The

Barren Realms Builder's Guide

v 2.0

written and compiled by:

Faustus

with special thanks to:

Kiri, Temper, Delton, Matai,
Mazrim, Brand, Ahab,
and Kytar

(for contributing to the previous version
of this guide, making this new version
much easier to create.)

Visit us at: barren.coredcs.com 8000

Original DIKU code by: Hans Henrik Staerfeldt,
Katja Nyboe, Tom Madson,
Michael Seifert, and
Sebastian Hammer
Original MERC code by: Kahn, Hatchet, and Furey
BR Supplemental code by: The BARREN REALMS Coding Crew

Part I
before you begin your new area

Barren Realms has been around since 1994. In that time it has constantly grown and developed, mainly through the contributions of the players who have volunteered their time and energy to making Barren Realms a better place. One of the ways that players like you can help to add to the Barren Realms experience is by building an area. Anyone can create an area. You don't need any programming experience of any special tools. All that it takes is a little creativity, dedication, and drive.

This guide is designed to help anyone who wishes to write and area for the mud. It is primarily intended to help someone who wants to use a text editor to create their area file, as opposed to one of the off-line building programs available on the internet. (Barren Realms does not offer on-line building and there are no plans to add this feature.) Even if you are planning to use something other than a text editor to build your area, you should read through this guide, as there is plenty of good information contained herein that can be applied to all builders.

If the idea of creating an area sounds appealing to you, ask yourself the following question: Why do I want to make an area?

There are several possible answers you might come up with. If your answer is something like:

-I want to be in a position of power on the mud and building an area will help me to get there.
-I want to create an area that has equipment far more powerful than anything else on the mud.

perhaps you should reconsider. Building an area will not directly net your character any benefits or promotions on the mud. Likewise, the area you write will be checked to ensure balance with the rest of the mud so things do not become to easy.

On the other hand, you might answer that question with something like:

-I want to try a new way of expressing myself creatively.
-I have an interesting idea that I think would enhance the mud.

by all means, give it a try. We welcome any contribution that you could provide for us.

Also, keep in mind that Barren Realms is a fantasy-themed mud. We are only interested in area that fit well in a fantasy theme. Even though you might like to create one very badly, we will not accept any areas that contain modern elements, sci-fi elements, or anything that does not fit with the mud. This does not mean that you can only have dragons, ogres, and fairy-tale creatures in your area. It just means that your area cannot contradict any fantasy elements.

Once you've decided that you do want to create an area, start to think about the following details for your area:

What is the theme of your area? What kind of things will a player encounter there?
Where is the area located? What can be found in the surroundings or your area?
What kind of npc's will populate your area? Who will players find there?
What are some of the major features of the area? What is the terrain like? Are there any significant landmarks?
How large and spread out will your area be? Will all of the rooms be tightly packed together, or will there be lots of empty space in the map? (Don't make the area overly large, especially if this is your first try.)

After you've thought about the details for your area, you'll need put together a brief proposal for your area. Be sure to include the following information in your proposal:

The name of your character on Barren Realms
Your e-mail address
A brief description of the theme of your area, and, if possible, a name for the area
A general idea of what the terrain of your area is like
An approximation of where in the mud your area will be located
The level range for which your area is intended
Pro's and con's of the area (what will your area add to the mud? what will make your area a challenge for players?)
An approximate number of rooms that you will need for the area (a high guess is better than a low guess)
and, if possible, a sample description for one room, one mob, and one object for your area

Send this proposal to Kiri. (You can find her e-mail address by profiling her on Barren Realms.) You should receive a response shortly. If everything looks good, your idea will be approved and your will receive vnums** that you can use to start writing your area. If we feel that your idea doesn't quite work for some reason, we'll let you know what you should change before you re-submit things. We might ask you to change your level range, or we might not currently need an area with the theme that you are proposing. Whatever the problems might be, we will work with you to develop your idea into one that works well for the mud.

Once you have received approval for your area, there

**What are vnums? If you are new to building you might be a bit confused when I mentioned 'vnums' above. Vnums are the ID numbers that identify each unique mob, object, and room in your area. When you are working on an area, you will be assigned a specific range of vnums, say for example, 1001 through 1100. Each number in that range can be used for 1 mob, 1 object, and 1 room. Thus, you can have a room with the ID number 1015 as well as an object with the ID number 1015 and a mob with the ID number 1015. You cannot, however have another object with that number. An ID number used for a mob is generally called an 'mnum'. An ID number used for an object is often called an 'onum'. An ID number used for a room is usually called an 'rnum'.
Most areas have more rooms than they have objects or mobs. That's why we want an idea of how many rooms your area will have. That way, we can be sure that we are assigning you a sufficient number of vnums.

are a few more things to do before you sit down and type out the area file. If you haven't done so yet, map the area out on a piece of graph paper. Creating a map ahead of time will make writing the area file much easier. Also, jot down some notes about the mobs and objects you want to include in the area. You don't have to have full details about everything that will be found in the area at this point, but at least start thinking about these things.

Then, once everything is in order, you're set to write out the area file. See Part II for details on how to write the area file.

Part II
writing the area file

This section will walk you through each of the sections that you need to write for the area file. The file itself is a basic text file. You can create the file in any word processing program or text editor. Whatever program you use, though, make sure it is capable of saving files in a plain text format. Most, if not all word processors will give you the option to "save as text only' when you are in the program's save screen. If you save your file as anything other than plain text, there will be extra formatting information that can confuse and crash the mud. Always save your area file as a text only document.

As you type out the area file, try not to let the text wrap from one line to the next. If you are typing something that is going to spill over onto a new line, be sure to hit return before the text reaches the right edge of the file. Then, continue with what you were typing on the next line. This will keep things formatted correctly on the mud. No line should have more than 75 characters. Also, try and type the file in a mono-spaced font. That will also help you to format things correctly.

The area file itself is broken into eight different sections, as follows:

#AREA
#HELPS
#MOBILES
#OBJECTS
#ROOMS
#RESETS
#SHOPS
#SPECIALS

Each of these sections goes into one area file in the order listed above. The following pages will detail each of those items in depth.

Area

The line that starts your area file is where you set the area's name as well as the level range for which it is intended. The line should be written as follows:

#AREA {<level range>} <your name> <area title>~

As an example, I could start an area with the following:

```
#AREA { 0 15} Faustus The Tavern~
```

Be sure to use brackets (like this '{}') and not parenthesis around the level range. This way, your area will

be consistent with all the other areas in the area list. Also, since the name of your area is a title, you should be sure to capitalize any and all important words.

The symbol at the end of this line is called a tilde (~). It is a very important symbol that appears often throughout the area file. The symbol tells the mud that certain lines are finished. In these instructions, pay special attention to where tildes are placed. The mud will always expect a tilde to be in that spot and will probably crash if you omit any tilde.

Helps

The next section of your area file is where you can include any help files, should you desire. Some writers like to create a help file that details the specifics of their area. Other like to use help files to include interesting or bizarre messages for the players.

The first line of the helps section should be typed as follows:

```
#HELPS
```

After that, use the following sets of lines for each help file your wish to include.

Line A: -<level>

The level (which should be typed with numerals) determines how high of a level a player will need to be in order to read the help file. If you want everyone to be able to read the help file, use -0 on this line. For hero and immortal only help files, put -51 here. If you wish to reserve the help file for the immortals only, use -152. You can also set minimum level to any number in between.

For my example, I'd like everyone to be able to read the file, so I'll put:

```
-0
```

Line B: <keywords>~

The keywords specify what a player has to type in order to see the help file. You can include one single keyword, or you could make a short list of several available keywords. For my example, I'll use:

```
sample area~
```

Thus anyone who types 'help sample' or 'help area' will see the this help file. Be sure to include the tilde at the end of the line.

Line C: <help file>

Here you will type out the text that you want players to see when they request this particular help file. You can type more than one line, but be sure to hit return at the end of each individual line. I'll continue my example with:

```
This is a sample area that Faustus has put together for the area writing
guide.  Hopefully, this file will allow you, too, to create areas that
we can use to make Barren Realms an even better place.
```

That message will be displayed to anyone who types ‘help sample’ or ‘help area’.

Line D: ~

On the line following the text of your help file, simply type a single tilde all by itself.

If you wish to include more than one help file, just repeat lines A through D once again. You may include as many help files as you wish.

On the line after the ending tilde of your last help file, type the following:

```
0 $~
```

That tells the mud that you are finished with the Helps section.

If you do not wish to include any help files, you mush still type ‘#HELPS’ on one line, and follow it with ‘0 $~’ on the next line.

Once everything is put together, your Helps section should look something like this:

```
#HELPS
-0
sample area~
This is a sample area that Faustus has put together for the area writing
guide.  Hopefully, this file will allow you, too, to create areas that
we can use to make Barren Realms an even better place.
~
-152
secret~
This information is for immortals only.  It’s so secret that I can’t even
type what it really is, for fear that is might fall into the wrong hands.
~
0 $~
```

Mobiles

The mobiles section is where you create the details of all of the people, creatures, animals, spirits, etc. that will inhabit your area. This is probably the easiest of all the major sections in your area file to write. As long as you follow the guidelines about which special characters go where, you’ll be fine. Since this part of the file is relatively easy, take the time to do a good job. Make your mobs original and put as much detail into them as you can. How interesting your mobs are corresponds directly to how much work you put into them. Spending a few extra minutes to make truly noteworthy descriptions for your mobs will pay off in the end.

The first line of the mobiles section of your area file must consist of the following printed on a line all by itself: #MOBILES (in capitals with no other punctuation). When the mud reads your area file, this is

how it knows where to find the mobs.

All of your mobs will follow that line, and each individual mob will follow the same pattern of lines.

Line A: #<mnum>

This tells the mud what id number the mob will have. Start with the first vnum you have been assigned. For each new mob that you add, continue in numerical order until you run out of mobs that you want to include. I'll start an example mob with '`#101`'.

Line B: <keywords>~

This line specifies the word (or more commonly, the group of words) that a player can use to interact with the mob. There is no limit to the number of keywords a mob can have, and generally the more it does have, the better. Be sure to include the tilde after the last keyword in the list. For my example mob, I'll write this line as '`guard elf elven young~`'. Do not include any articles such as a, an, or the on this line.

Line C: <short description>~

The mud displays the short description as part of a longer sentence whenever a mob enters or leaves a room, and whenever a players interacts with the mob. The short description should just be a noun or two, along with any appropriate adjectives and articles, such as a, an, or the. I'll continue my example with the line '`a young elven guard~`'. Do not capitalize this line, and do not use any sort of ending punctuation. Do be sure to include the tilde, though.

Line D: <long description>

The long description is a message that is displayed after the room description whenever a player enters a room or types look. This message alerts the player to the mob's presence. It always appears on its own line, so be sure that you write the long description as a complete sentence, including proper punctuation and ending punctuation. For my example, this line will look like '`A young elf stands here, watching over the hallway.`' with no tilde at the end.

Line E: ~

This line needs to be a single tilde all by itself.

Line F: <look description>

The look description can (and should) actually be more than one single line. This is the description that is shown when a player actually looks at the mob. Assuming your description spills over onto more than one line (and it should), be sure to hit return before your words drop down onto the next line. This will prevent the mud from doing ugly things with word wrap. The look description of my example mob will look like this:

```
An elf with sandy blond hair paces back and forth along the hallway.  Every
few steps, he stops and swivels his head from side to side.  Sighing, he
places his hand over the hilt of his sword and continues pacing.
```

You'll probably be using more than one sentence here. Be sure all of them are capitalized and punctuated properly.

Line G: ~

This line comes right after the last line of the mob's look description. All you need in a tile by itself.

Line H: <act bits> <affect bits> <alignment> S

This line will contain a bunch of numbers. Act bits determine any special sets of behavior that the mob might have. These can be found in TABLE A. Feel free to assign more than one act bit to you mob, but be sure to separate them with the | character. Affect bits determine any spells effects that will be permanently assigned to the mob. You can find the different affect bits in TABLE B. Again, you can use more than one affect bit, as long as you use a | to separate them. If you chose not to use any act bits or affect bits, just put a 0 in one or both of those spots. Alignment is a numerical value that falls between 1000 and -1000. Remember, high numbers mean good, low numbers mean evil. Finally, the S at the end of the line has no real meaning on the mud. Farther back in mud history, that spot was important, but our code just ignores it. If that S isn't there, though, the mud gets upset and crashes. Don't forget it. My example mob will have the following values written on this line: '`1|4|32 4|8 500 S`'. This means that it is an NPC, it will pick up objects off of the floor, it is aggressive, and it is affected by detect evil and detect invisible. Its alignment is 500, meaning that it is good aligned.

Line I: <mob level> 0 0 0d0+0 0d0+0

Mob level is, as you would expect, the level of the mob you wish to create. When the mob loads into the mud, its level will be within two of whatever number you assign here. All of those zeroes after the mobs level are needed, even though the mud never actually bothers to read those numbers. For my example mob, this line will be '`20 0 0 0d0+0 0d0+0`'. The mob will always load between levels 18 and 22.

Line J: 0 0 0 0 <gender>

The first four zeroes are ignored, but needed. For the last number, use 1 for male mobs, 2 for female mobs, and 0 for neuter mobs. Those are the only options you have on this line. I'll finish up with this line as: '`0 0 0 0 1`', meaning that my mob is male.

That's it. At this point we have a complete mob. On the line next line, you can start over again at Line A with the next vnum for the next mob. If you have finished that last of the mobs for your area, type the following on the line immediately following the gender of your last mob: #0 That signifies that the #MOBILES section is done.

Putting all of my example lines together, we get the following, which gives you an approximation of what the #MOBILES section will look like:

```
#MOBILES
#101
guard elf elven young~
a young elven guard~
A young elf stands here, watching over the hallway.
~
An elf with sandy blond hair paces back and forth along the hallway.  Every
few steps, he stops and swivels his head from side to side.  Sighing, he
places his hand over the hilt of his sword and continues pacing.
~
1|4|32 4|8 500 S
20 0 0 0d0+0 0d0+0
0 0 0 0 1
#102
etc. etc. etc.
#0
```

MOB TABLES

TABLE A - ACT BITS FOR MOBS

ACT	BIT
NPC	1
SENTINEL	2
SCAVENGER	4
MEMORY	8
ILLITHID PRACTICE	16
AGGRESSIVE	32
STAY AREA	64
WIMPY	128
PET	256
ENHANCE	512
ELF PRACTICE	1024
HEALER	2048
UNDEAD	4096
DRUID PRACTICE	8192
KENDER PRACTICE	16384
HUMAN PRACTICE	32768
DWARF PRACTICE	65536
IDENTIFY	131072
REPAIR	262144
SELL MID	524288
SELL MID-HIGH	1048576
SELL HIGH	2097152
STAY TERRAIN	8388608
AVIAN PRACTICE	16777216

notes:

All mobs should be set with the NCP bit.

SENTINEL mobs will not wander around at all.

SCAVENGER mobs will pick up any objects on the ground.

If a mob has MEMORY, it will remember any player that has attacked it and fled, and will attack that player on sight.

The various PRACTICE mobs will train skills that correspond to the specified race.

AGGRESSIVE mobs will attack any player lower than them in level.

STAY AREA mobs will not leave your area. This must be used if the rooms that connect your area to other areas are not set to be no-mob rooms.

If any of your mobs will be sold in a pet shop, set those mobs with the PET bit.

Mobs that have the ENHANCE bit can enhance a player's stats for the appropriate price.

HEALER mobs will heal all characters for a price.

When a player fights an UNDEAD mob, the mob's arms and legs may be lopped off and continue to fight on their own.

A mob set with IDENTIFY will, for a price, identify any object that a player gives them.

A REPAIR mob can fixed damaged equipment for a small fee.

SELL MID, SELL MID-HIGH, and SELL HIGH are used in conjunction with shopkeeper mobs. These bits will set the levels of items that load into the inventories of these shopkeepers to 10, 20, or 30, respectively.

Mobs with STAY TERRAIN will not wander off of the terrain type of the room where the start out. (Terrain type is set in the #ROOMS section.) This is useful for confining certain mobs to specific portions of your area.

Be aware that any mob set with a HEALER, REPAIR, IDENTIFY, ENHANCE, or PRACTICE bit will be protected and players will be unable to kill that mob.

TABLE B - AFFECT BITS FOR MOBS

AFFECT	**BIT**
BLIND	1
INVISIBLE	2
DETECT EVIL	4
DETECT INVIS	8
DETECT MAGIC	16
DETECT HIDDEN	32
SANCTUARY	128
FAERIE FIRE	256
INFRARED	512
CURSE	1024
POISON	2048
PROTECTION	4096
SNEAK	32768
HIDE	65536
SLEEP	131072
FLY	524288
PASS DOOR	1048576
PLAGUE	2097152
VISION	4194304
DIRT KICK	16777216
VERTIGO	33554432
FIRM GRASP	67108864

Each of these will permanently give the mob the effects of the spell listed. Most are self explanatory, and if you need any extra information, you can find them in the help files on the mud.

Note that shopkeepers, identifiers, repairmen, healers and the like should all have DETECT INVIS and DETECT HIDDEN, unless you don't want them to be able to interact with invisible characters/objects.

There are a few other options for mobs. You can chose special skills such as spell casting and pick pocketing to different mobs. These will be covered in detail in the #SPECIALS section, where those skills are assigned.

That's all there is to creating mobs. If you want to assign some of your mobs as shopkeepers, you will do so later in the #SHOPS section.

A few last thoughts on mobs:

- Make your list of keywords as thorough as possible. Too many people write areas where the mobs have a long description like 'An ugly witch cackles as she stirs her bubbling cauldron', but the only keywords

that word for the mob are ‘Brunhilda’ and ‘evil’. Players will get frustrated very quickly trying to interact with a mob like that. If you want to be a successful builder, do not be that vague with your keywords. A good rule of thumb is to include every noun and adjective from both the long and short description when you create the list of the mob’s keywords.

- Try to be as detailed as you possibly can in the mob’s look description. Close your eyes and try to get a good mental picture of the mob. Once you’ve done that, put into words as many of the details you came up with as you can.

- At the same time, be objective about your descriptions. Put less emphasis on any interacting between the mob and the character, and more emphasis on the mob by itself. Also, try not to come out and state the mob’s emotions directly. Don’t say that a create is “happy.” Say that it is “smiling and giggling.” If you let the players come to their own conclusions about the mob, they will enjoy your area better.

- Keep your mobs consistent with the theme of your area and with the overall theme of the mud. You mobs should not look out of place in out fantasy world.

- Be detailed with you mobs, but remember that not every mob needs to be extremely interesting. A typical village will probably have several ordinary children and adults who have nothing outwardly special about them. Likewise, you area will probably have its share of mobs that are rather ‘plain.’

- Be aware that any keyword you assign to a mob will prevent that word from being used as a character name. If you were to assign the keyword ‘Faustus’ to one of your mobs, I would not be able to log onto the mud any more. If you submit an area with mob keywords that overlap with existing character names, you will be asked to change those keywords.

- In general, try to aim for approximately one mob for every two rooms in your area. If there are a lot of sentinel, increase the ratio to something a little closer to one mob for every one and a half rooms.

Objects

The objects section can be somewhat tricky, because many builders tend to want to see every player walking around with equipment that they wrote. As such, it is tempting, while you are building, to create objects that are more powerful than anything that currently exists on the mud. Resist this temptation. Please. To keep things somewhat realistic, your area should contain more ‘mundane’ equipment that ‘exceptional’ equipment. It’s all right to include a few powerful items, but they should be the exception, rather than the norm. Yes, this does mean that you might be creating many objects that players will never use, but your area will be more well rounded and detailed as a result. Enough with my moralizing for now. Let’s move on to how you actually write the objects section.

The objects section of your area must begin with: `#OBJECTS` typed on a line all by itself. This tells the mud that the following lines will contain the info necessary for all of the objects in your area.

After the line that signifies the start of the objects section, repeat each of the following entries for each object you want to include:

Line A: #<onum>

This tells the mud what id number the object will have. Start with the first vnum you have been assigned. For each new object that you add, continue in numerical order until you run out of objects that you want to include. I'll start an example object with '`#101`'.

Line B: <name>~

The name (which can be more than one word) tells the mud what keywords can be used with this object. For example, if this line reads '`wand oak crooked~`', then a player could use any of those three words to interact with the object. Do not include any articles, such as an, a, or the on this line. Be sure to include the tilde at the end of this line.

Line C: <short description>~

The short description is what is shown when an object is in your inventory. It is also displayed as part of a longer message when any player picks up, drops, wears, etc. said object. For example, I could continue this object with the line '`a crooked oak wand~`' or something similar. Do not capitalize the start of this line unless the short description of the object begins with a proper noun. Do not use any sort of ending punctuation, but do be sure to end the line with a tilde.

Line D: <long description>~

The long description is what you see when an object is lying on the floor. Since the mud displays this on a line all by itself, you need to write the long description as a complete sentence, including capitalization and proper ending punctuation. I could, for example, continue my object with the line '`A crooked wand made from oak lies here in the dust.~`' Don't forget the tilde after the ending punctuation. Also, do you best to keep this long description from spilling over onto a second line.

Line E: ~

Yep. That's it. Just put a tilde all by itself on this line.

Line F: <object type> <extra bits> <wear bits>

The object type is a single number that determines how the object behaves. The number can be found in TABLE C. Extra bits are one or more numbers that determine if the object has any special properties, such as glowing, invisibility, etc. You can find a listing of extra bits in TABLE D. Wear bits are one or more numbers that determine whether or not an object can be picked up and if so, where it can be worn. You can find the different wear bits in TABLE E. Continuing on with my example object, my next line would be '`3 1|1024 1|16384`'. This means that the object is a wand, it glows and cannot be used by evil characters, and that it can be picked up and held in the hand. Note that if you use more than one number for the extra bits and wear bits, you must join them with a |. If you do not wish to include extra bits or wear bits, put a 0 in that spot.

Line G: <value0> <value1> <value2> <value3>

These four values help to determine exactly how the object works. This is where you set which spells a potion will cast on a player, or how much weight a container can hold, or what type of damage a weapon deals, etc. Each type of object uses these four values in different ways, so you'll need to have a good look at TABLE F. That table will tell you how to set these four values based on the object type you set in the previous line. For my wand, I can set this line to be '`20 8 8 10`'. This means that the wand cast

the spell colour spray at level 20, and has 8 charges.

Line H: <weight> <cost> 0

The weight is simple a single number that determines how heavy the object is. Use some common sense when designating weights. Obviously, a steel breastplate should weigh much, much, much more than a linen shirt. Objects weights should logically correlate to one another Pay special attention to the weight for weapons, since characters with low strengths may not be able to wield heavier weapons. See TABLE G to find out what strengths are needed to wield weapons of differing weights. Cost can be ignored unless the object is a potion or pill. For those object types, you must designate a value, or shopkeepers will not be able to sell them. The 0 at the end of this line gets ignored by the mud, but it has to be there, or the mud will crash. For my wand, I'll write this line as '`3 0 0`'.

Extra descriptions are where you get to tell the players a little bit more about your objects. You are not obligated to include any extra descriptions for any of the objects your write, but we highly recommend that you do use them for most, if not all of your objects. Otherwise, when someone looks at the object, the mud will just show them the long description again, even if the object is in the character's inventory. If you do not wish to include extra descriptions, skip lines I, J, and K.

Line I: E <extra description keywords>~

When players look at any of these keywords, they will see the corresponding extra description. Since I want players to be able to find out a little more about the wand, I will write this line as '`E wand oak crooked~`'. Again, don't forget the tilde.

Line J: <extra description>

This is the message that is displayed whenever a player looks at one of the keywords listed in the line above. Just use this space to give a little more detail about the object. For example, I'll put '`Tiny runes are carved into the length of the wand. The wood seems to radiate an aura of power.`' It's perfectly all right for this description to spill over onto more than one line, but just make sure that you hit return at the end of each line, rather than simply letting your word processor wrap the words onto the next line by itself.

Line K: ~

On the line just below the end of you extra description, put a tilde all by itself. Don't put a tilde here if you're not using any extra descriptions, though.

If you want, you can include multiple sets of extra descriptions. Just repeat lines I, J, and K for each different extra description you wish to add onto the object.

Line L: A <apply bit> <apply value>

Apply bits and values tell the mud to modify a player's stats when they are wearing the object in question. The different apply bits can be found in TABLE H. The apply value is simply how much the stat specified by the apply bit is altered. For my wand, I'll use '`A 13 15`'. This means that when the wand is held, the character's maximum number of hit points goes up by 15. If you want the object to affect more than one stat, repeat line L on a new line for each affect you wish the object to have. Line L is optional. If you do not wish for the object to affect stats, simply skip this line.

At this point the object is finished. The next line in the area file will be line A once again,

specifying the number of the next object. If you have just written the last object you wish to include in your area, type '#0'. That tells the mud that the #OBJECTS section is finished.

Putting together all of the lines from my example, the start of my objects section would look like this:

```
#OBJECTS
#101
wand oak crooked~
a crooked oak wand~
A crooked wand made from oak lies here in the dust.~
~
3 1|1024 1|16384
20 8 8 10
3 0 0
E wand oak crooked~
Tiny runes are carved into the length of the wand.  The wood seems to
radiate an aura of power.
~
A 13 15
#102
etc. etc. etc.
#0
```

OBJECT TABLES

TABLE C - TYPE NUMBERS FOR ITEMS

TYPE	NUMBER
LIGHT	1
SCROLL	2
WAND	3
STAFF	4
WEAPON	5
TREASURE	8
ARMOR	9
POTION	10
FURNITURE	12
TRASH	13
CONTAINER	15
DRINK CONTAINER	17
KEY	18
FOOD	19
MONEY	20
FOUNTAIN	25
PILL	26
SCUBA	27
FLY	28
QUIVER	29
PROJECTILE	30

TABLE D - EXTRA BITS FOR ITEMS

FLAG	BIT
GLOW	1
HUM	2
EVIL	16
INVIS	32
MAGIC	64
NODROP	128
ANTI GOOD	512
ANTI EVIL	1024
ANTI NEUTRAL	2048
NOREMOVE	4096
INVENTORY	8192
FLAMING	65536
INVISIFY	262144

Most of these are self-explanatory. Inventory is a little odd, though. If an object has the inventory bit, it will disappear when its owner dies. This means that it will be tricky for players to get such objects if a mob has it, and that once they do get one of these objects, they will lose it when they die.

TABLE E - WEAR BITS FOR ITEMS

SLOT	BIT
TAKE	1
FINGER	2
NECK	4
BODY	8
HEAD	16
LEGS	32
FEET	64
HANDS	128
ARMS	256
OFFHAND	512
ABOUT BODY	1024
ABOUT WAIST	2048
WRIST	4096
WIELD	8192
HOLD	16384
TWO HANDED	32768

*if you don't include the TAKE bit, players wont' be able to pick up the object.

TABLE F - OBJECT VALUES

Type	#	Value0	Value1	Value2	Value3
LIGHT	1	0	0	hours	0
SCROLL	2	spell level	spell # 1	spell # 2	spell # 3
WAND	3	spell level	max charges	curr. charges	spell #
STAFF	4	spell level	max charges	curr. charges	spell #
WEAPON	5	0	0	0	damage type
TREASURE	8	0	0	0	0
ARMOR	9	0	0	0	0
POTION	10	spell level	spell # 1	spell # 2	spell # 3
FURNITURE	12	0	0	0	0
TRASH	13	0	0	0	0
CONTAINER	15	max. wieght	state	ONUM of key	0
DRINK CONT.	17	max. capacity	cur. capacity	liquid type	poison
KEY	18	0	0	0	0
FOOD	19	nourishment	0	0	poison
MONEY	20	0	0	0	0
FOUNTAIN	25	0	0	liquid type	0
PILL	26	spell level	spell # 1	spell # 2	spell # 3
SCUBA	27	0	0	hours	0
FLY	28	0	0	0	0
QUIVER	29	max. weight	state	ONUM of key	0
THROW	30	0	0	0	0

NOTES:

HOURS designates how many hours of game time the object will last. If set this number to -1, the object will last forever.

Spell numbers for potions, staves, wands, pills, and container traps can be found in TABLE I.

MAX CHARGES and CURRENT CHARGES for wands and staves should generally be set to the same number.

DAMAGE TYPE for weapons can be found in TABLE J.

STATE options for containers and quivers can be found in TABLE K.

If a container has no key, use -1 for the key number.

If a container is not trapped, use o for the trap spell number.

MAX CAPACITY and CURRENT CAPACITY for drink containers tell how many units of liquid a player can drink from the container before it disappears. A typical drink is anywhere from 5 to 10 units.

LIQUID TYPES can be found in TABLE L.

POISON denotes whether the food or water is poisonous. 0 means it is fine. Any other number means it is not.

QUIVER type objects are used to hold THROW type objects.

TABLE G - WEAPON WEIGHT INFO

The following table lists the weight limits for weapons that a character can wield based on their strength. Keep these in mind when you are creating weapons.

STRENGTH	MAX WIELDABLE WEIGHT
13	13
14	14
15	15
16	16
17	22
18	25
19	30
20	35
21	40
22	45
23	50
24	55
25	60

Since only dwarves can have a strength of 23 or higher, any weapon that weighs more than 45 pounds is effectively a dwarf-only object.

TABLE H - APPLY NUMBERS

Use these bits to specify which attributes an object will affect,

ATTRIBUTE	BIT
STR	1
DEX	2
INT	3
WIS	4
CON	5
MANA	12
HIT POINTS	13
MOVEMENT	14
ARMOR CLASS	17
HITROLL BONUS	18
DAMAGE BONUS	19
SAVE VS. PARALYZATION	20
SAVE VS. RODS	21
SAVE VS. PETRIFICATION	22
SAVE VS. BREATH	23
SAVE VS. SPELL	24

For armor class and all the save vs. whatevers, a negative value is a bonus, and a positive value is a penalty. For everything else, positive is a bonus, negative is a penalty.

Object Apply Guidelines

One of the hardest things to do while building an area is balancing you objects to be sure that none of them are too over-powerful, giving an unfair advantage to anyone that uses certain items from your area. To ensure that none of objects are unfairly powerful, we ask that you use the following guidelines when you apply bonus affects to the objects in your area file.

To determine the limit of possible effects for each individual object, use the following procedure:

Take the level of the mob upon which that this item loads. That number is the number of "points" that you can use to assign bonus to the object.

Different bonuses can be added at the cost of different numbers of points. The costs are as follows:

str or dex	each +1 costs	4 points
any other attribute	each +1 costs	3 points
mana	each +3 costs	2 points
hit points	each +1 costs	1 point
movement	each +4 costs	1 point
armor class	each -3 costs	2 points
hitroll bonus	each +1 costs	4 points
damroll bonus	each +1 costs	7 points
any saving throw	each -1 costs	4 points

if you change any of the above bonus to a penalty, you can subtract half the number of applicable points from the total you have spent, For example, if one of the affects -1 to wisdom, you can subtract 2 from the number of points you have spent.

Effects that limit the use of the object can also give you extra points to spend on your bonuses. Refer to the list below to see how any extra points each limitation gives you.

if the object:	you can spend an extra:
is useable by only two alignments	1 point for every ten levels of the object
is useable by only one alignment	2 points for every ten levels of the object
is something other than weapon or armor	1 point for every ten levels of the object
is a shield	2 points for every ten levels of the object
is no-drop or no-remove*	1 point for every ten levels of the object
is held by an aggressive mob or a mob with sanctuary*	2 point for every ten levels of the object
is significantly far from recall**	1 or 2 point for every ten levels of the object

* Only apply this once. An object that is both no-drop and no-remove gets no more extra points that an object that is simply no-drop.
** Talk to the builder imms to see if this can apply to objects in your area.

An example of an object that falls within these guidelines appears as follows:

Mithril Gauntlets	(level 40)
+ 2 to strength	(+8 pts)
- 2 to wisdom	(-3 pts)
+ 3 to hitroll	(+12 pts)
+ 2 to damroll	(+12 pts)
+ 20 hitpoints	(+20 pts)
anti-evil	(-4 pts)
held by aggressive mob	(-8 pts)
total	37 points

Please note that only a few of the objects in your area should use their maximum allotment of points. It is possible to include one or two exceptional items in your area that exceed these limits, but do not go overboard. If you abuse these limits, you will be asked to revise the statistics for your objects.

TABLE I - SPELL NUMBERS

The following table lists which numbers correspond to which individual spells. Not all spells will work with all types of objects. This table also tells you whether or not any given spell will work when placed on different object types.

SPELL NAME	NUMBER	WAND/SCROLL	STAFF	POTION/PILL
acid blast	70	yes	yes	no
acid breath	200	yes	yes	no
adrenaline control	470	yes	yes	yes
agitation	471	yes	yes	no
armor	1	yes	yes	yes
astral	100	no	no	no
awe	473	no	no	no
ballistic attack	474	yes	yes	no
bamf	700	yes	yes	no
berzerk	92	yes	yes	yes
bless	3	yes	yes	yes
biofeedback	475	yes	yes	yes
blindness	4	yes	yes	no
buddha finger	600	yes	yes	yes
burning hands	5	yes	yes	no
call lightning	6	yes	yes	yes
cause critical	63	yes	yes	no
cause light	62	yes	yes	no
cause serious	64	yes	yes	no

SPELL NAME	NUMBER	WAND/SCROLL	STAFF	POTION/PILL
cell adjustment	476	yes	yes	yes
change sex	82	yes	yes	yes
charm person	7	yes	yes	no
chill touch	8	yes	yes	no
chin kang palm	601	yes	yes	no
colour spray	10	yes	yes	no
combat mind	477	yes	yes	yes
complete healing	478	yes	yes	yes
concentrate	602	yes	yes	yes
control flames	478	yes	yes	no
continual light	57	yes	yes	yes
control weather	11	no	no	no
create food	12	yes	yes	yes
create sound	480	no	no	no
create spring	80	yes	yes	yes
create water	13	yes	no	no
cure blindness	14	yes	yes	yes
cure critical	15	yes	yes	yes
cure disease	501	yes	yes	yes
cure light	16	yes	yes	yes
cure poison	43	yes	yes	yes
cure serious	61	yes	yes	yes
curse	17	yes	yes	no
death field	481	yes	yes	yes
detect evil	18	yes	yes	yes
detect hidden	44	yes	yes	yes
detect invis	19	yes	yes	yes
detect magic	20	yes	yes	yes
detect poison	21	yes	no	no
detect presence	618	yes	yes	yes
detonate	482	yes	yes	no
disintigrate	483	yes	yes	no
dispel evil	22	yes	yes	no
dispel magic	59	yes	yes	yes
displacement	484	yes	yes	yes
earthquake	23	yes	yes	yes
ectoplasmic form	485	yes	yes	yes
ego whip	486	yes	yes	no
enchant weapon	24	yes	no	no
energy containment	487	yes	yes	yes
energy drain	25	yes	yes	no
energy transfer	91	yes	yes	no
enflame weapon	97	yes	no	no
enhance armor	488	yes	no	no
enhanced strength	489	yes	yes	yes
expose	617	yes	no	no
faerie fire	72	yes	yes	no

SPELL NAME	NUMBER	WAND/SCROLL	STAFF	POTION/PILL
faerie fog	73	yes	yes	yes
fire breath	201	yes	yes	no
fireball	26	yes	yes	no
firm grasp	610	yes	yes	yes
flamestrike	65	yes	yes	no
flesh armor	490	yes	yes	yes
fly	56	yes	yes	yes
frenzy	609	yes	yes	yes
frost breath	202	yes	yes	no
gas breath	203	yes	yes	yes
general purpose	205	yes	yes	no
giant strength	39	yes	yes	yes
harm	27	yes	yes	no
heal	28	yes	yes	yes
high explosive	206	yes	yes	no
identify	53	yes	no	no
inertial barrier	491	yes	yes	yes
inflict pain	492	yes	yes	no
insight	611	yes	no	no
intellect fortress	493	yes	yes	yes
illusion	613	yes	yes	no
infravision	77	yes	yes	yes
invis	29	yes	yes	yes
iron monk	603	yes	yes	yes
know alignment	58	yes	yes	no
laughing buddha	604	yes	yes	no
lend health	494	yes	yes	no
levitation	495	yes	yes	yes
lightning bolt	30	yes	yes	no
lightning breath	204	yes	yes	no
locate object	31	no	no	no
magic missile	32	yes	yes	no
mass invis	69	yes	yes	yes
mass heal	608	yes	yes	yes
medicine	605	yes	yes	yes
mental barrier	496	yes	yes	yes
mind scramble	615	yes	yes	no
pass door	74	yes	yes	yes
plague	503	yes	yes	no
prayer	606	yes	yes	yes
poison	33	yes	yes	no
project force	498	yes	yes	no
protection	34	yes	yes	yes
psionic blast	499	yes	yes	no
psychic crush	460	yes	yes	no
psychic drain	461	yes	yes	no
psychic healing	462	yes	yes	yes

SPELL NAME	NUMBER	WAND/SCROLL	STAFF	POTION/PILL
quivering palm	607	yes	yes	no
refresh	81	yes	yes	yes
remove curse	35	yes	yes	yes
repel	612	yes	yes	no
sanctuary	36	yes	yes	yes
scry	614	no	no	no
sizzle	616	yes	no	no
share strength	464	yes	yes	no
shield	67	yes	yes	yes
shocking grasp	37	yes	yes	no
sleep	38	yes	yes	no
spite	90	yes	yes	no
stone skin	66	yes	yes	yes
summon	40	no	no	no
telepathy	619	no	no	no
teleport	2	no	no	no
thought shield	465	yes	yes	yes
transform energy	95	yes	yes	yes
ultrablast	466	yes	yes	no
ventriloquate	41	no	no	no
weaken	68	yes	yes	no
web	93	yes	yes	no
word of recall	42	no	yes	yes

TABLE J - WEAPON DAMAGE TYPES

TYPE	NUMBER
HIT	0
SLICE	1
STAB	2
SLASH	3
WHIP	4
CLAW	5
BLAST	6
POUND	7
CRUSH	8
GREP	9
BITE	10
PIERCE	11
SUCTION	12

The damage type determines what message will be displayed to the player when they do damage with their weapon. There is no actual game-play difference between any of these damage types, with the exception of piercing-type weapons. Weapons that pierce all the only weapons that can be used by a kender for the backstab skill.

TABLE K - CONTAINER BITS

FLAG	BIT
CLOSEABLE	1
PICKPROOF	2
CLOSED	4
LOCKED	8

Use a combination of these flags for value1 of container type objects. If you use more than one of these flags, be sure to separate them with the | symbol. For example, if you want to have a closeable object that is closed when it loads, but not locked, you would use '1|4'. The closed and locked flags will determine whether the container is closed and locked when the container loads into the mud. If you want the container to load in the closed state, be sure that you also give it the closeable bit as well, or no one will be able to open it. Lastly, it you don't want the container to be closeable at all, just put a 0 in value1 for the object.

TABLE L - LIQUID TYPES

0 - water	1 - beer*	2 - wine*	3 - ale*
4 - dark ale*	5 - whisky*	6 - lemonade	7 - firebreather*
8 - local specialty*	9 - slime mold juice%	10 - milk	11 - tea
12 - coffee	13 - blood%	14 - salt water%	15 - cola
16 - ice water	17 - stagnant water	18 - sewer water	19 - soda water
20 - tonic water	21 - ginger ale	22 - root beer	23 - cherry cola
24 - orange soda	25 - herbal tea	26 - lemon juice	27 - pink lemonade
28 - grapefruit juice	29 - orange juice	30 - tomato juice	31 - pickle juice
32 - banana juice	33 - watermellon juice	34 - cantaulope juice	35 - beet juice
36 - kraut juice	37 - cranberry juice	38 - pineapple juice	39 - cherry juice
40 - grape juice	41 - apple juice	42 - blueberry juice	43 - blackberry juice
44 - raspberry juice	45 - kiwi juice	46 - mango juice	47 - apple cider*
48 - coconut milk	49 - chocolate mile	50 - chocolate milkshake	51 - vanilla milkshake
52 -strawberry milkshake	53 - pink goat milk	54 - buttermilk	55 - clabbered milk
56 - cream	57 - vanilla extract	58 - tabasco	59 - melted butter
60 - olive oil	61 - barbecue sauce	62 - gravy	63 - beef stew
64 - vegetable stew	65 - broth	66 - borsht	67 - tomato soup
68 - chicken noodle soup	69 - beet soup	70 - mushroom soup	71 - clam chowder
72 - strange brew	73 - Reisling*	74 - Chenin Blanc*	75 - Gewrtztraminer*
76 - Pinot Blanc*	77 - Gris*	78 - Trebbiano*	79 - Muscat*
80 - Semillon*	81 - Chardonnay*	82 - Cabernay Fran*	83 - Sangiovese*
84 - Tempranillo*	85 - Syrah*	86 - Grenache*	87 - Zinfindel*
88 - Gamay*	89 - Nebbiolo*	90 - Cabernet Sauvignon	*91 - Pinot Noir*
92 - Merlot*	93 - Meritage*	94 - fermentation*	95 - house blend*
96 - sparkling wine*	97 - sherry*	98 - port*	99 - mead wine*
100 - stout*	101 - cream stout*	102 - oatmeal stout*	103 - porter*
104 - honey porter*	105 - lager*	106 - honey brown lager*	107 - summer brew*

108 - winter brew*	109 - honey wheat*	110 - honey brown ale*	111 -dark and dry cider*
112 - brown ale*	113 - chocolate amber ale*	114 - stock ale*	115 - stock lager*
116 - pub draft*	117 - pilsner*	118 - nut brown ale*	119 - mocha stout*
120 - amber ale*	121 - triple malt ale*	122 - wicked ale*	123 - pale ale*
124 - grain alcohol*	125 - vodka*	126 - gin*	127 - rum*
128 - sloe gin*	129 - dry vermouth*	130 - sweet vermouth*	131 - tequilla*
132 - liquer*	133 - bitters	134 - blended scotch*	135 - single malt scotch*
136 - 15 year old single malt*	137 -50 year old single malt*	138 - bourbon*	139 - sour mash*
140 - straight bourbon*	141 - bile%	142 - grape slushee	143 - cherry slushee
144 - cola slushee	145 - pepsi	146 - mountain dew*	147 - blood%
148 - saliva			

Liquids marked with an * will increase a player's drunk state.
Liquids marked with a % will actually increase a player's thirst.

There are a few other special characteristics you can odd to you objects, but they will be covered in the SPECIALS section.

***** ADVANCED OBJECT TOPICS *****

Once you are familiar with creating objects for your areas, try out some of these options. If you're new to building, you might want to skip this section and finish up with the general object hints at the end of the objects section.

SPELLED WEAPONS

It is possible to create a weapon that casts a certain spell sometimes when it hits. Spell weapons are very cool, but they shouldn't be too common. Moderation is the key. If you wish to create such a weapon, follow these steps:

1. Add the 'spell weapon' extra bit in the extras spot for the weapon. The bit for spelled weapons is 33554432.

2. Set value0 for the weapon to the spell number you wish to use. DO NOT, however, use the spell numbers from TABLE I. Use the numbers below, instead.

acid blast	1	fire breath	89
acid breath	87	fireball	46
adrenaline control	117	flamestrike	47
agitation	118	flesh armor	137
armor	2	fly	48

aura sight 119
awe 120
bamf 6
ballistic attack 121
biofeedback 122
blindness 7
buddha finger 8
burning hands 9
call lightning 10
cause critical 11
cause light 12
cause serious 13
cell adjustment* 123
change sex 14
chill touch 16
chin kang palm 17
colour spray 18
combat mind 124
complete healing 125
concentrate 19
continual light 20
control flames 126
create food 22
create spring 23
cure blindness 25
cure critical 26
cure disease 27
cure light 28
cure poison 29
cure serious 30
curse 31
death field 128
detect evil 32
detect hidden 33
detect invis 34
detect magic 35
detect presence* 194
detonate 129
disintegration 130
dispel evil 37
dispel magic 38
displacement 131
earthquake 39
ectoplasmic form 132
ego whip 133
empath* 186
energy containment 134
energy drain 41
frost breath 89
gas breath 90
general purpose 115
giant strength 50
harm 51
heal 52
high explosive 116
illusion 189
inertial barrier* 138
inflict pain 139
infravision 54
insight 187
intellect fortress* 140
invis 55
iron monk 56
laughing buddha 58
lend health 141
levitation 142
lightning bolt 59
lightning breath 91
magic missile 61
mass invis* 62
mass heal* 63
medicine 64
mental barrier* 143
mind scramble 191
pass door 65
plague 66
poison 68
project force 145
protection 70
psionic blast 146
psychic crush 147
psychic healing 149
quivering palm 71
refresh 72
remove curse 73
sanctuary 74
share strength 150
shield 75
shocking grasp 76
sizzle 192
sleep 77
stone skin 79
teleport** 81
thought shield 151
transform energy 82
ultrablast 152

energy transfer	42	weaken	84
enhanced strength	136	web	85
expose	193		
faerie fire	44		
faerie fog	45		

* - These spells will be cast on the weapon's wielder, not on the mob.
** - Teleport will work on a spell weapon, but will always result in a mis-teleport, sending the wielder to some random location. Use this very, very sparingly, if at all.

Be aware that, with the exception of those spells marked with an *, all spell weapons will affect the mob being fought. Yes, it is possible to create a weapon that will cast sanctuary on a mob in the middle of a fight, but you should have a very, very good reason for using a non-offensive spell on a spell weapon.

ENTERABLE OBJECTS

You can designate certain containers in you area as enterable. This means that you could create, say, a tent that players can sleep in, or a fireplace that a character can enter and root around in the ashes, looking for hidden items. Generally, enterable objects should not be takeable, but if you have a good reason that fits with the theme of your area, you can create enterable objects that can be picked up and carried around.

To create an enterable container, first create it like any other container. Add an extra flag with the value 133680 to the container's Value1 (the spot that determines that container's state).

If you want, you can create a special interior description for the object that can be seen by players who are inside. Simply add an extra description with the keyword: inside_description (the underscore is required). The extra description text will automatically be shown to players on the inside of the object.

You will probably have to set the Value0 of the container to a fairly high number, since the container's max weight will have to be great enough to hold the character *and* all of their equipment. If you want to be mean, keep that number fairly low so that a player will have to drop most, if not all of their equipment before entering the object. Meanness is always an admirable quality in a builder.

PERSONALIZED OBJECTS

You can set up an object so that only someone who kills the mob at originally owned said object is able to use it. The process is tricky and must be followed exactly, so read this part very, very carefully.

First of all, you have to create an owner list for the object. This list is just an extra description for the object with the keyword: owner_list (make sure you include the underscore). The description that goes along with this keyword will be a list of the keywords for all of the mobs you wish this item to load onto. You must use all of the keywords for these mobs as they appear in the #MOBS section of your area file. Each group of mob keywords must be surrounded by single quotes (‘). put a single space between each set of mob keywords, if there is more than one mob you wish to be able to use this item.

So.... Suppose you wish to create a personalized sword that will load onto three different mobs: a hobgoblin, a goblin, and an elf.
The hobgoblin’s full set of keywords is: hobgoblin hob green
The goblin’s full set of keywords is: goblin gob short
The elf’s full set of keywords is: elf short blond

You then have to set up an extra description for the sword as follows:

```
E owner_list~
‘hobgoblin hob green’ ‘goblin gob short’ ‘elf short blond’
~
```

Any player that kills a mob holding said item will have his or her name added to that list, so that they can use the object, but other people can’t.

The last step to making a personalized object is to set the object as speco_personalized in the #SPECIALS section. More info on that in the instructions for SPECIALS.

DAMAGED ARMOR

Using the normal object creation guidelines, any piece of armor that you create starts out in perfect condition. Only after fighting for a while will equipment get damaged. However, it is possible to make equipment that starts out in less than perfect condition.

TABLE F states that Value0, Value1, Value2, and Value3 should all be set to 0 if the item is armor. If you want a piece of armor to start out as damaged, though, you’ll have to change value1. The following chart shows the possible numbers for Value1 and their corresponding damage states:

0	Like New	6	Hammered
1	Broken In	7	Smashed
2	Scratched	8	THRASHED
3	Nicked	9	SHREDED
4	Dented	10	DESTROYED
5	Damaged		

If Value1 is set to 10 or higher, the piece of armor will be destroyed completely the next time it gets damaged. Players can, at any time, restore the armor to like new condition by having it repaired.

Number of repairs

Normally, armor can be repaired a total of ten times. When you create your objects, though, you can choose to set a lower limit for the number of times a piece of armor can be repaired.

Instead of setting Value3 to 0, as indicated in TABLE F, set it to 10 minus the total number of repairs that object can have. (Thus, if you want to limit the total number of repairs for an object to four, set Value3 to 6.) You can make an object that is unrepairable by setting Value3 to 10 or higher.

It is possible to used damaged armor and a limited number of repairs together on a single item. It is even possible to create a very fragile item that cannot be repaired and will be destroyed easily by setting both Value1 and Value3 to 10.

A few last thoughts on objects:

- The keyword for each of you objects had better include every noun and adjective in both the object's short and long descriptions. Few things irritate me as much as seeing "A short blade with a golden hilt lies on the floor", only to find that blade, hilt, short and/or golden cannot be used to pick up the object. If you don't make your list of keywords as thorough as possible, you'd better have a very good reason.

- You don't need to worry about setting the level of you equipment. The mud will automatically generate the level based on the level of the mob on which the equipment loads.

- Even if it seems like it's going to be boring, you should create some mundane, clothing-like equipment for your mobs. Generally, people don't wander around naked. Neither should you mobs, unless they're animals. Create a few tunics and jerkins and such so you humanoid mobs don't have to cover themselves with their bare hands.

- Likewise, create some 'plain' items to leave sitting around the rooms. If you create a woodworkers shop, there should probably be some saws, chisels, and other tools that a character could pick up and take with them, even if they serve no practical purpose in the game.

- Try to include extra descriptions for every keyword of your object. It will make things far more detailed and interesting. You don't have to assign each individual keyword its own description. You can link them all to one or two good extra descriptions.

- Not all objects have to be able to be picked up by players. No-take objects can add new layers of depth and realism to your rooms, even if they are never going to be used for any practical in-game purpose.

- Make sure your items fit within the theme of your area and within the overall blanket theme of the mud. Remember, we are a medieval-fantasy based mud. Guns, cars, walkmans, boxer-briefs, cell phones,

computers, or anything else modern should not be included in your area.

- Its possible to create “hidden” items. Just eliminate the words for the line where you create the object’s long description and use a lone tilde by itself. The object won’t show up on the players’ screens, but can still be manipulated. This works especially well if the room description includes something like “a chest of drawers.” You can create a hidden chest that clever players who read the room descriptions will open to discover the cool contents you put inside. Hidden items like this work best if they are no-take. Do not attempt to put two hidden items in the same room. The result is just plain ugly.

- Another example for the above. Say that you have a series of rooms with a river running through them. You could create an invisible fountain type object with the keyword ‘water’ or ‘river’ and the short description ‘a river’ and put that hidden item into each of those rooms. That way, it would be set up so that a player could drink from the river in any of those rooms.

- While you’re designing your objects, keep in mind that while a dragon probably has piles and piles of valuable goodies, several objects in its hoard are probably worthless junk. It’s all right, and sometimes desirable, to create objects that players won’t drool over.

- Don’t give everything away in the long or short description of an item. A phrase like “a yellow potion of see invisible” is not nearly as interesting or engaging from a game-play perspective as something like “a frothy yellow potion that bubbles.”

- Containers can be used in lots of very creative, clever ways. They don’t even have to be ‘containers’ in the conventional sense of the word. You could create a container called ‘a hole in the wall’ that players can reach into and remove stuff. You could create some eggs and place them in a container called ‘a bird’s nest’. Try naming a container ‘a tree’ and making some fruit to put inside it. You’ve got a fruit tree that players can ‘pick’ fruit from. The possibilities are almost endless. Note, though, that most of these creative containers should be no-take items.

Rooms

The rooms section is arguably the most important section in your area file. Sure, you can create plenty of interesting monsters and items, but without a good rooms section, the whole area is just boring overall. Connecting rooms can be a little bit tricky, since you have to keep very good track of which rooms connect to which and in what direction these connections lie. This is where designing a map of your area ahead of time comes in very handy. Some people tend to repeat room descriptions, giving identical details for multiple rooms. Generally, this reflects very shoddy workmanship. If you serious about building a quality area, you can spend the time it takes to write unique descriptions for each room. The only time that repeated room descriptions are acceptable are in a random maze. (More info on random mazed later.)

Your rooms section will begin with the following line: `#ROOMS` (again, all caps, no other punctuation). This tells the mud that the information that follows contains details on all of the rooms.

After that, follow the same pattern of lines for each room that you want to include in your area.

Line A: #<rnum>

This is the ID number for your room. As you create each room, take careful note of which room number is assigned to which room. When you link your rooms together, you will link them according to the room's ID number. The easiest way to keep from getting confused is to have a map of your area on graph paper with the numbers written in the space for each room. I'll start my example room with the following: '`#101`'.

Line B: <room name>~

This is the name that is displayed to the player above the room's description and above the list of exits. Keep in mind, the room name is to be written as a title. This means, do not use a complete sentence as you title. Do not use ending punctuation. Always capitalize the first word and also capitalize every other word in the room name with the exception of articles and prepositions. To show a title in the appropriate format, my example will be '`In a Dusty Wooden Hallway~`' (with that oh-so-important tilde at the end, but no other ending punctuation).

Line C: <room description>

This is definitely not the place to scrimp creatively. Here, you must write multiple lines of description. Area that are submitted with single sentence room descriptions will be returned to their authors for further work on this section. Once again, close your eyes and picture every detail of the spot you are trying to describe. Think objectively. Describe concrete details that can be experienced through the five senses. The more detail you can include, the better. An example of a decent looking room description in as follows:

```
A wood-lined hallway, roughly eight feet in height, extends away from a thick
oak door set into the southern wall.  A thin, ashen layer of dust coats the
hardwood panels of the floor, but this layer is disturbed in several places by
delicate footprints.  A low bench sits against the western wall next to an
open doorway.  As the musty air swirls about, a set of quiet footprints echoes
by from the north.
```

You should be sure to hit return at the end of each line before the words wrap down to the subsequent line. This will keep the mud from doing funny things to the word wrap of your description and keep things from looking ugly. Be sure that you use complete sentences and that you don't put a tilde at the end of the last line.

Line D: ~

This is where the tilde after your room description goes. It belongs all by itself one line down from the end of your room description.

Line E: 0 <room flags> <sector>

The zero at the start of this line will always be a zero. The mud doesn't actually care what number goes in that spot, so it's not worth wasting brain power to try and think of anything else to use there. Room flags are a series of numbers that define any special characteristics that the room will have. You can find the bits to use for the room flags in TABLE M. If you use more than one flag, be sure to join them with the | character. If you do no wish to use any room flags, simply enter another 0. Sector just a single number that determines the terrain type of the room. It affects how much movement a character must spend to move through the room, and in some cases can restrict which rooms a person may enter. For the sector, simply chose the number from TABLE N that corresponds to the terrain you would like to use and enter

that number at the end of this line. In the room I'm creating here, I'll use '`0 8|8192 1`'. This means that the room is indoors, players cannot recall from this room, and its terrain type is 'city'.

If this is a no-exit room, skip lines F through H. Otherwise, repeat those lines for each of the room's exits.

Line F: D<direction number>

All that you should have on this line is a capital D and a number between 0 and 5 with no spaces between them. The directions correspond to the following numbers: north=0, east=1, south=2, west=3, up=4, and down=5. These values are repeated in TABLE O.

If you have multiple exits, write them in numerical order (i.e. if your exits lead south, east, and up, first write out all of the info for the east exit, then everything for the south exit, and lastly all of the specification for the up exit). In my room description above, I mentioned that the hallway leads to the north, so I'll start the exit in that direction with the following info on this line: '`D0`' (with nothing else at all on the line).

Line G: <exit description>

The exit description is a brief message displayed to the player whenever they look in that particular direction from within the room. Exit descriptions are not required, but are recommended. They are an excellent way to give players hints and warning about what lies nearby. Here, I'll include the line '`The light from a nearby torch fills the hallway with flickering shadows.`' Write the exit description as a complete sentence and do not use a tilde. If you chose not to use an exit description, skip this line completely. Do not even leave a black space in it's place.

Line H: ~

Regardless of whether or not you chose to include an exit description, place a tile all by itself on this line.

Line G: <exit keywords>~

If you wish to use a door for you exit, this is where you will designate one or more keywords that can be used to open, close, lock, or unlock that door. If the exit has no door, there is no need to create keywords, but you still need the tilde. Since my north exit for this room has no door, I'll skip the keywords and just put '~' on this line.

Line H: <door type> <key #> <connected room #>

If the exit has no door, just use a 0 as the first number on this line. If you wish to use a regular door at this exit, start this line with a 1, and if you want to make a pick-proof door, put a 2 as the first number on the line. As of now, there are no other options for doors. The key number is whatever onum is assigned to the object that locks and unlocks this exit. If you have no door here, you don't need a key at all, so just put -1 in this spot. If you do have a door, you are not required to assign a key. If this is a door that is not going to be locked at all, just enter a -1 as the second number. Only enter a key number if this is a door that is normally going to be locked. Lastly, the connected room number is the rnum of the room that this exit leads to. As I mentioned earlier, its best to have a detailed map of your area written out so you can just glance at that to be sure of the room number you wish the exit to lead to. This line for my example will look like: '`0 -1 102`', meaning that there is no door, no key for this exit, and that this exit leads to room 102.

If you need to see an example of an exit that does include a door, just look a little further down. When I assemble all of the parts of my example room, I'll show the info for all of the exits, not just the one to the north.

If your room has more than one exit, repeat lines F through H for each exit.

Lines I, J, and K are optional, although we do encourage you to use them. These lines detail any extra information you might want to give to the more observant players concerning the room. You can include as many of these extras as you want. Simply repeat lines I, J, and K for each one. If you do not want to describe any extra things in the room, skip these lines completely.

Line I: <extra description keywords>~

This tells the mud which keywords will be linked with the extra description you are creating. Whenever a players looks at one of those keywords, they will see the message that you designate. Look back up at the description I wrote earlier for my example room. I mentioned a bench, and someone might be kind of curious about it. I'll write this line as '`bench low~`' (including the tilde) so that anyone who looks at either of those two words will see the extra description.

Line J: <extra description>

This is one or more lines of text displayed to the player when they actively look at one of the keywords listed above. This is an excellent place to give clues about nearby hidden objects or doors, or it can simply be a way to make your rooms more detailed and realistic. Either way, extra descriptions can help make your rooms come alive. I'll use the following paragraph to match with the keywords I designated in the previous line:

```
The finely carved scrollwork of this bench marks it as being of elven
craftsmanship, probably dating back over a hundred years.  Its surface shines
with signs of frequent polishings.  A small scrap of paper sits folded over
near the center of the bench.
```

Note that is this description spills over onto multiple lines, you should be sure to hit return before the words wrap down for each of the lines.

Line K: ~

If you have made an extra description, end it with a tilde on a line all by itself.

At this point you could go on and repeat lines I, J, and K to add another extra description (in this case I could add another description about the paper) or you can simply end the room description with the last item for each room.

Line L: S

All rooms need to finish with a capital S all by itself. Put it on the line after the tilde of your last extra description, or, if you have no extra descriptions, on the line after your last door. Every single room in your area must end with this S.

Just below the S, start over again with the rnum of your next room. If you have no more rooms to write, Just enter '`#0`" after the last `S`.

When we put all of the parts of the rooms section together, it should look something like this:

```
#ROOMS
#101
In a Dusty Wooden Hallway~
A wood-lined hallway, roughly eight feet in height, extends away from a thick
oak door set into the southern wall.  A thin, ashen layer of dust coats the
hardwood panels of the floor, but this layer is disturbed in several places by
delicate footprints.  A low bench sits against the western wall next to an
open doorway.  As the musty air swirls about, a set of quiet footprints echoes
by from the north.
~
0 8|8192 1
D0
The light from a nearby torch fills the hallway with flickering shadows.
~
~
0 -1 102
D2
A door fashioned from heavy oak beams hangs from polished brass hinges.
~
door oak heavy~
0 102 103
D3
A dimly lit sitting room lies beyond an open doorway.
~
~
0 -1 103
E bench low~
The finely carved scrollwork of this bench marks it as being of elven
craftsmanship, probably dating back over a hundred years.  Its surface shines
with signs of frequent polishings.  A small scrap of paper sits folded over
near the center of the bench.
~
E scrap paper~
This delicate slip of paper bears the following message:

   Roger,
     I am deeply sorry that I had to take your keys without telling you first.
   You may retrieve them at your convenience from the bartender at the tavern
   across the road.  Thanks for your understanding.
     -Stephan
~
S
#102
etc. etc. etc.
#0
```

ROOM TABLES

TABLE M - ROOM FLAGS

FLAG	BIT
DARK	1
NO MOB	4
INDOORS	8
NO MAGIC TO	16
NO MAGIC FROM	32
ARENA	64
ANTI-MAGIC	128
PRIVATE	512
SAFE	1024
SOLITARY	2048
PET SHOP	4096
NO RECALL	8192
BANK	16384
NORTH CURRENT	32768
EAST CURRENT	65536
SOUTH CURRENT	131072
WEST CURRENT	262144
UP CURRENT	524288
DOWN CURRENT	1048576

TABLE N - SECTOR TYPES

TERRAIN	VALUE
INSIDE	0
CITY	1
FIELD	2
FOREST	3
HILLS	4
MOUNTAIN	5
WATER (SWIMMABLE)	6
WATER (NON-SWIMMABLE)	7
AIR	9
DESERT	10
UNDERWATER	11

NOTES:
Players will need a fly spell or a vehicle to move through a non-swimmable water room.

Players will need a fly spell to move through air rooms.

Players will need scuba gear to move through an underwater room.

TABLE O - DIRECTIONS

DIRECTION	VALUE
NORTH	0
EAST	1
SOUTH	2
WEST	3
UP	4
DOWN	5

*** ADVANCED ROOM TOPICS ***

Some of the following items might be a bit tricky or confusing for first-time builders. If you'd like to try some, be sure to read the descriptions fully and don't be afraid to ask for help from one of the other more experienced builders.

PET SHOPS

Most shops are easy to make. Simple create a mob for a certain room and then set that mob's characteristics in the SHOPS section. Pet shops are more complicated, though.

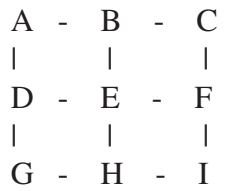
Firstly, you must assign the PET SHOP room flag (4096) to the room that you wish to use as your pet shop. Next, create a special storage room for the pets. This storage room must have the vnum immediately after the room that you created as the pet shop. There should be no exits that lead into or out of the storage room. Lastly, when you set up your mob resets, load the shopkeeper into the pet shop room, and the mobs to be sold as pets into the storage room.

RANDOMIZED MAZES

Random mazes, where all of the rooms have the same description and the exits rearrange themselves whenever the mud resets, can add a new challenge to your area. They make things more difficult for the players, but can also be a serious headache for you if you are trying to correctly incorporate one into your area.

Most random mazes are set up as two dimensional or three dimensional grids. If you want to make a

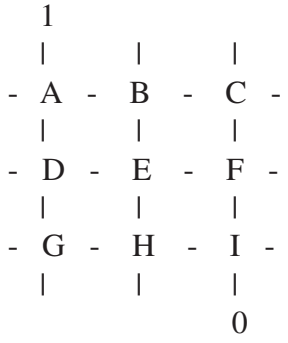
simple two dimensional random maze, start out with a series of identical rooms laid out and connected in a square, just like at the right (dashes show the connections between rooms):



Next, add exits to the outside rooms that "wrap around" to the opposite side. (I.e. C leads east to A, D leads east to F, B leads north to H, I leads south to C, and so on.) This way, each will have four different exits. Once all of the rooms on the outside are connected to each other, a player will be able to travel infinitely far in any direction even though there are only nine rooms.

After this, you need to attach an entrance and an exit to your maze so that players can get in and out.

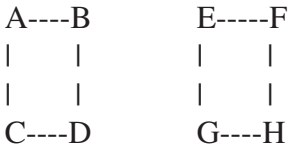
To do this, simply reassign a pair of your outside exits so that they connect the maze to outside rooms. We can take the previous grid and add an entrance room (which I'll label as 1) and an exit room (which I'll label as 0). It should look something like this:



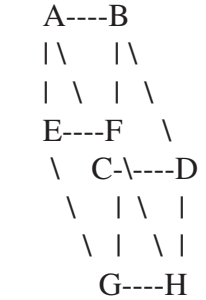
If you've already connect all of the wrap around exits, there are two extraneous exits that the mud won't like. The code doesn't like the idea of being able to head north from C, and then south to 0, or of being able to go south from G, and then north to 1. If you leave the exits like this, the mud will complain. To remedy the situation, simply change the south exit from G so that it leads to C, and the north exit from C so that it leads to G. Now all of the exits line up nicely.

The last step is to assign all of the rooms (A through H) as random rooms in the RESETS section. Once that is done, the exits in those rooms will randomly reassign themselves each time the mud repops. See the RESETS part of the guide for the final details.

If you wish to make a three dimensional maze, the process is basically the same, there are simply more exits to deal with. For the most basic of three dimensional mazes, lets start with two 2 by 2 grids:



Next, arrange things so that A is directly over E, B is directly over F, and so on:



As before, add "wrap around" exits, so that B leads east to A, H leads south to F, D leads up to H, G leads down to C, and so on. Change one of those

exits to an entrance for the maze and one to an exit. Make sure all of the exits balance out. Then finish by setting all of the rooms in the maze as randomized rooms in the RESETS section.

There is no set upper limit for the size of a random maze. It is possible to create a 20 by 20 grid of rooms and use them as a random maze, but it will be almost impossible to navigate. Players will avoid such a place and, if you create a maze like this, we'll probably send the area back to you to reduce the size of the maze.

Also, your maze does not necessarily have to be set up in a square or a cube, nor do all of the rooms have to have the same number of exits. Once you are used to creating randomized mazes, you may wish to experiment with unconventional styles.

A few last thoughts on rooms:

- One of the hardest things to learn about writing good room descriptions is learning the difference between 'telling' and 'showing.' It is rather dull and boring to simply tell a player the reaction that they have upon entering a room. Rather, focus on what brings about that particular reaction. Show the players those details, and let them come to the right conclusions on their own. Don't use phrases like "The hallway looks quite scary." Instead, it sounds much better to say something like "The hallway is filled with deep pockets of shadow, any of which could hide a creature waiting to strike." I think it's pretty clear which sounds better.

- Some muds forbid the use of the word 'you' in the room descriptions. We have no hard and fast rule against its use, but if you do chose to use it, do so sparingly. Especially avoid using it to tell players how they feel or to tell someone that they are performing some unusual action. Players are (usually) independently thinking people who don't like being told outright what to feel and what to do. Also, avoid starting the first line of the room description with the word 'you.' It's a weak beginning, and sounds a little too sloppy.

- Never indicate that a character is travelling in a specific direction in your room descriptions. Few things but me more than seeing a room that says 'you follow the road to the north' or 'a red brick castle lies ahead.' What if I am heading south along that road, or if I have just come from the castle and am now heading away from it? Remember that it is possible to travel through the area in multiple directions, and your descriptions should still fit no matter which route a person takes through your area. Likewise, phrases such as 'this room is warmer than the previous one' are to be avoided, since players won't always have entered from the same direction.

- Be sure to have a good thesaurus handy when you are writing your room descriptions. Too many builders develop a small group of words that they like to use, and then they use them over and over and over. Unless you're careful, you'll probably end up using certain words way too often in your writing. Words like 'dark' and 'small' tend to be too overused. Finding some good synonyms can avoid problems of repetition and can add an interesting, more literate feel to your rooms.

- When you wish to create a door, you must define it twice, once for each side. In other words, if you want a door that lies between room A and room B to the east, first you have to define a door for the east exit in room A. Then, you must define a door for the west exit in room B. The mud won't automatically create the door in both directions for you.

- Be sure to use up and down exits in your area wherever they fit. Too many people create flat areas that could easily be more interesting if they included a few rooms on an upper or lower level.

- Do what you can to insure that the rooms connect to one another in a logical manner. In other words, if you start a room 1, then go south, then east, then north, then west, you should end up back in room one. Players should easily be able to map your area with graph paper if they want to. At times, this rule can be broken if it fits the area, but you had better have a good reason for doing so.

- Make your room descriptions as detailed as possible, but don't make them too long. If all of your room descriptions are eight to ten lines or more in length, players will start to ignore what you've written. If you've got a lot of information, try putting some of it into extra descriptions.

- Avoid no exit rooms that players must recall from. People hate them and they are a good way to drive players away from your area.

- Do not describe any mobiles that load into the room in your room descriptions. It looks really odd to see a room description that says that a giant is guarding the door even after you've just killed said giant. Save descriptions of creatures for the MOBS section.

- ASCII art looks out of place in a room description. If you feel that you must use some, save it for the extra descriptions.

- Remember that a character can see and hear not only what is in their immediate surroundings, but also what lies farther away, particularly when they are outside. Don't be afraid to create a few major landmarks in your area that can be seen from several different places. It can also be possible to see in to surrounding areas, particularly if your area lies near a major city.

- Be aware of the flow of the terrain through your area. Deserts should not logically be right next to a swamp. If you wish to include multiple terrains in your area, be sure that they mix together in a way that makes sense.

Resets

The RESETS section tells the mud what the state of the area should be whenever the mud starts up or goes through a repop. In this section, you will specify which mobs load where, what equipment they are wearing, which doors will be locked, which rooms will have random exits, and other details as well. Most lines in this section will follow the same basic format, but you'll be working with many, many numbers. As such, you'll need to keep careful track of all of your mnums, onums, and rnums so that things work out the way you intend.

The mud executes each line of your resets when it first loads. It also checks each line whenever it repops. During a repop, it may run a particular reset, or may skip over it. I'll point out which ones may be skipped as I describe each type of reset below.

The first line of this section must look like '#RESETS' so that the mud knows to begin the resets section.

Each of the following lines will follow the same basic format:

<reset type> 0 <value1> <value2> <value3> <comment>

The reset type will always be a capital letter, and values 1, 2, and 3 will be numbers. Comments at the end of the line are optional, but recommended. Comments are generally used to remind yourself (or anyone else reading your area file) what each reset is supposed to do so that you can easily track down any resets that might be causing problems.

Below, you'll find details and examples for each type of reset that you can use. (Note that any 0 listed in the details for a particular reset will be ignored by the mud, but is still necessary.)

MOB RESETS

M 0 <mnum> <total number in the mud> <rnum> <comment>

The mnum tells the mud which mob to load. The total number tells how many of that particular mob belong in the mod. At each repop, the mud counts how many of this mob already exist. If the there are as many or more than the number listed here, the mud will skip this particular reset. Otherwise, it will load one copy of this mob. The rnum tells the mud which room the mob should load into. Finally, the comment is your change to leave yourself notes on what this reset line is supposed to do.

EQUIP RESETS

E 0 <onum> 0 <wear location> <comment>

This reset tells the mud to equip a mob with a certain piece of equipment. The onum tells the mud which particular piece of equipment is to be placed onto a mob. The wear location tells the mud which equipment slot to load the item to. (See the chart below for the list of equipment slots and there corresponding numbers.) Again, the comment is where you can leave yourself any pertinent notes.

WEAR LOCATIONS	VALUE
LIGHT	0
LEFT FINGER	1
RIGHT FINGER	2
NECK (1st slot)	3
NECK (2nd slot)	4
BODY	5
HEAD	6
LEGS	7
FEET	8
HANDS	9
ARMS	10
OFFHAND	11
ABOUT BODY	12
WAIST	13
LEFT WRIST	14
RIGHT WRIST	15
WIELD	16
HOLD	17

Note that an equip reset must come just after the mob you wish to equip.

GIVE RESETS

G 0 <onum> 0 0 <comment>

A give reset loads a particular piece of equipment into a mob's inventory. The onum tells the mud which item to put among the items that the mob has. The only other detail to worry about on this line is the comment at the end. Like the equip reset, a give reset must come just after the mob that is to receive the item.

Equip and give resets are only executed when a mob is actually loaded into the mud. If the preceding mob does not load for any reason, the mud will skip the G and E resets.

OBJECT RESETS

O 0 <onum> 0 <rnum> <comment>

This type of reset will load an object that will be placed on the ground in a particular room. The onum tells the mud which object to load, while the rnum tells the mud in which room it should load the object. The mud will ignore this reset if a copy of the object already exists in the room. Thus, you cannot load two copies of a particular objects into a room.

PUT RESETS

P 0 <contents onum> 0 <container onum> <comment>

Put resets load a certain item to the interior of a container. The contents onum tells the mud what object to load. The container onum dictates which container will receive the contents. If the object you designate to receive the contents is not actually a container, the mud will simply ignore this reset. Also, like an O reset, the mud will skip this reset if there is already a copy of the contents object inside the container. As such, two or more copies of the same object cannot load into a single container.

Also, be aware that the mud cannot tell the difference between multiple copies of the same container. This, if you create one single chest and instruct the mud to load that chest in two different locations, you will not be able to put items into both chests. Avoid using multiple containers with the same onum.

It doesn't really matter where in the RESETS section you place your P resets. The only thing you have to be careful about is that the container in question must load somewhere higher up the list of resets that the object you wish to place into it.

DOOR RESETS

D 0 <rnum> (direction> <door state> <comment>

A door reset tells the mud what state a particular door should be in whenever the mud repops. The rnum tells the mud what room this reset will affect. The direction is a number between 0 and 5 that tells the mud which exit direction you wish to set. Exits directions correspond to the following numbers:

DIRECTION	**NUMBER**
NORTH	0
EAST	1
SOUTH	2
WEST	3
UP	4
DOWN	5

The door state dictates whether the door should normally be open, closed, or closed and locked. Whenever the mud repops, the door will return to this default state. Use the following to set the state of the door:

DOOR STATE	**NUMBER**
OPEN	0
CLOSED, UNLOCKED	1
CLOSED, LOCKED	2

You must set the reset state for both sides of any particular door. Thus, if there is a door that closes off an exit between rooms A and B, you must create one reset for the door that leads from A to B, and another reset for the door that leads from B to A.

Also, be aware that if you try and set the door state for an exit that has no door, the mud will crash.

Make sure that any door that you try and set with a door reset has already been defined as a door in line H of your ROOMS section.

RANDOMIZE RESETS

R 0 <rnum> <highest exit number> 0 <comment>

The randomize reset is used to rearrange the exits in a room to create a random maze. Obviously, the rnum is the number of the room whose exits you would like to randomize. When the mud rearranges the exits in a room, it starts with the north exit, and mixes up all the exits up to and including the exit with the highest number that you designate. (If you need to remember which exits have which numbers, look back a few paragraphs.) If you have a flat, two dimensional maze, you should set the highest exit number as '3'. This will make the mud rearrange all of the exits between north (0) and west (3). For a three dimensional maze, set the highest exit number to '5', so the mud will rearrange the exits in every direction at each repop.

COMMENT LINES

Any line in the RESETS section that begins with an asterict (*) will be ignored by the mud. This means you can dedicate a line entirely to comments by beginning with an asterict. It is not necessary to have comment lines, but we highly recommend that you do. Comment lines can be used to group certain resets together to make this section easier to read and debug, should the need arise.

IMPORTANT - After the last of your resets, type 'S' on a line all by itself.

A typical RESETS section might look like the following:

```
#RESETS
* room 101
M 0 101 2 101       load a copy of mob 101 into room 101
E 0 101 0 16        mob 101 wields object 101
E 0 102 0 6         mob 101 wears object 102 on head
O 0 103 0 101       Load object 103 to the floor of room 101
P 0 104 0 103       Put object 104 into the contents of object 103
D 0 101 2 2         close and lock the southern door in room 101
*
* room 102
R 0 102 3 0         randomize north, south, east, and west exits in room 102
M 0 102 2 102       load a copy of mob 102 into room 102
E 0 105 0 17        mob 102 holds object 105
G 0 106 0 0         put object 106 in mob 102's inventory
*
* room 103
M 0 101 2 103       load a copy of mob 101 to room 103
E 0 107 0 8         mob wears object 107 on it's feet
G 0 108 0 0         put object 108 in mob 101's inventory
D 0 103 0 2         close and lock the northern door in room 103
S
```

That's it. Keep in mind, though, that your resets section will probably be a lot longer, but it should still look basically the same.

Shops

In the shops section, you will designate which of your mobs will be shopkeepers, as well as what items they will buy, when they will conduct business, and other important information.

Begin this section with the following on its own line: #SHOPS

Each subsequent line will define a particular shopkeeper and must contain the following information:

<mnum> <type0> <type1> <type2> <type3> <type4> <%sells> <%buys> <open> <close>

Mnum tells the mud which particular mob you wish to set as a shopkeeper.

Type0 through type4 determines which types of items a shopkeeper will buy. Use the object type numbers from TABLE C to define these values. For example, if you want the shopkeeper to buy wands, set one of these values to 3. If you would like it to buy armor, set one of the values to 9. You chose to have your shopkeeper buy up to five different types of items. If you do not wish to use all five of these fields, simply enter a zero (0) for any of these fields you chose not to use.

Be aware that you do not need to tell the mud what types of objects a shopkeeper will buy. It will automatically sell any item that is in its inventory. Thus, it is possible to create a shopkeeper that will refuse to buy anything, but will still sell any object that normally loads into its inventory.

%sells and %buys will affect the price that a shopkeeper will offer for of charge for an item. The mud naturally assigns a monetary value to all items. If you would like the shopkeeper charge this normal value for anything he sells, set the %sells to 100 (100% of the normal value). If you want to shopkeeper to mark up prices, set the %sells to a higher value (for example, a %sells of 120 means that the shopkeeper will mark up all values by 20 per cent). Likewise, a number less than 100 means that the shopkeeper will sell items at a discount. Naturally, %buys affects the amount of money that a shopkeeper will offer when it buys an item. If you set %buys to 100, the shopkeeper will pay the full normal value when it buys an item. If %buys is set to 50, it will only offer half of the objects normal value. Not that you should never, ever set %sell to a lower number than %buys.

Open and close tell the mud what time mob will start and stop buying and selling items. All times are based on a 24 hour clock with 0 representing midnight and 12 representing noon. If you wish the shopkeeper to do business between the hours of 7 AM and 10 PM, you would set the open to 7 and the close to 22.

After the last shopkeeper you wish to define, type a zero (0) all by itself on its own line.

Once everything is put together, your SHOPS section might look like this:

```
#SHOPS
110 2 10 3 4 0 125 85 10 18
115 5 9 0 0 0 110 65 8 20
0
```

This would mean that mob 110 buys scrolls, potions, wands, and staves, sells them at a 25% profit, pays 85% of full value for them, and is open between the hours of 10 AM and 6 PM. Mob 115 buy weapons and armor, sells them for a 10% profit, pays 65% of full value for them, and is open between the hours of 8 AM and 8 PM.

If you wish to create a shopkeeper that sells pets, define them as normal in this section, but do not set any object types for it to buy. Then, make sure that this particular shopkeeper loads into a room that has been set as a pet store in the ROOMS section.

Lastly, be aware that on our mud, any mob that you designate as a shopkeeper cannot be attacked by players. If you want a certain mob to be killable, do not make it a shopkeeper.

If you don not want to include any shopkeepers in your area, you still must have a SHOPS section. Simply begin with the opening line (#SHOPS) and follow that with the number zero (0) on a line by itself.

Specials

In the SPECIALS section, the final section of your area format, you can assign certain special attributes to any of the mobs and objects in your area. You can chose to have certain mobs cast spells during combat or interact with players in unusual ways, and you can make objects do unusual things to a player that uses them.

The opening line of this section must be '`#SPECIALS`' all by itself. Each subsequent will consist of either a mob special or an object special.

If you want to assign a special function to a mob, the line will follow this format:

M <mnum> spec_<special type> comment

The mnum is the number of the mob to which you wish to assign the special function. The special type determines what unusual properties this mob will exhibit. (Note that the underscore (_) after the word spec is necessary.) As an example:

```
M 115 spec_breath_fire
```

would give mob 115 the ability to breathe fire in combat. A complete list of specials available for mobs will come later in this section.

For special functions assigned to objects, use the following format for the line:

O <onum> speco_<special type> comment

The onum tell the mud which object to assign the special function to. Again, the underscore is needed in each line. A line written as:

```
O 120 speco_social
```

would mean that object 120 will make any character that holds it execute random socials. There will also be a complete list of object socials later in this section.

It does not matter what order the lines in your SPECIALS section are written in. Although most people arrange them in the order of mnums and onums, there is no reason that this section must be written this way.

After the last special you wish to assign, write '`#$`' on a line all by itself. This will be the last line of your area file. If you do not wish to assign any of these special function, the you still need to write '`#SPECIALS`' and '`#$`' on their own lines. The mud expects there to be a specials section even if it does not contain anything.

Be aware that mobs and object cannot have more than one special assigned to them. If you try to assign multiple specials, the mud will ignore all but the last one.

****
The following specials are available for use with mobs:

spec_align_change

The mob will change a character's alignment during a fight.

spec_breath_any

The mob will be able to use all five of the available breath weapon spells during combat.

spec_breath_acid

The mob can spit acid onto a character during combat.

spec_breath_fire

The mob can breathe fire as a attack in combat.

spec_breath_frost

The mob can breathe a cone of frost during a fight.

spec_breath_gas

The mob can breathe toxic gas as an area attack during combat.

spec_breath_lightning

The mob can shoot lightning bolts from its mouth during combat.

spec_buddha

The mob will be affected by both spec_breath_any and spec_cast_cleric.

spec_cast_adept

The mob will randomly cast armor, bless, cure blindness, cure poison, cure light, and refresh on characters of up to level 10.

spec_cast_cleric

The mob can use blindness, cause serious, earthquake, cause critical, dispel evil, curse, change sex, flamestrike, harm, and dispel magic against a player during combat.

spec_cast_judge

The mob can cast the spell ‘high explosive’ in combat.

spec_cast_mage

The mob will cast blindness, chill touch, weaken, teleport, color spray, change sex, energy drain, fireball, and acid blast during a fight.

spec_cast_undead

> The mob can use curse, weaken, chill touch, blindness, poison, energy drain, harm, and teleport while in combat.

spec_executioner

> The mod will push players labeled as killers and thieves out of the room. It will also clean up blood stains.

spec_fido

> The mob will eat any corpses that it finds.

spec_grue

> The mob will destroy any light that the character is using or holding in their inventory.

spec_guard

> The mob will attack players labeled as killers and thieves. It will also join an existing fight on the side of whichever fighter has the higher alignment.

spec_hate_avian, spec_hate_druid, spec_hate_elf, spec_hate_human, spec_hate_illithid, spec_hate_kender

> The mob will automatically attack any character of the indicated race that it sees.

spec_hate_evil, spec_hate_good, spec_hate_neutral

> The mob fights any character it finds that falls within the given alignment range.

spec_janitor

> The mob picks up any objects that it finds on the ground. It will also clean up blood trails.

spec_kungfu_poison

> The mob can use the ‘poison palm’ technique to poison a character during combat.

spec_love_evil, spec_love_good, spec_love_neutral

> The mob fights any character it finds that does not fall within the given alignment range.

spec_poison

> The mob has a poisonous bit that is can use to poison a character during a fight.

spec_thief

The mob will attempt to steal gold from any character in the same room.

****
The following specials are available for use with objects:

speco_airfill

The object can be used to refill the air in scuba type objects.
*This special is only to be assigned to no-take objects.

speco_attach

The object will automatically jump into a character's inventory and attempt to force the characters to wear said object.

speco_burper

When equipped, the object will cause the character to use the 'burp' social at random intervals.

speco_drain_hp

The object will cause physical damage to any character that uses it as equipment.

speco_drunker

The object will periodically make any character that uses it drunk.

speco_recycler

This special is for use with containers. If a character put a trash type item into the container, the object will be destroyed and the container will create a small amount of gold in return.
*This special is only to be assigned to no-take objects.

speco_social

Any character that uses this object or has it in their inventory will execute random socials.

speco_personalized

Limits which characters will be able to use the object. See the OBJECTS section for more information.

If your area needs some sort of special function to assign to a mob or an object that is not covered above, it is possible for us to add new specials to the mud. The easiest way to accomplish this is to write a new special yourself. Of course, this isn't always an option since not everyone has the programming know-how. If there is a new special that you absolutely need, but cannot create by yourself, talk to the coders on the mud. It is possible for us to create new specials for you, but we aware - our coders are very busy people who are usually tied up with other important business for the mud. The special that you might need would have to been very, very essential to the mud for us to be able to add it to the to-do list of our coders. It never hurts to ask if it's possible though, as long as you ask nicely and are willing to accept rejection.

The '#$' right after the SPECIALS section marks the end of your area file. Once everything is done, all you have left to do is look for any errors and submit the finished process.

If you would like to see what a full and complete area file looks like, please see the appendix to this file.

Part III
after you've finished
the area file

After many long hours of work and frustration, you've finally completed your area. Typing the '#$' at the end of the area file certainly gives you a great sense that you can finally take a break and relax. There still is a little more work to do, though.

Despite your best efforts, some spelling and grammar mistakes have undoubtedly slipped past you. Look through your area once again from start to finish. Do not rely on your computer's spell checker to find mistakes for you. Dozens of misspellings, if not more, will slip past. Take the time to actually read through everything line by line. Not only will you find many of your spelling and grammar mistakes, you might even find some ways to rewrite and improve your descriptions.

Yes, I know that spelling mistakes are not uncommon on the mud, but we want to keep them to a minimum. If you submit an area with too many blatant spelling and grammar errors, we will send the file back to you and ask you to clean up the writing yourself. One of the biggest favors that you can do for yourself is to make sure that all of your writing looks good before you submit the file you have written.

In addition to spell checking your area, you should try to debug the file as well. If you access to a copy of Merc, you can try to add your area into the collection of areas included with Merc. Running the program will automatically search the area for any errors and tell you not only if it runs onto any problems, but also where in the area file it found those problems.

If you don't have your own copy of Merc, you should still look through the area file to determine if you have omitted any tildes or zeros. Any missing characters could cause the mud to crash. If you have no access to Merc yourself, we can run your area through the debugging process ourselves, but that will probably delay the implementation of your area.

Once you've checked and double checked the area file (which should, if you remember, be saved as a text only file) send a copy to Kiri. (Again, you can find her current e-mail address by profiling her on Barren Realms.) Then, sit back and wait. Your area will be examined by one or more of the other builders on the mud. If there are any problems with your area, we'll let you know what they are and return the area to you so that you can fix them. Problems may include, but are certainly not limited to: too many mobs, too few mobs, overpowered objects, poorly connected rooms, sub-standard descriptions, and descriptions that are repeated to often. If your area is returned to you for any reason, don't be discouraged. We strive for high quality areas on the mud and we are committed to helping you write the best areas that you can.

Before long, your area will most likely be accepted, either upon its first submission or after one or more revisions. At this point, we will work with you to find the best spot for connecting your area with the rest of the mud. We will do our best to work with you, but be aware that we cannot always honor your exact wishes.

Soon, your area will be added to the mud, and you and everyone else on the mud can enjoy your handiwork. It may take a few days before your area is finally added, or it may take a few weeks. Be patient. It's worth the wait.

A few disclaimers.

Just because you complete and submit an area does not mean that we are obligated to add it to the mud. We would love to add everyone's work to the mud, but we also have certain standards to maintain. Occasionally, there will be problems with an area that simply cannot be fixed. We cannot guarantee that every area submitted to us will be added to the mud.

After an area has been added to the mud, it may become necessary for us to make certain changes. Spelling mistakes may be found, or we may discover that your area if unbalanced for some reason. For the well being of the mud, we will correct any problems that we find. Assuming that you have provided us with up-to-date contact info, we will work with you to implement these changes. If we cannot reach you, we will still go ahead with necessary changes to your area as we see fit.

Once an area is submitted to us, the administration of Barren Realms will have final say over whether an us added or removed from the mud. Occasionally, it may become necessary to remove an area for any number of reasons. Once again, if you have provided us with accurate contact info, we will let you know if your area must be removed.

Now, a few words about the completeness and of this guide. I've done my best to cover all possibilities in area building, but be aware that new code is always being developed for Barren Realms. By the time you read this, there may be new spells, or new options for mobs and objects. Whenever any significant changes are made to the mud, we will update this guide to reflect those changes. These changes may take some time, though. If you know of something that has been added to the mud that you don't see reflected in this guide, by all means ask about it.

Lastly we value the what you can bring to us as a builder. Anyone who is interested should talk to us about any ideas you might have. Questions are welcomed and encouraged. We're always on the lookout for new builders and would love the opportunity to add someone to the team.

DIKUMUD LICENSE:

****
DikuMud License
****

Program & Concept created by

Sebastian Hammer
Prss. Maries Alle 15, 1
1908 Frb. C.
DENMARK
(email quinn@freja.diku.dk)

Michael Seifert
Nr. Soeg. 37C, 1, doer 3
1370 Copenhagen K.
DENMARK
(email seifert@freja.diku.dk)

Hans Henrik St}rfeldt
Langs} 19
3500 V{rl|se
DENMARK
email bombman@freja.diku.dk

Tom Madsen
R|de Mellemvej 94B, 64
2300 Copenhagen S.
DENMARK
(email noop@freja.diku.dk)

Katja Nyboe
Kildeg}rdsvej 2
2900 Hellerup
31 62 82 84
DENMARK
(email katz@freja.diku.dk)

This document contains the rules by which you can use, alter or publish parts of DikuMud. DikuMud has been created by the above five listed persons in their spare time, at DIKU (Computer Science Institute at Copenhagen University). You are legally bound to follow the rules described in this document.

==========
* Rules: *
==========

****
!! DikuMud is NOT Public Domain, shareware, careware or the like !!
****

You may under no circumstances make profit on *ANY* part of DikuMud in any possible way. You may under no circumstances charge money for distributing any part of DikuMud - this includes the usual $5 charge for “sending the disk” or “just for the disk” etc.
By breaking these rules you violate the agreement between us and the University, and hence will be sued.
You may not remove any copyright notices from any of the documents or

sources given to you.
This license must *always* be included “as is” if you copy or give away any part of DikuMud (which is to be done as described in this document).
If you publish *any* part of DikuMud, we as creators must appear in the article, and the article must be clearly copyrighted subject to this license. Before publishing you must first send us a message, by snail-mail or e-mail, and inform us what, where and when you are publishing (remember to include your address, name etc.)
If you wish to setup a version of DikuMud on any computer system, you must send us a message , by snail-mail or e-mail, and inform us where and when you are running the game. (remember to include your address, name etc.)
Any running version of DikuMud must include our names in the login sequence. Furthermore the “credits” command shall always contain our name, addresses, and a notice which states we have created DikuMud.
You are allowed to alter DikuMud, source and documentation as long as you do not violate any of the above stated rules.
Regards,

The DikuMud Group

Note:

****
We hope you will enjoy DikuMud, and encourage you to send us any reports on bugs (when you find ‘it’). Remember that we are all using our spare time to write and improve DikuMud, bugs, etc. - and changes will take their time. We have so far put extremely many programming hours into this project. If you make any major improvements on DikuMud we would be happy to hear from you. As you will naturally honor the above rules, you will receive new updates and improvements made to the game.
****

Subscribe to the barren realms newsletter by emailing contacting Kiri.
For the builders forum, email barren@chaos.coredcs.com.
-
-
| Barren Realms: barren.coredcs.com 8000 |
-

APPENDIX
sample area file

The following is an example of a complete area file. It is rather small, but it shows all parts of the file combined together. Use this a a reference as you are working on your area file.

```
#AREA   { 5 15} Faustus Dragon Egg Tavern~


#HELPS
-0
tavern~
The Dragon Egg Tavern is written as a small example area to accompany
the Barren Realms builder's guide.  Chances are, this area will never
actually be implemented anywhere, so this help file will most likely
never been read on the mud.  Nevertheless, this shows you what a help
file looks like.
0 $~


#MOBILES
#501
klaus bartender tall man heavy~
Klaus the bartender~
A tall, heavyset man stands behind the bar, awaiting orders.
~
Klaus stands here with a jolly countenance and a gleam in his eye.  He
takes a moment to wipe his hands on his dirty apron and then turns to
laugh at a joke made by some other patron.  As he moves about behind the
bar, it is possible to see the powerful muscles that hide beneath his
portly body.
~
1|2|4 4|32 750 S
99 0 0 0d0+0 0d0+0
0 0 0 0 1
#502
cook shabby short~
a shabby cook~
A short cook scurries about the area, busily preparing food.
~
This short man regards you with a gruff grunt as you look at him, then
returns his attention to the half-prepared food before him.  His dark is
pulled back into a tight knot and he wears a thin, greasy apron.
Despite the cramped quarters, he moves about with ease.
~
1|2 0 500 S
11 0 0 0d0+0 0d0+0
0 0 0 0 1
#503
patron hungry stout man~
a hungry patron~
A stout man is here, waiting to be served.
~
```

```
Faint grumbling sounds are audible from the stomach of this portly man.
He paces back and forth, impatiently looking around for something to
eat.  He pauses for a moment to wipe some sweat from beneath his
thinning brow.
~
1 0 150 S
5 0 0 0d0+0 0d0+0
0 0 0 0 1
#504
patron quiet young woman~
a quiet patron~
A young woman standing nearby watches everyone around her.
~
This woman gazes about, her face mostly covered by her long, curly hair.
She stands rather still and does not draw much attention to herself.
Her eyes dart over to the bar for a moment, but her face betrays no sign
of emotion.
~
1 0 150 S
8 0 0 0d0+0 0d0+0
0 0 0 0 2
#505
smell rancid rotten cloud stink~
a rancid smell~
A rotten cloud of yellowish stink permeates the room.
~
The yellowish cloud swirls amorphously through the air, despite the fact
that no fresh air blows into the room.  When it comes into contact with
the water on the floor, the water starts to sizzle and bubble slightly.
This cloud moves about almost as if it is seeking out life.
~
1|2|8|32 8|32 -750 S
15 0 0 0d0+0 0d0+0
0 0 0 0 0
#0


#OBJECTS
#501
beef plate slices~
a plate of beef~
A plate covered with slices of beef sits here.~
~
19 0 1
25 0 0 0
3 0 0
E beef plate slices~
Several slices of beef, covered in a thick, juicy sauce, lie atop a
cracked clay plate.  A rich aroma rises from the meat, tantalizing those
in the vicinity.
~
#502
key worn scratched~
a worn key~
A scratched and worn key lies here, discarded.~
~
18 0 1
0 0 0 0
1 0 0
E key worn scratched~
This key is about three inches in length.  Several deep scratches run
from one end to the other.  Despite these marks, the key feels quite
solid and sturdy.
~
```

```
#503
ale bottle brown liquid~
a bottle of ale~
A dark brown bottle filled with amber liquid sits on the floor.~
~
10 1 1
15 28 1 36
2 125 0
E ale bottle brown liquid~
The liquid in this bottle fizzes with rolling effervescence.  A rich,
heady aroma escapes from within and fills the air.
~
#504
cleaver meat knife large~
a meat cleaver~
A large knife has been stuck into the ground.~
~
5 2|64 1|8192
0 0 0 1
5 0 0
E cleaver meat knife large~
The ten-inch long blade of this cleaver has been sharpened to a finely
honed edge.  It is capable of hewing easily through meat, shaving off
paper-thin slices.
~
A 18 2
A 19 1
#505
tunic brown loose pile fabric~
a loose brown tunic~
A pile of brown fabric has fallen in a rumpled heap.~
~
9 0 1|8
0 0 0 0
5 0 0
E tunic brown loose pile fabric~
Despite the rough weave of the fabric, this tunic drapes easily about
the body, making it rather comfortable.
~
A 5 1
A 17 10
#506
boots muddy pair~
a pair of muddy boots~
A pair of boots coated in mud lies nearby.~
~
9 0 1|64
0 0 0 0
4 0 0
E boots muddy pair~
These boots are made with stout, thick soles designed to withstand a
great deal of wear.  Most of the mud that coats these boots is old and
dry, but a few spots are still slick.
~
A 2 1
A 13 10
#507
coatrack coat rack tall~
a coatrack~
A tall coatrack stands in the corner.~
~
12 0 0
0 0 0 0
25 0 0
```

```
E coatrack coat rack tall~
A wooden pole, slightly taller than the average human, rests upon a
sturdy base.  A pair of tan cloaks hang from small hooks atop the rack.
~
#508
stove iron cast black~
a cast iron stove~
A large black stove is placed against a wall.~
~
15 0 0
100 1|4 -1 0
200 0 0
E stove iron cast black~
Drops of grease coat the surface of this stove, attesting to its
frequent use.  The hinges on its door look oiled and well maintained,
though.
~
#509
pork roast roasted piece~
a roast pork~
A roasted piece of pork sits here with steam rising from it.~
~
26 2|64 1
20 19 74 36
2 250 0
E pork roast roasted piece~
A light glaze glistens on the surface of this juicy meat.  It has a rich
golden color and smells delicious.
~
#510
toilet old broken cracked~
an old, broken toilet~
A cracked toilet stands in the back corner.~
~
15 0 0
150 0 -1 0
1000 0 0
E toilet old broken cracked~
This toilet is badly stained, and there appears to be something growing
on one side.  Its interior has long since dried out and currently stands
empty.
~
#0


#ROOMS
#501
On a Muddy City Street~
The dirt roads of the city have become slick with mud following a recent
rainstorm.  This particular stretch is especially messy, as the mud has
been churned up by passing pedestrians and horses.  On the western edge
of the street, a few wooden steps lead up to a one-story building.  A
crude, but welcoming sign hangs outside of the door to this building,
offering refuge from the muck and dirt of the streets.
~
0 4 1
D3
A heavy wooden door provides access to a nearby building.
~
door wooden~
1 -1 502
E sign crude~
The sign hangs from a pair of rusty iron chains that creak softly as the
sign sways in the breeze.  A set of letters carved into the sign reads
```

“Welcome travelers to the Dragon Egg Tavern.” Next to these letters, someone has painted a colorful egg from which protrudes the brightly painted head of a dragon.
~
S
#502
The Common Room of the Tavern~
The sounds of glasses clinking together, of forks scraping against plates, and of jovial conversation fill the air. Warm air fills the air of this tavern, creating a comfortable, welcoming atmosphere. To the east, a thick wooden door leads out to the neighboring street. Inside the tavern, there is a bar to the west and a few tables that sit to the north. Underfoot, a few mud streaks mark the wooden floor panels.
~
0 8 0
D0
A few tables sit in the corner to the north.
~
~
0 -1 505
D1
A wooden door, marked by a few scratches, is set into the east wall.
~
door wooden thick~
1 -1 501
D3
A bar stands against the western wall of the tavern.
~
~
0 -1 503
E floor panel panels mud streak streaks~
Much of the mud on the floor has been tracked in from the street. Boot prints, some still slick with wet mud, track from the door and lead both to the north and to the west.
~
S
#503
In Front of a Tall Bar~
Several mugs of ale, some dented, others overflowing with froth, sit atop the nearby oak bar. Large barrels sit behind the bar against the back wall of the tavern. A shelf that stands above these barrels holds a small collection of cheap wine bottles. The air in this corner of the tavern holds a mixture of alcohol fumes and roasted meat aromas. A pair of swinging doors to the south leads into a bustling kitchen, the source of these cooking smells.
~
0 8 0
D0
A few tables have been set up in a corner to the north.
~
~
0 -1 506
D1
East of here is the entrance to the tavern.
~
~
0 -1 502
D2
The swinging doors sway gently to and fro on their hinges.
~
door doors swinging~
1 -1 504
E barrel barrels~
The barrels behind the bar are three to four feet in diameter each.

Amber drop of ale slowly drip from the poorly closed tap of the largest
of these barrels.
~
E shelf wine bottle bottles~
Many of these bottles are covered in a layer of gritty dust. Only two
of the bottles appear to have beet touched in the past year. A close
look reveals cobwebs clinging to a few bottles.
~
S
#504
A Cramped Kitchen~
This room, no more than a scant few square feet, is packed with cooking
implements. A cast iron stove takes up much of the space against the
southern wall, while the other walls are occupied by tables covered with
pots, pans, and dishes. One of the tables also holds a few chunks of
bread and various slices of meat. The single small window set high
into the west wall does little to provide ventilation, but some fresh
air does filter in through the swinging doors to the east.
~
0 8 0
D0
A pair of swinging doors leads to the heart of the tavern.
~
pair doors swinging door~
1 -1 503
E bread chunk chunks~
This bread is heavy and has quite a thick crust, but it looks fresh and
filling. A freshly baked aroma rises from the bread to mingle with the
aromas in the room.
~
E meat slice slices~
Warm, juicy drippings seep from the surface of the meat, puddling in a
few spots on the table top. The aroma of this deep-brown, well roasted
meat is positively mouth-watering.
~
S
#505
Inside the Tavern~
Candles that sit atop the tables in this corner cast in dim light upon
the surroundings. A few patrons sit around two of the tables, engaged
in a deep conversation. The wooden floorboards are strewn with bits of
food, and spilled drinks have stained numerous spots. After a moment,
the conversation grows louder and more urgent. A large painting hangs
upon the northern wall, taking up most of the available space.
~
0 8 0
D2
To the south lies the entrance of the tavern.
~
~
0 -1 502
D3
A dim corner of the tavern sits nearby.
~
~
0 -1 506
E patron patrons~
The patrons are made up of a mixture of adventuring types and city
dwellers. After a moment, the voices of some of these patrons rise to a
shout. The yell is accompanied by the sharp crash of breaking glass.
Quickly, though, the atmosphere calms down and returns to normal.
~
E painting~
This rough paint sketch depicts a brightly speckled egg sitting in a

straw nest. The egg is cracked open on one side, and the scaly head of
an infant dragon pokes tentatively through the opening.
~
S
#506
The Back of the Tavern~
A few unoccupied tables have been placed about randomly in this corner
of the tavern. The patrons and employees of the tavern seem to shun
this area. Periodically one wanders by, but leaves quickly in an
uncomfortable silence. The dark wooden floorboards have not been swept
in quite some time. One of the tables partially blocks a door in the
northern wall with a small paper sign hanging from it. An unpleasant
odor seeps in from the direction of this door.
~
0 8 0
D0
This thick door is partially blocked by one of the tables.
~
door thick~
1 502 507
D1
Several patrons sit at the tables to the east.
~
~
0 -1 505
D2
A bar, offering several drinks, stands nearby.
~
~
0 -1 503
E sign paper~
Someone has nailed a piece of paper to the door. In flowing script, it
says that “This room has been closed by order of the city.” Underneath
this writing, a shakier handwritten script reads “See the bartender if
you really need to get in. Enter at your own risk.”
~
S
#507
A Filthy Restroom~
This small, cramped chamber had probably once been used as a restroom,
but untold neglect has transformed it into something unspeakable. An
inch of brackish, foul-smelling water covers the floor, staining the
bottoms of each wall. The remnants of a wooden stall, now smashed and
broken, rest against the northern and western walls. An awful odor
fills the small room. A door in the southern wall offers a chance to
escape from here.
~
0 1|8|512|8192 0
D2
This door offers the only route out of here.
~
door~
1 502 506
E water~
An unidentifiable oily substance floats on the surface of the water.
Its dull, yellowish color contrasts starkly with the thick brown tint of
the rest of the water.
~
S
#0

#RESETS
* 501

```
D 0 501 3 1         door resets closed, unlocked
*
* 502
D 0 502 1 1         door resets closed, unlocked
M 0 503 2 502       load hungry patron
E 0 505 0 5         hungry patron wears tunic
O 0 507 0 502       load coatrack
*
* 503
M 0 501 1 503       load bartender
G 0 501 0 0         beef to bartender’s inventory
G 0 502 0 0         key to bartender’s inventory
G 0 503 0 0         ale to bartender’s inventory
*
* 504
M 0 502 1 504       load cook
E 0 504 0 16        cook wields cleaver
O 0 508 0 504       load stove
P 0 509 0 508       put pork in stove
*
* 505
M 0 503 2 505       load hungry patron
E 0 505 0 5         hungry patron wears tunic
M 0 504 1 505       load quiet patron
E 0 505 0 5         quiet patron wears tunic
E 0 506 0 8         quiet patron wears boots
*
* 506
D 0 506 0 2         door resets closed, locked
*
* 507
D 2 507 2 2         door resets closed, locked
M 0 505 1 507       load smell
O 0 510 0 507       load toilet
S


#SHOPS
501 19 26 10 0 0 110 85 8 23
0


#SPECIALS
M 501 spec_janitor         bartender
M 504 spec_thief           quiet patron
O 510 speco_recylcer       toilet
S


#$
```