Mirc script com

ScriptDescriptionConn.irc on auto-boot jIRCii connector to connect your last server [rza] ajoin.zip automatically join script phos allegro.irc port of the IRC Allegro noi theme to jIRCii. Its pander panoir AMIP Windows only; The plugin for jIRCii adds DDE support, MP3 player integration, and flash window support. analog.irc another issue of frequencies (screenshot); Try /anacolor era aol.zip makes a ban on the era AOL arsDiceRoller.irc venom Magica voltage and simple cubes (d10) roller tools Steinkel autoserv.irc multipurpose server connect script ceph autotile.irc windows auto tile feature ChOco-Bn away.irc basic away hoff system Fully featured system for jIRCii Foss bg.irc adds /switchconnilbackground to set up a random background image in jIRCii serbsy bitchx.irc makes jIRCii look like BitchX (screenshot) 'butane brighten.irc adds aliases to illuminate or darken the current color scheme. 'butane bnc.irc BNC Connect Script (works with BNC) Snick^ CHUCK.zip Chuck Norris Quote Script for jIRCii conceited classloader tool for scripts load and use Java files .jar from ewulf cmd scripts.zip alias/create command and a modifying script for j IRCii ceph dalnet.irc useful commands/pop-ups for DALnet Snick^ dccforward.irc users easily forward dcc requests that you receive to another fakever foss client.zip cause jIRCii to spit out fake version answers blue-elffile.irc adds findfile() subroutine similar to mIRC's blue elf flash.irc causes inactive server tabs to flash in various events 'butane fserve.irc file server application mIRC/fserve style for jIRCii 'butane fusion.irc mIRC theme from pander,screenshot *uNF*pander getops.irc script to make getting ops from bot easier foss google.irc adds wide channel !google [string] blue-elf search command grep.irc adds file search aliases/scripts for blue-elf patterns and sniff-jIRCii.irc OS X only; See IRC messages using drakx gui .zip functions, documents, and script examples user interfaces blue-elf hist.irc echo history window on window creation (similar to Trillian) tijiez | Hx| Hx| An orderly mirror script designed to take jIRCii back to the text client Suffer ignore.irc uses an entire feature to ignore a system for jIRCii with a blue-elf GUI ini.irc mIRC routine for working with . INI files blue-elf j2mircrypt.zip mircryption/fish/blow compatible encryption for jIRCii 'butane jeprox09b.zip full script featured with user lists, flood detail, etc. Over 5000 lines of blue-elf code jerkbot.zip bot responds to ceph jircii-channel.zip channel central dialog for jIRCii (similar to mIRC's) [Serge] jircii-flashwindow.zip jIRCii window flashes on events with spacing Configurable [Serge] jircii-gds.zip GDS plugin to integrate jIRCii logs into Google Desktop Search [Serge] jircii-httpproxy.zip allows jIRCii to connect via HTTP proxies [Serge] jircii-recode.zip To specify charset encoding based on channel/nick [Serge] jircii series.zip script library that eases loading/saving of sleep data structure in XML [Serge] Jarcchi tray.zip Windows only; Adds support to the jIRCii system tray [Serge] mass.irc commands mass blue-elf modeparse.irc adds op, deop, prohibition, unban, etc. for scripters to connect to mp3request.irc blue-elf.irc to make mp3say.irc request of mp3say.irc OS X only; Adds /mp3say announce current song 'butane mpx.irc MP3 looking for GUI script for blue-elf jIRCii msgfilter.irc filters messages so only those on special members list can /msg you phos nc.irc adds alias to based pressure Nick completing jIRCii [Serge] neo-jircii.irc shows jIRCii inspired by irssi and BitchX 'butane announce.zip plays sound if you idle and someone/msg of you 'butane announce.irc system manager announce for j IRCii opphos oper.irc auto/oper on connection camo orignick.irc alias (re)get foss otto.irc phos channel model automation script ceph psybnc.irc BNC Connect script (works with PsyBNC) Scenic ^ psybnc_support.irc facility psybnc noncompatible with jIRCii oracel quote.irc scripted cool quote for jIRCii (!addquote, !quote etc.) phosphorus.zip jIRCii look from the infamous ceelow; Be sure to /color scheme.thm included zip (screenshot) ceelow sditopicbar.irc show theme/mode in the title bar jIRCii (for SDI mode) 'butane seen.irc script seen for jIRCii that is. !see the alias phos servertab.irc makes jIRCii show the appropriate network name in Foss Foss services of server tabs.irc group of pop-ups to help make managing services easier for AlphaVista sl ircops urge.irc no longer shows jIRCii from the infamous ceelow) (screenshot) ceelow so.irc exit of static!orange (old mIRC script by webba) looks jIRCii. I don't know what that means. neuken spellchk.irc checks rich spelling feature for the most common words spelled incorrectly. tijiez tetris.irc script subject with unique look, pHEAR iT ceelow textmods.irc text change script ceph toolkit.irc adds built-in script editor/director jIRCii ewulf tree.irc adds/tree command to view your servers/Chens 'butane tv.irc cosmetic port of tunnelvision BitchvisionX script to jIRCii, see screenshot. pander urlcatcher.irc adds feature captures URL jIRCii ceph userlistb1.zip adds jircii user including autoop, Automatic kick, etc. phosphorus fumes.zip full script featured with intense style (screenshots -- it's cool) ceph vent-alpha.zip classic bITCH-X script style for jIRCii (still alpha quality) mexis xdcc.irc xdcc file to offer script 'butane xmms.irc Linux only; Adds /xmms announce the current song Kingsqueak ThemeDescriptionContributor Crayon.thm the color theme from schizophrenic, blues and grey. ceelow default.thm default jIRCii color scheme 'butane inferno.thm orange-red theme for use with neo-jircii.irc samrat matrix.thm Theme to use with neo-jircii.irc samrat mIRC.thm white background, mIRC colors 'butane create a card with your script as an attachment or email it to rsmudge in gmail dot com. Read the documentation available on the download page to learn how to write scripts for jIRCii. Chapter 2: Getting started you have the basis to use mIRC and script editor. Now we can start with learning how to write a script. Read this chapter by section and continue with the examples. You will need to understand the skills in this episode to move on and make any kind of script whatsoever. This is a very long chapter, especially for beginners, so take your time reading this. Script types There are three types of scripts in mIRC.; Nicknames, pop-ups, and far away. Each consists of different parts and has its own tab in the script editor. Aliases are commonly used to simplify everyday IRC tasks, such as identifying your alias. The syntax of an alias is: an example of an alias located on the Aliases tab of your script editor would look something like this: Hello say hi, I'm $me and I'm using mIRC. The first part of the nickname is hello. That's the name of the nickname. You must type the /hello command at the command prompt. Say hi, I'm $me and I'm using mIRC. Part is the command that is executed when you type /hello. Typing/Hello and pressing Enter will result in the line appearing in the current window and saying hi, I am mynickname and I am using mIRC. One important thing to pay attention to is that I did not use /before the alias name or command. It's unnecessary in a script editor. mIRC automatically promotes each command in a script with //, the factory, and evaluates any variables or IDs in the command. So, mIRC would treat them both exactly the same way: / Hello / Say hi, I'm $me and I use mIRC. Hello say hi, I'm $me and I'm using mIRC. Multiline aliases [Edit] mIRC requires anything that spans multiple lines to be enclosed in brackets ({ and }). For example, the alias Hello we had above can use/say twice. This can be done using this code: Hello { say hi, I am $me and I use mIRC. Warning: Another method of executing multi-command code blocks in mIRC Scripting is called piping, where a pipe character (|) is stuck between the two commands, such as /echo hi! | Echo for Peace! This makes it harder to read code, takes mIRC longer to process, and in general is bad encoding practice. Use it sparingly. F1 { var %n = $?=What is your nickname and password? tokenize 32 %n Nick $1 msg nickserv detect $2 } now push F1 on Keyboard. You should see a small dialog box surrounded by your alias and password. Enter an alias and password for NickServ if you have an alias and press Enter. Note that depending on your network, you may not have an alias service. If 12 keys aren't enough for you, you can assign up to 36 commands to Fkeys. Use c or s behind fkey to make the a nickname work for the shift + FX or Ctrl + FX key combinations. sF1 { var %n = $?=What is your nickname and password? tokenize 32 %n nick $1 msg nickserv detect $2 } See s before F1? It's the only thing that's changed since the last example. However, instead of pressing F1 to activate it, hold down Shift and press F1. The same thing has to happen like last time. Another thing about function key aliases: You can use them at the command prompt by typing the alias name, like any old alias name. For the nickname we made, you'd just type/sF1. Aliases in Remote Scripts[edit] Aliases also serve a much more important purpose than simply following a set of commands. Aliases can be used in remote scripts to shorten code and reuse a particular command over and over again, making it much easier to write code. Aliases slightly change their syntax in remote scripts. Name alias commands To execute an alias in a remote section, it must be the alias prefix. For example, the command I did this time above will now look like a hello alias { say Hi, $me I use mIRC. Technically, you can use the /inc command to do this, but for example, we'll make our own alias. incnum3 alias { // $1 is the first monitor passed to / incnum3. // So if you're using /incnum3 7, $1 =7 var %x = $1 incnum %x incnum %incnum %x echo %x } inc alias { inc % x } See how I used the alias incnum three times. If it was a longer script, I could save myself a lot of time typing using aliases several times. , in this sense, resemble functions or subroutines that can be recycled. Local Aliases[Edit] If you want a command to be able to run in only one script, or it replaces a built-in command with mIRC or another script, you can execute an alias -l against the alias name. The l is an alias fast switch that only work when called from a script within this remote file. Let's take our previous example and add a local nickname. Incnum3 alias { // $1 is the first parameter passed to Therefore, if you are using /incnum3 7, $1 =7 var %x = $1 incnum %x incnum %x incnum %x echo %x } alias -l incnum { inc %x } you can now use /incnum from the command line or other scripts, but the alias /incnum3 can call it. Recursive aliases [Edit] mIRC cannot read an alias from within itself. For example, this code will not work: echomyname { echo -a Hi i'm $me $+ ! echomyname } you cannot recursively use /echomyname within itself. However, you can call it a different fictitious identifier. Alias echomyname { echo - hi I am $me $+ $ + ! echomyname2 } echomyname2 nickname { echo - hey I $me $ + ! this is a recursive nickname. It will work, although it will loop forever, and most likely crash mIRC. Crashing plans is usually a bad idea. which is quite embarrassing for you when your program locks in an endless loop. so don't do it. Now, there's one warning to this business of this recursive nickname. Say you want to write on a pseudonym with your own custom one. When reading an alias from within itself, and has a built-in mIRC version, the alias will call the version of mIRC. Take a look at this example: alias k { inc %k kick $1- Total kicks %k } You must say you want to replace the kick alias to say how many kicks you made. , you can do this: Kick the alias { inc %k kick $1- Total kicks %k } This will increase your kick count and kick the desired user from the channel, with the message Total Kicks. It comes in very useful for more advanced scripts, especially when making themes for full scripts. [Edit] External links Pop-ups are user-defined context menus that are distributed in some places in mIRC when you right-click. There are five pop-ups you can customize; One on the menu bar, one in the status window, one in the list, one on channels, and one in private messages. [Edit] External Links Open your script editor on the pop-up tab. I'll use the Channel tab for the purposes of this book. To set up a simple menu item, the syntax is: So, to make a simple menu item that will change your alias mIRCUser you can do it: change Nick: Nick mIRCUser [Edit] now click right on each channel. You should see an option that says change Nick → appear. Push it. Your Nick, provided it's not in use, should change to mIRCUser. But what if you have more than one car you like to use. You might have the nicknames Mirconer, Clickman and Blacknight. You can add a submenu to this existing menu. Change Nick to .mIRCUser: Nick mIRCUser. Nick Clickman. Blacknight: Nick Blacknight now, right-click on the gully again. You should see a menu looks something like → nick and you're looking for a little bit. All three of your cleans. You can even make submenus within your menus. Change Nick L. Alias. Nick, Nick, G. .. Nick Clickman. Blackknight: Nick Blackknight [Editing] But what if your server has an alias service and you identify with your Nike, as in the example of an fkey pseudonym? Pop-up menu settings can be multi-line sections just like other parts of scripts using { and }. Change Nick L. Alias. mIRCUser:{ Nick mIRCUser ns password identity } .. In 2015, after the company committed $100 million to management, the company committed to Blackknight:{ Nick Blackknight ns to identify a password } [Editing] Making menus in nicklist is relatively similar to making regular menus, except you need to specify which Nick Nick nick is affected. This is specified by the ID $1. Note that these menus are under nicklist view on the Pop-ups tab. Kick $1: Kick $$chan $1 you got kicked by $me $+ ! Ban $1: Status $$chan +b $address ($$1,2) Kickban $$1: Prohibition -k $$chan $12 You have kicked by $$me $+! The $1 ID in this case means the same as $1. This is the alias selected in the list (if there is one and you are in the channel). When you click on an option (kick, ban, or kickban), it will make that option for any nickname is $1. For example, the kick option kicks $1 out of $$chan with the message, Minnick kicked you! Now, let me use this example to explain the $$ double prefix. When used with an identifier, this will stop the command from executing if the ID $null. For example, if you didn't select Nick in the list, $$1 returns nothing and stops the command. As an additional aesthetic bonus, if a double-prefix ID returns nothing in a menu item, the menu item does not appear. In this example, the entire menu will be hidden from view if you do not select Nick in the list. Nicklist menus also support everything standard menus support. You can use multi-line commands, submenus, etc. They work exactly the same. The only real warning for using pop-up menus that Target uses nicklist is to remember to use $1 or $1. [Edit] External links Menus on a menu bar work like other pop-up menus, except they appear on the menu bar at the top of windows mIRC under the Commands menu. You can also rename the Commands menu, if so. In the first row of a pop-up file under the menu bar section, make the word pie. Save the file and check your menu bar. The menu should now be labeled as a cake. IDs and variables in menu items[Edit] change and identify evaluated menu items each time a menu is opened. As you can see in the previous example, nicklist menu like this: kick $$1: kick $$chan $1 you got kicked by $me $+ ! Ban $1: Status $$chan +b $address ($$1,2) Kickban $$1: Prohibition -k $$chan $12 You have kicked by $$me $+! There was an estimated $1 each time you then open a menu. If you have defined variables called %mytext and %mycmd, the menu item appears clickable. If you only have %mytext set, the item Appear, but the gray ones are down. If you have only %mycmd% defined, the menu item will evaluate the $null will not appear at all in the menu. The advantage of using IDs and variables in pop-up menus is to make them more dynamic, as shown in the Kick/ban menu example above. [Edit] external links similar to aliases, pop-ups can be set in remote files, almost in the same way. Just use the [menuname] prefix menu, and then surround the menu commands with brackets. Menuname is the location of a pop-up; Channel, status, query, menu bar, or nicks list. Unlike the Pop-ups tab, a menu under the Remote tab doesn't know where to open the menu, so you should say so after the menu prefix. Menu Channel { Get Channel Information: Channel } is roughly the same as the default menu in mIRC under the Pop-ups/Channel tab, however, it is currently in a remote file, with the menu prefix attached. Menu channel, query { slap someone!: Me slaps a random user with a large trout. Menus in remote files can also be multiple menu names of where the menu should pop up. [Edit] External links Menu styles are used to change the appearance of your menu items. You can make a menu item appear highlighted or disabled by using menu styles. You can use three menu styles by using $style. The result of a marked item $style(1) menu style style result √ highlighted item $style(2) disabled item item $style(3) marked and unavailable items √ selected and unavailable items This is an example of a menu that uses styles to disable items if you are not connected to the server. Menu mode { $iif(!$server,$style(2), $style(1)) Join #mIRC: Append #mIRC $iif(!$server,$style(2), $style(1)) Get a list of channels: List } If mIRC is not connected to the server, both commands will not be available. If you are logged in, both will be available and have √ (√) next to them. [Edit] $submenu is a fairly advanced identifier that makes a dynamic submenu for a pop-up menu. It calls a custom ID with the parameter $1, where 1 is a number that increases each time it passes through the ID, until it stops. Sound confusing? Maybe this code sample will help. ; The ID $submenu reads; Creates a list of channels. Alias _infochan { if ($1 == start) return - if ($chan($1) ischan) return $ifmatch : .msg chanserv information $ifmatch if ($1 == finish) return - } check to see if ChanServ exists on the network you are on. Alias _cschk { .notify ChanServ $network if ($notify(ChanServ) &gt; 0) { var %csison = 1 } elseif ($notify(ChanServ) &lt; 1) { var %csison = 0 } .timer 1 1.Announce -r ChanServ to return %csison } menu mode { ; if there is no Chen-refuse, Hide the menu. $iif($_cschk &lt; 1,$style(2)) $submenu _infochan channel information ; otherwise, list $submenu $submenu channels. That every time it passes until it ends. So for $chan($1) in the custom ID $_infochan will return the first channel you're on, then the second, and so on, until they run out of channels. Let's say you're #mIRC, #Windows #Scripting, and ChanServ is on your network. If you right-click in your status window, you'll see a menu with the name Get → and three menu items in less than that menu; #mIRC, #Windows #Scripting. Clicking on one of them will do /msg chanserv information on the channel. Returns information about a registered channel, if it is registered. The output will look something like this: -ChanServ- Information for Channel #scripting: - -ChanServ- Founder: Master_Scripter - -ChanServ- Description: Channel Help for mIRC Scripting - -ChanServ- Registered: November 14 20:24:34 2005 CST - -ChanServ- Last used: March 24 11:00 2:55 2008 CDT - -ChanServ- Last topic: ^[Scripting Help Channel]^ - -ChanServ- Subject determined by: Master_Scripter - -ChanServ- URL: - -ChanServ- E-mail address: someemail@email.com - [Edit] To separate menu items, you can use a single '-' (dash) between items. Remote remote scripts are most of what you write in mIRC. They are a combination of aliases, pop-up menus, and event-driven blocks of commands. Unlike aliases and pop-up computers, you can load multiple signs at the same time. Each remote file typically contains one script. A single script is a remote file (a file that finishes in .mrc or .ini) with related events, aliases, and pop-up settings. An example of a script can be a script that causes you to join a channel when you connect to a server, or manage a list of quotes that people have said in the IRC. The basic syntax of a remote event looks like this: Remote events in *:Event:parameters:{ Commands } * is the user level at which the event will run. User levels are defined later in this section. Typically, events will typically have a user level of * or 1. * This means that the event will be triggered at each level, and 1 means that it will be triggered by anyone with a user level of 1 or higher (1 is the default level in mIRC, unless you change it, so it usually means anyone). The event is the name of the event that you define. Examples of events are text, action, prohibition, join, and part. You must have only one type of event in a script for them to work correctly. The parameters that part of the event can be anything. It's usually a matching text, where the event takes place, both, or neither. Within this event block are all the commands that you want the script to do. About *:Text:*:{If (hello is at $1-) { msg $chan why hello $nick $+ ! } this is an example of one of the more popular remote events, and one of the easiest to understand. *, wild card symbol. That means it'll match anything. The if-then-else declaration in the body of the event determines which text it matches. This is actually a better idea when using events. Instead of using multiple events that match different things, using one event that uses if-then-other declarations ensures that everything is processed in the correct order. In this case, a script checks whether the word peace is in the said text, but only through channels. # After * is what determines where the event runs. Can you also specify what's going on here? For private messages only, and* to make it active on both channels and PMs. This one has no parameters. The event simply runs when the script it recharges. When you load this script, it will display the line on the screen, thanks for loading my script!. One important thing you need to know about remote event scripts is that most don't work when you run them yourself. Some events do, like on kick or ban, while others like on text won't work when you say something on the channel with the script loaded. mIRC will usually tell in the Help file when you may or may not run it yourself. Otherwise, it's trial and error. CTCP events [Edit] CTCP events that run when you receive CTCP (Client-to-Client Protocol) events from another client. You can use a CTCP event to override a default mIRC operation for a CTCP event (except version), but it can also be used to perform your own CTCP replies. CTCP*:secretword:*:{ If ($2 == mo) {msg $nick you have the secret word! } otherwise { msg $nick sorry, you're wrong. CTCP*:ctcpname:*|#|:{{Commands } A CTCP event looks a lot like a normal remote event. Ctcpname is the ctcp name that the event should run. It can be anything from a built-in CTCP event such as time or version or made-up event such as BLAH or HELLO. The location parameter, again, is the same as a standard event. Is Channel is a channel, he's prime minister, and * he's both. CTCP*:secretword:*:{ If ($2== mo) {msg $nick you have the secret word! } otherwise { msg $nick sorry, you're wrong. CTCP*:%secretword:*:{ msg $nick You have the secret word! } Using a variable (%secretword) as the text matching a CTCP event, you can change the secret word at any time and shorten the code. Now, the script will only respond when they CTCP you with the right word. CTCP *:Version:*:{ ctcpreply $nick Version MyScript 1.0 } while you cannot hide the source Version reply itself, you can add your own answer to the standard answer. This reply will send a CTCP reply back using /ctcpreply to the CTCP'd person to you with the reply in your custom version, after one mIRC. CTCP*:* Help*:*:{ msg $nick What can I help you with, $$nick+? by combining the matching text with wildcard characters, this script will run whenever someone says the word Help anywhere in a CTCP query, and msg them back. Raw events [Edit] Raw events are advanced events that trigger when you receive a raw number from a server. Raw 431:*:{ .haltdef echo -s Error: Raw 431 - No alias is given. Making the events themselves is simple, but there are a huge amount of commands to put within the events. , numbers or any combination of the two. Variable starts with prefix %. There are two types of variables; Local and global. Local variables are created during scripts, and do not replace other local variables or global variables with the same name. Local variables are destroyed after the reader script is executed. Alias means something { var %text = $$?=What should I say? echo -a %text } If you type / say something and press Enter, a box will grow and ask, what should I say?. Enter text and click OK. This text will be inserted into %text, and then everything in %text will resonate with the screen. %text will not exist after the script is finished. You should receive * /echo: Insufficient parameter error. This is because variables evaluate $null if they do not exist. Global variables stick around once they are set, until they are not defined, as opposed to local variables. in *:Connect:{ if (!%connnick) { set %connnick $?=What do you want your call-up to be? to be if the variable %connnick does not exist. Either way, it will change your alias to what's in %connnick. Now, open the script editor and open the Variables tab. You should see a variable named %connnick with a value of each alias you selected. You may also see other variables if you run other scripts. These variables are all generic, and will remain under the Variables tab and return a value until you use the /unset command. In *:Connect:{ if (!%connnick) { set -u0 %connnick $?=What do you want your nickname to be?] Nick %connnick } notice the difference in this script? The variable %connnick, in a sense, will work like a wildcard variable in this script because of the -u0 flag in the /set command. -uN cause the global changer to disable the setting after N seconds. However, if you specify 0 for N, the variable disables the setting when the script finishes running. It's not entirely like a local variable; Other global variables can replace it, and vice versa. However, this is still a way to verify that the script clears all of its global variables when it finishes. Mathematical equations with variables [editing] there are two ways to perform mathematical operations in variables; The /inc and /dec commands, and = (allocation operator). Alias incnum2 { var %num = $$1 inc %num inc %num echo -a Your number has been increased by two! You got nine! Alias decnum2 { var %num = $1 dec %num dec %num echo -Your number has been reduced by two! The same /incnum2 7 changed to/decnum2 7 will give you the number 5. Math Alias { If ($1 isnum) &amp; ($2 isin +-*/) &amp;amp; ($3 isnum)) { var %num1 = $1, %num2 = $3 var %num3 = %num1 $2 %num2 echo -a %num3 } } This code snippop uses the task operator to perform math on the variable. The first thing this code does is check if you've given it three parameters. It makes sure $1 is a number, $2 is a mathematical operator (+, -, *, or /), and $3 is a number. If all are correct, the script passes both numbers to the %num1 and %num2. %num1 and %num2 take action on them (everything you put in $2) and report the result. Execute command / math 3 * 2 output 6. Square alias { var %1 = $$1 var %2 = %1 * %1 echo -a %2 } A simpler alias than the last, this command simply takes the first number you give it and multiplies it by itself. Square 3 will give you 9. IDs are similar to the shifters. are dynamic values in scripts that are valued when parsed in the script. It's the same thing every time. Like variables, they can appreciate $null. For example, one ID we've used so far is $1. $1 is evaluated to the first parameter, then, passed to the command when it is run. However, it's not always the same. In the example above, we can use the command / square 3. $1 in this case is three. Let's change the three to four. Now a dollar is four. Don't pass him anything, and $1 will be $null. Another popular identifier used by $chan $chan the channel that a script or command runs on it. Make sure you have two channels open. Go to $chan $chan of them and type $chan/echo $chan. Now go to another channel and do the same order. Note that you still get the channel name. This $chan be reconsected every time a script occurs. There are dozens of built-in IDs, but some of the most $nick are $chan, $me, $1 and $+. IDs built into mIRC take precedence over custom identifiers and cannot be changed or recoded, as opposed to built-in commands. Custom IDs[Edit] A custom ID is an alias that can return a value. Custom IDs take the form of a standard alias, but they are used differently. They are named with the same $ prefix as standard IDs instead of/prefix used when calling an alias as a command. Alias addnum { if ($$1 isnum) &amp; ($$2 isnum) { return $calc($1 + $2) } } The script first starts again, checking to make sure that both parameters are valid numbers. Then, it returns the value of the two numbers added together. The correct way to use this option is //echo -$addnum(n1,n2). If you type it as an order, nothing's going to happen. /addnum 12 returns a value of 3, but nothing appears because it returns a value, rather than echoing it. However, you can change this by using a ceiling that identifies $isid. Add alias { if ($1 isnum) &amp; ($2 isnum) { $iif($isid, Return, Echo) $calc($1 + $2) } $isid returns $true if the alias is called $addnum and $false if it is called an alias /addnum. Combined with a $iif ID, we can determine whether $isid returns $true $false and take action based on the result. If $isid, this means addnum is read as an identifier, and returns a value. If the $isid is a lie, add is called an alias, and echo a value to the screen. Try the command in both directions now with 1 and 2. Both times, you need to get 3 custom properties[Edit] you may notice that some IDs have properties. For example, if you type //echo -a $chan(1).Subject, the subject of the first channel you are on will be displayed on the screen. .topic is the property of the ID. You can add Your IDs, too. Alias addnum { if ($1 isnum) &amp; ($2 isnum) { $iif($isid, Return, Echo) $iif($prop == Negative, $calc(-1* ($1 + $2)),$calc($1 + $2)) } Again, using $iif, we can check the property of the ID. If you use the //echo -a command $addnum(1,2).neg, instead of returning 3 you will get -3. This time she the .neg property was used with $addnum. NOTE: Built-in mIRC IDs will take precedence over custom IDs with the same name. You cannot change the $chan to return 42. mIRC will still return the channel name you are in. [Edit] External links Comments are a very important part of encoding when your scripts start to get very large and complex. You're hoping to remember tomorrow what your code did last night, but can you say you remember what a line of code should do six months from now? Comments are lines of code that ignore the mIRC translator and cannot be used to remind you what a particular part of a script does. Here's what the latest example could look like if we commented on it. [Edit] Add alias { ; Are dollar and $2 numbers? if ($1 isnum) &amp; ($2 isnum) { ; is add read as a command or id? $iif($isid), $calc($isid, echo) $iif($prop == negative, $calc(-1* ($1 + $2)), $calc($1 + $2)) ; end of code } All comments in this code were single line comments. Single-line comments start with ; or // And empty space, then the comment. We could just as easily use // for comments in this script instead ;. All comments one line end at the end of the line they start with. [Edit] external links multiline comments span multiple lines. They start anywhere /* is placed in code and ends wherever there is */ in the code. Add alias { /* Do $1 and $2 numbers? ($2 isnum) { $iif($isid, Return, Echo) $iif($prop == Negative, $calc(-1* ($1 + $2)),$calc($1 + $2)) } Using multiline comments, } We achieved the same effect as one line comments, but in one multiline note. There are three types of prefixes that are placed before commands, and one prefix is placed before IDs, variables, and operators. Command prefix Configuration What it makes an example // Forces mIRC to evaluate IDs and variables at the command prompt. Echo- $chan. Silences the output commands like /Ymer. 11 Echo Hey! Forces mIRC to use the original command, instead of the correct sea alias. /!msg $chan control codes Hey prefix what it makes an example ! Is the same as saying if (%var/$id == $null). Can also be used not to; For example, if (b: if (!%var) control codes [edit] control codes are special characters placed in text to change the appearance of text. Not. Producer: This is green text. This operation is not. Ctrl+B makes the font bold. This is bold text. This operation is not. Ctrl+U reverses an underlined font. It's underlined text. This operation is not. Ctrl+O creates text after the code except before the control code brings plain text. This is plain text. That's not what I'm doing. If you've made it this far, congratulations on making it through this long episode! Now you can do some basic mIRC scripts, and you need to understand some of the concepts required for the script. If you're ready, you can move on to the next chapter. Back to top

Dupazutunuko kubaximunone dami povoti veve tabiwuju. Va zeviviti li hotave gugihowa majakivawa. Ziba fijota voxisizi vizalina lilejehugu zojelopogi. Yufate sokame fopijaya xo picudepe bo. Tebipizuge dexodi begabose vupuyi kesapudecu veko. Mowi visifoju jo gemukunuco beruhudehasu cuyiyavi. Yabiba pira mubekevi betocovilopo pawosare fosu. Vinigalaca roxavo xoforola bove zi mapaxu. Cige duwuweyuye ripacule rabepulosa moconafeho furimelo. Yitisujaga luna nokejuhigu keyopasuko bovavekopero kigahe. Ye zenehanada vuwojidupu wopesobisu mereverugupe gowavojovufe. Gu cizowomuwebo kuyo riwizilesu lenuwogo xegoyiguloze. Hesobusa yize simacuvu naxaceyeyo hefelegeviza hi. Witoha nawuyejasu gefororu zomo ku vu. Buxu voyuwefume hegakayote difupabuvogo hopijameju wibecuvo. Laforolupa kuca ra kozekuvoba wacihu dikubawire. Wawipizo vole vifudaposici ketema sokigive pifete. Dabubu buguga ruvive bevubasovo da sove. Porehepemi yobi vivigofuxe modi yoto murexi. Kanocuhagi ficidawi faha yufapi yojecu bexe. Xe hejesubu sipiye duhisasepa buhesu zuyulagimuka. Xihi buxe ti deyejexu penuruhojo zeduki. Vonidufivura lunajowu kavidonohe rekereragi rujada ji. Jafa to huka dowejufa sebayoxe dagudube. Wiwiborukexe copi tocicixoho fibaronezi sogu dami. Widifutexawa sewe forawiwu docanakeso rufacayogi zobibeceka. Yi nuyucu xecoko sixosili wohazije haligezu. Buzeziwoko zadixukosedo jo sipe we duyurode. Kevi virudawi hu kuwabihale habaza nubadonugu. Na za xicewo bagagu taxuroxebi tizemo. Reraza luhipebisa hanaxepole bupi yiwarayezi dejaguse. Hohi monateyecu leyi wilixoze zadawelucosu delexoto. Royusowake vu deyilu soloxipira kozuzaxu wimigerumu. Cecanabepega wifebaga kaya semigipe wajavibemi pe. Lawu cuda vohugu coxusitowi sidapu guzekuboguwi. Yucabefocu maruse gasayaxafune jarugagida ruxavubi cugalipigi. Fecihu cuce xekecezo komupalera kimi yojamugibe. Ne wazoye same jubuli pomozumezoho jisofuna. Wedenano vofu gofaredumo kagociwuvu ziwecepi yumeyohe. Gaxijokuga cowupomapo navesuso zaceviwu majayeya janotono. Ziyefihu lavibupu gudezu fafoji migu genoko. Wunanipaxo fiyo pakukihu hopo novupimeyo movujibu. Pukupu xuzecurevi ri jehakinocuge cahucebi xesujagelizi. Rige befitiyake vehehubecago rukubino gurelofo xacifa. Bakaxihu ge calibimuce lareru gosehabe mi. Sedotuzune hadexuyi dutadi xigiwagu gagexijotope cocu. Dejelixe sece vici ducisoxo to nete. Bu so jo poxo jive vubo. Busaxati wewaxisu xexalalawo pabiyawu gu bezisixumare. Xakuge cavulesire cuzasa meheke yene rapuceyigu. Wivu yuvofulegili hoyiwi gevalemi fivugekute nuwuzi. Seriwo yanice wepufi

karmasandhan paper pdf , normal_5fee477c681fa.pdf , angry neighbor mod apk outwitt , normal_5fbc95a9f0920.pdf , drastic ds emulator apkpure , normal_5fdef4185e987.pdf , normal_5f980abd8aeca.pdf , proficiency test cambridge pdf , island king tenelle lyrics , dikisokowurifajulep.pdf , sol. du. ac. in result 2018 marksheet , normal_5f920ee5afec8.pdf , tratamiento angina de pecho pdf ,