bind…" and the whole string that follows. Edit it to suit, press ENTER, and then use ENTER to start new chat lines with the edited characteristics. You've changed your keybind for ENTER by doing so. This can create a complete mess if you're not careful, so… be careful.

Option Settings for Chat Bubbles

In theory, chat bubble colors can be set using /optionset commands. In practice, the format of the color code arguments has completely eluded me – it's not decimal, it's not hex and it's not color names. The codes in the option.txt file can be edited using the format `RRGGBBXX`, a hex color value in which the last two digits are always written `FF` and seem to be ignored. The lowest value for any color is written as `01` but using `00` works fine. A leading zero is not written but can be used.

By the way… has anyone else noticed that chat bubbles vary slightly with the different channels? The team channels are slightly transparent and have a zig-zag pointer. All the others are solid and have a straight pointer. Odd.

Appendix G: Saving & Loading Interface Settings

With Issue 11 or 12, City of Heroes/Villains finally resolved one of the most annoying oversights in its design. Each new alt that you designed started with a generic user interface setup, and there was no way to duplicate a favorite layout and setup without laboriously configuring each element, every time. Now, however, there are not one but three separate "save/load" functions to save an aspect of a customized user interface and reload it into another character's interface.

There are three sets of customization commands, for chat, window layout, and the grab-bag "options." All work much like the process for saving and loading binds and macros, so any user who has mastered those basics should have no trouble with these facilities.

One cool use for a straightforward save/load process is to keep all of your alts' interfaces identical. Save from the tweaked setup; load to the others. But then… you can use six more commands to keep customized versions of the interface for each alt.

G.1 Chat Configuration Save and Load

Saving a carefully designed chat window setup is now trivial.

1. Set up your chat windows as you like them, down to the last detail, on any of your characters.
2. Save the chat window configuration. The `\chat_save` command will save the chat configuration in the default game folder, in the file `chat.txt`.
3. Load the new configuration into each character's interface with the `\chat_load` command.

From there, you can save tweaked versions of the chat layout for each alt, if you like (e.g., you might want different channels for a soloing scrapper and a teaming tank). Use `\chat_save_file MyAlt-chat.txt` and `\chat_load_file MyAlt-chat.txt` to save and load specific files from the default location.

Warning: It may be possible to directly edit the `chat.txt` file, but one look at it showed some cryptic components (like numeric strings that likely reference specific channels). All but the most advanced users are recommended to leave the file contents alone and do all chat configuration from within the user interface. I would not even save these files with other bind and option files – use the standard location.

G.2 Window Configuration Save and Load

Saving your individual preference for window layout and arrangement is now trivial, and will save tons of time and frustration as the game, your mistakes and temporary preferences mess things up.

1. Set up your user interface windows as you like them, down to the last detail, on any of your characters.
2. Save the window configuration. The `wdw_save` command will save the window configuration in the default game folder, in the file `wdw.txt`.
3. Load the new configuration into each character's interface with the `\wdw_load` command.

As above, from there you can set specific combinations of window layout for each alt. Use `\wdw_save_file MyAlt-windows.txt` and `\wdw_load_file MyAlt-windows.txt` to save and load specific files from the default location.

Warning: As with the chat files, the window configuration files are pretty cryptic. It may be possible for advanced users to directly edit the `wdw.txt` file, but all but the most advanced users are recommended to leave the file contents alone and do all window configuration from within the user interface. I would not even save these files with other bind and option files – use the standard location.

G.3 Option Configuration, Saving and Loading

Ah. Now the good stuff – the feature that lets you set any of several dozen game parameters, either individually or by loading a saved file. The Devs decided to call this grab bag "options."

Simply saving and loading option configuration files is the same as saving and loading bind, chat and window configurations. Let's summarize that quickly:

1. Set all of your options in the configuration menu, down to the last detail, on any of your characters.
2. Save the option configuration. The `\option_save` command will save the option configuration in the default game folder, in the file `options.txt`.
3. Load the new option configuration into each character's interface with the `\option_load` command.

Unlike the above config choices, the `option.txt` file seems to be readily editable, as the contents are merely the option keywords and the status or values. So there is a bit more value in saving the file—or files for each alt—in an accessible location. Use `\option_save_file .\GABB\MyChar-options.txt` to save a unique options file in the standard bindfiles folder, and `\option_load_file .\GABB\MyChar-options.txt` to load that file for an alt.

Setting Options

You can set options in the extensive panels of the Menu | Options menu. You can also set individual options via the slash command `\option_set`, which takes two arguments: the option keyword and the new value. For example, you can toggle on dirty word bleeping with the following command:

```
\option_set allowprofanity 0
```

And return to seeing every word your angry tank teammate wants to type by using:

```
\option_set allowprofanity 1
```

Even simpler, most options can be toggled from one state to the other using optiontoggle:

```
\option_toggle allowprofanity
```

will simply flip the setting from one state to the other.

There are a number of options, and option values, that can *only* be set using this process—they are either not in the menu list, or do not offer the full range of option settings.

Option Keywords

Ah, but you ask, what *are* the available option keywords? There are two simple ways to determine them, besides looking at the handy table below. The command `/option_list` will dump out a list of option keywords in the System chat tab. You can use `logchat` or `copychat` to capture the stream for offline examination. I discovered, though, that this list is incomplete. So the more thorough way to learn all the option options is to save an option file, as above, and open it for examination. This will tell the savvy player many things about his or her UI setup.

In the list below, nearly every keyword is self-explanatory and can be mapped to items in the Option menu. If not, ask in the forums or experiment! All of the keywords in blue are *believed to be* 0/1 toggle, on-off, and can be set using the `\option_toggle` command, or `\option_set 0|1`.

The keywords in orange use more complex numerical arguments, some in decimal values, some in float values and some in hexadecimal. Only a few of these have been sorted out, so there is much experimentation, archive searching and code digging to do for a complete explanation. (The two keywords to set the chat bubble color have proven particularly frustrating, as even putting the numbers from the saved file back in as arguments produce erratic results.)

Complete argument sets for two groups of commands follows the table, to give savvy users a head start.

Option Keywords		
AdvancedPetControls	AllowProfanity	ArchitectAutoSave
ArchitectBlockComment	ArchitectNav	ArchitectToolTips
AutoAcceptTeamLevelAbove	AutoAcceptTeamLevelBelow	AutoDeclineSuperGroupInvite
AutoDeclineTradeInvite	AutoFlipSuperPackCards	BlinkCertifications
BuffSettings *n*	CamFree	Chat1Fade
Chat2Fade	Chat3Fade	Chat4Fade
ChatBubbleColor1 *RRGGBBxx*	ChatBubbleColor2 *RRGGBBxx*	ChatDisablePetSay
ChatEnablePetTeamSay	ChatFade	CompassFade
ContactSort	CursorScale *0-2.0 float*	DeclineGifts
DeclineGiftsFromTeammates	DefaultChatFontSize *n*	DisableCameraShake
DisableDrag	DisableEmail	DisableLoadingTips
DisableMouseScroll	DoNotSeeEnemyLocal	EnableChatLog
EnableClickToMove	EnableJoystick	FadeExtraTrays
FriendSGEmailOnly	GmailFriendOnly	gShowPetBuffs
HideButtons	HideContactIcons	HideConvertConfirmPrompt
HideEnhancementFullMsg	HideFee	HideHeader
HideInspirationFullMsg	HidePetNames	HidePromptCoop
HidePromptDeleteEnhancement	HidePromptDeleteRecipe	HidePromptDeleteSalvage
HidePromptPlaceEnhancement	HidePromptUnslotEnhancement	HideRecipeFullMsg
HideSalvageFullMsg	HideSalvageWarning	HideStorePiecesState
HideUnclaimableCert	LogPrivateMessages	LoyaltyTreeAccessButton
MapOptionRevision *n*	MapOptions *hex*	MapOptions2 *hex*
MouseButtonReverse	MouseInvert	MousePitchSetting
MouseScrollSpeed *float*	MouseSpeed *float*	NewCertPrompt
NoXP	NoXPExemplar	OpenSalvageWarning
PreventPetIconDrag	PromptTeleportFromTeammates	RecipeHideMissingParts
RecipeHideMissingPartsBench	RecipeHideUnowned	RecipeHideUnownedBench
SeeEnemyBroadcast	ShowAllStoreBoosts	ShowArchetype *n*
ShowAssistReticles *n*	ShowBallons	ShowEnemyTells
ShowOwnerName	ShowPetControls	ShowPets
ShowPlayerBars *n*	ShowPlayerName	ShowPlayerRating *n*
ShowPlayerReticles *n*	ShowSupergroup *n*	ShowVillainBars *n*
ShowVillainName *n*	ShowVillainReticles *n*	SpeedTurn *float*
StaticColorsPerName	StoreAccessButton	TeamComplete
ToolTipDelaySec. *float*	UseOldTeamUI	UseToolTips
VoucherPrompt	WebHideBadges	WebHideFriends
WindowFade		

BuffSettings Argument List

The complex stacking arguments for the `BuffSettings` option, which controls what buff icons are shown on the player, pet and group buff display and how, was worked out back in the Live era and distributed on the various wikis and discussion groups.

Kudos to whomever worked them out and took the time to present them to the community; thanks to Adeon Hawkwood for bringing this list to my attention in the game.

One of the frustrating things about direct manipulation of the option values is that they are evidently based on hexadecimal numbering within the game code, but may be written to the options file in either hex or decimal. The `BuffSettings` control values are decimal equivalents of hex, but hex numbers cannot be used to set the option despite how simple that would be. So users who want to carefully control their buff settings, including some choices not allowed in the Options menu, have to use really big numbers to do it, and carefully add together these values to get their desired result.

A `BuffSettings` setting of 0 means all default buff icon display settings will be used.

There are three groups of buff-display settings, which work identically with different ranges of numbers. All of the numbers have to be combined into one value to set the display as the user wishes.

BuffSettings argument values			
Buff Icon Status	Player Display (Status Bars)	Teammate Display	Pet Display
`Hide Auto Buffs`	1	256	65536
`Hide Toggle Buffs`	2	512	131072
`Do Not Blink Icons`	4	1024	262144
`Do Not Stack Icons`	8	2048	524288
`Enable Numeric Stacking`	16	4096	1048576
`Hide Buff Numbers`	32	8192	2097152
`Stop Sending Buffs`	64	16384	4194304
`NOT USED`	128	32768	

To disable buff-icon stacking for the player, an argument of 4 would be used. To hide auto power buffs and toggle power buffs, an argument of 3 (1+2) would be used. The same method would be used for the Team set of values and the Pet set of values.

So, to hide auto and toggle powers for all three groups, you would add together the relevant values:

`1+2+256+512+65536+131072 = 197379`

And *carefully* enter that totaled value as the argument. Anyone with some programming experience can see how much easier this would be in hex!

And again, a value of 0 means default settings will be used.

Character Display Options & Argument List

The Options menu has a number of settings that allow you to control how you see other characters—players, villains and NPCs—and under what conditions. By default, some information is shown all the time, some is shown when the mouse pointer hovers on a character and some is shown when the character is selected (targeted). All of these combinations are configurable… and some configurations not allowed in the menu can be set by /setoption commands, mainly Villain element to "show always."

Note that all available information is shown in the Target window when any character is selected, so these options control only the heads-up UI display.

The essential settings for how you see other characters are:

Setting / Menu Item	Default	Menu Option Range	Gunner's Choice
ShowArchetype	2	All	2
ShowSupergroup	2	All	0
ShowPlayerName	1	All	1
ShowPlayerBars	6	All	6
ShowVillainName	6	0 2 4 6	1
ShowVillainBars	6	0 2 4 6	6
ShowPlayerReticles	6	0 2 4 6	4
ShowVillainReticles	6	0 2 4 6	4
Reticles cannot be set to 1, "show always"			

Each of these can be *independently* set to the following options:

Status	Menu Option	/setoption Value
Off / Not Displayed	Hidden	0
Always Displayed	Show Always	1
Display on Mouse-Over	Show on Mouse-Over	2
Display when Selected	Show when Selected	4
Display when Mouse-Over or Selected	Mouse-Over or Selected	6

The simple 3-digit binary nature of the values should be obvious; any value that sets the 1 bit will produce an "Always" display, and so forth.

My personal choice for the cleanest, most informative character display is given in the red numbers in the list above. Try them yourself, and edit as you like. There are more display options related to pets, as well.

Further Information

Between this start and examination of your own options files, it should be possible to sort out all the other numeric options—maybe even the chat bubble colors, someday. Try setting different values in the Options menu, saving the file and examining the value changes… and once you have all those mapped, try using other values to see what happens.

Note that some options do indeed write their values out in hex, and may want hex arguments.

Reports on any complete, verified findings and interesting results solicited, with credit!

Appendix T: Gunner's Targeting Secrets

Targeting in City of Heroes and City of Villains can be an extremely useful adjunct to your character's eyesight - a bionic eye to help spot those pesky glowies, bosses, hostages and friendlies across vast and confusing outdoor maps.

If you've played very long, you've gotten an outdoor mission that you had to search and search to find the objectives... and you haven't played much longer if you've run into one in which the objectives remain stubbornly hidden, usually as the clock ticks down and your patience frays.

Gunner to the rescue: Here's how to use the advanced targeting commands to make those hidden suckers come out and play. As well as streamline more common needs like finding and locking onto the right foe.

Basic Targeting Commands

Okay, you probably know the targeting that's been in the game since Issue 1:

- `target_enemy_near`
- `target_enemy_far`
- `target_enemy_next`
- `target_enemy_prev`

These commands, which take no arguments, will target any foe in your visible range (about 180 degrees wide and either at map limit or about 300 yards) who is, respectively, the closest, farthest, next farthest from the one currently targeted, or next closer from the current target. The first two will select only one target at any one time, while the second two will cycle through the visible foes, one in nearest to farthest order and the other the other way around.

You can do the same thing for friendlies:

- `target_friend_near`
- `target_friend_far`
- `target_friend_next`
- `target_friend_prev`

Which does the same thing as above for any player or NPC that shows a blue or green reticle.

None of these commands will let you target objects or NPCs with a white reticle.

Custom Targeting Commands

There are a variety of useful binds that can be written with the fixed commands, but they don't quite cover all the bases. So in Issue 4 or 5, the following custom targeting options were added:

- `target_custom_near`
- `target_custom_far`
- `target_custom_next`
- `target_custom_prev`

These commands work as described above with the exception that each requires one or more arguments to tell it what to target. The arguments are:

- `friend`
- `enemy`
- `mypet`
- `notmypet`
- `base`

- **notbase**
- **alive**
- **defeated**
- **teammate**
- **notteammate**

Some of these options are identical to the fixed targeting equivalents:

target_custom_near friend

is identical to

target_friend_near

target_custom_next enemy

is identical to

target_enemy_next

…and so forth. There isn't really any reason to use these longer commands in place of the fixed ones except individual preference. But you could eliminate your use of the older commands to use the more consistent and flexible custom commands all around.

The New(ish) Targeting Options

It's these argument keywords that add new functionality. (Okay, it was new a long time ago.) But the actual operation of these commands and their keywords is not straightforward. There is a hierarchy to the commands that is still muddy to me after much experimentation. Here, to the best of my knowledge, is an accurate description of the keyword functions:

- **friend** will restrict targeting to any blue-reticle (other player) or green-reticle (teammate) character.
- **enemy** will restrict targeting to any orange-reticle (foe) character.
- **teammate** will restrict targeting to any green-reticle (teammate) character, including both your and others' pets.
- **notteammate** will exclude all green-reticle characters from the targeting cycle.
- **mypet** will restrict targeting to any of your own pets.
- **notmypet** will exclude any of your own pets from the targeting cycle

You can further define what the above keywords will select with these two secondary keywords. Note that these keywords do not work reliably unless paired with one of the above primary keywords.

- **defeated** (or **notalive**) will restrict targeting to any figure, friend, enemy or NPC with zero hit points.
- **alive** will restrict targeting to any figure, friend, enemy or NPC with at least one hit point.

The other commands are... peculiar. Both `base` and `notbase` appear to function identically, for one thing, but what they do is open targeting to every single live object within view. Instead of being limited to live game elements like friends, foes and pets, using base allows you to target civilians, neutrals, NPCs, and even objects like doors, glowies and terminals.

Unfortunately, there doesn't seem to be any good way to make this selection selective; you either target all objects or none. But they can still be used to great, useful and even amusing effect.

Using Custom Targeting Commands

It is important to understand that the “custom” commands begin with an assumed "target all" and are restricted to smaller sets of targetable items by the various commands. This might seem trivial but it helps in understanding how the keywords interact and stack.

There HAS to be at least one keyword. These commands won’t work by themselves.

Argument Stacking

To find friends, enemies or pet, those keywords have to be included. Again, they duplicate other fixed commands, but the rest of the keywords can add new capabilities. You can add the `alive` or `defeated/notalive` keywords to make the targeting more selective. For example, a character with a rez power could make good use of a bind that targets a defeated teammate. There may be other reasons to select pets, or even living teammates only.

So, the custom targeting commands permit more than one argument to be stacked - such as:

```
target_custom_near friend alive
```

which will target only blue- and green-reticle characters who have at least one hit point.

```
target_custom_near friend alive
```

will target green or blue reticle figures who have 1 hit point or more.

```
target_custom_near enemy defeated
```

will target enemies who have zero hit points.

Actually, those examples are backwards from any useful ones, so let's flip them around:

```
target_custom_near teammate defeated
```

will target the nearest teammate who's defeated and needs rez or tp out of the battle.

```
target_custom_near enemy alive
```

will target only enemies who have not been defeated - which would be useful for a grapple bind written with the custom targeting commands, since there's no point in a scrapper locking on to a defeated foe.

String Targeting

The final argument that the custom targeting commands will accept strings - any character name or part of a name. Unfortunately, this won't work with foe group names or titles, so you can't seach for "Family" or "boss," for example... wouldn't THAT be nice! However, if you're on a hunt for specific types of enemy - such as that damnable hunt for Marcone Capos to get the Gangbuster badge - you can write a quick bind with the appropriate string and greatly simplify your hunting:

```
target_custom_next enemy capo
```

If you're searching for more than one exact character name, you'll have to analyze the spread of names for each foe type to see if there's a substring that will cover them all.

All of the following are valid binds:

```
targetcustomnext sorc
```
(Tsoo Sorcerers)

```
targetcustomnext outcast
```
(Any Outcast minion)

```
targetcustomnext lead  (Outcast Lieutenant or Boss)
```

...etc. Have fun. This is particularly useful for those with macro keyboards, where a whole slew of keys can be bound to specific searches. You'll likely need a cheat sheet to keep them straight, though.

Now let's put it together.

Advanced Targeting

Here's how to use the custom targeting command to simplify those damned hunting missions, whether they're kill-alls, hostage rescues, glowie hunts or any other mish that requires you to laboriously search the whole darned map.

Put this bind on a key you can whack almost continuously while manipulating the mouse and movement keys.:

```
/bind ADD "targetcustomnext base"
```

I specify (and use) the numpad plus key because I can whack it while my hand is still on the trackball. (Yeah, I use a trackball, what's it to ya?)

You will need to be able to move, control air movement and whack this key, so choose a key or mouse button that works for you.

Now, when you're in a map that requires searching, or hunting in a zone...

- (Optional) Drag your targeting window to the center of the screen, either just below or just above the center of view. (Optional, but helpful.)
- (Optional) Make the target window large enough to see easily with the

  ```
  /windowscale target 1.5
  ```

  command. Adjust the value from 1.0 to 3.0 to find a comfortable size. You might bind this and the return to normal size to a pair of minor keys for convenience.
- Get to a good central place (among obstacles) or a high place (through jump, teleport or flight).
- Spin slowly while whacking the targeting key. Watch the target window carefully. When you see your desired target, freeze and cycle the targeting slowly until you have it targeted.
- Do whatever heroic or villainous thing you must.
- Repeat steps 3 through 5 as necessary throughout the map.
- Substitute a character name string, as described above - and strings like "hostage" might work just fine as long as all the scared little guys have the same name - for very selective targeting.

Tips, Secret Things & A Conclusion

Some targeting is kind of subtle. Things you aren't supposed to be able to target will show odd text, and may not show a selection reticle. This includes clickable doors (which will show a white "Dr" in the target window but no reticle).

What's really odd is that that "Dr" appears sometimes on non-door items, often key bosses or figures in missions. The only way to find these objects is to hit your Follow key and let it drag you to the item, which can be dangerous if it's a purple boss. (Don't complain to me if the last thing your character sees is an Aberrant's ugly face.)

Appendix W: The Way-Cool Binds List

Over the years, I've found, learned and created a whole bunch of useful binds. This section is a compendium of those that many players might find useful. There is still more info on the Heroica! Web site that I might fold in here, if there's time and interest.

Each uses a specific key that maps to my preferences – you're of course free to use others.

For a completely new bindset updated to modern gameplay, see the companion GABB – Gunner's Advanced Basic Bindfile, the same place you got this guide. Only a few samples from it are in this list!

And for more, see the separate WCBL page on the website, which I am updating more regularly than this Appendix.

Contributions welcome and will be credited!

Enjoy… and *to victory!*

GENERAL BINDS

Boss!

```
/bind CTRL+F9 "quit$$dialog_yes"
```

Bang, you're at the desktop when YOUR boss walks in. Be sure your char is in a safe place, though…

While you're at it, add these:

```
/bind F9 "requestexitmission"
/bind CTRL+F9 "quittocharacterselect"
/bind ALT+F9 "quittologin"
```

The first bind exits you from a completed mission – a useful alternative to finding and clicking the teeny EXIT button. It also gives you a fast exit when you're just doing XP mop-up and the situation turns ugly. (A nice insurance key when you decide to see if you really can solo a purple Aberrant…)

Last two functions should be obvious. All will give you an abort time. You could eliminate the `$$dialog_yes` on the first one if you want an abort time for quitting to the desktop, as well.

Zoom!

Make faster travel easier and reduce endurance cost when necessary.

```
/bind R "powexec_toggleon Sprint$$++autorun"
/bind CTRL+R "powexec_toggleoff Sprint$$autorun 0"
/bind Space "+up$$autorun 0"
/bind MOUSECHORD "+up"
```

The first bind turns on Sprint and initiates autorun on the first keypress, and will toggle autorun off and back on with successive presses. (If you have Super Speed, substitute that power for Sprint.)

Second bind cancels speed power and autorun, independently.

Third bind cancels autorun while still providing a quick jump key. (Hitting `back` (S) will also halt autorun.)

The fourth bind gives jump action when both mouse keys are pressed. If you have initiated speed autorun with the first bind, you can steer and jump obstacles with just the mouse hand.

Beam Me… Over There, Scotty!

There are a number of binds that make Teleport powers much faster and easier to use. The most basic, which turns Teleport into a one-hand point-and-click travel power, is:

```
/bind LeftDoubleClick “powexecname Teleport”
```

…and travel with any succession of point-and-doubleclicks.

The companion bind is to make Recall Friend (Teleport Teammate) quick and easy:

```
/bind CTRL+LeftDoubleClick “powexecname Recall Friend”
```

This has a slight limitation in that the range limit for Recall Friend is quite short, and if you click at a point outside that range, you'll get a red targeting ring that requires repositioning and another click. So you could use this alternate:

```
/bind CTRL+LeftDoubleClick “powexec_location 0:20 Recall Friend”
```

…which will TP your teammate to a spot just in front of you, and the pointing action will be irrelevant. You could also bind this to a key.

To prevent confusion and allow the teleported one to opt out, try:

```
/macro TTM "g Teleporting $target!$$Recall Friend"
```

The string "Teleporting [teammate name] will appear on the chat and your targeting circle will appear. You can pause for the target to comment or decline before clicking to complete the action.

This is a bind to a macro button, but you can also bind it to a team-mode key.

Another useful and amusing bind for teleporter is this one:

```
/bind U “powexec_location up:max Teleport”
```

Punch U (or the key of your choice) and your alt teleports vertically at his or her maximum range. Useful to bounce quickly out of a bad combat situation, or to jump way high to start cross-zone travel past buildings, cliffs, etc.

I said FROG!

Super Jumper or other jump power? Use this set:

```
/bind J "powexec_toggleon Super Jump"
/bind K "powexec_toggleon Combat Jumping"
/bind CTRL+J "powexectoggleon Super Jump$$up 1$$autorun 1"
```

The first two binds give single-key start of jump powers. Since the powers are mutually exclusive, they will toggle each other. The third bind sets you jumping across the zone; you can steer with the mouse. Usethe SPACE bind above to cancel forward travel.

And while we're here, I bind Fly to the Y key for convenience:

```
/bind Y “powexecname Fly”
```

COMBAT & MELEE BINDS

Follow!

```
/bind F "follow"
```

A default bind, but worth mentioning here. Binds you onto the selected target, be it friend, foe or NPC. Non-melee types should be cautious with this key, or even rebind it to ALT+F so that you aren't accidentally yanked into melee range of a foe.

Engage!

```
/bind G "target_enemy_near$$follow"
```

The essential melee bind for tankers and scrappers - target the nearest enemy and lock onto him. Bind to G for “Grapple” or “Get 'em”" and keep F for Follow when you have the desired target already selected.

I.C.U.!

```
/bind T "target_enemy_near"
/bind CTRL+T “target_enemy_next”
```

An essential bind for all types - helps you find and target slightly hidden foes, even at a distance. The first bind finds only the closest foe; Repeated presses of the second one will cycle through all visible foes, from nearest to furthest.

Alternately, use this bind:

```
/bind CTRL+T "target_friend_next"
```

…which will cycle through all friendly alts, on your team or not. Or use this one:

```
/bind CTRL+T “target_custom_teammate”
```

…to cycle through teammates as targets.

QuickInsp

```
/bind F1 "inspexec_name resurgence$$inspexec_name dramatic improvement$$inspexec_name respite"
/bind F2 "inspexec_name second wind$$inspexec_name take a breather$$inspexec_name catch a breath"
/bind F3 "inspexec_name phenomenal luck$$inspexec_name good luck$$inspexec_name luck"
/bind F4 "inspexec_name righteous rage$$inspexec_name focused rage$$inspexec_name enrage"
/bind F5 "inspexec_name uncanny insight$$inspexec_name keen insight$$inspexec_name insight"
/bind F6 "inspexec_name robust$$inspexec_name rugged$$inspexec_name sturdy "
/bind F7 "inspexec_name escape$$inspexec_name emerge$$inspexec_name break free "
```

Each of these binds will fire off the selected Inspiration type, from lowest power to highest. VERY useful for Health and Endurance - I don't find the others as useful but you might. Some players might prefer to reverse the order of Insps so that the most powerful ones fire first. Rearrange the specific key bindings to suit yourself - but be sure to make the first two, and perhaps "break free," easy to find and hit fast.

SneakyZapp!

```
/bind CTRL+Z “target_enemy_near$$powexecname Thunderbolt$$follow”
```

This cutie will target the nearest enemy, trigger an attack power and then move in to strike… but *stop at the absolute maximum range point to fire the attack*.

It’s best used with Blaster, Defender and Corrupter ranged powers, and allows a fast, controlled attack with maximum safety. Instead of trying to figure out how close to get before firing, and possibly drawing aggro and return fire, this combo lets you slide in and attack in the most efficient way.

You can omit the targeting command if you want to choose your target ahead of time.

The best way to use this power is to target, activate… and then hit S-for-backwards as soon as the power fires, so you can dash back out of range and escape. There is, unfortunately, no way to add auto-runaway to the bind.

USER INTERFACE BINDS

Google (the) Map

```
/bind F12 "window_scale map 0.6"
/bind CTRL+F12 "window_scale map 2.0"
```

This bind set will let you zoom the map to huge (2x normal size) with one key, and back to a small, out of your way helper with another. Getting rid of the map goes to the easier key. Adjust the small value to your preference, and increase the large value up to 3.0 if you like.

NOTE: Works best if you park the map window in the upper left corner.

I'm Talkin' Here!

```
/bind ENTER "afk Hold on, I'm speaking to someone…$$beginchat"
```

This text will put you in the current chat entry dialog and put an AFK bubble over your head telling other players what you're doing. Different binds for current chat and tells can be used.

To control your tells more accurately, this bind pair differentiates beween the last tell you *received* and the last one you *sent*. That way, if you're going back and forth with one player, a random tell from someone else won't divert your comments.

```
/bind BACKSPACE "autoreply"
```

…starts a reply to the last tell you were sent.

```
/bind CTRL+BACKSPACE "tell_last"
```

…adds a reply to the last tell you sent someone.

```
/bind ALT+BACKSPACE "t $target, "
```

…opens a quick tell to any player you have targeted, like the one who just gave you a passing buff or heal.

You can add the AFK chat bubble to each of those, with different messages to those around you.

I'm Going, I'm Going!

When you stop to check in with contacts and then your mission list, you end up with one or both of those windows plus the contact-dialog window open, and it gets tedious to close them so you can get on to heroic or villainy. This bind slams all three closed so you can get to it:

```
/bind F11 "windowclose contact$$windowclose mission$$windowclose contactdialog"
```

HEALER/BUFFER BINDS

These bind sets are intended to put healing, buffing and general team-support commands on the keyboard numpad. You'll have to go from mouse+keyboard control to two-handed keyboard control in combat, but I've found it very workable. You may sometimes find yourself having to press these keys twice to select and then affect - if there's a consistent reaction from the console, I haven't found how to get it.

In each case, the number-pad 1 through 8 are bound to select and affect a specific teammate, and numpad 9 is the same action on the currently-selected 'mate. Two keys are bound to function the same in all shift-bindings.

I also strongly suggest that defenders and controllers detach the team-status window and drag it right to the center of the screen, at a height that lets you see the action but lets you watch your teammates' health and status at the same time.

The powers referenced here are for an Empathy defender - adjust the power names and bindings to suit other models.

This numpad-per-character scheme can also be adapted to Pet control for Masterminds.

Heal All

```
/bind numpad0 "powexec Healing Aura"
/bind shift+numpad0 "powexec Healing Aura"
/bind ctrl+numpad0 "powexec Healing Aura"
/bind alt+numpad0 "powexec Healing Aura"
```

This bind fires your basic area healing power no matter which shift key you might have pressed.

Heal One

```
/bind numpad1 "unselect$$teamselect 1$$powexec Heal Other"
                                ...
/bind numpad8 "unselect$$teamselect 8$$powexec Heal Other"
/bind numpad9 "powexec Heal Other"
/bind add "powexec Absorb Pain"
```

This bind fires your basic heal-other power at the specified teammate. Note that the number pad plus key is bound to the immediate "power heal" command, as it is in all shift settings.

Power Heal

```
/bind shift+numpad1 "unselect$$teamselect 1$$powexec Absorb Pain"
                                ...
/bind shift+numpad8 "unselect$$teamselect 8$$powexec Absorb Pain"
/bind shift+numpad9 "powexec Absorb Pain"
/bind shift+add "powexec Absorb Pain"
```

This bind fires your power-healing power at the specified teammate.

You can repeat this bind model for the CTRL and ALT keys for other useful team-related powers - Recall Friend (use the "announced" version above), Fortitude, Clear Mind, etc.

(Note: the ADD key is the numpad plus key.)

ROLLOVER BINDS

There are many reasons to have a command change with each execution – to alternate forms of a power, or an emote, or whatever. One method for having a single key execute a series of actions is found in section **3.2, Macros Using Tray Rollover**. That method is best for one or two fixed alternations – more than that can create too much complexity and a very fragile tray organization system. It is, however, unlimited by local and net speeds.

An alternate approach is to use keybinds that load overwriting binds. This method is effectively unlimited but might be hampered by slow local or network access. (That said, I've never had it fail for me.)

To use rollover binds for a command, start with a bind like this in your master load file:

```
/bind CTRL+G "l Gratz!"
```

Now every time a teammate levels, you can gratz them with a keyflick. And probably get tired of saying the same thing over and over, as will your team.

So do this instead:

```
/bind CTRL+G "l Gratz!$$bindloadfilesilent gratz02.txt"
```

What's in the file `gratz02.txt`, **which should be prefaced by any local path string needed**? This one line:

```
/bind CTRL+G "l Congratz!$$bindloadfilesilent gratz03.txt"
```

And in `gratz03.txt`?

```
/bind CTRL+G "l Gratz-a-roonie!$$bindloadfilesilent gratz04.txt"
```

…and so forth. Eventually, a file in this chain should point back to a gratz01.txt file that resets the bind to the loadfile original. There is effectively no limit to the number of iterations for this process, and it could be used for other fun stuff like random dances or emotes, or for serious rotation of powers and attacks.

Revision History

Note that letter suffixes will be used to distinguish very minor interim updates but not noted here.

0.50	18 Feb 2005	First release.
	(Seven Live-era updates omitted here.)	
2.00	13 April 2009	(Issues 9-14 update.) Wow, getting to this a little late. Fortunately, the command and emote base has stayed relatively stable over the last several releases. This will likely be the last major update of this guide; I hope you've found it useful!
2.50	12 May 2019	Issue 14+/post-Live server update, and damn glad to do it!
2.52	14 May 2019	Added three missing binds and powexec_location usage.
2.55	16 May 2019	Added cc_emote usage.
2.56	18 May 2019	Added /monitorattribute usage and some other material.
2.60	29 May 2019	Reformatted, rewrote and extended Appendix W. Updated Slash Command group listing.
2.61	4 June 2019	Updated email slash commands, added base-edit slash commands, other tweaks.
2.65	5 June 2019	Updated the key names and mouse action names list a whole bunch.
2.70	9 June 2019	Added the controller button keybind section, expanded button names section. Corrections and cleanup, especially to the /showtime bind. Why didn't anyone tell me the footer title hadn't been updated?
2.71	15 June 2019	Added /macro_image usage in Section 3.3, and the name list on the website.
2.72	21 June 2019	Updated pet emotes in bind usage (section E.1).
2.75	24 June 2019	Thoroughly updated Section G, options saving and loading.
2.76		Expanded option-set examples with character display features.

YOU HAVE REACHED THE GUIDE LEVEL CAP.
TURN AROUND.
NO INCARNATE LEVELS AHEAD.