Building FAQ and general help

V3.1

18th January 2002
(shops updated Oct 9, 2016)

Reasons for the FAQ

In the days before the DoN website was updated, there was an online guide to some of the functions of Deadedit, the primary building tool for Dead of Night.
Now that the website has been updated and altered significantly, that online guide has been removed. This document contains advice and information about every dialog box involved in building an area using Deadedit as well as information about various optional “tricks” and ideas to add functionality to your areas.

Most of the information applies to both version 0.23 and 0.25Beta but some functionality and bug fixes can only be found in the beta version. It is in the interest of the builder community that they download the beta executable and use that rather than the 0.23 version of Deadedit.

If you have questions about any part of building that you cannot find the answers to within this FAQ, you can email questions to builder_q@dead.greenwing.com or you can contact the builder Immortals using the PAGE command on Dead of Night.

I hope that you find this document both informative and useful.

 January 2002

Rooms

Firstly, to add a room to a new area file, either left click on rooms in the left hand window and then right click, choosing ADD.
Or, click and drag outside any existing rooms in the map window and release after a short distance.

Both of these create a blank room with a terrain type of inside.

The most common mistakes with Rooms found on checking are spelling mistakes, grammar mistakes, incorrect room flags and procedures, incorrectly set up exits, "telling" the player what they do or feel rather than describing what they can sense, and setting strange terrain types.

With spelling and grammar, all you can really do is to be very careful when typing out a room description. Some people use another text editor, with a spellchecker and then copy and paste across into deadedit.
Make sure that the room name is capitalized other than small words such as "of" and so on (e.g. The Central Square of Pangaea)

Room descriptions are one of the first things we look at when checking an Area, and if they are inventive, colourful and generally interesting, your area is more likely to be implemented. Descriptions should have 2 spaces at the start of new paragraphs and after every period.

Terrain types set certain things such as if Druidic spells can be used there (forest), if you need FLY to get into the room (air, water swim and water noswim), if you need WATERBREATH to get into the room (underwater), or if you need a boat to get into the room (water noswim and water swim).

Water noswim is a special terrain type as it represents a river that carries along the player, like a teleport. The time delay and direction for this flow is set in the RIVER box and the numerical box to the right of this in the main room dialog. The time delay for teleports and rivers is in ¼ second intervals. So a number of 8 is a 2 second delay

All rooms, when first created begin with a terrain type of inside. Many people make the small mistake of thinking this will be a building that can stop the sun's rays (important for vampires). This actually won't do that. To set a room to be a haven for vampires and so stop the external light conditions from affecting the light conditions inside the room, you need to set a room flag.

Room flags are often not understood properly, and sometimes misused.

DARK	light level set to –100 (i.e. pitch black)
DEATH	Kills anything that enters the room
!MOB	Mobiles are unable to enter the room
INDOORS	Room is not affected by external light conditions i.e. day and night
PEACEFUL	No violent actions are allowed, including spells and skills
!STEAL	Steal doesn't work
!OUT	Cannot Astral walk, Nature walk or Gate out of the room
!MAGIC	No magic or magic like skill will work
TUNNEL	Only a certain number of mobiles and players can be in the room at any one time. Number allowed set in the MOB LIMIT box in the main Rooms dialog
!IN	Astral walk, Nature walk and Gate cannot be opened into this room
SILENCE	No noise, including scroll reading or spell casting can be made
!PUSH	Push doesn't work
IMMORTAL_RM	Mortals cannot enter this room. Imms only ☺
GOD_RM	High level Imms only. Note – do not use this or the previous flag unless given specific permission.
!RECALL	Recall scroll, Word of recall, or similar will not work in this room
HOSPITAL	Healing and regen rates quadrupled in this room

LARGE	Some Mobiles have the HUGE flag (see later). This room is big enough for them to fit in
!DEATH	Anyone killed in this room reappears in their inn without losing a level or items
SACRED	Room is permanently consecrated
CIVILIZED	No undead will rise in the room and if a Pkill or Psteal happens in this room, the bounty hunters will search for the perpetrator
!SUMMON	Summoning abilities do not work in this room. I.e. Monsum, Ghost etc

Teleports are often misunderstood as well.
These move everything in the room to another room in the MUD after a set time interval.
By setting the flags properly, these elements can be controlled.
Clicking on the blank box next to TELEPORT brings up 4 choices for flags to be applied.

LOOK	This automatically forces the player to LOOK at the room they arrive at.
COUNT	This counts the time set in the main room dialog before teleporting the room's contents
RANDOM	This sets a random time before the room teleports.
SPIN	Do not use unless very experienced.

In the main room dialog, there are 3 numerical boxes involved in teleports. ROOM is the destination room's number, TIME is the count that the teleport uses and DELAY is the time delay before the TIME count or RANDOM count starts.

As with Rivers, Teleports use ¼ second intervals for the TIME and DELAY boxes. I.e. 8 = 2 seconds

Room exits can take a long time to set up and an increased length of time to check, so it's important that they are set up correctly.
Left clicking on <<NEW>> in the EXITS box, opens the exit dialog.
In this, you can set the direction the exit goes in, a description of the exit (if required and should follow the guidelines for room descriptions), which room number the exit goes to, and the OBJECT number of a key needed for the exit (if any).

The flags on exits are very important and will make a huge difference to the mobility of people around the Area.

IS-DOOR	The exit is a door (many people forget to set this) and must be set up in the Zone section of the file (see later)
SECRET	The exit is not visible to mortals without the use of a skill.
PICKPROOF	Cannot be opened using Pick, Bash or Knock
CLIMB	Door can be climbed over. I.e. a gate

Deadedit versions 0.25B5 and earlier allow you to set other exit flags. These should not be used and should be set in the Zone section using D lines (see later)

All exit flags should be the same on each side of the door, with the exception of SECRET. Most secret doors are only hidden on one side.

A rapid method of adding exits between two rooms can be done using the map.
Press and hold CTRL and click and drag from inside one room to another on the map screen.
This generates a simple two way exit between those two rooms, and pops up an exit dialog box, in which you can set the flags necessary.

Descriptions are additional things in the room that players can type <look XXX> and will get a description. These things should be noted or hinted at in the room description and include signs in

shops. This is a way of giving additional information to the player that would not be normally readily available, but without using an OBJECT.
Clicking on <<NEW>> gives a dialog with Keywords and the description. Keywords are the XXX that the player should type to see the description and can be several words.
EG poster sign paper for a paper poster on the wall of a shop

Procedures should be avoided unless the builder is experienced and gets permission to use them as they include several special effects that are very rare.

Mobiles

Mobiles are the various non-player creatures that inhabit the Realms of DoN.
Left clicking on Mobiles in the left hand menu in the main Deadedit screen and then right clicking and choosing add, creates a blank mobile.
Double clicking on this allows you to change the various fields to create the creature required.

Name is the box that defines the mobile's keywords. These are the words that are used when the mobile is the target of a command, such as LOOK or an attack of some sort.
This box should contain no punctuation marks at all and should not have any "small operator" words such as "a", "the", "and" etc. It should, however, include all nouns and adjectives which appear in the "Act" and "View" fields.

Act is the line that is used by the game whenever the mobile does something. This should not have a period at the end of it as the game will add that in as necessary.

View is what everyone in the room will see as they enter the room. E.g. "a small, unkempt man sits here, rocking back and forth."
This line should be fully punctuated to the best of your abilities.

Description is the box that shows what a character will see when they LOOK at the mobile.
Every mobile must have a description, without exception. But there is a great deal of flexibility to what it can say. As with room description, this box must be fully spell checked, punctuated and checked for grammar.

The next two boxes set the gender and the race of the mobile
Gender for a mobile is not very important but should be chosen sensibly to be self-consistent with the description of the mobile. E.g. a noble woman should not have the gender male.
Race has some effect on what the mobile has I the way of special abilities, but not a great deal.
Note Elven mobiles DO have immunities to charm and sleep and infravision.
Undead mobiles crumble into dust on death.

Position and **Default** set how the mobile stands in the room and should both be set to the same value.

Level is an obvious box, and sets the values of the following boxes -
Speed, THAC0, Damage, AC and HPs.
Once a level has been chosen for the mobiles, press the **Standardise** button at the bottom of the dialog. This sets the boxes mentioned to appropriate values.
Note. All mobs will have standardise run on testing unless you specifically request otherwise but in the case you don't want specific details, please standardise before sending so as to avoid it being missed.

Align is a numerical value from –1000 to +1000 that sets it's alignment.
-1000 to –350 is evil and the mobile will have a dark aura.
-350 to +350 is neutral and the mobile will have no aura.
+350 to +1000 is good and the mobile will have a light aura.
Note Paladin mobiles cannot use all of their abilities if their alignment is not good.

Gold is the amount of gold that a mobile carries.
This value must not be any greater than 30000, and should usually be less than 20000.
The maximum value should be reserved for rare, tough creatures that have a reason for that amount of gold. E.g. dragons

Acts set various flags that determine how the mobile will behave in the presence of player characters. These flags are as follows.

SENTINAL	The mobile stays in the room it is in and does not wander.
SCAVENGER	The mobile will attempt to pick up anything on the floor.
NICE THIEF	The mobile will only shout over rumor if stolen from, rather than attacking.
AGGRESSIVE	The mobile will attack any non-polymorphed character that it can see.
STAY-ZONE	The mobile will not leave the zone it loads in. All mobiles without the Sentinal flag should have this flag set unless there is a very specific reason for it.
WIMPY	The mobile will attempt to flee if damaged to a third of its HP
ANNOYING	The mobile will be attacked by other mobiles in preference to other targets.
HATEFUL	Do not use this Act flag. Adding hatred to mobiles should be done with the H line in the Zone section.
AFRAID	Do not use this Act flag. Adding fear to mobiles should be done with the F line in the zone section.
IMMORTAL	The mobile cannot be killed by mortals. Only Immortals can harm it.
DEADLY	Do not use this Act flag.
POLYMORPHED	Do not use this Act flag.
META_AGG	Will attack any player characters or mobiles it sees, polymorphed or not. Will not attack any player characters or mobiles of the same race NB Needs the AGGRESSIVE flag setting as well.
GUARDING	Do not use this Act flag. To get a mobile to guard its master, see the description of C lines in the Zone section
LIQUID	The mobile is a liquid, like a shifter under the MELT skill, and so can move under doors.
STEED	The mobile can be ridden by the use of the RIDE skill.
HUGE	The mobile is very large, and can only fit into outdoors rooms or inside LARGE flagged rooms

Affects are a set of flags that are commonly misunderstood. They are skills or spells with are in effect on the mobile.
They should be chosen sensibly and with reasons and there are a couple of guidelines.
Do not give a mobile the flag of LIFE PROTECTION unless you obtain permission first.
Also, only Psi mobiles should be allowed the flag GREAT SIGHT without good reason.

Many builders have selected the flags POISON or PARALYSIS thinking that means that the mobile will use those spells in combat. The mobiles end up poisoned, or paralysed themselves.

Immunities, Resistances and Susceptibilities are a set of flags that should be chosen with care.
A creature immune to an attack form will take no damage what so ever from that attack.
A resistant mobile will take half damage and a susceptible mobile will take double damage.

These flags should only be used rarely and only if there is a good reason for them.
E.g. most undead and many supernatural creatures are immune to charm, sleep and/or hold but most natural creatures will not be.
One Immunity used a fair amount is that to SHIFT. This simply prevents a shifter from taking on the form of this mobile.

Sound is a message that will be sent in the room that the mobile is to everyone else in the room.
E.g. A bard singing.
Distant is what people in the room directly connected to the room the mobile is in will hear.
E.g. for our bard, it could be the faint sounds of singing.

Both of these are currently disabled in Dead of Night, so should be avoided.

Procs are procedures that further set what skills, spells, and behaviour a mobile will use.
They include generic_class flags that simply make the mobile a member of that particular class, Guild master flags, XXXXcaster flags that allow the mobile to only use spells of the type XXXX (e.g. fire caster), various elemental flags, and a whole host of others.

Certain procedures of note include

Air, fire, water and earth Elemental	Mobile casts the respective elemental spell with every attack it makes.
Bounty_Hunter	Mobile will hunt Pkillers and Pstealers if their crime is performed in civilized rooms. Will also hunt anyone deemed "Open Season" by the Immortals.
BreathWeapon	Mobile uses a breath weapon if attacked by more than 1 opponent
Death Knell	Mobile sends out a SHOUT message telling who killed it.
Janitor	Mobile cleans up the rooms it is in.
Receptionist	Mobile can be used to rent at.

Options to add to the Receptionist proc to control how the receptionist behaves are

-normal=<num>	percentage rent on non-rare items defaults to 0 - feel free to make this higher.
-limited=<num>	percentage rent on rare items - defaults to 45 below 25 not allowed anywhere 25 only for clan halls close to 25 should be out of the way - not near healing & hunting 35+ is reasonable in most places.
-limit=<num>	number of items allowed to rent - defaults to 50 - above 50 highly likely to be disallowed - lower is fine
-home=<vnum>	specify room vnum for where the receptionist will work - defaults to 0 which will set it to where the receptionist loads.
-guild=<name>	the number of the guild allowed to rent.
-builder	this flag means that only builders can rent here. Do Not Use

Produces	The mob with this proc produces items to its inventory. This should generally be used only for shopkeepers.
-obj=<num>	Object vnum to be produced. REQUIRED
-home=<num>	Room vnum that the mob must be in to produce the item. 0 - mob will produce it no matter what room it's in Default: 0
-max=<num>	Will only produce more of the object if it has this many or less in it's inventory. Default: 0
-rate=<num>	The number of pulses between reproduction of the item if the object is below its load limit. There are 1800 pulses per hour. Default: 3600
-limit_rate=<num>	The number of pulses between new items being produced if the object is above its load limit. There are 1800 pulses per mud hour. Default=7200
-starts_with=<num>	The mob will start with this many of the object in its inventory. Default: 0

Shop_keeper Mobile is a shopkeeper.

Options to add to the shop_keeper proc to control how the shop operates:

<num> is an integer

<real> is a decimal number - ie. 1.1

<msg> is text, and MUST begin with %s and include quotes (“”)

<bitvector> is a number, which is the sum of the number for each object type

Object types are:

Light	1		Scroll	2	
Wand	4		Staff	8	
Weapon	16		Fireweapon	32	ie. bows
Missle	64	ie. arrows	Treasure	128	
Armor	256		Potion	512	
Worn	1,024		Other	2,048	
Trash	4,096		Trap	8,192	
Container	16,384		Note	32,768	
Drink Container	65,536		Key	131,072	
Food	262,144		Money	524,288	
Pen	1,048,576		Boat	2,097,152	
Audio	4,194,304		Board	8,388,608	
Instrument	16,777,216		Swimmers	33,554,432	

Default: if the argument is included, the mud treats it like the argument was included as specified

-guild=<guildname> will deal only with members of guild <num>
Default: NULL [not in a guild]

-not_in_guild_msg=<msg>
what the shopkeepers will tell people not in the guild specified by the -guild option
Default: “%s I can’t seem to find your membership!”

-shop_selling_margin=<real>
the multiplier of cost used for player selling
Default: 0.1

-shop_buying_margin=<real>
the multiplier of cost used for player buying
Default: 1.1

-keeper_no_item_msg=<msg>
the message sent if you try to buy something the shopkeeper doesn’t have.
Default: “%s I just ain’t got that.”

-player_no_item_msg=<msg>
the message sent if you try to sell something that you don’t have
Default: %s You can't sell what you ain't got.”

-keeper_no_cash_msg=<msg>
the message if the shopkeeper can’t afford to buy the item
Default: “%s Sorry, but I just can't afford it right now.”

-player_no_cash_msg=<msg>
the message if the player can’t afford to buy the item
Default: “%s Be nice if you had the cash, freak.”

-do_not_buy_msg=<msg>
the message if the shopkeeper doesn’t buy that item type
Default: “%s Try somebody who cares.”

-player_buy_msg=<msg>
the message if the player buys an item
Default: “%s That'll be %d gold.”

-keeper_buy_msg=<msg>
the message of the shopkeeper buys an item
Default: “%s Here's %d gold for your trouble.”

-closed_msg=<msg>
the message if the shop is closed
Default: “%s Come back later.”
-buy_no_arg_msg=<msg>
the message if ‘buy’ is used with no argument
Default: “%s What do you want to buy?”
-sell_no_arg_msg=<msg>
the message if ‘sell’ is used with no argument
Default: “%s What do you want to sell?”
-value_no_arg_msg=<msg>
the message if ‘value’ is used with no argument
Default: “%s What do you want me to valuate?”
-player_item_limit=<msg>
the message if the player buys but cannot carry the item(s)
Default: “% s Your hands are full already.”
-player_weight_limit=<msg>
the message if the player buys but cannot carry the weight of the item(s)
Default: “% s You’re burdened already.”
-player_gold_limit=<msg>
the message if the player sells but could not carry more gold
Default: “%s You wouldn’t be able to carry the gold I give you.”
-not_home_msg=<msg>
the message if the shopkeeper isn’t at the <-home=<num>>
Default: “%s If only I was at my shop I might be able to help.”
-list_msg=<msg> the message begining the list of items the shopkeeper has
Default: “%s Here’s a list of what I’ve got.”
-value_msg=<msg> the message when the shopkeeper values an item

NOTE: “%d” is the placeholder for the number of coins, and MUST be included in the message

Default: “%s I’ll give you %d coins for that!”

-closed1=<num> Options for what time the shop opens and closes.
-closed2=<num>
-open1=<num>
-open2=<num> A shop open during the day needs only used open1 for the hour it opens, and closed1 for the hour it closes. Should be 0 to 23.
open1=8, closed1=20
A shop which is open 8pm to 8am:
open1=0, closed1=8, open2=20, closed2=24
Defaults: open1=8, open2=0, closed1=20, closed2=0
-trades=<bitvector> The types of objects the shop buys and sells.
Default: 0 (everything)
-temper_no_cash=<num> 0 - the shopkeeper pukes on the buyer who couldn’t afford to buy
1 - the shopkeeper grins happily when the buyer couldn’t afford it
<other> - the shopkeeper does nothing else
Default: 0
-home=<num> The shopkeeper’s home.
0 - the shopkeeper will work in ANY room
<room number> - the shopkeeper will work ONLY in this room
Default: 0
-deals_with_npc=<num> Whether the shopkeeper will deal with poly, tree, shift, etc.
0 - no, 1 - yes
Default: 1
-no_npc_msg=<msg> The message when the shopkeeper won’t deal with a poly, shift, tree, etc.
Default: “%s I don’t deal with your kind.”

Objects

Everything in DoN that is not a mobile, a player, or a description in a room is an **object**.
This part of DoN causes a great deal of problems but hopefully this section will help to explain it so that more and more great items are added to the already huge number in the game.

Like mobiles, making a new object is done by firstly adding a new object in the same manner as rooms or mobiles.
Double clicking on this unnamed object brings up a dialog box.

Name is the box for the keywords of the object. As with mobiles, this must not have an punctuation or "small" words in it. It should, however, have all nouns and adjectives which appear in the "Act" and "View" fields.

Act is what the object looks like when it is in an inventory or equipped by a character or mobile and should not end with a period. The game will add this afterwards.

View is what the object looks like when on the ground, and should be fully punctuated.

The **type** of the object is probably the most important part of the dialog, but we will skip this and return to it in a while

The **weight** of an object is simply the mass of it, and everything needs to have a value in here. Very small items may have a 0, but anything much bigger than a scroll (weight 1) should have a positive value. A long sword is roughly a 15 – 20 whilst a dagger would be roughly a 5.

The **cost** of an object determines its value in regards to selling and buying at a shop.

The **rent** of an object is how much you would be charged per day at an inn with 100% rent if the item has the RARE flag (see later). As most inns have a rent of 35%, this value may seem extraordinarily large.

The **level** of an item sets who may use it. Anyone with a level equal to or greater than the level of the item MINUS 5 may pick up and use the item as long as other conditions set allow it.

The 4 **value** boxes work with the Type box to define the basic properties of the item.
What each value means for each object type can be found in the following table.

Item Type	Value 0	Value 1	Value 2	Value 3
LIGHT	0	0	0	0
SCROLL	Caster Level	Spell 1 number	Spell 2 number	Spell 3 number
WAND	Caster Level	Max Charges	Charges	Spell number
STAFF	Caster Level	Max Charges	Charges	Spell number
WEAPON	0	Die number	Die Size	**Dam Type **
FIREWEAPON	Min Str	Max Range (in rooms)	0	Missile Type (must match that of the missile it fires)
MISSILE	% Breakage	Die number	Die Size	Missile Type
INSTRUMENT	***Type***	Hit points	Max hit points	0
TREASURE	0	0	0	0
ARMOR	AC-Apply (a positive number)	Max-Apply (a positive number)	0	0
POTION	Caster Level	Spell 1 number	Spell 2 number	Spell 3 number
WORN	0	0	0	0
OTHER	0	0	0	0
TRASH	0	0	0	0
TRAP	***Flags***	Spell number	Caster Level	Charges
CONTAINER	Max Content (amount of items)	***Flags***	Key number (if locked, else 0)	0
NOTE	0	0	0	0
DRINKCON	Max Content	Contents	***Type***	Poisoned? (0=no 1=yes)
KEY	0	0	0	0
FOOD	Amount of food (up to 20)	0	0	0
MONEY	Coin value	0	0	0
PEN	0	0	0	0
BOAT	0	0	0	0
AUDIO	DO NOT USE THIS ITEM TYPE			
BOARD	0	0	0	0

With this table, several values have been framed by * signs. These have sub tables to explain what goes in them.

Weapon Damage Type should be one of the following.

0 smite
1 stab
2 whip
3 slash
4 smash
5 cleave
6 crush
7 pound
8 claw
9 bite
10 sting
11 pierce
13 spear
14 punch

Instrument type should be one of the following

0	Do not use
1	Woodwind
2	Stringed
3	Percussion
4	Percussion strings e.g. piano

Trap flags determine under what circumstances the trap goes off.
Add up the values of the conditions required and enter in the box

1	MOVE	Trap goes off if character moves in a certain direction.
2	GET	Trap goes off if it is the target of a GET or PUT command.
4	ROOM	Trap affects everything in the room.
8	NORTH	Combined with a MOVE, these set the direction a MOVE will
16	EAST	activate from.
32	SOUTH	
64	WEST	
128	UP	
256	DOWN	

Container flags determine what its natural state is. If closed or locked, container needs the closable state as well.
Add up the values and enter in the box

0	UNCLOSABLE	Cannot be closed.
1	CLOSABLE	Can be closed
2	PICKPROOF	Cannot be opened except with it's key.
4	CLOSED	Object is naturally closed.
8	LOCKED	Object is naturally locked.

Drink type is one of the following and determines what effect the drink has on the imbiber.

0	Water
1	Beer
2	Wine
3	Ale
4	Dark ale
5	Whiskey
6	Lemonade
7	Fire breather
8	Local speciality
9	Slime
10	Milk
11	Tea
12	Coffee
13	Blood
14	Salt water
15	Cola
16	Vodka

With scrolls and potions, the keywords should be scroll or potion and then the keywords of the spells it contains.

The spell numbers to use in the Value boxes can be found by typing ALLSPELLS in DoN. This lists all of the spells and skills in the game along with their spell number.

Flags help determine the base properties of the item, as in its composition, and some things about it's nature.

GLOW	Item gives off light when used and reduces HIDE chances.
HUM	Item gives off a hum when used and reduces HIDE chances.
METAL	Item is made of metal.
MINERAL	Item is made of a mineral. E.g. a rock or gemstone
ORGANIC	Item is made of organic matter. E.g. a branch or piece of food
INVISIBLE	Item is naturally invisible.
MAGIC	Item has a magical aura and no other spells will hold on it. E.g. enchant weapon or create light will not work on this item.
NODROP	Item has been cursed and cannot be dropped or removed without magical help.
BLESS	Item has been blessed and so vampires cannot use it. Should only be used for holy artefacts or similar.
!PURGE	Item cannot be PURGED from a room simply. Use with caution.
BRITTLE	Item will scrap immediately if hit by a scrapping spell or blow.
RARE	Item has a rent cost and should have, if also useful, a low load limit (see the Zone section)
RESISTANT	Item has a better saving throw against scrapping effects.
IMMUNE	Item is immune to scrapping effects of spells or blows.
!LOCATE	Item cannot be found using Locate Object. Use with caution.
TWO-HANDED	Item, usually a weapon, needs two hands to use it.
PLAYER-MADE	Item has been made by a player. This flag should not be used.

Worn determines which positions on a player the item can occupy.
All of these are self-explanatory and should be used sensibly, (e.g. boots should have FEET) but some need explaining.
All objects should have the TAKE flag, or else they cannot be picked up. Only certain very heavy or special items should not have the TAKE flag. E.g. a fountain.
Do not use the SHIELD or LIGHT worn positions.

Rest determines which classes, sexes, alignments and levels are exempt from using the item.
The flag ONLY-CLASS needs some explaining.
This flag reverses the normal class restrictions inputted so that only characters with those classes and only those classes may use the item.
E.g. !mage !cleric !fighter only-class means that only mages, fighters and clerics or multiclasses of those three can use it. A ranger mage, for example, could not use it.

Affects are special effects that affect the user of the item and form the main part of an item's power.
Certain of the flags should not be used such as sex, class, level, char_height, char_weight, race_slayer, align_slayer, exp, numdice etc.
A rule of thumb. If it looks odd, don't use it.
When assigning Affects to an item, you need to bear in mind two things.
1 the level of the item
2 the sensibility of that affect for the item. E.g. a set of boots should not improve a person's BHD.

Affects that raise the HP, Mana, move, age or skill chances of a character should be in steps of 5
Affects that raise the hnd, stats, saving throws or similar should be in steps of 1

For every increase of 5 hp, mana, move, age, or skill chances, the level of the item should go up by 5
For every increase of hnd, stats, saving throws or similar, the level of the item should go up by 10
Adding restrictions to an item can lower its level.

No item should have more than a +1 to speed and even then, only very rare and expensive items should have any increase to speed.

Spell affect needs to use one of the following numbers.

	Hex Number	Decimal Equivalent
BLIND	0x00000001	1
INVISIBLE	0x00000002	2
REGENERATE	0x00000004	4
DETECT_INVISIBLE	0x00000008	8
SENSE_AURA	0x00000010	16
SENSE_LIFE	0x00000020	32
BERSERK	0x00000200	512
CURSE	0x00000400	1024
FLYING	0x00000800	2048
POISON	0x00001000	4096
INFRAVISION	0x00008000	32768
WATERBREATH	0x00010000	65536
DODGE	0x00040000	262144
SNEAK	0x00080000	524288
HIDE	0x00100000	1048576
SILENCE	0x00200000	2097152
TRUE_SIGHT	0x02000000	33554432
MEDITATE	0x20000000	536870912

Add up the decimal numbers of the affects you want and input the number.

Weapon spell uses the ALLSPELL number of the spell you want, as does **eat spell**.

Procs adds special features to an object.
The most commonly used of these is Fountain, which set the item to make an unlimited supply of water.

If you look at an object, all you see normally is nothing.
To add a LOOK description, you use the **Extra Descriptions** box.
Enter the keywords of the item and then a description. Descriptions should have 2 spaces added before the start of every new paragraph or sentence to help them be displayed properly.

Shops (OBSOLETE - use the shop_keeper and produces procs instead)

Shops in Deadedit require five things to make them work properly.
They need a **PEACEFUL** room in which the shop will be set up, a **SENTINAL** mob which has been given the **SHOP_KEEPER** procedure, the items that the shop will sell designed in the same area file as the shop, the shop dialog properly filled in, and finally the items for sale given to the mobile as set up in the **Zone** part of the **Area** file (see later for this)

The shop dialog seems very different from the other dialogs and can get very confusing.
As with all of the other parts of Deadedit, adding a new shop is easy.

Mobile is the number of the mobile that you have given the shopkeeper procedure to and that you want to use the shop settings you are setting up.

Room is the number of the room that you want the shop to be located in.

Keeper No Item is the message the shopkeeper tells the player if she tries to sell an item he does not have in his inventory.

Seller No Item is the message that the shopkeeper tells the player if she tries to buy an item that the shop doesn't have in stock.

Wrong Type is the message that the shopkeeper tells the player if she tries to sell it an item type that the shop does not trade in (see below).

Keeper Broke is the message that the shopkeeper tells the player if the player doesn't have enough money to buy an item.

Seller Broke is the message that the shopkeeper tells the player if the shopkeeper doesn't have enough money to buy an item that the player is selling.

Tell Buy is the message that the shopkeeper tells the player when she buys something from the shop.

Tell Sell is the message that the shopkeeper tells the player when she sells something to the shop.

Note that the last two of these are, in the default set up of a new shop, set the wrong way around.
The details in this guide are the correct way around. This mistake should be fixed in future versions of Deadedit.

Two points about these messages. They should always begin with **%s** as this directs the game to the interacting player.
And if the shop should say the amount of money involved, **%d** will put this into the message.

Buy is the multiplier applied to the cost of an item to get the cost to buy it in this shop. It should, in general, be greater than 1.0 but not much more than 1.7 or 1.8

Sell is the multiplier applied to the cost of an item to get the amount that the shop will pay for it when a player tries to sell it. It should, in general be less than 1.0 and not much less than 0.5

Trades are the types of item that the shop will buy from players if they attempt to sell to them.
These should be chosen sensibly, as, for example, it makes no sense for a food shop to have a trades of SCROLL. A more sensible choice for this imaginary food shop would be FOOD and LIQUID CON.
If left at undefined, the box is ignored.

Produces are the numbers of the items that you want the shop to stock. These items should be ones that are made in the same file, and should be noted down and then ensured that the shopkeeper mobile is given them in the **Zone** file.

Open and **Close** are the hours that the shop is open for. The shops in the large cities on Pangaea (the main continent) are open effectively all of the time. New shops that are open for most of the time should have something similar to **open 2 close 12 open 13 close 23** set.

Zones

The Zone file is perhaps the most important part of the area file, in terms of making the area work, as this is the part that tells DoN how to use the area file. Each area file needs a single zone file.

Add a zone file exactly the same way as with rooms etc and double click on the new zone file in the left hand window. This will pop up a dialog box with several very important parts.

Name should be the name of the zone, as it will appear in use in DoN.

Author should include your builder character's name and your email address, ICQ number and/or AIM name so that we can get in touch if needed.

Reset is the number of minutes the game waits to repopulate the area once it is working. The default for this is 30 minutes and this number should be between 20 and about 50 at most.

The drop menu to the left of **Reset** shows a final corollary to when the area will repopulate. **Always** means it always will, **never** obviously means never and **empty only** repopulates if no PCs are in the zone.

End room should be the number of the last room number you have in the rooms section.

In the middle of this dialog box is the most important section.
Once complete, this section will have several lines of "code" describing where in the area which objects and mobiles will appear and in what quantities.

Right click in the central area and a dialog pops up.
There are two main types of lines in a zone file.
Sibling and Child. Child lines are offshoots from a Sibling line.

Select add sibling and a further box appears with a drop menu of types of zone line.
These help to describe how the zone file is made up.

The main types of lines you will need are:

*** lines** These are comments on the structure of the zone and should be added to describe what you are trying to achieve.

C lines These lines only work if using Deadedit 0.25Beta release 5 or later.
They are always child lines of an M line and load a mobile charmed by the parent mobile. The load limit on this line is the same as if it were an M line (i.e. one for each instance of that mobile type to be loaded in the zone at least)
To make the charmed mobile guard it's master, set Add Act Flags to a value of 65536
Otherwise, leave Add Act Flags at a value of 0
C lines should be added as the last child line in any set of child lines. C lines also cannot have C lines as child lines from themselves. Attempting to do so will cause bizarre results.

D lines These set the state of a door in your zone.
Every door you have set in the rooms section needs a D line for each side. IE two D lines per door.
Each D line has the room number the door is in, which direction it goes and what state it is.
Direction 0=n
1=e
2=s
3=w

4=u
5=d

State 0=open
1=closed
2=closed and locked

E lines Usually added as a CHILD line from an M line (see later)
These EQUIP a mobile with a certain piece of equipment from the Items section of your area file. The object number and location to equip are straight forward, and the limit field is the load limit on that object in the entire of DoN. (See later for info on load limits)

F lines These should be added as a child line to either an M line or a C line.
They make the parent mobile fearful of certain other mobiles or players. The first parameter gives a TYPE of mobile and the second gives the value of that type.
Types
1 - Sex - value 0 1 2 neuter - male - female
2 - Race - same as race number
3 - char - do not use
4 - class - value 1 2 4 8 16 32 64 128 256 512 1024 2048
mage cleric warrior thief paladin druid psi ranger shifter bard monk blackguard
add them together and they will fear all of the choices.
5 - evil - value ignored
6 - good - value ignored
7 - vnum - value is vnum of feared mob type.

NB multiple F lines are possible to make a mobile fear more than one type of mobile but multiple lines of the same type will override each other with only the last one taking effect.

G lines These are similar to E lines in that they are added as CHILD lines to an M line.
They act in the same way, but the object is given to the mobile and put in their inventory.
NB Shop keepers need to have a G line for each piece of equipment that they will stock.

H lines These should be added as a child line to either an M line or a C line.
They make the parent mobile hateful of certain other mobiles or players. The first parameter gives a TYPE of mobile and the second gives the value of that type.
Types
1 - Sex - value 0 1 2 neuter - male - female
2 - Race - same as race number
3 - char - do not use
4 - class - value 1 2 4 8 16 32 64 128 256 512
mage cleric warrior thief paladin druid psi ranger shifter bard monk blackguard
add them together and they will hate all of the choices.
5 - evil - value ignored
6 - good - value ignored
7 - vnum - value is vnum of hated mob type.

NB multiple F lines are possible to make a mobile fear more than one type of mobile but multiple lines of the same type will override each other with only the last one taking effect.

M lines The most widely used zone lines, M lines tell DoN to load a Mobile.
For every individual creature that you want to load each repopulation, you require an M line. The load limit in the M line is the load limit for that TYPE of mobile.

E.g. to load 4 guards each repop of the same type and have a maximum of 12 of them loaded at any time, you would need 4 M lines, each with a load limit of 12.

O lines These act just like M lines, in that they load something in a room, but they load objects instead of mobiles.
Usually used for things such as fountains and notice boards, they can also be used for such items as Chests, or piles of gold on the floor.

X lines These remove a certain item from the floor of a certain room.
Used in conjunction with an O line, X lines can prevent such incidents as 30 fountains loading in a single room.

NB When it is time for an area to repopulate, DoN looks along the zone file at each line in turn. If that object or mobile is not at its load limit, it will put one of that thing into the area and move onto the next line.
Child lines are only ever looked at if the line they branch from is looked at and acted on.

With X and O line combinations, the way to set them up us to have the X line first and then have the O line afterwards.

To get child lines of M lines working properly, they should be added in the following order
E
G
H
F
C
M
Any other lines

Please, to increase the ease of checking, collect M lines, O and X pairs, O lines and D lines together in different sections, separated by * lines. Also, ensure that you add a comment to each line, in the space provided, to indicate what the line should be doing or loading.

Please note the following about Load limits.

A load limit of **0** means that the thing will have a small chance of loading each time its line is used.

A load limit of **1** means that the object will load once and once only. All keys should have a load limit of 1 and have their rent value set as –1. This will prevent people renting out with the item, and only allow a single key to ever load.

A load limit of **greater than 1** means that the object will load every time it's line is referenced, until that limit is reached DoN-wide. After this, it will load with a small chance each time, just like an object with a limit of 0.
Objects with the RARE flag should have fairly low load limits e.g. 20 or less
Objects that are very useful but not RARE should have limits below 100
Standard objects, such as food and most items sold by shops should have very large limits. E.g. most bread types have a load limit of 9999

Mobiles **NEVER** load past their load limits.

ACHECK

Acheck is a new feature added to Deadedit in the 0.25 beta release 5.
It allows you, using the simulator mode of Deadedit, to check the rooms, mobiles and items for common mistakes generating a report of the errors it has found.
Acheck is not infallible but can provide a good guide to various common mistakes

To use acheck, change from the map view to the simulator view of the main Deadedit window.
From the prompt, type “zload <x>” where <x> is the number of the Zone section of your file (by default this will be 1000)
That loads the zone file into memory and places objects and mobiles as if the zone had just repopulated in the mud.
Type “goto <x>” where <x> is the number of the first room in the area (default 1000) followed by “acheck”

The function will process each room, room extra description, room exit, mobile, item and item extra description and report a warning if it spots anything that might be a problem.
You can then go back and check on those warning points.