Come
to
Valhalla

in your own time...


Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3, published by the Free Software Foundation. A copy of the license is included in the section entitled “GNU Free Documentation License”.

http://www.hoboes.com/NetLife/Valhalla/

October 4, 2011
# Table of Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beginning</td>
<td>1</td>
</tr>
<tr>
<td>Getting To Valhalla</td>
<td>1</td>
</tr>
<tr>
<td>Internet</td>
<td>1</td>
</tr>
<tr>
<td>Getting Into Valhalla</td>
<td>1</td>
</tr>
<tr>
<td>Getting Around</td>
<td>1</td>
</tr>
<tr>
<td>Virtual Self</td>
<td>2</td>
</tr>
<tr>
<td>Interacting With Other People</td>
<td>2</td>
</tr>
<tr>
<td>Talking</td>
<td>3</td>
</tr>
<tr>
<td>Emoting</td>
<td>3</td>
</tr>
<tr>
<td>Paging</td>
<td>3</td>
</tr>
<tr>
<td>Whispering</td>
<td>4</td>
</tr>
<tr>
<td>Mail</td>
<td>4</td>
</tr>
<tr>
<td>Discussion Groups</td>
<td>5</td>
</tr>
<tr>
<td>Talking With Text</td>
<td>6</td>
</tr>
<tr>
<td>Interacting With Objects</td>
<td>7</td>
</tr>
<tr>
<td>Picking Them Up and Dropping Them</td>
<td>7</td>
</tr>
<tr>
<td>Looking At Them and Examining Them</td>
<td>7</td>
</tr>
<tr>
<td>Getting Help</td>
<td>7</td>
</tr>
<tr>
<td>What is Valhalla?</td>
<td>7</td>
</tr>
<tr>
<td>Purpose</td>
<td>7</td>
</tr>
<tr>
<td>Technical Stuff</td>
<td>8</td>
</tr>
<tr>
<td>Where to Go for Help</td>
<td>9</td>
</tr>
<tr>
<td>The Newbie Arch-Wizard FAQ</td>
<td>9</td>
</tr>
<tr>
<td>The Moo-Cows Mailing List</td>
<td>9</td>
</tr>
<tr>
<td>The Original MOO</td>
<td>9</td>
</tr>
<tr>
<td>The Wide World of MOO</td>
<td>9</td>
</tr>
<tr>
<td>The Cerebus MOO Object Archive</td>
<td>10</td>
</tr>
<tr>
<td>Come to Valhalla</td>
<td>10</td>
</tr>
<tr>
<td>Mail</td>
<td>11</td>
</tr>
<tr>
<td>Getting Around in Mail</td>
<td>11</td>
</tr>
<tr>
<td>What’s Available?</td>
<td>11</td>
</tr>
<tr>
<td>Reading a Lot of Mail</td>
<td>11</td>
</tr>
<tr>
<td>Deleting Old Mail</td>
<td>11</td>
</tr>
<tr>
<td>Setting Your Mail Options</td>
<td>11</td>
</tr>
<tr>
<td>Stickiness</td>
<td>12</td>
</tr>
<tr>
<td>Forwarding Mail to an Internet Address</td>
<td>12</td>
</tr>
<tr>
<td>Building Discussion Groups</td>
<td>12</td>
</tr>
<tr>
<td>Moderated Groups</td>
<td>12</td>
</tr>
<tr>
<td>Private Groups</td>
<td>13</td>
</tr>
<tr>
<td>Building</td>
<td>14</td>
</tr>
<tr>
<td>Building Things</td>
<td>14</td>
</tr>
<tr>
<td>Creating Objects</td>
<td>14</td>
</tr>
<tr>
<td>Describing Objects</td>
<td>15</td>
</tr>
<tr>
<td>Making Rooms</td>
<td>16</td>
</tr>
<tr>
<td>Hooking Your Rooms To Valhalla</td>
<td>18</td>
</tr>
<tr>
<td>Advanced Building</td>
<td>19</td>
</tr>
<tr>
<td>Quotas and Audits</td>
<td>19</td>
</tr>
<tr>
<td>Talking to the MOO</td>
<td>19</td>
</tr>
<tr>
<td>Sentence Structure</td>
<td>19</td>
</tr>
<tr>
<td>Where Does Valhalla Look for Verbs?</td>
<td>20</td>
</tr>
<tr>
<td>Yourself</td>
<td>20</td>
</tr>
<tr>
<td>Home, Sweet Home</td>
<td>20</td>
</tr>
<tr>
<td>Password</td>
<td>20</td>
</tr>
<tr>
<td>Page Messages</td>
<td>21</td>
</tr>
<tr>
<td>Pronoun Substitution</td>
<td>21</td>
</tr>
<tr>
<td>Keys</td>
<td>22</td>
</tr>
<tr>
<td>Objects</td>
<td>23</td>
</tr>
<tr>
<td>Drop and Take Messages</td>
<td>23</td>
</tr>
<tr>
<td>Containers</td>
<td>24</td>
</tr>
<tr>
<td>Notes and Letters</td>
<td>24</td>
</tr>
<tr>
<td>Rooms</td>
<td>25</td>
</tr>
<tr>
<td>Arrival and Departure Messages</td>
<td>25</td>
</tr>
<tr>
<td>Who Location</td>
<td>26</td>
</tr>
<tr>
<td>Free Home and Residents</td>
<td>26</td>
</tr>
<tr>
<td>Locking Your Doors</td>
<td>26</td>
</tr>
<tr>
<td>Making Children</td>
<td>27</td>
</tr>
<tr>
<td>Using Universal Resource Locators (URLs)</td>
<td>27</td>
</tr>
<tr>
<td>Programming</td>
<td>28</td>
</tr>
<tr>
<td>Becoming a Programmer</td>
<td>28</td>
</tr>
<tr>
<td>Properties</td>
<td>28</td>
</tr>
<tr>
<td>Strings</td>
<td>28</td>
</tr>
<tr>
<td>Numbers</td>
<td>28</td>
</tr>
<tr>
<td>Objects</td>
<td>29</td>
</tr>
<tr>
<td>Lists</td>
<td>29</td>
</tr>
<tr>
<td>Verbs</td>
<td>29</td>
</tr>
<tr>
<td>Classes of Objects</td>
<td>29</td>
</tr>
<tr>
<td>Sentence Structure</td>
<td>30</td>
</tr>
<tr>
<td>Editing Properties and Verbs</td>
<td>30</td>
</tr>
<tr>
<td>Properties: Notedit</td>
<td>30</td>
</tr>
<tr>
<td>Verbs: Edit</td>
<td>31</td>
</tr>
<tr>
<td>The Generic Swinging Weapon</td>
<td>31</td>
</tr>
<tr>
<td>The Advantages of Being Object-Oriented</td>
<td>33</td>
</tr>
<tr>
<td>A Better Generic Swinging Weapon</td>
<td>35</td>
</tr>
<tr>
<td>Forks and Tasks</td>
<td>36</td>
</tr>
<tr>
<td>Using Files</td>
<td>36</td>
</tr>
<tr>
<td>The File Utilities Object</td>
<td>36</td>
</tr>
<tr>
<td>Wizards</td>
<td>37</td>
</tr>
<tr>
<td>Creating Players</td>
<td>37</td>
</tr>
<tr>
<td>Creating Wizards, Programmers, and Builders</td>
<td>37</td>
</tr>
<tr>
<td>Administration</td>
<td>39</td>
</tr>
<tr>
<td>Compiling</td>
<td>39</td>
</tr>
<tr>
<td>Choosing Wizards</td>
<td>39</td>
</tr>
<tr>
<td>Choosing Administrators</td>
<td>39</td>
</tr>
<tr>
<td>Login Stuff</td>
<td>39</td>
</tr>
<tr>
<td>Welcoming Message</td>
<td>39</td>
</tr>
<tr>
<td>Player Creation</td>
<td>40</td>
</tr>
<tr>
<td>Maximum Users</td>
<td>40</td>
</tr>
<tr>
<td>Monitoring Players</td>
<td>40</td>
</tr>
<tr>
<td>@Net-Who</td>
<td>40</td>
</tr>
<tr>
<td>Checkpoints</td>
<td>40</td>
</tr>
<tr>
<td>Dump Interval</td>
<td>40</td>
</tr>
<tr>
<td>Backups</td>
<td>40</td>
</tr>
<tr>
<td>Database</td>
<td>40</td>
</tr>
<tr>
<td>Objects</td>
<td>41</td>
</tr>
<tr>
<td>Discussion Groups</td>
<td>41</td>
</tr>
<tr>
<td>Creating A New Discussion Group</td>
<td>41</td>
</tr>
<tr>
<td>New Generics</td>
<td>41</td>
</tr>
<tr>
<td>New Help</td>
<td>41</td>
</tr>
<tr>
<td>Internet</td>
<td>42</td>
</tr>
<tr>
<td>Mail</td>
<td>42</td>
</tr>
<tr>
<td>Gopher</td>
<td>42</td>
</tr>
<tr>
<td>World-Wide Web</td>
<td>42</td>
</tr>
<tr>
<td>Usenet News</td>
<td>42</td>
</tr>
<tr>
<td>Blacklists</td>
<td>42</td>
</tr>
<tr>
<td>Newsys</td>
<td>42</td>
</tr>
<tr>
<td>Toads</td>
<td>42</td>
</tr>
<tr>
<td>Blacklists</td>
<td>42</td>
</tr>
<tr>
<td>Open source license</td>
<td>43</td>
</tr>
<tr>
<td>Gnu Free Documentation License</td>
<td>43</td>
</tr>
</tbody>
</table>
BEGINNING

GETTING TO VALHALLA

Valhalla used to be on a computer at the University of San Diego. It no longer exists. But this document will still be useful for beginners to MOO.

Internet

In order to get to a “MOO”, you need a “MOO Client”, just like you need a web client to get to the web, or an e-mail client to get e-mail. If you don’t have a MOO client, don’t panic: you can use any old telnet program at first.

You also need to know the “host” and the “port” of the MOO you want to get to. For example, if the MOO says to use the host valhalla.hoboes.com, and port 4444, you would ‘telnet’ to:

```
telnet valhalla.hoboes.com 4444
```

Note that Valhalla MOO isn’t actually running right now: it only runs when I have need of it. But I’ll be using it for examples because it’s the one I use most often.

GETTING INTO VALHALLA

When you arrive at Garm’s cave, you’ll type

```
@connect your-player your-password
```

to get past the cute little puppy. If you don't have a player yet, you'll need to create one, by thinking up a name and password, and typing:

```
@create your-player-name your-password
```

Valhalla will tell you if the player name you've chosen is already being used. You'll then have to use @create with a new name. Once you're in Valhalla, type

```
news
```

to see the latest edition of the Valhalla News-Rune, for the latest information about Valhalla and the people who use it.

GETTING AROUND

When you want to see what’s around you, type

```
look
```

and type
look object
to look at a particular object. For example, try:
look me
Whenever you say me, Valhalla assumes you’re talking about yourself.
Usually, there will be some directions you can go, such as north, door, or docks. Type
go place
to go somewhere. Look in the description of where you are to see possible directions to go in.
You can also get a ‘compass rose’ by typing @rose.
If you have a home, you can type home
and you’ll be transported there just like in the Wizard of Oz.
You can get help by typing help. You can get help on a topic by typing help topic. We have no psychiatrists in Valhalla. Yet.

VIRTUAL SELF
You are malleable. You yam what you yam and you ain’t no more. When you first use your player, you will have something like the following generic description:
look me
‘You see a player who should type ‘@describe me as ...’.
It is awake and looks alert.
You can describe things that you own (and you own yourself) with the command:
@describe object as "Description"
You’ll also need to set your gender if you are male or female.
For example:
@describe me as "Loki, half-giant, half-Aesir, is a dashing fellow with a sly smile."
@gender male
The next time someone looks at you, they’ll see:
look me
Loki, half-giant, half-Aesir, is a dashing fellow with a sly smile.
He is awake and looks alert.
You are ‘awake’ because you are here. When you leave Valhalla you are asleep. You are ‘alert’ because you are doing something. If you leave your player unattended for a while, you are staring off into space.

INTERACTING WITH OTHER PEOPLE
The most common ways of interacting with other people are by ‘talking’, ‘emoting’, and ‘paging’.

"Hi everybody! Where’s the ale?
You say, "Hi everybody! Where’s the ale?"
:smiles.
Balder smiles.
:chugs a tanker of ale.
Balder chugs a tanker of ale.
page Thor "Hey, Thor, there’s a great party over at the circus."
Your message has been sent.

Meanwhile, Thor sees:

You sense that Balder is looking for you in The Fields of Valhalla.
He pages, "Hey, Thor, there’s a great party over at the circus."

Talking

"Hi everybody! Where’s the ale?
First, I talked. To talk, you type a quotation mark (double-quote) and what you want to say. Everyone else sees:

Balder says, "Hi everybody! Where’s the ale?"

Emoting

:smiles.

Then, I emoted. This uses the colon. Everyone else saw:

Balder smiles.

The ‘emote’ can be used to do things as well. For example, chugging a tanker of ale. There's a difference, however, between chugging a tanker of ale by emoting, and chugging an actual tanker of ale. To really chug a tanker (and what does ‘really’ mean in a virtual reality?), you need to have an object called tanker, and this object must contain a verb called chug. More about objects and verbs later, though.

Paging

When you talk, only the people who are in the same room can hear you. If there’s someone else in Valhalla, you can page them.

The syntax is:

page player *text of page

For example:

page Thor "Hey, Thor, there’s a great party over at the circus."

Thor sees:
You sense that Balder is looking for you in Balder’s Chimney. He pages, "Hey, Thor, there’s a great party over at the circus."

To find out who is awake in Valhalla, type

@who

<table>
<thead>
<tr>
<th>Player name</th>
<th>Connected</th>
<th>Idle time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balder (#78)</td>
<td>an hour</td>
<td>0 seconds</td>
<td>The Fields of Valhalla</td>
</tr>
<tr>
<td>Loki (#79)</td>
<td>20 minutes</td>
<td>10 minutes</td>
<td>Jotunheim</td>
</tr>
<tr>
<td>Thor (#3)</td>
<td>2 hours</td>
<td>10 seconds</td>
<td>Bifrost Bridge</td>
</tr>
</tbody>
</table>

**Whispering**

You can also whisper to someone if you want to have a private conversation without finding an empty room.

> whisper "What’s with the funny hat?" to Thor
> You whisper, "What’s with the funny hat?" to Thor.

And, Thor sees:

Balder whispers, "What’s with the funny hat?"

Nobody else in the room sees anything at all.

**Mail**

You can send and receive mail within Valhalla. You send mail with the @send command. You’ll be asked for a subject, and then you can type in your message, one line at a time.

@send Fred
Subject:
[Type a line of input or '@abort' to abort the command.]
Party at Thor’s
Mail Room

Do a 'look' to get the list of commands, or 'help' for assistance.

Composing a letter to Balder (#78) entitled "Hello, Fred."
*There is a party at Thor’s on Tuesday night, 7:30 pm Valhalla time.*
Line 1 added.
*Can you bring your keg-of-20-drinks?*
Line 2 added.

Note that you have to use a quotation mark, as if you are saying something, to add text. That’s because you are saying it, and the ‘Mail Room’ is dutifully writing down what you say.

If you want to see what you’ve typed so far, you can use list.

List
1: There is a party at Thor’s on Tuesday night, 7:30 pm Valhalla time.
2: Can you bring your keg-of-20-drinks?
To change something you’ve written, use subst (for substitute). The syntax is subst /text-as-it-is/ text-as-you-want-it/lines-the-text-is-in. For example,

```
sub /your/my/2
```

changes the your in line 2 to my. You can also specify a range of lines, for example, 2-5. If you don’t know what the last line is, you can use the dollar sign $. Valhalla interprets $ as ‘the end’.

Use send when you’re ready to send the message on its way. If you change your mind and decide not to send any mail, use abort.

```
send
Sending...
Mail actually sent to Fred (#99)
Chimney
The stone chimney is smokey and warm, and a bit cramped for anyone of normal size. Dirty elfs skitter about as they see you coming.
```

You see the Chimney description again because, when you are composing a mail message, you are actually in a different room—the ‘Mail Room’. After you ‘send’ the message, you return to wherever you were before composing the message.

When Fred receives the message, he sees:

```
You have new mail (1) from Balder (#78).
Type `help mail' for info on reading it.
```

You can list your mail by typing:

```
@mail on me
```

```
1:+ Jul 15 11:37 Fred (#99) Re: Hello, Fred.
2:+ Jul 15 12:59 Thor (#3) Party Canceled
```

You can read a message by typing

```
@read # on me
```

where # is the number of the message you want to read. Delete a message with

```
@rm 1 from me
```

For more information about mail, type help mail.

**Discussion Groups**

There are a number of discussion groups on Valhalla. You are already a member of *Life, a group for general discussion of life in Valhalla. If you want to see a list of mail on *Life, type @mail on *Life. To read message 7 on *Life, type @read 7 on *Life. Discussion groups are just like mail, except that instead of the word ‘me’, you use the name of the discussion group. All of the mail verbs work with discussion groups.

To see a list of all of the discussion groups available to you, type @unsubscribed. To take part in a discussion, you need to subscribe by typing
If you want to stop taking part in a discussion group, use

@unsubscribe discussion-group

and use @subscribed to see a list of the groups you’re currently subscribed to.

Talking With Text

Using pure text can be somewhat limiting. You can’t squiggle pictures in the margins, you can’t wave your arms or smile and expect the person at the other computer to see it. In Valhalla, you can wave your arms and smile using the *emote* verb already described:

:B:smiles disdainfully. *Funny as a crutch, Thor.*

Balder smiles disdainfully. ‘Funny as a crutch, Thor.’

You can add **emphasis** to a word or phrase by surrounding it with an *asterisk*. If you surround each word *separately* with asterisks, it’s a more staccato an emphasis.

Balder says, **‘Where* did you put it?!’**

Balder says, **‘Where did you put it?!’**

Balder says, **‘Where* *did* *you* *put* *it*?!’**

You can yell (but you shouldn't very often) by using **all capitals**. (And if you accidentally hit your shift lock key, everyone will think you are yelling.)

Balder says, ‘WHERE DID YOU PUT IT?!’

If you want to set of a title without as much emphasis, use the *underscore*. It’s similar to *italics*, and is probably derived from the editor’s mark for italics.

Balder says, ‘Have you seen _Gone With the Wind_?’

And, of course, you can combine them as much as you want…

Balder looks at his tape collection quizzically. ‘Have you seen _Gone With the Wind_? *Bitchin’* movie.*WHERE* did you *put* *it*?!’

…but don’t overdo it, you’ll burn your keyboard out.

You can also use *emoticons*, although they should be limited to electronic mail. The most common emoticons are the *smiley* and the *frown*. They look sort of like a smiling and a frowning person… if you turn your head to the side… and use some imagination.

:*)  :*(  :^)  :^ (  =*(  =*

Other emoticons exist. There’s an exhibit in the museum in Balmooa Park.

---

1 The editing mark for italics is to underline the text. The editing note for underlines is to underline the text with a squiggly line. You do occasionally see the *tilda* used to set off titles, and this, too, is probably derived from the editing mark.
Picking Them Up and Dropping Them

The basic commands that apply to objects are look, take, drop, and give.

look staff
take staff
drop staff
give staff to Thor

In Valhalla, these commands are called ‘verbs’. In order to ‘do’ something to or with an object, either that object or one of it’s ancestors must have that ‘command’ as a ‘verb’. Only programmers can create and program verbs.

For more information about using objects, type:

help manipulation

Looking At Them and Examining Them

You can look object to see an object’s description. Usually, looking at an object will tell you everything that the object’s creator wants you to know. If you want to see a list of the verbs on the object, and some of the properties, type @examine object. ‘Verbs’ are things you can do to or with the object. Drop and look are both verbs. ‘Properties’ are information about the object. Names, aliases, and descriptions are all properties.

I’ll talk a little more about this under Building and Advanced Building.

Getting Help

Valhalla has a built in help system. Type help to see it. Some parts of help that you may find helpful are:

help movement
help communication
help players
help manipulation

What is Valhalla?

Purpose

Valhalla’s purpose is to provide a meeting ground in ‘virtual space’. It is a space on the Internet where people can interact in real time or by leaving messages. People in Valhalla can personalize
their interactions by creating special rooms and objects. Valhalla is meant to provide a place for
the University of San Diego community to meet with other communities on the Internet.
Members of the USD community can sponsor on-line festivals, professional conferences, regular
get-togethers, readings, and anything else that humans meet for, using Valhalla and the Internet.
Valhalla is open to anyone on the Internet. However, ‘programmer’ status will generally only be
granted on the request of a member of the USD community, and then only if it will contribute to
the general needs of Valhalla’s USD users.

Technical Stuff

From a technical standpoint, Valhalla is what is known as a MOO. MOO is a ‘programming
language’ for creating multi-user dungeons (MUD). There are many other MUDs around. MOO
is object oriented (thus, MUD, Object Oriented, or MOO). Don’t worry about what that means.
MOO was created and is currently maintained by Pavel Curtis and Xerox PARC, who are to be
commended for their work. They provide the MOO environment free of charge.

Valhalla is a MOO on one of the IBM compatibles at the University of San Diego, using the
Linux operating system. Linux is a Unix look-alike. There are different versions of Linux
available, but all are provided free of charge by the people who write and maintain them.

Finally, Valhalla is maintained by Thor and Balder, who can occasionally be seen wandering
about the Circus Bazarre and the rest of Valhalla’s universe.
WHERE TO GO FOR HELP

There are many more MOOs out there, and many people using them. They’ve developed quite a support group.

THE NEWBIE ARCH-WIZARD FAQ

On the Net, an ‘FAQ’ is a collection of Frequently Asked Questions—and their answers. The Newbie Arch-Wizard FAQ is for people setting up a MOO who don't know anything about MOOs. It's on a gopher site. Get to it via gopher at gsep.pepperdine.edu, and look in the directories technical and then MUD-MOO. The file is new-archwiz-faq.txt. If you're using World-Wide Web, the URL is:

http://www.fringenet.net/MOO/new-archwiz-faq.txt

THE MOO-COWS MAILING LIST

MOO-COWs is a group of people who use electronic mail to discuss MOO-related issues. In order to take part in the mailing list, you need to subscribe. Get more information about MOO-Cows at:

http://www.moo.mud.org/moo-faq/

THE ORIGINAL MOO

Pavel Curtis’ MOO, LambdaMOO, is running at lambda.moo.mud.org, on port 8888. There is also an ftp site designated for MOO issues, at ftp.lambda.moo.mud.org, in /pub/MOO. The URL for World-Wide Web is:

http://ftp.lambda.moo.mud.org/pub/MOO/

The official programmer's manual is there in text and postscript formats.

THE WIDE WORLD OF MOO

There is a special World-Wide Web server set up for MOOs at:

http://www.fringenet.net/MOO/
THE CEREBUS MOO OBJECT ARCHIVE

There is a web site at www.hoboes.com for MOO objects. The World-Wide Web URL is:
http://www.hoboes.com/pub/MOO%20Stuff/

If you create interesting objects that you want to share with the rest of the MOO world, write me via that web page.

COME TO VALHALLA

The full on-line Come to Valhalla includes the basic information you have here, plus more advanced information about building things, programming, and MOO management. Look for it at

http://www.hoboes.com/NetLife/Valhalla/
We’ve already discussed a little bit about mail inside of Valhalla. *Mail* is a very important feature—it allows you to carry on discussions with other players even if you can’t get inside of Valhalla at the same time. It also allows you to take part in group discussions with a wide variety of Valhalla’s denizens.

**Getting Around in Mail**

**What’s Available?**

When you come into Valhalla, you’ll be told which discussion groups have mail on them that you haven’t read. If you want to double check later, on, you can type

@rm

to see a list of all of your discussion groups that have unread mail on them.

**Reading a Lot of Mail**

After you read the first piece of mail, you can simply type

@next on me

or

@next on *discussion-group*

to see the next message.

**Deleting Old Mail**

You can delete mail that you no longer need by typing

@rm *number* on me

Your message numbers will continue to climb, however. You can reset the message numbers back to one with

@renumber

**Setting Your Mail Options**

There are a number of things that the *mail agent* assumes about you, and you can change these assumptions. Type *help @mail-option* for more information.
Stickiness

Normally, mail assumes that you’re talking about yourself. If you type @next, it’ll assume you mean @next on me. If you set mail to be sticky, it’ll assume you meant whatever ‘mail box’ you used last time. If the last thing you typed was @next on *Life, another @next will also deal with *Life. Use

@mail-option +sticky

to make your mail verbs sticky, and

@mail-option -sticky

to ‘unstick’ your mail verbs.

Forwarding Mail to an Internet Address

Many people like to get all of their mail in one place. You can have your Valhalla mail forwarded to another Internet address (including valhalla.hoboes.com if you have an address there) by setting your netmail mail-option:

@mail-option +netmail

Building Discussion Groups

You can build your own discussion group. The ‘parent’ discussion group is $mail_recipient. Here’s a basic, general access discussion group:

@create $mail-recipient named discussion-group-name

Your discussion group name cannot have any spaces in it.

@set discussion-group-name.readers to 1

By setting this property to 1, you’re giving everybody in Valhalla access to the group.

@describe discussion-group-name as *Description of discussion group

@move discussion-group-name to $mail_agent

Until you move the discussion group into the mail agent, it doesn’t matter that anyone else can take part in it, because they can’t see it, and won’t know it exists. Once it’s inside the mail agent, anyone can @subscribe to your discussion group. Your group will show up when anyone else in Valhalla lists discussion groups with the @unsubscribed verb.

Moderated Groups

A moderated group is one that can be read but not sent to. You might, for example, create a newspaper as a discussion group. Journalists would send their articles to you (the editor), and you would edit them and send them to the group, at which time all the readers will see it. To make a group moderated, set the ?? property to the names of the people who will be allowed to send to the group. Only these people can send.
You need to use the object identifier numbers because Valhalla can’t ‘see’ either the discussion group or the players.

Private Groups

You can make a group *private* by telling it exactly which people are allowed to read it. Instead of setting `.readers` to 1, set it to a list of the players who will be allowed to read it:

```plaintext
@set #discussion-id to [player_1-id, player_2-id, ..., player_n-id]
```

The discussion group will not allow anyone who is not on the list to join the discussion.
**Building**

In Valhalla, you can *build*, or *create* objects for your player to carry around. You can also build rooms, buildings, or surfboards. There’s a special discussion group on Valhalla for talking about building things, called *Everything*.

**Building Things**

Here’s an example:

```bash
@create $thing named Staff
You now have Staff with object number #80 and parent generic thing (#5).
look staff
You see nothing special.
@describe staff as "Balder’s staff is a gnarly branch of an oak, tied at both ends by hemp yarn and inscribed with strange runes."
Description set.
look staff
Balder’s staff is a gnarly branch of an oak, tied at both ends by hemp yarn and inscribed with strange runes.
look me
Balder the Brave strides across Valhalla with a strong purpose and a light in his eyes.
He is awake and looks alert.
Carrying:
Staff
```

**Creating Objects**

What did I do there? First, I *created* the object and named it. Then, I *described* it and fooled around with it.

```bash
@create $thing named Staff
Things with the dollar sign in front of them are ‘generic objects’. The object called $thing is the most basic thing you can have. In Valhalla, every object has parents. In this case, I created a child of $thing named staff. When I created the staff, Valhalla told me that it has object number #80. Sometimes, when Valhalla can’t tell what object you’re talking about (if there are two objects with the same name, or the object isn’t visible), you’ll need to use the object number instead. Look staff and look #80 are both the same thing, but I can look #80 even when the staff isn’t in the same room as I am.
@parent staff
Staff(#80)   generic thing(#5)   Root Class(#1)
```
I can see the ancestors of an object with the @parent command. The staff is a child of *generic thing*, which is a child of the *Root Class*.

```
@parent me
Balder(#78) generic wizard(#58) generic programmer(#59) generic builder(#4) generic player(#6) Root Class(#1)
```

An object’s parents determine what the object can do. My staff can do anything that a generic thing can do (which is, not much). I (Balder) can do anything that a generic wizard can do; anything that a generic programmer can do; anything that a generic builder can do; and anything that a generic player can do.

```
@parent here
The Fields of Valhalla(#11) generic room(#3) Root Class(#1)
```

When you say “here”, Valhalla assumes you mean “the place you currently are”. The Fields of Valhalla are a generic room.

You can build objects based on other objects you have created. I could build a second staff based on the first (although I see no point to it). I could also create a type of room called ‘field’ and base the Fields of Valhalla on ‘field’ instead of on ‘room’. Here are the standard generics that Valhalla has:

- $thing: Most things you’ll create will be *things*.
- $container: *Containers* are objects that can hold other objects.
- $note: You can create *notes* with this. Try help $note.
- $player: People are all based on the *generic player*. Check your heritage!
- $room: Places where people can travel should be based on *room*.
- $exit: Valhalla creates these when you do an @dig.

Look in *Balder’s Basement* in *Asgard* for more generics that you can use.

## Describing Objects

After I @created the staff, I @described it. Until you describe an object, no one else can see what it is by *looking* at it. Until you describe it, they will see

```
You see nothing special.
```

The description of an object is a *property* of that object. We’ll talk more about properties later on. For now, whatever you

```
@describe object as...
```

is what other people see when they *look* at it.
Making Rooms

Rooms are just another type of object. The basic (generic) room is $room.

@create $room named "Balder's Hall","Hall"
You now have Balder's Hall (aka Hall) with object number #81 and parent generic room (#3).

@move me to hall
Balder's Hall is inside of Balder!
drop hall
Dropped.

@move me to hall
Balder's Hall
You see nothing special.
Moved.

First, I created the room.

@create $room named "Balder's Hall","Hall"
I gave it two names. The first name is the object’s real name. The second name is an ‘alias’. An object can have many aliases. Aliases are simply other ways of referring to an object. It’s easier for people to type hall than to type Balder’s Hall.

I tried to go into the room, but Valhalla wouldn’t let me. When you create an object, you are automatically carrying that object. That means that the Hall was inside of me. Valhalla does not allow objects to be inside of the objects that are inside of them. First, I had to drop the Hall so that it was no longer ‘inside’ of me.

There’s nothing special about the Hall yet. I need to describe it.

@describe here as "The ceiling is made of huge vaulting oak beams. The walls are the hide of some great scaled lizard, and its scales glisten in the light of the flickering fire in the stone fireplace at the hall's north end.
Description set.

look
Balder's Hall
The ceiling is made of huge vaulting oak beams. The walls are the hide of some great scaled lizard, and its scales glisten in the light of the flickering fire in the stone fireplace at the hall's north end.
Let’s make another room, and make an entrance between the Hall and the new room.

@dig fireplace, fire|down, fire to Chimney
Chimney (#82) created.
Exit from Balder’s Hall (#81) to Chimney (#82) via [“fireplace”, “fire”] created with id #83.
Exit from Chimney (#82) to Balder’s Hall (#81) via [“down”, “fire”] created with id #84.
go fireplace
Chimney
You see nothing special.
go down
Balder’s Hall
The ceiling is made of huge vaulting oak beams. The walls are the hide of some great scaled lizard, and its scales glisten in the light of the flickering fire in the stone fireplace at the hall’s north end.

The dig command is a bit complicated. Let’s break it down:

@dig fireplace, fire|down, fire to Chimney

The syntax is:

@dig entrances/exits to room

So, in the example, @dig created a room called Chimney. It created an entrance from here (Balder’s Hall) to the Chimney, called fireplace, with alias fire. It created an exit from Chimney to here called down, with alias fire. Anyone inside Balder’s Hall can use the fireplace or fire to move between the Hall and the Chimney. From the Chimney, they can use down or fire to move to the Hall.

They all need descriptions. Since I’m currently in Balder’s Hall, the only object I can see is the fireplace entrance. I need to refer to the chimney and the down exit with their object numbers. Valhalla told me what the object numbers were when I dug them.

@describe #82 as “The stone chimney is smokey and warm, and a bit cramped for anyone of normal size. Dirty elfs skitter about as they see you coming.”
Description set.
@describe fireplace as “A blazing fire from the fireplace heats the entire room. You occasionally see rodents’ faces popping down from the chimney and then disappearing.”
Description set.
@describe #84 as “The fire below spews smoke and ash up the chimney and past you. What in the world are you doing here?”
Description set.

look fire
A blazing fire from the fireplace heats the entire room. You occasionally see rodents’ faces popping down from the chimney and then disappearing.
go fireplace
Chimney
The stone chimney is smokey and warm, and a bit cramped for anyone of normal size. Dirty elfs skitter about as they see you coming.
look down
The fire below spews smoke and ash up the chimney and past you. What in the world are you doing here?
Note that, because the ‘fireplace’ is a passageway, when I ‘go fireplace’ I end up in the Chimney. Likewise, when I ‘go down’ I end up in Balder’s Hall. You never actually spend any time inside of passageways.

**Hooking Your Rooms To Valhalla**

When you create a room, no one else can get to it. Even *you* can only get to it by using the @move verb. You can’t *go Balder’s Hall* because there’s no entrance to Balder’s Hall. You can only create passageways between two areas that you own. So, only a wizard can hook things up to Valhalla. When you have your rooms ready for other people to use, tell a wizard. You can see if there are any wizards currently available by typing @who. Otherwise, send mail to either Balder or Thor.

You can hook up passageways between two rooms you *do* own, so you can create a mansion of a hundred rooms and carry it in your pocket if you want.
**Advanced Building**

*Help building* and *help gen-index* will give you a list of topics that might interest you. The command *@classes generics* will give you a list of all the built-in generic objects available for parenting.

**Quotas and Audits**

To see a list of the objects that you have, along with their ID number and current location, type *@audit me*.

You can only create a certain number of objects. This is your *quota*. Once you create this many objects, you cannot create any more without getting rid of some of the ones you have. To get rid of an object, type *@recycle object*. If you can’t see the object, you’ll have to use its object number. Once you recycle an object, it is *gone*. You *cannot* get it back. Don’t recycle an object that’s important to you.

If you ask a wizard nicely, you can get your quota increased.

**Talking to the MOO**

**Sentence Structure**

It helps to know how Valhalla interprets what you type. The simplest way to do something is to type a single verb, such as *home*. ‘Home’ is a verb that only works on yourself, so it doesn’t need to know any more. A *verb* is the word you use to tell Valhalla that you want to *do* something.

You can also have direct objects, indirect objects, and prepositions. For example, in

```
give chocolate to Garm
```

the *chocolate* is the direct object, and *Garm* is the indirect object of the verb *give*. Valhalla first attempts to find a verb. ‘Give’ is the verb in this case, and the verb is always the first single word typed. Then, it looks for a *preposition* (in this case, ‘to’), and assumes that what is on the left (chocolate) is the direct object, and what is on the right (Garm) is the indirect object. If one of your objects contains a preposition, you can confuse Valhalla. To keep Valhalla from looking at the prepositions inside your object, enclose the object text in quotation marks. For example:

```
show "Like Water For Chocolate" to Garm
```

If you keep in mind how Valhalla works, you’ll be better able to phrase things when you try to do something.

If there is no preposition, everything after the verb is the direct object.
Where Does Valhalla Look for Verbs?

Verbs can only be found on objects. You are an object, and you have quite a few verbs on you. So, most likely, does the room that you are in. When Valhalla looks for a verb, it looks on the player who typed the verb first. If it doesn't find a matching verb there, it looks in the room that the player is in. If it still doesn't find a matching verb, it looks on the direct object of the sentence, if there is one, and finally on the indirect object of the sentence, if there is one.

If it hasn't found any verb at all, you will be told something to the effect of “Valhalla doesn't have any idea what you're trying to do.” Sometimes, however, Valhalla will find the verb that you typed, but with a different sentence structure. If so, Valhalla will suggest a different way of typing your sentence to get the desired results.

YOURSELF

Home, Sweet Home

Every player (and most objects) have a home. For players, your home is where you go when you sleep. When you @quit Valhalla, your player is whisked away to home. You can also teleport home by typing home. By default, your ‘home’ is The Fields of Valhalla. If you create your own room, or find another place you’d rather call home, go there and type:

@sethome
Balder's Hall is your new home.

From now on, this is your home.

home
You click your heels three times.
Balder's Hall
The ceiling is made of huge vaulting oak beams. The walls are the hide of some great scaled lizard, and its scales glisten in the light of the flickering fire in the stone fireplace at the hall's north end.

Clicking your heels three times may not be appropriate for a great warrior of Valhalla. You can change this. It’s what is called a message. Type

@home me is "You smash your boots to the ground and the earth opens up."
You set the “home” message of Balder (#78).

and your ‘home’ message is changed.

Password

You can (and should regularly) change your password. Use:

@password old-password new-password
For example,

@password iqoqoj LoveSexy
New password set.

You should choose a password that is easy for you to remember, and hard for anyone else to guess. Your password keeps other people from using your player when you’re not looking.

Page Messages

*Home* is an example of a message. Your player has many messages that you can change. To see a list of the messages that you can change, type @messages me. Many people commonly change the page messages, *page_absent*, *page_origin*, and *page_echo*. *Page_absent* is the message other people get when they try to page you but you’re asleep. *Page_origin* is the message other people get when you page them, and *page_echo* is the message other people get when they page you.

The syntax for changing a message on yourself is

@message me is “New Message.”

For example,

@page_absent me is “%N is sleeping in %l, dreaming of pillaging the European countryside.”

You set the “page_absent” message of Balder (#78).

When someone else pages you and you’re not in, they’ll see:

page balder “Hey, Balder, what’s up?
Balder is sleeping in Balder’s Hall, dreaming of pillaging the European countryside.

The “%N” and “%l” are *pronoun substitutes*. I’ll talk more about those later. For the moment, use %N to refer to your name when it appears at the beginning of a sentence, and %n when it appears inside of a sentence. The same goes for %L, which is the location of the person who owns the message. Use %l (lowercase L) when it appears inside a sentence, and %L at the beginning of a sentence. Pronoun substitutes only work with certain verbs (of which *page*, obviously, is one).

Pronoun Substitution

‘Pronoun subs’ are ways to make your messages more personal. They are ‘place-holders’ for the name of the person using an object, the location of the object, or whatever personal or possessive pronoun the person using the object needs. For a full description of how to use pronouns, type @help pronouns.

Pronouns come in two flavors: capitalized and uncapitalized. Usecapitalized pronouns at the beginning of sentences, and uncapitalized ones inside of sentences. You’ve already seen “%L” and “%l” for location. It’s the location of the object that the message is on. All pronouns are preceded by a “%” sign. If you really want a “%” sign, use “%%%”.

Let’s say we’ve got a drinking fountain, and once in a while it sprays the drinker embarrassingly. We could use the message:
@embarrass fountain is "%N sprays water all over %r, and a puddle forms around %o and the %t on the floor of %l.
@embarrass fountain is "The %t sprays water all over you. %L grows quiet as everyone turns to look at you.

This is overdoing it a just a bit. Let's say the drinking fountain is in “the Fields of Valhalla”, and Balder takes a drink, triggering the embarrassing message. Balder sees:

The drinking fountain sprays water all over you. The Fields of Valhalla grows quiet as everyone turns to look at you.

Everyone else sees:

Balder sprays water all over himself, and a puddle forms around him and the drinking fountain on the floor of the Fields of Valhalla.

Here are the commonly used pronoun substitutions. Each one also has a capitalized version.

%n the player
%t this object (i.e., the object that holds the message,... usually)
%d the direct object from the verb line
%i the indirect object from the verb line
%l the location of the object
%s subject pronoun, either 'he', 'she', or 'it'
%o object pronoun, either 'him', 'her', or 'it'
%p posessive pronoun (adj), either 'his', 'her', or 'its'
%q posessive pronoun (noun), either 'his', 'hers', or 'its'
%r reflexive pronoun, either 'himself', 'herself', or 'itself

Keys

Objects can require keys in order to ‘use’ them. The player who attempts to use the object must either be or have the key.

<table>
<thead>
<tr>
<th>Object Class</th>
<th>Key required for:</th>
</tr>
</thead>
<tbody>
<tr>
<td>$thing</td>
<td>taking</td>
</tr>
<tr>
<td>$note/$letter</td>
<td>taking or reading</td>
</tr>
<tr>
<td>$container</td>
<td>taking or opening</td>
</tr>
<tr>
<td>$exit</td>
<td>leaving</td>
</tr>
<tr>
<td>$room</td>
<td>entering</td>
</tr>
</tbody>
</table>

If you @lock an object with yourself as the key, then you are the only person who can use that object. If you want to specify more complex keys, you have to understand the ‘language’ of keys, which is a bit weird. It's sort of like saying “let Bob or Jo or Bill but not anyone carrying my donut use this object”. But you have to say it in computer talk:

@lock here with (Bob || Jo || Bill) &!donut

It’s like that logic stuff you got from algebra. Use ‘||’ to say ‘or’, ‘&&’ to say ‘and’, and ‘!’ to say ‘not’. And you can group parts together with parentheses. In the above example, if it’s Bob, Jo, or Bill trying to get in, the part between parentheses comes out true. And as long as none of them are carrying your donut, !donut comes out true as well (because donut comes out false, and not donut is the opposite). Which lets them in. If anyone besides Bob, Jo, or Bill tries to come in, that part turns out false, and the room doesn’t let them in. Likewise, if anyone is carrying your
donut, that part comes out false and the room doesn't let them in, even if they are Bob, Bill, or Jo. At least, until they drop your donut.

The Truth Table, in case you’ve forgotten your high school algebra:

<table>
<thead>
<tr>
<th>English</th>
<th>Computerese</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>true and true</td>
<td>true &amp;&amp; true</td>
<td>true</td>
</tr>
<tr>
<td>true and false</td>
<td>true &amp;&amp; false</td>
<td>false</td>
</tr>
<tr>
<td>true or true</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>true or false</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>not true</td>
<td>!true</td>
<td>false</td>
</tr>
<tr>
<td>not false</td>
<td>!false</td>
<td>true</td>
</tr>
</tbody>
</table>

Here are some complex examples:

(Bob || (Jo && Balder’s Key) || me) && !#13

Anyone who is carrying whatever object has number #13 is locked out, even me: suppose I’m carrying object #13 and Balder’s Key:

(false || (false && true) ||true) && !true

Bob can come in as long as he’s not carrying #13, and Jo can only come in if she’s carrying my key. No one else can come in at all. Let’s say Fred, who is carrying my key but not object #13, tries to come in:

(true || (false && true) ||false) && !false

More commonly, you’ll probably use simple things like only allow yourself into the room:

@lock here with me

or only allow your friends and yourself into the room:

@lock here with me && CapVideo && Thor && Jenni

or lock a particular obnoxious person out of your room:

@lock here with !Fred

You can also use keys to make it so that only certain people can read notes and letters. I’ll talk about that in a moment. Type help keys for more detailed information about keys.

**OBJECTS**

To see a list of messages on an object, type @messages object-name. You can get a list of some of the verbs and properties on an object by using @examine object-name.

**Drop and Take Messages**

If you @lock a thing, people who don’t meet the criteria you specify cannot take the object. The message they’ll see when they try to take it is take_failed. The message that everyone else sees when they try to take it is otake_failed. When a message is exactly the same as another message,
but begins with o, the first message is for the player, or the person using the verb, and the second message (with the o) is for everyone else in the same room.

@take_failed Platter of Stale Donuts is "You pick up the entire %t, but your guilty conscience forces you to put it back. How about just eating a single donut?"

@take_failed Platter of Stale Donuts is "%N tries to hoard the %t, but %p conscience gets the better of %o."

@take_succeeded "Sword in the Stone" is "You pull the sword from the stone. Everyone else is looking at you *awfully* funny."

@take_succeeded "Sword in the Stone" is "With a majestic flourish, %n jerks the sword from the stone. A shaft of light from the sky illuminates %o with a divine glow."

You can also set the drop messages, which the player (drop_succeeded, drop_failed) and others (odrop_succeeded and odrop_failed) see when the player drops or throws an object.

Containers

A container is a thing that can hold other things.

@create $container named steamer trunk;steamer;trunk
put staff in trunk
You put the staff in the steamer trunk.

Containers normally have an opacity of 1, which means that you can see into them when they’re open, but not when they’re closed. You can use

@opaque container is 0
to make a clear container, and

@opaque container is 2
to make a container that is opaque even when it is open. Containers can be opened and closed. If you want to restrict who can open a container, you can use keys, with:

@lock_for_open container with key
Unlock a container with @unlock_for_open. Since containers are $things, @lock works the same for containers as for other things, restricting who can and cannot take the container.

Notes and Letters

A note is a thing you can write on. A letter is a note that can be given to someone and then burned (thus saving your quota).

You can encrypt a note or letter with a key, so that only you or people you specify can read the note:

encrypt Balder's Notepad with me || CapVideo
allows only me (Balder) or Captain Video to read Balder’s Notepad. See the part about keys, on page 22, for more information about ‘constructing’ encryption keys. To make it so that anyone can read the note, use:

decrypt object
This erases the keys you’ve constructed for the object. @Locking a note or letter is the same as @locking a thing.

**Rooms**

To see a list of messages that you can change for your room, go into the room and type @messages here. For a list of messages on an exit, type @messages exit-name.

**Arrival and Departure Messages**

Entrances and exits are normally quiet. When a player goes through a door, the door simply plops the player out the other side. In some cases, you’ll want an entrance to say something, either to the player or to the other players in the room that the player is leaving or entering. Here are the messages for exits (which, on the other side, are entrances):

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>leave</td>
<td>What the player sees when he leaves the room.</td>
</tr>
<tr>
<td>oleave</td>
<td>What the other players see when the player leaves.</td>
</tr>
<tr>
<td>arrive</td>
<td>What the player sees when he enters the room.</td>
</tr>
<tr>
<td>oarrive</td>
<td>What the other players see when the player enters.</td>
</tr>
</tbody>
</table>

For example, I might want to set the `leave` and `arrive` messages on the fire (that leads to the chimney of Balder’s Hall) as:

```
@leave fire is "No fear of getting burned, eh, %n?"
You set the "leave" message of fireplace (#83).
@arrive fire is "The flame burns your butt as you hang in %l."
You set the "arrive" message of fireplace (#83).
```

And, so that people in the room see what’s going on, I might:

```
@oleave fire is "takes leave of %p senses and crawls into the burning fire."
You set the "oleave" message of fireplace (#83).
@oarrive fire is "climbs up out of the fire, smoke rising from %p eyebrows."
You set the "oarrive" message of fireplace (#83).
```

Note that I did *not* use “%N”. The `oleave` and `oarrive` messages automatically put the player’s name in front of the message, whether you want it or not, so you’ll have to want it. When I go up the chimney, the people in Balder’s Hall see:

```
Balder takes leave of his senses and crawls into the burning fire.
```

If there’s already someone in the chimney, they see:

```
Balder climbs up out of the fire, smoke rising from his eyebrows.
```
Note the use of ‘%p’ for possessive pronoun. This takes care of whether or not the player using the exit is a him, a her, or an it.

**Who Location**

When people are in your room, your room is listed whenever anyone does @who. *How* it appears depends on the rooms *who_location* message. Normally, it is simply ‘%T’, and that’s recommended. But you can change it:

<table>
<thead>
<tr>
<th>@who</th>
<th>Player name</th>
<th>Connected</th>
<th>Idle time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Balder (#78)</td>
<td>11 minutes</td>
<td>0 seconds</td>
<td>Balder’s Hall</td>
</tr>
</tbody>
</table>

Total: 1 player, who has been active recently.

@who_location here is ‘By the fire in %t.
You set the ‘who_location’ message of Balder’s Hall (#81).

<table>
<thead>
<tr>
<th>@who</th>
<th>Player name</th>
<th>Connected</th>
<th>Idle time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Balder (#78)</td>
<td>11 minutes</td>
<td>0 seconds</td>
<td>By the fire in Balder’s Hall</td>
</tr>
</tbody>
</table>

Total: 1 player, who has been active recently.

**Free Home and Residents**

Normally, the rooms you create will allow anyone to make it their home. You can lock people out by setting the property *free_home* to 0. (In computer talk, ‘zero’ is often used for *no* or *false*. ‘One’ is used for *yes* or *true*.) If you have *free_home* set to 0, the @resident verb will allow specific players to use your room as their home.

@set here.free_home to 0
@resident Carol, Bob, Ted, Alice

Now, the players Carol, Bob, Ted, and Alice can set your current location as their home. You can only do set properties and residents for rooms that you own.

**Locking Your Doors**

You can restrict access to your room to certain people, or restrict specific people from your room. For example,

@lock here with me

will make it so that only *you* can get into your room.

@lock here with !capvideo &!thor

---

1 This is a feature of Valhalla. In other MOOs, the default is usually that rooms will *not* allow others to make it their home.
will lock both CapVideo and Thor out of your room.

@lock here with me || (bible & cross)

will lock everyone out except you or someone carrying your bible and your cross. See page 22 for more information about making keys to lock your rooms up with. Use

@unlock here

to unlock a room so that anyone can get in. @unlock erases your key requirements, so if you want to remember them, write them down.

**Making Children**

Messages (properties) get transferred through children. So, if you want a new object with similar messages to an object you already have, make the new object a child of the other object, either with

@create other object named new object

or by using

@chparent new object to other object

on an object you’ve already created. If you can’t currently see the other object or the new object, you’ll need to refer to them by object number. Only you can create children from objects that you’ve made, unless you make the object **fertile**:

@chmod object to +f

Fertile objects can be used in @creates by anyone.

**Using Universal Resource Locators (URLs)**

Objects in Valhalla can have ‘URLs’ attached to them. If someone looks at the object and has the correct client software (such as HyperMU* for the Macintosh), the URL will be displayed at their computer. A URL can be any standard Internet service that is covered under the World Wide Web.
PROGRAMMING

So you want to become a MOO\(^1\) programmer? Run, do not walk, to the LambdaMOO ftp site and get the latest text version of the LambdaMOO Programmer’s Manual. You will need to continually reference this book. If you want to print it out, get the postscript version, but I also recommend keeping the text version on your personal computer so that you can refer to it and do searches on it. I’m only going to talk about the basics of MOO programming here. You'll need the official manual as a reference, and you should read through it at least once.

Inside the MOO, *help programming* and *help prog-index* will give you a list of topics that might interest you. The command *@classes utilities* will give you a list of all the built-in utility objects available for programming. There’s a special discussion group for programmers on Valhalla called *Universe.*\(^4\)

BEComing a programmer

In order to become a programmer, you need to talk to a friendly wizard. Most likely, you will be given a new player to use as a programmer, and you will keep your current player so that you can use it as a guinea pig for your new creations.

Properties

You’re already quite familiar with properties. The *messages* you’ve been setting are properties, as is your name and your description. Most of the properties you’ve dealt with so far have been *string* properties. There are four basic kinds of properties in MOO.

Strings

not written

Numbers

not written

---

\(^1\) From now on, I'm going to talk about MOOs in general, rather than specifically Valhalla.

\(^2\) You can still read this even if you’re not a programmer. However, only programmers can *send* messages to the group.

\(^4\)
Objects
not written

Lists
not written

Verbs

Verbs are the ‘commands’ of the MOO. You’ve been using verbs left and right. When you ‘look’ or ‘go’, when you talk, emote, page, or whisper, you’re using verbs.

Classes of Objects

Verbs must be attached to objects, whether the object is a player, a room, or some thing. Verbs should be attached to appropriate objects. If you have a cat named Lucretia, and you want people to be able to kick it, you should attach the kick verb to Lucretia, and not the room you happen to be in. People are going to want to kick your cat no matter where they find it. A kick verb on a room might be appropriate if the room is a kick-boxing arena.

MOO is an object oriented programming language. This means that it has classes of objects. When you look at the parents of an object, you’re looking at the classes that the object belongs to.

@parents Lucretia
   Lucretia(#392)   generic cat(#259)   generic animal (#258)   generic thing(#5)   Root Class
   (#1)

Lucretia is a member of the class generic cat. Cats, in general, are members of the class generic animal, which are members of the class generic thing. Everything in the MOO is based on the Root Class.

Suppose you type kick Lucretia. Assuming that there is no kick verb on you or on the room you are in, the MOO looks to see if Lucretia has a kick verb. If she does, the MOO ‘runs’ that kick verb. If not, the MOO checks to see if cats can be kicked; then, it checks animals, and then things, and finally the Root Class.

This order of verb-searching means that higher-order objects can override verbs in lower-order objects. If both generic thing and generic cat have a kick verb, only the verb on generic cat is used by the MOO.

---

1 Remember the order that the MOO looks for verbs: player, room, direct object, indirect object.
Sentence Structure

There are three basic sentence forms in MOOs:

```
verb
verb direct-object
verb direct-object preposition indirect object
```

When you create a verb, you need to tell the MOO what kind of sentence the verb belongs in, and what kind of direct or indirect objects are allowed. For example,

```
@verb here:cough none none none
```

creates a `cough` verb in the current room. The cough verb only works if the player doesn't type anything except ‘cough’.

```
@verb staff:swing this at any
```

creates a `swing` verb on the staff. This verb needs the direct object to be the staff (this), the preposition to be at\(^6\), and the player must swing it at `something`, although this something can be anything.

The LambdaMOO Programmer’s Manual discusses verbs in much more detail. The important thing to remember here is that, when the MOO starts looking for a verb, it basically ignores any verbs whose sentence structure doesn't match what the player typed. If the player types `cough loudly`, the MOO will not use the cough verb we created above. If the player types `swing staff in time`, the MOO will not use the swing verb we created.

In both cases, if the MOO doesn't find an appropriate verb/sentence structure combination, it will make suggestions to the player. In the above examples, we might expect:

```
cough loudly
  I don't understand that.
  Try this instead: cough
swing staff in time
  I don't understand that.
  Try this instead: swing staff to time
```

Editing Properties and Verbs

Properties: Notedit

You can edit string properties, or lists properties that are lists of strings, with the `@notedit` verb: `@notedit object.property`. For example, if you want to give the current room a description that has more than one line—i.e, that is a ‘list’ of strings—use `@notedit`:

```
@notedit here.description
```

The property editor works just like the mail editor, except that you use `save` when you’re finished, instead of `send`. `Save` saves the changes you’ve made to the property. When you’re done

---

1 Because prepositions are applied in groups, `to` will also work. `To` is part of the same group as `at`. 
editing and want to exit the editor, use `done`. The `done` verb doesn’t save the changes you’ve made—that’s what `save` is for—but it *does* keep track of what you’ve done if you haven’t saved. You can return to your editing by typing `@notedit` without anything else.

**Verbs: Edit**

When you need to edit a verb, you use `@edit object:verb`. It’s the standard editor that you’ve used to send mail and possibly to edit properties. When you’re ready to save the changes you’ve made, type `compile`. You will be told if there are any ‘syntax’ problems in the verb, and what line these problems are in. Use `done` to leave the editor.

**The Generic Swinging Weapon**

Even though I’m going to leave the majority of programming instruction to the official manual, I’m a firm believer that you can never have too many examples. This is Valhalla. Lots of gods with short tempers. We will have swords, staves, and hammers coming out of (and into) our ears. Let's make a *generic swinging weapon* to base them all on.

```moowl
@create $thing named generic swinging weapon, swinger
@describe swinger as "a nondescript weapon of untold power; be careful with it: you might knock someone’s eye out!
```

We need a verb of some sort so that we can *hit* people with it. What are the possibilities?

- hit Jack with sword
- strike Thor with staff

There is a *big* problem with both of these. Remember the order of searching? The MOO is going to look at *Jack* and *Thor* (the direct objects) before it looks at *sword* or *staff*. If I bother them too much, they’ll create a *swing* verb on themselves:

```moowl
@verb me:"Hit Strike" any with this
@edit me:Hit
*player:tell("You miss.");
*player.location:announce($string_utils:pronoun_subs("%N strikes at %d with %p %i. %D dodges nimbly, and %n strikes the ground, leaving %r embarrassed and revealing %p utter incompetence.");
compile
done
```

Since the MOO will get to *this* verb before it gets to *ours*, whenever we try to *hit* or *strike* Thor, we’ll see:

```
You miss.
```

And everyone else will see:

```
Balder strikes at Thor with his staff. Thor dodges nimbly, and Balder strikes the ground, leaving himself embarrassed and revealing his utter incompetence.
```

Do we really want *that*? How about, since this is a *generic* swinging weapon:

```moowl
swing staff at Thor
```
Okay.

@verb swinger:swing this at any
@edit swinger:swing

The @verb creates the verb ‘swing’ on ‘swinger’. And then @edit allows us to edit it, and add instructions:

@if (valid(iobj))

If the player typed a valid object—something that exists—do the rest of this. Otherwise, Valhalla will jump down to the ‘else’ a couple lines down.

@player:tell("You hit *.iobj.name,* with your *,this.name,.*");
@player.location:announce_all_but({player,iobj}, $string_utils:pronoun_sub("%N hits %i with %p %d. It is a crushing blow.")

Here, we first tell the player who swung the weapon that the weapon (‘this.name’) hit the target (‘iobj.name’). ‘Name’ in each case is a property of the object referred to by ‘this’ and ‘iobj’. You refer to object properties with object.property. Tell is a verb that exists on player. We use it by typing player:tell. Announce_all_but is a verb that exists on the room that the player is in. We know the room because it is in the player’s property location. So, player.location:announce_all_but uses that verb. First, Valhalla interprets ‘player.location’, and then looks for the verb announce_all_but on that location. This is how we refer to verbs: object:verb(information for the verb).

Then, we tell everybody in the room—except the player, who we already told, and the target (‘iobj’), who we’ll tell in a moment—the same thing. Notice that we also call the pronoun_sub verb to interpret all those ‘%’ things. The pronoun_sub verb is on an object called $string_utils.

@if ($object_utils:has_callable_verb(iobj,"tell")

$iobj:tell($string_utils:pronoun_sub("%N swings %p %d at you and deals an ugly wound. Good lord, %i, are you going to stand for that?");
@endif

Here’s a complete if statement. If the target (‘iobj’) has a verb that we can use called ‘tell’, Valhalla will use the verb after the if statement and before the endif statement. In this case, it tells the iobj that it has been hit by the player swinging the weapon.

@else

@player:tell("I don’t see any *.iobjstr,* here. Feeling a little troubled?");
@player.location:announce($string_utils:pronoun_sub("%N appears to be developing an interesting rapport with %p %d");
@endif

This else and endif belong to the first ‘if’ at the beginning of this verb. The part between the else and the endif here will only be used if the if (valid(iobj)) comes back and says “no, the player didn’t refer to a valid object.” In other words, the MOO uses the first part (between the if and the else) if the statement valid(iobj) is ‘true’; else it uses the second part (between the else and the endif).

And what we’ve decided to do is, first, tell the player that we couldn’t find any ‘iobjstr’ to swing at, and, second, announce to everyone in the room that the player is doing something strange. Iobjstr is the text after the first preposition. It’s like iobj, except that it’s text instead of an object
number. *Announce* is a verb that all rooms have, and it *tells* everyone in the room—except for the player who called the verb—whatever we put between the parentheses. In this case, that the player “appears to be developing an interesting rapport with” a generic weapon.

*compile*

*Compile* tells the MOO to check the verb for errors, and then save the verb so that it can be used. You must compile your verbs before what you typed can be used.

*dona*

*Dona* returns you to wherever you were before you started editing the verb.

@end parent staff to swinger

Balder already created a *staff* in order to demonstrate creating objects. The *@chparent* command makes that staff a child of *swinger*, as if Balder had originally created it with *@create swinger named staff*. The staff now can do anything that a generic weapon can.

`swing staff at Captain Video`

*You swing the staff at Captain Video.*

Everyone else (except Captain Video) sees:

*Balder hits Captain Video with his staff. It is a crushing blow.*

And Captain Video sees:

*Balder swings his staff at you and deals an ugly wound. Good lord, Captain Video, are you going to stand for that?*

Now, this isn’t really that interesting of a generic. You can *@create* all the staves, swords, and maces you want, they’ll still all “hit” and “deal ugly wounds”. At the end of this section, I’ll show you a better form of generic, that includes messages—properties that the owner can change, like all of the messages we fooled around with in *Advanced Building*.

**The Advantages of Being Object-Oriented**

Let’s make a present for Thor. First, we need to complete one more thing on the *generic swinging weapon*. We need to make it *fertile*.

* @chmod swinger to +f

This adds ‘fertility’ to the swinger. Until an object is fertile, only the owner can make children from it. *Anyone* can make children from a fertile object. Now that swinger is fertile, anyone in Valhalla can *@create* *swingers* named whatever they want.

* @create swinger named “Thor’s Hammer”, hammer, Mjolnir

Now, as anyone who reads comics—I mean, studied Norse mythology—knows, Thor’s Hammer is not your average weapon. One of its biggest properties is that it returns to Thor after Thor throws it.

Right now, Thor’s hammer can be ‘thrown’ just like any other object. It’s a *$thing*, and all *$things* have a *throw* verb which allows them to be thrown. Because Valhalla is object-oriented, we can *override* the basic *throw* verb with our own:
I use the @show command to find out what the exact syntax for throw is. If I don’t copy it exactly, my ‘throw’ may not override all instances that players will use.

```
@verb hammer:throw* this none none
@edit hammer:throw
*if (player == this.owner)
  *pass(@args);
*else
  *player:tell("You fling the *,dobjstr,* across the sky.");
  *player.location:announce(player.name, " flings *,player.pp,* *,dobjstr,* out of sight!");
  *player.location:announce("The *,dobjstr,* flips back around and returns to *,player.name,*.");
  *player:tell("Your *,dobjstr,* returns to your grasp.");
*endif
compile
done
```

The important thing to look at here is the pass(@args); verb. This passes the sentence to the parent verb. If anyone other than the hammer’s owner throws the hammer, it throws normally. If the owner (Thor) throws it, we take control and send out our special messages. By passing control on to the parent verb when we do want the hammer to be thrown normally, we don’t have to worry about what goes on in throwing an object, and, if the administrators ever improve their ‘throw’ verb, our own ‘throw’ verb will reflect these improvements.

Also, the original ‘throw’ verb is combined with drop. We didn’t override drop at all, so if Thor drops his hammer, it won’t return to him.

If you have any other questions about the statements in this verb, see the LambdaMOO Programmer's Manual, available at all fine virtual bookstores.
A Better Generic Swinging Weapon

@create $thing named generic swinging weapon:generic swinging weapon, swinger
@describe swinger as "a nondescript weapon of untold power; be careful with it: you might
knock someone’s eye out."
@prop swinger."swing_msg" "You &hit %i with your %d, &wounds."
@prop swinger.tswing_msg "%N &ohits you with &p %d, &wounds. &tcomment"
@prop swinger.oswing_msg "%N &ohits %i with &p %d, &wounds."
@prop swinger.wounds_msg "dealing a deadly wound"
@prop swinger.no_target_msg "I don’t see any %i here. Feeling a little troubled?"
@prop swinger.ono_target_msg "%N appears to be developing an interesting rapport with
%p %d."
@prop swinger.hit_msg "hit"
@prop swinger.ohits_msg "hits"
@prop swinger.tcomment_msg "Good Lord, %i, are you going to stand for this?"
@prop swinger.help_msg ""
@notedit swinger.help_msg

*The generic swinging weapon has the following messages:
  * tswing: What the target sees when the weapon is swung.
  * swing: What the swinger sees when the weapon is swung.
  * oswing: What others see when the weapon is swung.
  * no_target: What the swinger sees when the weapon is swung at a target that doesn’t
    exist.
  * ono_target: What others see when the weapon is swung at a target that doesn’t exist.

*In addition, the following ‘ampersand’ substitutions can be placed in the above messages:
  * &wounds: Should be set to the kind of wounds the weapon does as a phrase: "dealing a
deadly wound"
  * &hit: What the weapon does when it hits as the swinger sees it: "hit"
  * &ohit: What the weapon does when it hits as others see it: "hits"
  * &tcomment: An extra comment for the target. Should be a complete sentence. "Good
Lord, %i, are you going to stand for this?"

*For the current state of the above messages, use the @messages verb.

*Remember to be careful what you do with this thing! We don’t want any ownerless eyeballs
floating around Valhalla.
@verb swinger:swing this at any
@program swinger:swing
if (valid(lobj))
    player:tell($string_utils:pronoun_sub(this:hitting_sub(this.swing_msg)));
    player.location:announce_all_but({player, lobj},
    $string_utils:pronoun_sub(this:hitting_sub(this.oswing_msg)));
    if ($object_utils:has_callable_verb(lobj, "tell"))
        lobj:tell($string_utils:pronoun_sub(this:hitting_sub(this.tswing_msg)));
    endif
else
    player:tell($string_utils:pronoun_sub(this:hitting_sub(this.no_target_msg)));

Forks and Tasks

Let's go back to Thor's hammer. We're not really letting him throw his hammer. We just tell everybody he threw it, which is almost the same thing. But we can also have the hammer actually be thrown, and then return after a short bit of time. We'll re-write throw so that when Thor throws his hammer, it waits a bit at the destination, and then sails through the air back into his hand.

MOOs do not allow long-playing verbs. Any verb that takes too much time is terminated abruptly, with nary a polite goodbye. If your verb needs to wait before doing something, you have to tell it to go to sleep.

Using Files

The File Utilities Object

not written
Wizards

Creating Players

To create a player, you need to specify the player’s name (which must be one word), the player’s e-mail address, and an optional comment:

@make-player name e-mail@address comment

For example, to create a player for Captain Video, with name Captain, whose electronic mail address is capvideo@example.com, use:

@make-player Captain capvideo@example.com Captain Video

You will be told what the player’s password and object number is. Write them down. You’ll need to give the password to the person so that they can log in with their player, and you may need the number to refer to the player when you can’t see them.

Creating Wizards, Programmers, and Builders

Players can’t do a whole lot except interact. Most people will probably want to be, at least, builders\(^7\). This means that they can build objects, although these objects must be based on other objects that a programmer has made. A programmer is a builder, and a wizard is automatically a programmer and a builder. Most people should not be wizards. Wizards have the ability to kill other players, to muck with other players’ objects, and to cause considerable havoc if they aren’t responsible and competent.

You also do not want to create wizards out of long-existing player. When you turn a player into a wizard, all of the verbs that player created have all the rights of wizards. Including the ones they wrote before they knew how to program. If you want to give wizard status to a long-existing player, make a new player, turn it into a wizard, and give it to the person. This also ensures that the person has a ‘non-wizarded’ player to use when testing new verbs.

You should generally leave the task of designating who is and is not a wizard to the administrator. It is their responsibility to delegate authority to the lesser wizards of the MOO.

Create builders, programmers, or wizards with the following commands:

@chparent #playernumber to $builder
@programmer #playernumber
@chparent #playernumber to $wiz

---

\(^1\) On Valhalla, all players start as builders, although they have a quota of zero objects.
In addition, for wizards, the *wizard* property must be set. Wizards must be programmers before they can be wizards:

```
@set #playernumber.wizard to 1
```

A wizard can take away all programmer and builder status with the last command, which turns the player into a non-builder.

```
@chparent #playernumber to $player
@set #playernumber.wizard to 0
@set #playernumber.programmer to 0
```

Only the grand-high wizard, *Thor*, can take away wizard status.\footnote{MOOs other than Valhalla, of course, will have different grand high wizards.}
When you get the latest version of MOO, it’s in source code format. You need to compile the source code so that it will run on your computer. If you are using a Unix computer, it should be easy: type make and press return. If it runs into some errors, you’ll have to talk to a programmer about fixing them.

You’re not going to be able to do everything yourself. You may not even personally know much about programming. That’s what you’ll need wizards for. Wizards are MOO players to whom you delegate authority. Wizards can modify any part of the MOO, and can edit and create verbs that affect everyone. Obviously, you should only choose wizards who you can trust, and who you know are reasonably competent.

On Valhalla, there’s a property called #0.admins that lists which players are administrators. You can edit this list with @notedit. When your administrators write verbs that only administrators should be able to use, they need to have their verbs check against this list.

You will almost certainly want to personalize what potential members of your MOO see when they first connect. The $login object contains everything that can be seen or done while in the ‘waiting area’ before logging in to your MOO.

Your welcome message explains what the purpose of your MOO or the theme of your MOO, or both. You can also add a little help into the welcome message, for new members who may not remember exactly how to get inside. The welcome message property is $login.welcome_message. You or one of your wizards can use @notedit $login.welcome_message to edit and change the message text.
**Player Creation**

When players are created by you or one of your wizards, and an electronic mail address is specified for the player, Valhalla sends the player a message over electronic mail, telling them that the player is ready, and how to get to it. The subject of the message is in $wiz_utils.new_player_subject, and the text of the message is in $wiz_utils.new_player_message. You can @notedit these properties if you need to modify them. Type help $wiz_utils:new_player_message for more information.

This mail message is only sent if an external electronic mail address (outside of the MOO) is given, and (because no e-mail address is specified) is not sent to people who create their own players at the Valhalla’s entrance.

**Maximum Users**

not written

**MONITORING PLAYERS**

@Net-Who

not written

**CHECKPOINTS**

Dump Interval

not written

**BACKUPS**

Database

not written

---

1 LambdaMOO 1.7.7 requires a special ‘patch’ in order to do this. Valhalla has this patch installed. It was posted to the MOO-Cows mailing list, and is available on the Cerebus MOO Object gopher/ftp site.
Objects
not written

Discussion Groups
The MOO has it’s own set of discussion groups. They operate through the MOO’s mail functions. You can create and maintain groups for any topic you desire. The basic discussion groups in Valhalla are Life, the Universe, and Everything.

Creating A New Discussion Group
You’ll want to read the help, by typing help $mail_recipient. You need to create a child of $mail_recipient, describe the child, set the readers property to 1 (so that anyone can read it), and then move the child to $mail_agent:

```
@create $mail_recipient named Life-in-Asgard
@describe Life-in-Asgard as "A discussion group specifically for discussing the trials and tribulations of living in the city of Asgard."
@set Life-in-Asgard.readers to 1
@move Life-in-Asgard to $mail_agent
```

If you want to make the discussion group available only to a certain group of players, follow the help for $mail_recipient.

New Generics
On Valhalla, there are five Valhalla classes that ‘intercept’ the standard generic classes. These are:

ValhallaRoomClass  Intercepts Sroom
ValhallaThingClass  Intercepts Sthing
ValhallaExitClass  Intercepts Sexit
ValhallaPlayerClass  Intercepts Splayer
ValhallaRootClass  Intercepts the root class.

If you want to make changes to an entire class of objects (such as all players, for example), you can ask one of your wizards to change the corresponding Valhalla class.

Each time you upgrade to a newer version of MOO, the standard generics are replaced with newer versions. By making your changes to a separate class that has been inserted into the class hierarchy, it will be easier for you to make upgrades.

New Help
not written
INTERNET

If you want Internet access in your MOO, it has to be enabled twice: first, whoever compiles the MOO code on your machine has to enable it, and second, you need to set $network.enabled to 1.

Mail
not written

Gopher

The current standard for gopher access in MOOs is the *Generic Gopher Slate*. The MOO code for it is available at the LambdaMOO ftp site, in /pub/MOO/contrib. Gopher is, of course, pretty much dead. Support for this may be disappearing from Valhalla.

World-Wide Web
not written

Usenet News
not written

BLACKLISTS

Newts
not written

Toads
not written

Blacklists
not written
GNU Free Documentation License

Version 1.3, 3 November 2008
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book.

We recommend this License principally for works whose purpose is instruction or reference.

1. Applicability and definitions

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed verbatim, in a machine-readable Copying in quantity

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.
If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. Modifications
You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document’s license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version. If there is any section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. Combining documents
You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. Collections of documents
You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. Aggregation with independent works
A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.
8. Translation
Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. Termination
You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. Future revisions of this license
The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. Relicensing
“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

“Eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.