OpenSim サーバシステム構築入門 (v0.7.2 対応)

-- & ネットワークサーバ構築入門 --

OpenSim サーバシステム構築入門 (v0.7.2 対応)

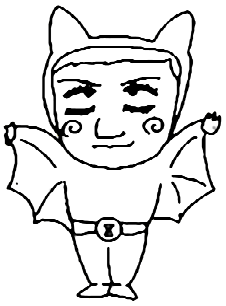
-- & ネットワークサーバ構築入門 --

『キャラクター・プロフィール』

名前 リサ(Lisa)
性格 Macintosh という林檎が好き
OpemSimにはついてそこそこ詳しい. しっかりモノである.

名前 クラリス
性格 Linux については比較的初心者
すぐに何でもやろうとするが大体失敗する.

名前 ソナタ(Sonata)
性格 普段は Linux を使ってるが実は Mac 好き
OpenSimの管理をやってるがおっちょこちょいで, 昔MOにファイルをバックアップするとき, MacにSCSIドライブを刺したらマシンがクラッシュして, 原稿が全部吹っ飛んだこともあるらしい.



名前 Fumi Hax
昭和3x年生まれのハッカー崩れ.
自称NSLの鬼教官
最近老眼が.... 何の, まだまだ若いもんにゃ負けん!!
うりゃ〜!!
未熟者め. 修行が足らん!!!!! タイヤ交...じゃない. Linux インストール・設定 100 回じゃ!!!!!!!!!

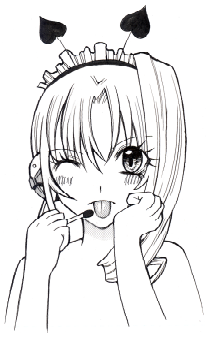
東京情報大学 ネットワークシステム研究室 メタバース研究会
キャラ原案・イラストレーション 「サミッティー」

目次

1. はじめに

この章では Opensim とは何なのか．また，本書の目標と対象となる読者について記述したいと思います．

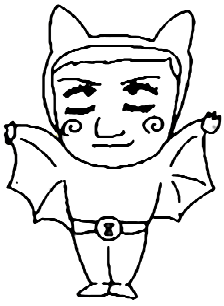
そもそも，OpenSimって何なのよ？



案内役なんだからそれくらい知ってなさい
OpenSimはフリーの3次元仮想空間
構築用サーバーのことよ！

じゃぁ早速やってみよう！
Windows 起動!

まってまって！
この本は基本 Linux での設定で解説
いくんだよ．



Windows でのセットアップは
付録 B だ．

1.1 OpenSimとは何か

OpenSim (OpenSimulator) はSecond Lifeサーバと互換性のある3次元仮想空間構築用のオープンソースのサーバです(http://opensimulator.org/).OpenSimはクライアントとの通信に Second Lifeの通信プロトコルと同じプロトコルを使用しますので, Second Life用のビューア (Viewer) をそのまま使用することも可能です (ただし, 本書執筆時点では Second Life Viewer v2.xは使用不能です).

筆時点でのバージョンは 0.7.2で, まだαバージョンですが, 一部, Second Lifeを超える機能も実装されています. ただし, ODE (Open Dynamics Engine) の物理エンジン部分がまだ弱く, 物理オブジェクトに関するスクリプト(銃器などのスクリプト)を実行しようとするとサーバプロセスが落ちてしまう場合もあります.

OpenSimはフリーのオープンソースですので,自分のサーバ上に自由に3次元仮想空間を構築し,公開することも可能です. OpenSimは OpenSimプロジェクトにより精力的に開発が行われており, 近い将来 WEBサーバと同じように,各サイトで独自の3次元仮想空間が構築されるようになるかもしれません (なるといいな).

OpenSimプロジェクトの他に, OpenMeataverseというプロジェクトもあります (http://openmetaverse.org/). OpenSimは OpenMestaverseの成果を取り入れて構築されています. OpenSimおよびOpenMetaverseプロジェクトの目標は「オープンなメタバースの構築」です.

メタバースとは上位概念を表すメタと, 世界と表すユーバースを接合した造語で, ニール・スチーブンスンのSF小説「スノー・クラッシュ」の中に登場する3次元仮想世界 (空間) のことを指します. 転じて, 今日では, 3次元仮想空間全般をメタバースと呼ぶようになりました.「スノー・クラッシュ」の世界は宅配ピザ用大学があるなどかなりぶっ飛んだ設定で,辻褄が合わないと思われる箇所もありますが, 全体的には非常に面白く, 3次元仮想世界 (空間) も魅力的に描かれています. ご興味のある方は是非ご一読を (「スノー・クラッシュ」ハヤカワ文庫SF, 2001).

現在実在する3次元仮想空間サービスの中で,「スノー・クラッシュ」で語られるメタバースにもっとも近いとされているのが Second Lifeです. そこで, OpenSimプロジェクトでは, オープンなメタバース作成の手本として, とりあえず Second Lifeを目標に開発が進められているのです.

OpenSim (Sencond Life) を利用するにあたり最も注意すべきことは, これはいわゆるネットワークゲームではないということです. OpenSim (Second Life) には敵もモンスターもいません. やらなければならないクエストもありません.最初にあるのは,ただ土地だけです.それは,例えば MS Windowsなどの OSと同じ基本システムです. 他のあらゆるものは自分で作り上げるか, 他から持ってこければなりません.

従って,「自分で何か作ってみたい」,「仮想世界を構築したい」と思う人にはとても魅力的ですが, そこでただゲームをしたいと思っている人にとってはこれ程つまらないものはありません. 我々はOpenSimやSecond Lifeなどのメタバースはメディアの一種であると考えています.

1.2 本書の目標と大まかな内容

本書では LinuxへのOpenSimのインストールから, OpenSimの公開と運営までの技術的な手法について解説しています. OpenSimの公開までの過程で, いくつかのネットワークサーバを構築する必要がありますが, それらについても出来る限り詳しく説明しています. また, 本書でのターゲットOSはCentOSですが, CentOS固有の部分はなるべく少なくなるようにしていますので, 他のLinuxディストリビューションをお使いの人にも十分の役立つと思っています.

本書では重要なネットワークプログラムについては, Unixの伝統を重んじ, aptや yumなどのパッケージ管理システム(正確に言えば,aptや yumはパッケージ管理システムのネットワーク用インターフェイスです) を使用せずにソースコードからコンパイルする方法を紹介します. つまり本書の内容は OpenSimシステムの構築を第一の目標としていますが,読者のネットワークスキルの向上も視野に入れて構成されています.「Ubuntu をPCにインストールしたが, Windowsとどこが違うのだ」などと

思っている人にはお勧めです.

ただし，どうしてもコンパイルが旨く行かない場合や，早急にOpenSimシステムを構築したい場合などは，aptやyumなどを使用してインストールしても良いでしょう.

1.3 対象とするユーザ（読者）

本書の読者としては，以下のスキルレベルを想定しています.

1） 自分のPCにLinuxをインストールすることができる.

手元のPCにLinuxがインストールされていなければ話になりませんので，少なくとも自分でPCにLinuxをインストールする位のスキルは必要です．ただし，最近のLinuxディストリビューションのインストーラは非常に良くできていて，殆ど「次」ボタンをクリックして行けばインストールは完了します.MS Windowsとデュアルブートなどを行うというのでなければ,簡単にインストール可能です.

2） Linuxのコマンドラインで基本的なコマンドを使用することができる.

ネットワークサーバにはグラフィカル・ユーザ・インターフェース（GUI），つまりWindowシステムは不要です．GUIはCPUやメモリなどのリソースを大量に消費します．通常ネットワークサーバは設定が完了すれば，ディスプレイやキーボード，マウスは取り外し，電源とネットワークケーブルのみ繋いで机の下にでも転がしておきます.

本書の説明でも最初にGUIは止めますので，Linuxのコマンドラインで基本的なコマンドが使用できるくらいのスキルは必要です．chmod や vi の使い方を理解しているレベルであれば問題ないでしょう．もし 「chmod なんて聞いたこともない」というレベルであれば，別途Linuxの入門書やコマンドリファレンスなどの書籍の購入をお勧めします．何なら，そのまま勢い余って LPIC(http://www.lpi.or.jp/）の資格を取ってしまうというのも有りでしょう.

1.4 想定するコンピュータ環境

1） MS Windowsマシン以外のPC.

MS Windowsと同じマシンにLinuxをインストールしてデュアルブートにする手もありますが,使い勝手などを考慮すれば別のPCを用意した方が良いでしょう．ただし，前節にもある通り，このマシンではGUIなどの負荷のかかるプロセスは全て停止させますので,大規模なシステムを組むのでなければ,マシンのスペックとしてはそんなに高くなくても大丈夫です.CPUは Core2 Duoで十分(Pentium4でもOK）ですし，グラフィックカードは文字がでればOKです．ただし，メモリはできるだけ積んだ方が良いでしょう（1GByte以上).

2） ネットワーク接続.

ネットワークサーバの構築を行いますので，インターネットへの接続環境は必須です．自宅でブロードバンドルータを使用した接続でも構いませんが,ルータの種類によってはOpenSimを外部に公開するのが難しい場合もあります．大学や企業などでOpenSimを外部に公開する場合は，ネットワーク管理部門にファイアウォールの設定変更をお願いして,通信ポートを開けてもらう必要があります.

1.5 記述に関する注意

本文中のコマンド例で **$** は一般ユーザのコマンドプロンプト，**#** はrootのコマンドプロンプトをあらわします．コマンドプロンプトが $ の場合は，一般ユーザでも起動可能なコマンドであることを示しています.

本文中で示しているコマンド例はあくまでも標準的な例であり,サーバマシンの設定や環境,Linuxのディストリビューション，OpenSimのバージョン，展開場所などによって変化する可能性があります.

また，内容に関してましても，万全を期しているつもりですが，間違いや勘違いなどがあるかもしれません．予めご了承ください.

なお,NSLは東京情報大学のネットワークシステム研究室の略号です.

2. Linux のインストール

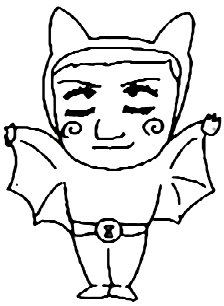
この章では OpenSim サーバを Linux で構築する場合に必要なアプリケーションのインストール方法について紹介します．Linux のディストリビューションは CentOS5.5 を基本としています．その他のディストリビューションをお使いの場合は本書を参考にしながら Google大先生などに尋ねてみてください ;-)

よーし，インストールするぞ！
Linuxの設定なんて全部「次へ」でいいや！

そんな、雑なことしちゃだめだよ（泣
細かい設定もちゃんとやらないとエラーで止まっちゃうよ＝

とくに FireWall や SELinux の設定はインストール時に設定すれば後で慌てないからじっくりやっていこう！

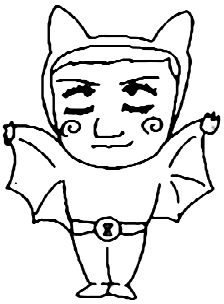
Linux/Unix の基本はコマンドラインだ！
ビシビシ行くぜ！！
目指せハッカー一番星！



え～．でも犯罪者は嫌だよ～

それはクラッカー！

は～い．



2.1 インストール

CentOSは通常の方法にてインストールを行います. ただし, ファイアウォールの設定はOpenSimの通信を妨害するため行わない方が良でしょう. また, 後でFTPサーバを構築する場合には, SELinux(Linuxのセキュリティ機能)が有効になっていると通信を妨害してしまうので「無効(Disable)」にしておきます.もし,ファイアウォールの設定を行ってしまった場合や,SELinuxを「有効(Enforcing)」にしてしまった場合は,「2.4.2 不要なサービスの停止」を参照してください.

2.2 パッケージの更新

インストールの終了後, **yum** コマンドでパッケージの更新を行います(図2.1). ただしインストール直後のアップデートは, 終了するまでかなりの時間がかかる場合があります. また, **apt-get** を使用してアップデートを行うシステムでは 図2.2のようにしてアップデートを行います.

```
# yum -y update
```

図2.1 yum コマンドによるアップデート

```
# apt-get update
# apt-get upgrade
```

図2.2 apt-get コマンドによるアップデート

2.3 開発環境のインストール

大抵のLinuxディストリビューションでは,インストール時には開発環境はインストールされません.CentOSの場合は,開発環境のインストール用スクリプトが http://www.nsl.tuis.ac.jp/DownLoad/SoftWare/Linux/centos-devel.sh に用意されていますので, それを利用して開発環境のインストールを行います(図2.3). 他のディストリビューションについては, 別途それぞれの説明サイトなどを参考にしてインストールしてください.

なおこのCentOS用のスクリプトは, yum を使用する他のLinuxディストリビューションでも使用できる可能性があります （実行しても, パッケージがインストールされないということ以外の問題は発生しません).

```
# wget http://www.nsl.tuis.ac.jp/DownLoad/SoftWare/Linux/centos-devel.sh
# bash centos-devel.sh
```

図2.3 CentOSでの開発環境のインストール

2.4 サーバの設定

2.4.1 Run Level の変更 （オプション設定）

もし, サーバを Run Level 5 (GUI) 以外で稼動させるのであれば, Run Level の変更を行います(マシンをOpenSimのサーバとしてのみ運用するのであれば, Run Level 3を推奨します). Run Level の変更を行うには /etc/inittab をエディタで編集し (図2.4), reboot コマンドなどでサーバを再起動します.

```
# vi /etc/inittab
      id:5:initdefault:     の5を適当なRun Levelに変更する（3を推奨）
# reboot
```

図 2.4 Run Level の変更

ただし，Run Level を 3 ではユーザインターフェイスは CUI（キャラクタ・ユーザ・インターフェイス）となり，GUI(X Window)が起動しなくなるため作業の効率が悪くなる可能性があります．このような場合の対策として，

1. **全ての作業終了後に Run Level を 3 にする．ただし，次項の setup コマンドでの設定はレベル変更後にやり直す必要がある．**
2. **リモートマシンから SSH で接続して作業を行う．（MS Windows から Poderosa や Putty を使って接続するのもお勧め）**
3. **随時，init 5 コマンドなどで，Run Level を 5 に戻して作業する．**

などが考えられます．それぞれの環境に合わせて選択すると良でしょう．

2.4.2 不要なサービスの停止

インストール直後の Linux では，不要なサービス（**デーモン**）が多数起動しています．どうしてもというわけではありませんが，少しでもサーバのリソースを節約するためには，不必要なサービス（デーモン）は全て停止させた方が良いでしょう．CentOS などの RedHat 系の Linux では **setup コマンド**が利用できますので，これを利用して不要なサービス（デーモン）を停止させます（図 2.5, 2.6）．停止させるサービス（デーモン）はサーバの運用形態によっても変化しますので，運用形態に合わせて取捨選択を行います．

また，表 2.7 に参考として通常必要と思われるサービス例（あくまでも例）を示します．

```
# setup
# reboot        （リブートしないと設定は有効にならない）
```

図 2.5 setup コマンドの起動とリブート

- **上下カーソル移動キーで「システムサービス（System service)」を選択する．**
- **Tab キーで「実行ツール（Run Tool)」を選択し，スペースバーを押す．**
- **サービス選択画面では，上下カーソルでサービス選択し，スペースバーで ON, OFF を行う．**
- **設定が終わったら，Tab キーで「OK」ボタンを選択し，スペースバーを押す．**
- **setup を終了するにはメニュー画面で，Tab キーにより「停止（Quit)」を選択し，スペースバーを押す．**

図 2.6 setup コマンドの操作法

acpid	電源管理デーモン
anacron	cronの補助デーモン．指定された日時にOSが落ちていた場合，OSの再起動時に該当プログラムを実行できる
atd	時間指定でコマンドを遅延実行するデーモン
autofs	ファイルシステムを自動でマウントするデーモン
crond	指定された日時にプログラムを起動させるデーモン
irqbalance	マルチCPUマシンにおいて，複数のCPUからの割り込みを可能にする
kudzu	新しいハードウェアを自動認識するデーモン
network	ネットワークサービス
ntpd	ネットワーク上でシステムの時計を設定するデーモン
sshd	リモートマシンからの接続用の sshサーバデーモン
syslog	システムログの収集デーモン
xinetd	スーパーデーモン

表 2.7 通常， 最低限必要と思われるデーモンの例

なお，OS のインストール時に**ファイアウォールを設定した場合**には **iptables** が有効になっている筈ですので，これは必ずOFFにしておいてください．ただし setupコマンドで変更した内容はリブートしないと有効にならないので，その場でファイアウォールを無効にしたい場合は図2.8のようにして手動で停止させると良いでしょう．

```
# /etc/init.d/iptables stop
```

図 2.8 ファイアウォールの停止

また，OSのインストール時にSELinuxを有効(Enforcing)にしてしまった場合は，/etc/sysconfig/selinux を編集し，SELINUX=enforcing を **SELINUX=disabled** に修正します．さらに /boot/grub/menu.lst を編集し kernel オプション行の末尾に **selinux=0** を追加し（図 2.9），マシンを再起動させます．ただし，この修正はFTPサーバを立てるのでなければ特に必要はありません．ここでは設定せず，FTPサーバがどうしても必要になった時点で設定しても良いでしょう．

```
kernel /vmlinuz-..... ro root=..... selinux=0
```

図 2.9 /boot/grub/menu.lst の追加箇所 （太字の部分が追加箇所）

2.4.3 ブート時のセキュリティ設定 （オプション設定）

シングルユーザモードにおいて，パスワードなしでrootとしてログインすることを防止するために /etc/inittab を編集し，図 2.10 の一行を追加します．

実際には図2.10の設定を行っても，起動オプションでinit を別のプログラム（通常はシェル）に置き換えると，パスワードなしで root としてログインできてしまうので，**Grub** の起動オプション変更に対してパスワードによる保護を掛ける方が良いでしょう．具体的には Grubの設定ファイルである /boot/grub/**menu.lst** の先頭部分にパスワード（password）行を追加します（図 2.11）．

```
co:S:respawn:/sbin/sulogin /dev/console
```

図2.10 /etc/inittabへの追加 （追加場所はどこでも良いが，最終行が無難）

```
default=0
timeout=5
password XXXXXXXXX          （← この行を追加）
```

図 2.11 /boot/grab/menu.lst の追加部分 （XXXXXXXXX 部分にパスワードを設定する）

2.4.4 接続制限

ssh を利用したリモートマシンからのログインを許可する場合には，ssh の接続制限を行います．特に組織外部からもssh接続を許可する場合には，接続制限は必須です．/etc/hosts.deny には **ALL: ALL** と一行だけ記述し，一旦（/etc/hosts.*に対応するプロセスの）すべての通信を禁止します．次に，/etc/hosts.allow には **sshd:** に続いて，接続を許可するIPアドレスを，**IPアドレス／サブネットマスク** の形式で指定します（図 2.12，2.13）．ただし，**IPアドレス／サブネットマスク**の形式で記述する場合，指定するIPアドレスはサブネットマスクの範囲で規定されるIPアドレスのうち，一番最初のものでなければなりません（つまりネットワークアドレスを指定しなければなりません）．

例えばローカルループバックは，127.0.0.1/255.0.0.0 ではなく，127.0.0.0/255.0.0.0 と記述しなければなりません．

```
# vi /etc/hosts.deny
# vi /etc/hosts.allow
```

図 2.12 sshd の接続制限用ファイルの編集

```
sshd: 58.158.97.64/255.255.255.224 202.26.159.140
```

図 2.13 /etc/hosts.allow の例（サブネットマスク部が /255.255.255.255 の場合は省略可）

2.4.5 個人設定

個人用（root を含む）のシェル環境の設定ファイル（~/.bashrc, ~/.bash_profile）を用意します. これらは個人の好みによって設定が大きく変わりますが, CentOSではrootの初期設定では /sbin などにコマンドパスが通っていないので, 最低限コマンドのパスなどはちゃんと設定しておくべきでしょう.

なお, 下記に **.bashrc**, **.bash_profile** のサンプルへのリンクを示します. 実際の使用では, 使用環境や個人の嗜好に合わせて, これらを変更して使用してください.

.bashrc のサンプル: http://www.nsl.tuis.ac.jp/etc/setting/bash/.bashrc
.bash_profile のサンプル: http://www.nsl.tuis.ac.jp/etc/setting/bash/.bash_profile

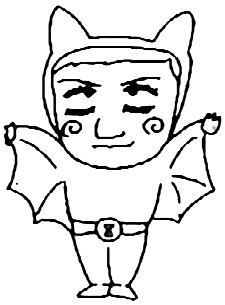
3. 必要なソフトウェアの準備

ここでは OpenSim のインストールの前準備として，必要なソフトウェアのインストール方法の解説を行います. これらのソフトウェアは yumや apt-getなどでインストールしても良いのですが,「最新版を使用する」「スキルアップを図る」と言う意味で,FTPサーバ以外のソフトウェアに関してはソースコードからのインストール方法を示します（FTP サーバは 「2.3 開発環境のインストール」の centos-devel.sh でインストール済みです).

各ソフトウェアのインストールでは，ダウンロードURL（またはサイトURL）より最新のソースコードをダウンロードして，コンパイルおよびインストールを行います.

ふぅ，インストールと設定できた～！

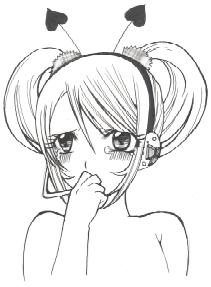
まだまだ，ここからが本番だ!!



じゃぁ,「yum install ･･･」っと

確かに，yumを使った方法のが楽に早くできるけど，OpenSimは開発バージョンなのもあって最新のパッケージに対応することが出来るコンパイルでやっていくよ.

すこし時間もかかるけどマニュアル通りじっくりやれば難しくないよ！



が、 がんばるね＞＜！

3.1 pkgconfig のライブラリディレクトリ

幾つかのソフトウェアではインストールの状態をシステムに通知するために pkgconfig を使用します. CentOS にインストールされている pkgconfig では /usr/lib/pkgconfig に各ソフトウェアの情報を保存しますが,ソースコードからコンパイルしてインストールした場合には,/usr/local/lib/pkgconfigにインストール情報が保存される場合があります.

この不整合を解消するために, ソフトウェアをインストールする前に /usr/local/lib/pkgconfig から /usr/lib/pkgconfig にシンボリックリンクを張り, ソースコードからコンパイルした場合でもインストール情報が /usr/lib/pkgconfig に保存されるようにします (図3.1).

```
# ln -s /usr/lib/pkgconfig /usr/local/lib
```

図 3.1 pkgconfig へのシンボリックリンク

3.2 FTP サーバの設定 （オプション）

「2.4.1 Run Level の変更」で Run Level を3 にした場合, サーバ上では GUI (X Window) が起動しませんので WEBブラウザを動かすことができなくなります.そのためソフトウェアのダウンロードに支障をきたす場合があります. このような場合はサーバマシンで FTPサーバを起動し,MS Windows のWEBブラウザでダウンロードしたものをサーバマシンに FTPクライアントソフト(FFFTPなど)を使用して転送します.

もし他の方法でサーバマシンにソフトウェアを転送することが可能な場合や, Run Level を5のまま使用する場合にはFTP サーバは設定しなくてもかまいません. ソフトウェアによっては, MS Windows 上の WEB ブラウザでダウンロードURL を確認して, サーバ上で wget を使用してダウンロードするという手法も使えます.

FTP サーバソフトとしてはここでは vsftpd を使用します. また vsftpd を管理するスーパーデーモンとしては xinetd を使用します (vftpd, xinetd は「2.3節」の centos-devel.sh の実行によってインストール済みです).

まず vsftpd の設定では, **/etc/vsftpd/vsftpd.conf** を編集し, **anonymous_enable** と **listen** に **NO** を設定します. **anonymous_enable=NO** は匿名 FTP サーバ機能を使用しないことを表し, **listen=NO** は vsftpd が xinetd の管理下で動作することを表します.

また「2.4.4 接続制限」で説明した sshd と同様に /etc/hosts.allow に **vsftpd:** で始まる行を追加し, 接続を許可するマシンを列挙します (図3.2).

次に xinetd の設定ですが, vsftpd 用のファイルを http://www.nsl.tuis.ac.jp/ に用意してありますので, このファイルを xinetd の設定ディレクトリにダウンロードし, xinetd を再起動すれば良いでしょう (図3.3) . FTP サーバを無効にするには /etc/xinetd.d/vsftpd の **disable** を **yes** にします.

```
sshd: 58.158.97.64/255.255.255.224 202.26.159.140
vsdtpd: 202.26.159.140 202.26.159.144/255.255.255.240
```

図 3.2 vftpd を追加した /etc/hosts.allow の例 (サブネットマスク部が /255.255.255.255 の場合は省略可)

```
# cd /etc/xinetd.d
# wget http://www.nsl.tuis.ac.jp/etc/setting/xinetd/vsftpd
# /etc/init.d/xinetd  restart
```

図 3.3 /etc/xinetd.d/vsftpd 用ファイルのダウンロードと xinetd の再起動

3.3 OpenSSL

OpenSSLは暗号化ライブラリです.yumで OpenSSLの開発環境をインストールしても良いのですが,OpenSSLではしばしばセキュリティホールが発見されるので,通常の場合でもディストリビューション用にパッケージ化されるのを待つのでは無く，常に最新版をインストールすることを勧めします(図3.4).

なお，OpenSSL1.0 がリリースされていますが，1.0は 0.9から大きくAPIの仕様が変更されたため，1.0を使用すると従来のソフトウェアが動作しなくなる恐れがあるので，ここでは 0.9.x の最新版を使用します.

サイトURL: http://www.openssl.org/
ダウンロードURL: http://www.openssl.org/source/

```
# tar zxfv openssl-0.9.8o.tar.gz
# cd openssl-0.9.8o
# ./config shared      (configure ではないので注意)
# make
# make test            (注： マシンによっては make test は非常に時間がかかる場合がある)
# make install
```

図3.4 OpenSSLのインストール手順 (openssl-0.9.8o の場合)

make installを実行すると OpenSSLは/usr/local/ssl にインストールされます.OpenSSLの(ダイナミックリンクまたは共有)ライブラリも /usr/local/ssl/lib にインストールされます.OpenSSLの共有ライブラリが /usr/local/ssl/lib にあることをシステムに知らせるために,/etc/ld.so.confを編集して，**/usr/local/ssl/lib** の一行を追加します．ついでに後で必要になりますので，**/usr/local/lib** も追加しておきます.

この状態で **ldconfig** コマンドを実行すれば，共有ライブラリのデータベースが作成され，システムが共有ライブラリのあるディレクトリを知ることが可能となります(図3.5, 3.6)．/etc/ld.so.conf を編集しないで 環境変数の **LD_LIBRARY_PATH** で共有ライブラリのパスを指定することもできます.

rootシェルでファイルパーミッションのマスクを適切に設定している場合は,/usr/local/ssl に対して other のパーミッションが設定されない場合があります．その場合は図3.7のようにして,otherに対する読み取り許可等のパーミッションを設定します.

```
# vi /etc/ld.so.conf
# ldconfig
```

図3.5 共有ライブラリのデータベースの更新

```
include ld.so.conf.d/*.conf
/usr/local/lib
/usr/local/ssl/lib
```

図3.6 /etc/ld.so.conf の設定例 (太字が追加部分)

```
# cd /usr/local
# chmod -R o+r ssl
# find ssl -type d | xargs chmod o+x
```

図3.7 /usr/local/ssl のパーミッションの変更

3.4 Glib2

Glib2はグラフィックツールGTK+用のライブラリであり,libgdiplusをコンパイルするために必要となります. 図3.8に従ってコンパイルとインストールを行います.

インストールコマンド終了後, /usr/local/lib にライブラリがインストールされます.

サイトURL: http://www.icewalkers.com/Linux/Software/515980/GLib2.html

```
# tar zxfv glib-2.24.1.tar.gz
# cd glib-2.24.1
# ./configure
# make
# make install
# ldconfig
```

図3.8 Glib2のインストール手順 (glib-2.24.1 の場合)

3.5 libgdiplus

libgdiplus は GDI(Graphic Device Interface: MS Windowsのグラフィック用API)の拡張ライブラリであり,Monoで必要となります. なお, 使用する ligdiplusのバージョン番号はMonoのバージョン番号と合わせた方が良いでしょう (リリース番号は合わせる必要はありません).

図3.9に従ってコンパイル, インストールを行います. インストールコマンド終了後, /usr/local/lib にライブラリがインストールされます.

サイトURL: http://www.mono-project.com/
ダウンロードURL: http://go-mono.com/sources-stable/

```
# tar jxfv libgdiplus-2.6.tar.bz2
# cd libgdiplus-2.6
# ./configure
# make
# make install
# ldconfig
```

図3.9 Libgdiplus のインストール手順 (libgdiplus-2.6 の場合)

3.6 Mono

monoは Linux/Unix上で動作する MS .NETの実行環境です. MS Windows用のC#などの中間コードをUnix/Linux上で動作させる事が可能であり, OpenSimをLinux上で実行されることができます.

なお, システム内に古いmonoがインストールされている場合, 新しいmonoをコンパイルする際に古いmonoのライブラリを使用してエラーを起こすことがありますので, 古いmonoのライブラリは削除しておいた方が良いでしょう. 古いmonoもソースからコンパイルした場合は, /usr/lofcal/lib/mono を削除あるいはリネームします (図3.10).

サイトURL: http://www.mono-project.com/
ダウンロードURL: http://go-mono.com/sources-stable/

```
# mv /usr/local/lib/mono /usr/local/lib/mono-     (古い monoをリネーム)
# tar jxfv mono-2.6.4.tar.bz2
# cd mono-2.6.4
# ./configure --with-libgdiplus=/usr/local/lib/libgdiplus.la
# make
# make install
```

図3.10 Monoのインストール手順 (mono-2.6.4 の場合)

monoのコマンドは /usr/local/binに,ライブラリ関連のファイルは /usr/local/libまたは/usr/local/lib/mono などにインストールされます.

また, monoのソフトウェア情報は /usr/lib/pkgconfig に保存されますが, その内容の整合性が崩れている場合があります. mono-2.6用の pkgconfig用ファイルを別途用意してありますので, このファイルを /usr/lib/pkgconfig に展開してください (図3.11).

```
# wget http://www.nsl.tuis.ac.jp/DownLoad/SoftWare/Linux/mono.pc-2.6.tgz
# zcat mono.pc-2.6.tgz | (cd /usr/lib/pkgconfig/ && tar xfv -)
```

図 3.11 Mono の pkgconfig 用ファイルの修正

3.7 Nant

nant は mono (MS .NET) 用の構築ツールで, C言語のmakeやJavaのantに相当します. 図3.12に従ってコンパイルとインストールを行います. nantコマンドは /usr/local/bin に保存されます.

サイトURL: http://nant.sourceforge.net/
ダウンロードURL: http://sourceforge.net/projects/nant/files/nant/

```
# tar zfxv nant-0.90-src.tar.gz
# cd nant-0.90
# make
# make install
```

図 3.12 Nant のインストール手順 (nant-0.90 の場合)

3.8 MySQL サーバ

MySQL Community Server(以下MySQLサーバ)はフリーのデータベースサーバです. OpenSimではSQLite3と呼ばれるファイルベースのデータベースも使用できますが,OpenSimをグリッドモードで動作させる場合には MySQLサーバが必須となります(スタンドアロンモードでもMySQLは使用できます). また,10章で紹介する WEBインターフェイスをインストールする場合にも MySQLサーバは必須です.

MySQLサーバは 5.1系と5.5系がダウンロードできますが('10 7/20現在), 5.5系はこれまで非推奨とされていた機能がばっさりと削除されたようで,既存のアプリケーションでは作動しないものもあります. 従って, ここでは5.1系を使用します.

なお, MySQLサーバはOpenSimを作動させるマシンとは別のマシンで稼動させることも可能です.

サイトURL: http://dev.mysql.com/
ダウンロードURL: http://dev.mysql.com/downloads/mysql/
セレクトボタンで Source Code -> Generic Linux Compressed TAR Archive を選択.

3.8.1 MySQL のインストール

図3.13にしたがってコンパイル,インストールを行います.関連ファイルは /usr/local/mysql にインストールされます.

```
# tar zxfv mysql-5.1.48.tar.gz
# cd mysql-5.1.48
# ./configure --prefix=/usr/local/mysql --without-readline
# make
# make install
```

図 3.13 MySQL のインストール手順 (mysql-5.1.48 の場合)

3.8.2 起動スクリプトの準備

まず，MySQL の起動スクリプトを準備し，適当な Run Level で自動起動するようにシンボリックリンクを張ります．「2.4.1 Run Level の変更」の例での Run Level は **3** でした．

図 3.14 の例では，MySQL の起動順序を **90** に設定しています（ネットワークサービスの開始より後であれば 90 以外でも可ですが，大体 90 前後が適当かと思われます）．

次に，/etc/init.d/mysql を編集し，データベースの格納ディレクトリ（ここでは **/var/mysql**）を指定する変数 **datadir** を変更します（**datadir=/var/mysql**）（図 3.15）．

```
# cp support-files/mysql.server /etc/init.d/mysql
# chmod a+rx /etc/init.d/mysql
# ln -s ../init.d/mysql /etc/rc3.d/S90mysql
# ls -l /etc/rc3.d/S90mysql                    （確認）
```

図3.14 MySQLの起動用スクリプトの準備

```
# vi /etc/init.d/mysql
          datadir=/var/mysql     と変更する
```

図 3.15 MySQL のデータベースディレクトリを準備

3.8.3 mysql ユーザの作成

/var/mysql をホームディレクトリとした， mysql 用のユーザとグループを作成します．手動で作成しても良いですし，**adduser** を使用してもかまいません．図3.16の例で使用するユーザ番号の103（グループ番号も同じ）はあくまでも例ですので（CentOS では未使用の番号），実際は /etc/passwd, /etc/group をチェックして，ユーザ番号およびグループ番号が他のユーザやプロセスと被らないようにしなければなりません．

adduserで mysqlユーザを作成した場合は，ホームディレクトリ上に不要なファイルが作成される場合があるので，これらは消しておきます（図3.17）．なおこのコマンドは，入力ミスした場合悲惨な結果を招きますので注意深く入力してください．

次に，作成した mysql ユーザが /usr/local/mysql ディレクトリ内を読めるようにしなければなりません．また一般ユーザでも mysqlコマンドを起動できれば便利ですので，ここでは /usr/local/ssl と同様に other に対して読み取り許可を与えることにします（図3.18）．

```
# adduser mysql -u 103 -d /var/mysql -s /sbin/nologin
```

図 3.16 mysql ユーザの追加

コマンド入力注意!!!

```
# ¥rm -rf /var/mysql/.*          （* の前に . が付きます）
```

図 3.17 mysql ユーザのホームディレクトリ（データベースディレクトリ）上の余分なファイルを削除

```
# cd /usr/local
# chmod -R o+r mysql
# find mysql -type d |xargs chmod o+x
```

図 3.18 /usr/local/myswl のパーミッションを変更

3.8.4 データベースの初期化と設定

MySQL のデータベースを図 3.19 のコマンドで初期化します.

MySQL データベースの設定は /etc/**my.cnf** ファイルでも行うことができます. 必須の設定ではありませんが, OpenSim のデータベースで日本語を使用したい場合には, 図 3.20 のようにサーバ, クライアント共に **character set** として **UTF-8** を指定しておきます. また同図の [mysqld]セクションのコネクションに関する記述は OpenSim 0.7 での推奨設定です (公式 Wiki より).

```
# /usr/local/mysql/bin/mysql_install_db --user=mysql --ldata=/var/mysql
```

図 3.19 MySQL データベースの初期化

```
[mysql]
default-character-set = utf8

[mysqld]
default-character-set = utf8
open-files-limit = 20000
interactive_timeout = 999999
wait_timeout = 999999
max_connections = 2000
```

図3.20 日本語(UTF-8)を扱うための設定とタイムアウトなどの推奨設定 (/etc/my.cnf)

3.8.5 管理ユーザとパスワードの設定

まず, MySQL サーバを手動起動し, ps コマンドでプロセスが正常に起動したかを確認します (図 3.21). 次に MySQL サーバの管理ユーザ名とそのパスワードを **mysqladmin** コマンドで指定します. 図 3.22 の例では **root** が管理ユーザ名, **SQLPass** がパスワードとなっています.

サーバへのアクセス制限がある場合, 図 3.23 の **mysqladmin** コマンドではエラーが発生しますので, -h オプションでマシン名を指定して再度設定を行います (図3.23).

```
# /etc/init.d/mysql start
# ps ax
```

図 3.21 MySQL サーバの起動と確認

```
# /usr/local/mysql/bin/mysqladmin -u root password SQLPass
```

図 3.22 MySQL サーバの管理ユーザとパスワードの設定

```
# /usr/local/mysql/bin/mysqladmin -u root -h localhost password SQLPass
```

図 3.23 サーバ名を指定した場合の MySQL サーバの管理ユーザとパスワードの設定

3.8.6 セキュリティ設定

セキュリティレベル維持のため, MySQL に予め登録されている匿名ユーザ, パスワード無しユーザの削除を行います (練習などでセキュリティを気にしないのであればやらなくてもかまいません).

mysql コマンドでデータベースに接続し, 直接ユーザの削除コマンドを入力します (図 3.24).

3.8.7 OpenSim 用ユーザの作成

MySQL サーバ上に OpenSim 用のデータベースを作成し, さらに接続用ユーザを作成してアクセス権

限を与えます（図 3.25）．これ以降の例では，OpenSim 用データベースの名前を **opensim_db**，管理ユーザ名を **opensim_user**，パスワードを **opensim_pass** として説明を行います．

```
$ /usr/local/mysql/bin/mysql -u root -p
Enter password: SQLPass    (MySQL の root のパスワード)

mysql> use mysql;
mysql> delete from user where user='';
mysql> delete from user where password='';
mysql> exit
```

図 3.24 匿名ユーザとパスワード無しのユーザの削除

```
$ /usr/local/mysql/bin/mysql -u root -p
Enter password: SQLPass    (MySQL の root のパスワード)

mysql> create database opensim_db default character set utf8;
mysql> grant all on opensim_db.* to opensim_user identified by 'opensim_pass';
mysql> flush privileges;
mysql> exit
```

図 3.25 OpenSim 用データベースの作成と権限の設定

mysqlコマンド入力中に間違えてEnterを押しちゃって，
「->」っと表示してしまったら「;」キーを押してEnterを押すんよ．

3.9 SQLite3

OpenSimをスタンドアロンモードで動かす場合には,データベースとして SQLite3が使用できます．MySQLに比べて事前設定がほとんど無いのが魅力的ですが,OpenSimをグリッドモードで動かす場合には使用できません．

SQLite3 はほとんどのLinuxディストリビューションでシステムインストール時にインストール済みですが，OpenSim の 0.7 で使用する場合にはバージョンが 3.5.0 以上である必要があります．システムにインストール済みの SQLite3のバージョンを調べるには 図3.26のようにします（パッケージ管理システムにRPMを使用している場合）．

バージョンが 3.5.0以上であっても，ディストリビューションによっては libsqlite3.so共有ライブラリに sqlite3_column_origin_name()関数がリンクされていない場合があります．sqlite3_column_origin_name()関数がリンクされていなるかどうかを確認するには,図3.27にようにします．関数名が表示されれば，ライブラリにsqlite3_column_origin_name()関数が含まれていることを示します．

```
# rpm -q sqlite
  または
# rpm -qa | grep sqlite
```

図 3.26 SQLite3 のバージョン確認 （RPM の場合）

もし，バージョンが 3.5.0より古い場合や sqlite3_column_origin_name()関数がリンクされていない場合は(CentOS5.5ではバージョンが古いようです)，最新のソースコードをダウンロードし，図3.28のようにしてコンパイル，インストールを行います（makeコマンドのオプションに注意）.

サイトURL: http://www.sqlite.org/

```
# objdump -T /usr/lib/libsqlite3.so | grep sqlite3_column_origin_name
```

図3.27 sqlite3_column_origin_name 関数の存在確認

```
# tar zfxv  sqlite-3.6.23.1.tar.gz
# cd sqlite-3.6.23.1
# ./configure
# make OPTS=-DSQLITE_ENABLE_COLUMN_METADATA=1
# make install
```

図3.28 SQLite3のインストール手順 （sqlite-3.6.23.1 の場合）

3.10 Git （オプション）

Gitは分散型のプロジェクト（ソースコード）管理ツールです．GitはOpenSimの開発バージョンをダウンロードする場合に使用します．従って，OpenSimのリリースバージョンのみを取り扱う場合には必要はありません．

因みにCentOSでは Gitは標準パッケージとしては用意されていないので，必ずソースコードからコンパイルしなければなりません（図3.29）.

サイトURL: http://kernel.org/pub/software/scm/git/

```
# tar zfxv git-1.7.1.tar.gz
# cd git-1.7.1
# ./configure
# make
# make install
```

図3.29 Gitのインストール手順 （git-1.7.1 の場合）

3.11 Openjpeg-dotnet （オプション）

Linuxのディストリビューションによっては，OpenSimの起動時に稀に libopenjpegのエラーメッセージが出力される場合があります.この場合はOpenSimを起動できたとしても,マップ上でリージョンのアイコンが表示されないなどの不都合が生じます.

libopenjpegのエラーが出た場合は，OpenMetaverseから libopenmetaverseのソースをsvnを使用してダウンロードし，そこ中の openjpeg-dotnetをコンパイルし直します．図3.30に従ってコンパイルを行い，生成された共有ライブラリ(*.so) を OpenSimの binディレクトリにコピーします.

```
# svn co  http://libopenmetaverse.googlecode.com/svn/libopenmetaverse/trunk libopenmetaverse
# cd libopenmetaverse/openjpeg-dotnet
# make
# cp *.so [OpenSimのインストールディレクトリ]/bin
```

図3.30 Openjpeg-dotnetのインストール手順

3.12 ODE（オプション）

Openjpeg-dotnetと同様に，Linuxのディストリビューションによっては，OpenSimの起動時に稀にODE（Open Dynamics Engine: OpenSimのデフォルトの物理エンジン）のエラーが出て起動できない場合があります．この場合は，ODEのライブラリを新しく作成します．下記のダウンロードURLから最新版のソースをダウンロードし，図3.31に従ってコンパイルを行います．

サイトURL: http://ode.org/
ダウンロードURL: http://sourceforge.net/projects/opende/files/

```
# tar zxfv ode-0.11.1.tar.gz
# cd ode-0.11.1
# ./configure --with-trimesh=gimpact --enable-shared
# make
# make install
# vi /etc/ld.so.conf      (確認)
# ldconfig
```

図3.31 ODEのインストール手順（ode-0.11.1の場合）

図3.31 の vi /etc/ld.so.conf では中に /usr/local/lib の一行があることを確認します．もしなければ「3.3節」を参考に追加します．

4. OpenSim のコンパイルとインストール

この章では，OpenSimのコンパイル方法とインストール方法を解説します．また，OpenSimをより使いやすく，高機能にするためのパッチの適用についても解説をおこないます．

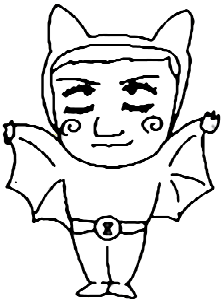
ちゃんと，ここまで出来たかな??

もちろん，でもMySQLがちょっと
大変だったよ～！

ここからは，いよいよOpenSimのインストー
ルだよ！ コンパイル方法が今までちょっと
違う方法だから気をつけてね～

MonoとNantはMS .NETでつくられたものを
コンパイルするときに必要なんだ～
あと，このパッチも使ってね！

やり方， また違うんだ（泣
ところでこのパッチってなに～？



なんだ．そんなことも知らんのか!! パッチはプログラムのソースコード内容を変更するものだ．NSLではプログラムの修正や機能拡張などもやっているんだ． これはpatchコマンドで簡単に適用できる．もっとも私も超初心者だったころ，patchコマンドを知らないで，パッチファイルの内容を見て，手動でプログラムを書き変えたこともあったよ．ふっ（遠い目 ……）

ほぇ～～

4.1 リリースバージョン

4.1.1 リリースバージョンのソースコードのダウンロード

‘11 12/21 現在の最新リリースバージョンは **0.7.2** です．ソースコードのダウンロード URL は以下の通りです．

http://opensimulator.org/dist/opensim-0.7.2-source.tar.gz

または　http://opensimulator.org/dist/opensim-0.7.2-source.zip

```
# cd /usr/local
# tar zfxv [DL]/opensim-0.7.2-source.tar.gz
# mv opensim-0.7.2-source opensim
```

図 4.1 OpenSim のソースコードの展開 （opensim-0.7.2-source.tar.gz の場合）
（ただし [DL] は opensim-0.7.2-source.tar.gz をダウンロードしたディレクトリを表す）

なお，図 4.1 のコマンド例で opensim-0.7.2-source を opensim に変更している理由は以後の説明上の都合によるもので，必ずしもディレクトリ名を変更する必要はありません．また展開するディレクトリも /usr/local 以外でもかまいません．ただし，以後の説明では OpenSim は /usr/local/opensim に展開されているものとします．

4.1.2 リリースバージョン用非公式 NSL パッチの適用

0.7.2 のリリースバージョン用のパッチ（非公式 NSL パッチ）の URL は以下の通りです．

http://www.nsl.tuis.ac.jp/xoops/modules/d3downloads/index.php?page=singlefile&cid=8&lid=29

このパッチでは以下の修正が施されています．

1) **多様な OS でグリッドを構成する場合，OS の違いだけで「OpenSim のバージョンが違う」というダイアログが表示するのを防ぐ**
2) **OpenSim.ConsoleClient のバグ修正（8.2 節参照）**
3) **Mega Region での llGround() 関数の不具合を修正**
4) **Offline メッセージモジュール用の修正**

```
# cd /usr/local/opensim
# patch -p1 < [DL]/opensim_nsl_0.7.2.patch
```

図 4.2 非公式 NSL パッチの適用 （opensim.nsl.patch-0.7.2 の場合）
（ただし [DL] は opensim.nsl.patch-0.7.2.patch をダウンロードしたディレクトリを表す）

4.1.3 リリースバージョン用マネーサーバパッチの適用

0.7.2 でDTL/NSL マネーサーバを使用する場合にはパッチは必要ありません．

0.7.2 用の DTL/NSL マネーサーバのダウンロード用の URL は以下の通りです．なお，このバージョンにはコンパイル済みバイナリーも同梱されています．

http://www.nsl.tuis.ac.jp/xoops/modules/d3downloads/index.php?page=singlefile&cid=8&lid=26

（図 4.3 は更新時に削除されました）

4.2 開発バージョン

4.2.1 開発バージョンのソースコードのダウンロード

開発バージョンのダウンロードでは，Gitを使用する方法と Subversionを使用する方法があります.初期のころは Subversionがメインリポジトリでしたが,現在では Gitがメインリポジトリとなっています．一方，Subversionのリポジトリは一定時間ごとに Gitのリポジトリと同期が取られるように設定されていますが，時々同期しなくなる場合もありますので，Gitを使用する方をお勧めします.

```
# cd /usr/local
# git clone git://opensimulator.org/git/opensim opensim
```

図 4.4 Git を使用した開発バージョンのダウンロード

```
# cd /usr/local
# svn co http://opensimulator.org/svn/opensim-track/trunk opensim
```

図 4.5 Subversion を使用した開発バージョンのダウンロード

4.2.2 開発バージョン用非公式 NSL パッチとマネーサーバパッチ

NSLでは,OpenSimの開発バージョンを追いかけながらパッチ(NSLパッチとマネーサーバ用パッチ)の作成を行っています．この開発バージョン用パッチは Subversionで管理されており，図4.6，4.7の方法でダウンロードできます．ただし，OpenSimの最新開発バージョンに対してパッチの開発が追いつかない場合もありますので,その点を注意してください.パッチの開発がOpenSimの最新開発バージョンに追いついていない場合，パッチの適用は失敗に終わります.

OpenSimの開発バージョンをダウンロードしてパッチを適用した場合,svnおよびgitコマンドではソースをアップデートできなくなる場合があります.特に Gitの場合は少しでも変更があった場合，変更箇所をリポジトリに登録しないと，OpenSimを最新バージョンに更新できません（Subversionでは最悪の状態でもパッチを適応したファイルを削除すれば良い）．この場合の対処方法については「4.4 Gitでのアップデートとパッチの適用」を参照してください.

```
# cd /usr/local/opensim
# svn co http://www.nsl.tuis.ac.jp/svn/opensim/opensim.nsl.patch/trunk
  opensim.nsl.patch
# patch -p1 < opensim.nsl.patch/opensim_nsl_latest.patch
```

図 4.6 Subversion を使用した開発バージョン用非公式 NSL パッチのダウンロードと適用

```
# cd /usr/local/opensim
# svn co http://www.nsl.tuis.ac.jp/svn/opensim/opensim.currency/trunk
  opensim.currency
# patch -p1 < opensim.currency/opensim_currency_latest.patch
```

図 4.7 Subversionを使用した， 開発バージョン用 Money サーバパッチのダウンロードと適用

4.3 OpenSim のコンパイル

OpenSimのコンパイルでは nantを使用します（図4.8）．一度コンパイルして，再度コンパイルし直すには，nant を実行する前に **nant clean** を実行します．

マネーサーバのパッチを適用している場合は，図4.9のようにしてマネーサーバおよびクライアント用モジュールのコンパイル，インストールを行います．

```
# cd /usr/local/opensim
# bash runprebuild.sh
# nant
```

図 4.8 OpenSim のコンパイル

```
# cd /usr/local/opensim
# cd opensim.currency-0.7    または    # cd opensim.currency
# ./build.sh
```

図 4.9 Money サーバとクライアントモジュールの構築

4.4 Git でのアップデートとパッチの適用

開発バージョンをGitでダウンロードしてパッチを適用した場合，Gitでのアップデート（git pull コマンド）ができなくなります．これを防ぐには Gitでブランチを作成し，パッチは作成したブランチに適用するようにします．一方でアップデートについては masterブランチに戻って行えば，問題なく OpenSimを最新バージョンに更新することができます（図4.10）．

```
# git branch    ブランチ名          （ブランチの作成）
# git checkout ブランチ名           （ブランチの移動）
# patch -p1 < ......                （パッチの適用）
# bash runprebuild.sh && nant clean && nant   （コンパイル）
......
......
# git commit -a -m "dummy"          （パッチによる変更をコミット）
# git checkout master               （master ブランチに戻る）
# git pull                          （更新）
# git branch  新しいブランチ名      （新しいブランチの作成）
以降繰り返し．
```

図 4.10 Git を使用した場合のアップデートとパッチ適用

5. OpenSimの設定と起動 （スタンドアロンモード）

スタンドアロンモードはOpenSimの最も単純な動作モードです. スタンドアロンモードであっても の1台のマシンで複数のリージョン（マルチリージョン）を起動することも可能で，リージョンサーバ（リージョンを管理するサーバ）を複数台使用しないのであればスタンドアロンモードでも十分です.

スタンドアロンモードってなんだろう？？

あれ，質問なんて珍しいね!

いつも,グリッドモードってやつでやってるから設定
したことが無いんだ（泣

どうやら，スタンドモードは，OpenSim でのサーバを
全部ひとつにまとめちゃったやつみたいだよ～

そうだね. ただし今は,グリッドモードでも本来あった複数
のサーバがひとつのサーバ(ROBUSTサーバ)にまとめら
れたから，管理者の負担はだいぶ減ったんだ.
グリッドモードについては次章で解説するよ.

お手軽に構築できるサーバってわけだね！

5.1 設定ファイルの準備

スタンドアロンモードで用意するファイルは bin/OpenSim.ini，bin/config-include/StandaloneCommon.ini および bin/config-include/FlotsamCache.ini の３つです．これらのファイルは，それぞれの example ファイルからコピーして作成します（図 5.1）．なお，FlotsamCache.ini は StandaloneCommon.ini から読み込まれるアセット（オブジェクト）のキャッシュ設定用ファイルですが，内容を変更する必要はありません．

0.7.2 のスタンドアロンモードで起動する場合は，OpenSim.ini を一箇所だけ書き換える必要があります．[Architecture]セクションの ， Include-Architecture = "config-include/Standalone.ini" を有効にします（図 5.2）．

リージョン（SIM）の設定ファイル bin/Regions/*.ini も必要ですが，これはサーバプロセス起動時に対話的に作成されます（または手動で作成することも，サーバコマンドで作成することも可能です）．

```
# cd /usr/local/opensim/bin
# cp OpenSim.ini.example OpenSim.ini
# vi OpenSim.ini
# cd config-include
# cp StandaloneCommon.ini.example StandaloneCommon.ini
# cp FlotsamCache.ini.example FlotSamCache.ini
```

図 5.1 StandAlone モードで必要なファイルの準備と書き換え

```
[Architecture]
    ........................
    ........................
    ;;
     Include-Architecture = "config-include/Standalone.ini"
    ; Include-Architecture = "config-include/StandaloneHypergrid.ini"
    ; Include-Architecture = "config-include/Grid.ini"
    ; Include-Architecture = "config-include/GridHypergrid.ini"
    ; Include-Architecture = "config-include/SimianGrid.ini"
    ; Include-Architecture = "config-include/HyperSimianGrid.ini"
```

図 5.2 OpenSim.ini の書き換え

5.1.1 OpenSim.ini（SQLite3 を使用する場合）

スタンドアロンモードでデータベースとして SQLite3 を使用する場合は図 5.2 の変更以外，設定ファイルを変更する必要はありません．

5.1.2 OpenSim.ini（MySQL サーバを使用する場合）

図 5.2 のように bin/OpenSim.ini の Include-Architecture = "config-include/Standalone.ini" を有効にします．

5.1.3 StandaloneCommon.ini（MySQLサーバを使用する場合）

StandaloneCommon.ini の変更点の例を図5.3に示します．StandaloneCommon.iniでは[DatabaseService]セクションのデータベースへの接続部分を変更します．つまりSQLiteを無効にしてMySQLを有効にし，**ConnectionString**に接続用パラメータを記述します．MySQLサーバがリモートマシンの場合は，storage_connection_stringの **Data Source** にlocalhostの代わりに MySQLサーバのIPアドレスかFQDN（正式なホスト名）を指定します．

```
[DatabaseService]
    ; SQLite
    ;Include-Storage = "config-include/storage/SQLiteStandalone.ini";

    ; MySql
    ; Uncomment these lines if you want to use mysql storage
    ; Change the connection string to your db details
     StorageProvider = "OpenSim.Data.MySQL.dll"
      ConnectionString = "Data Source=localhost;Database=opensim_db;User
ID=opensim_user;Password=opensim_pass;Old Guids=true;"
```

図5.3 StandaloneCommon.iniの[DatabaseService]セクション（太字の部分が変更箇所）

5.2 起動と停止

MySQLサーバを使用する場合は MySQLサーバの設定(3.8節)，StandaloneCommon.iniの設定(5.1.1項），OpenSim.iniの設定（5.1.2項）を行い，その後 MySQLサーバを起動させてから（図5.4)，OpenSim.exe を monoを使用して起動します（図5.5)．MySQLサーバは必ずOpenSim.exeより先に起動しておかなければなりません．

一方，SQLite3を使用する場合の設定は何もありません．そのまま図5.5のようにして リージョンサーバ（OpenSim.exe）を起動します．もし起動時にSQLiteのエラーが発生する場合は，SQLiteのバージョン等に問題のある可能性があります．その場合は，「3.9節」の内容を確認してください．

```
# /etc/init.d/mysql start    （MySQLサーバの手動起動）
# /etc/init.d/mysql stop     （MySQLサーバの手動停止）
```

図5.4 MySQLサーバの手動起動と手動停止

```
# cd /usr/local/opensim/bin
# mono OpenSim.exe
```

図5.5 OpenSimの起動

5.2.1 リージョン設定ファイル

OpenSim.exeの起動時に bin/Regions/*.ini が存在しなければリージョンの設定のために幾つかの質問が表示され（図5.6)，その結果として **bin/Regions/Regions.ini** が生成されます（図5.7)．

New region name にはリージョンの名前を設定します．また，**Region UUID** は通常はシステムが生成したものをそのまま使用します（そのままエンターキーを押す）．**UUID**は128bitの識別コードで，OpenSimではほとんど全てのリソースにこのIDが割り振られ，一意的に管理されます．そもそも，UUIDは世の中の全てのリソースを一意的に管理するために考え出されたIDです．

Region Locationはリージョンの座標です．一台のマシンで複数のリージョンを稼動させる（マルチリージョンにする）場合は，それぞれのリージョンに違う座標を割り振る必要があります．一番

最初のリージョン，またはリージョンが一つしかない場合はデフォルトのままでよいでしょう．

Internal IP address にはリージョンサーバの IP アドレスを指定します．0.0.0.0（デフォルト）を設定した場合には，リージョンサーバ内部で自動的にサーバのIPアドレスが設定されます．ただし，Internal IP addressに127.0.0.1 を指定した場合は，リージョンサーバは正常に作動しなくなりますので注意する必要があります（ビューアをサーバと同じマシンで起動する場合に限って，127.0.0.1を使用することが可能です）．

Internal port はリージョンサーバがUDPのデータ通信を行うためのポート番号です（デフォルトは9000）．一方，HTTP(TCP)通信を行うポート番号は OpenSim.ini の[Network]セクションの **http_listener_port** で指定します（デフォルトはUDPと同じく9000）．マルチリージョンの場合，同一マシン上のリージョン間でこのUDPのポートの番号が被らないようにしなければなりません．一方，HTTPポートはリージョン間で共有されます．

Allow alternate ports は実験的な機能ですので，通常は False のままにします（そのままエンターキーを押す）．

External host name にはサーバマシンの IP アドレスか **FQDN**（正式なホスト名）を指定します．FQDNを指定する場合は，DNSなどで解決できる名前でなければなりません．デフォルト(SYSTEMIP)のままエンターキーを押した場合は，自動的にサーバの IP アドレスが設定されます．

ここで，**Internal IP address** と **External host name** の違いですが，ソースコードを見る限りでは，**Internal IP address** はUDP通信を行う場合のリージョンサーバのアドレス通知に使用され，**External host name** は主にHTTPでのCAPSのリクエスト時のアドレスに用いられる場合が多いようです．従って，Internal IP address と External host nameに同じIPアドレスとを入力しても何も問題はありません．

一般家庭のADSL回線でBBルータ（ブロードバンルータ）を使用している環境においては，**Internal IP address** と **External host name** にBBルータのグローバルアドレスを指定する場合もあります．ただし，実はこの設定は非常にややこしい問題をはらんでいます．この種の設定の問題点に関しては「11章 サーバのNAT(NAPT)越えの問題」をご覧ください．

```
New region name []: TEST_SIM
Region UUID [7f35da2d-c02e-4567-b92f-56bcd8fca6e7]:
Region Location [1000,1000]:
Internal IP address [0.0.0.0]:
Internal port [9000]:
Allow alternate ports [False]:
External host name [SYSTEMIP]: 202.26.159.211
```

図 5.6 Regions.ini 作成のための対話モードの例（太字が入力した部分）

```
[TEST_SIM]
RegionUUID = 7f35da2d-c02e-4567-b92f-56bcd8fca6e7
Location = 1000,1000
InternalAddress = 0.0.0.0
InternalPort = 9000
AllowAlternatePorts = False
ExternalHostName = 202.26.159.211
```

図 5.7 生成された Regions.ini の例

```
$ uuidgen
b186d73a-b9a6-4de1-85b4-fa2224e1368d
```

図 5.8 uuidgen コマンド

これらの情報を 図5.6のように入力した場合は，図5.7のような **bin/Regions/Regions.ini** が生成されます．図5.7のファイルフォーマットに従っていれば，手動（エディタ）でリージョン設定ファイルを作成することも可能です．この場合，RegionUUID は **uuidgen** コマンドで生成したものを使用します（図5.8）．

また先に述べた様に，サーバのリソースが許す限り一台のリージョンサーバで複数個のリージョン(SIM)を稼動されることも可能です(**マルチリージョン**)．マルチリージョンにするには，複数のリージョン設定ファイルを bin/Regionsディレクトリに置くだけです．この場合，リージョン設定ファイルの名称は，拡張子が .ini であれば何でも結構です．ただし，当然のことながら，RegionUUID, Location および InternalPortはこれらのファイル間で重複してはいけません．

5.2.2 エステートの設定

エステートは土地の管理単位です．初回起動時には Region.iniの情報入力後に，そのリージョンのエステートの名前を設定のための情報入力を促されます．**New estate name [My Estate]:** と表示されますので，適当な名前を入力します（そのままエンターを押して，デフォルトの名前でも結構です）．

5.2.3 アバターの作成

エステートの設定に続いて，エステートの管理アバターを作成するための情報入力が促されます．管理アバターのファーストネーム（**Estate owner first name**），セカンドネーム（**Estate owner last name**），ログイン用パスワード（**Password**）を入力します．パスワードを入力しなかった場合，パスワードなしでログインできるアバターが作成されます．なお，パスワードは，入力中は画面に表示されませんが，エンターキーを入力すると画面に表示されます!?

なお，ユーザのメールアドレス（**Email**）の入力は任意です（図5.9）．

```
The current estate has no owner set.
Estate owner first name [Test]: TEST
Estate owner last name [User]: User
Password********
Email []:
```

図5.9 アバター作成画面

New region name	リージョンの名前.
Region UUID	リージョンのUUID．通常はデフォルトを使用する.
Region Location	リージョンの座標.
Internal IP address	UDP通信で使用される，サーバのIPアドレス．0.0.0.0 の場合は自動的にリージョンサーバのIPアドレスが設定される.
Internal port	UDP通信で使用される，サーバのポート番号．デフォルトは9000.
Allow alternate ports	実験的な機能．通常はFalse.
External host name	HTTPのCAPSで使用されるIPアドレス，またはFQDN．デフォルト（ SYSTEMIP）の場合はリージョンサーバのIPアドレスが設定される
Do you wish to join an existing estate?	作成中のリージョンのエステートを既存のエステートと結合させるか？ 通常はnoで良い.
New estate name	エステートの名前.
Estate owner first name	エステートのオーナ（ アバター）のファーストネーム．通常，エステートはリージョンと一致するので，エステートのオーナはリージョンのオーナーと同一と考えても良い.
Estate owner last name	エステートのオーナ（ アバター）のセカンドネーム
Password	エステートのオーナ（ アバター）のパスワード
Email	ユーザの E-mailアドレス

表5.10 初回起動時に入力を促される項目

5.2.4 コマンドプロンプトと停止コマンド

全ての情報が揃えば,その設定に従ってリージョンが起動します.初回起動に入力を求められる情報を表5.10にまとめました.

起動中のメッセージはコンソールに出力されると共に bin/OpenSim.logにも保存されます.カラーのエスケープシーケンスが有効なコンソールでサーバプロセスを起動した場合には,エラーメッセージは赤字で表示されます.エラーが表示された場合は,OpenSim.logでエラーの内容を確認します.エラーが表示された場合でもリージョンがそのまま起動する場合がありますが,なるべくエラー表示が少なくなるように設定を見直します.

正常に起動すれば，最後に **Region（リージョン名）#** のコマンドプロンプトを表示して，コマンド入力待ちとなります（図5.11）．ただし，マルチリージョンの場合は **Region (root) #** のコマンドプロンプトが表示されます.

データベースとしてSQLite3を使用した場合は，データベースファイルとして bin/*.dbが自動的に作成されます．またMySQLを使用した場合は，MySQLサーバにテーブルが自動生成されます.

サーバプロセスを停止させるには，リージョンサーバのコマンドプロンプトに対して **quit** または **shutdown** コマンドを入力します.

```
10:22:01 - [SCENE]: Loading land objects from storage
10:22:01 - [PRIM INVENTORY]: Starting scripts in scene
10:22:01 - [LLUDPSERVER]: Starting the LLUDP server in synchronous mode
10:22:01 - [UDPBASE]: SIO_UDP_CONNRESET flag not supported on this platform, ignoring
10:22:01 - [SUN]: Sun Settings Update: Fixed Sun? : False
10:22:01 - [SUN]: Sun Settings Update: Sun Hour   : 20.20167
10:22:01 - [SUN]: PosTime : 1278530521
10:22:01 - [WATCHDOG]: Started tracking thread "Outgoing Packets (TEST_SIM)" (ID 6)
10:22:01 - [PERMISSIONS]: Groups module not found, group permissions will not work
10:22:01 - [!]:STARTUP COMPLETE
Currently selected region is TEST_SIM
10:22:01 - [STARTUP]: Startup took 0m 5s
10:22:01 - [WATCHDOG]: Started tracking thread "Heartbeat for region TEST_SIM" (ID 7)
10:22:01 - [WATCHDOG]: Started tracking thread "Incoming Packets (TEST_SIM)" (ID 8)
10:22:02 - [REGION]: Enabling logins for TEST_SIM
10:22:02 - [GRID SERVICE]: region TEST_SIM has 0 neighours
10:22:02 - [INTERGRID]: Informing 0 neighbours that this region is up
Region (TEST_SIM) #
```

図5.11 OpenSimの起動画面の一部

5.2.5 エラーが出力されて起動できない場合

Linuxのディストリビューションによっては，稀ではありますが，ODEや libopenjpegのエラーが出て起動できない場合があります．その場合は，「3.11節」または「3.12節」を参照してください.

5.3 ビューアの設定とログイン

OpenSim のスタンドアロンモードでの接続 URL は http://**ExternalHostName:InternalPort**/ となります. ExternalHostName と InternalPort は リージョン設定ファイル bin/Regions/*.ini で指定したものですので. たとえば 図 5.7 の例では http://202.26.159.211:9000 となります.

Second Life のビューアを用いての OpenSim に接続する場合は, MS Windows ではショートカットのプロパティを表示させ,「リンク先」にコマンドの引数として

-loginuri http://ExternalHostName:InternalPort/

を追加します. 今回の例では -loginuri http://202.26.159.211:9000 です (図5.12).

図 5.12 MS Windows でのショートカットの設定

名　　称	特　　徴	URL
Hippo OpenSim Viewer	OpenSim公式ビューア. 255x255x255m までの巨大プリムを作成可能.	http://mjm-labs.com/viewer/
Snowglobe	テクスチャのダウンロードが早い. HTTP Get Texture機能をサポート.	http://snowglobeproject.org/
Emerald Viewer	あやしい機能満載. アバター名で相手の位置にテレポート可能.	http://emerald.modularsystems.sl/
Cool VL Viewer	高速描画. メモリ消費が少ない. 独自機能満載. 陰影機能付き.	http://sldev.free.fr/
Kirstens Viewer	高速描画の超軽量ビューア. 最大描画距離 1024m. さまざまな景観設定が可能. 陰影機能付き.	http://www.kirstensviewer.com/
Meerkat	プリムのバックアップとリストアが可能. 巨大プリムを作成可能.	http://code.google.com/p/meerkat-viewer/

表 5.13 主なサード・パーティによるビューアの例

Second Life のビューアのビューアの他に，サード・パーティ製のビューアも使用することができます．これらのビューアは，Second Life のビューアがオープンソース化されたことにより生まれたものですが，本家の Second Life ビューアを超える機能を持つものもあります．表 5.13 に主なサード・パーティ製のビューアを載せましたので，いろいろと試してみてください．

なお，ログイン時に使用するアバターのファーストネーム，ラストネームおよびパスワードは，初回起動時に入力したエステートの管理アバターのものを使用します．アバターを追加登録するには，リージョンサーバのコマンドプロンプトに対して **create user**コマンドを入力し，アバターのファーストネーム，セカンドネーム，パスワードを入力します（Email アドレスの入力は任意）．（6.1.2 項の図 6.5 を参照）

5.4 リージョン，エステート(estate)，パーセル(percel)

リージョンはリージョンサーバのスレッドの制御対象となる区画で，通常は 256x256 の大きさを持ちます．一方エステートは管理用の区画で，複数のリージョンにまたがることが可能です．OpenSim ではリージョンの管理者といった場合は，正確にはエステートの所有者（オーナ）アバターを指し場合がほとんどです．言い換えるとリージョンは物理的な区画で，エステートは論理的な区画です．エステートが一つのリージョンに限定される（通常は大体この形態）場合は，リージョンとエステートを同一視しても特に問題はありません．

エステートは，細かく分割して他のアバターに販売することが可能です．この細かく分割した土地をパーセルと呼びます．パーセルを購入したアバタはパーセルのオーナとなります．リージョン，エステートおよびパーセルの関係を図 5.14 に示します．

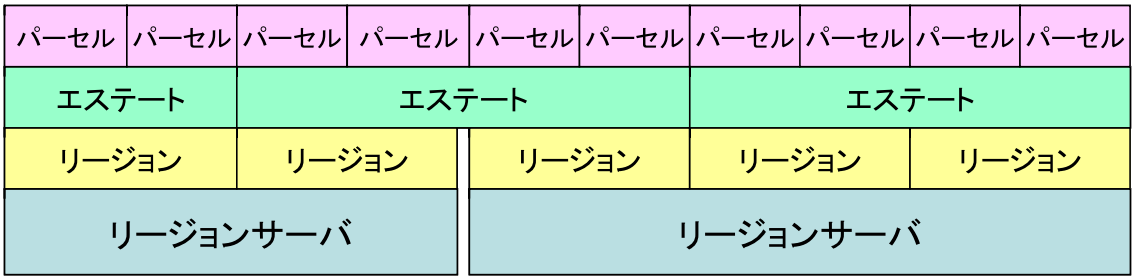


図 5.14 リージョン，エステート，パーセルの関係

6. OpenSimの設定と起動 （グリッドモード）

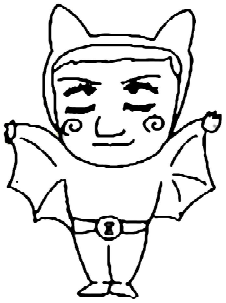
複数台のサーバマシンでリージョンサーバを稼動させる場合にはグリッドモードを用います(勿論サーバマシンが1台のみでもグリッドモードは可能です).グリッドモードではリージョンサーバの他に別のサーバプロセスを稼動させる必要があります.以前のバージョンの OpenSimでは,グリッドサーバ,ユーザサーバ,インベントリサーバ,アセットサーバおよびメッセージングサーバなどの複数のサーバプロセスを起動する必要がありましたが, v0.7 ではこれらのサーバプロセスは全て **ROBUST サーバ**（Redesigned OpenSim Basic Universal Server Technology サーバ）として統合されました(オプションでマネーサーバを起動させることも可能です. 6.3節参照).

グリッドモード構築しよーよ～！

あせらないで昔よりは簡単になったとはいえ,
やっぱり設定は難しいよ～

コツはやっぱり設定ファイルにあるよ

でもグリッドモードで動作し始めたらリージョンサーバ増やしてSIMも増やし放題だよ!



それじゃあ,いよいよ山場のグリッドモードだ.
気合を入れていくぞ～！！

はーい！

ROBUST サーバはデフォルトで TCP の 8003 のポートを使用してサービスの提供を行いますが，アバターの認証などのユーザサービスについては 8002番も使用します．サービスのプロトコルにはHTTPが使用されます．図 6.1 にグリッドモードでの OpenSim のサーバ構成の例を示します．なお，このような OpenSim のシステムの一つの単位を**グリッド**と呼びます．

通常，リージョンサーバやROBUSTサーバのプロセスはフォアグラウンドで作動します．従って，Run Level が 3(CUI)のマシン 1 台でリージョンサーバや ROBUST サーバなどの複数のサーバプロセスを稼動させる場合は **Ctrl+Alt+F1** ～ **Ctrl+Alt+F6** でコンソール画面を切り替えると良いでしょう．また各サーバプロセスは図 6.1 のようにそれぞれ違うマシンで稼動させることも可能です．

なお，各サーバプロセスのバックグラウンド起動については，「8 章 サーバプロセスのバックグラウンド起動」を参照してください．

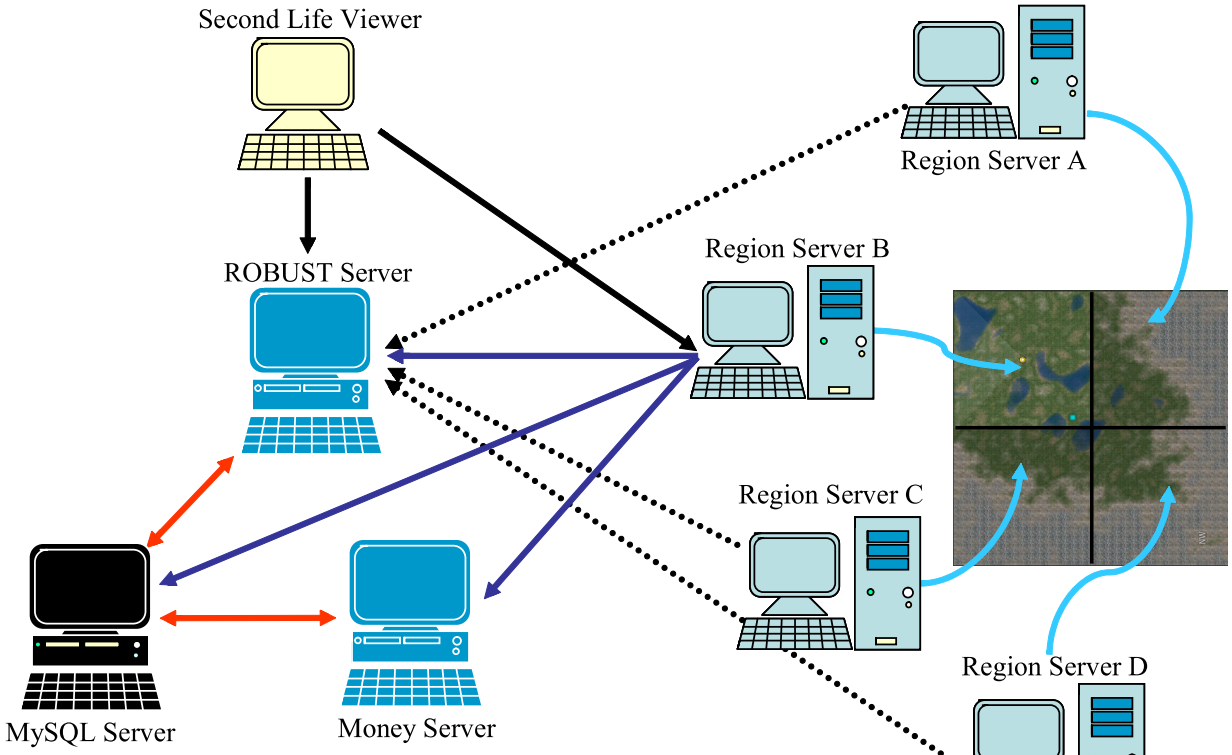


図6.1 グリッドモードでのサーバ構成

```
# cd /usr/local/opensim/bin
# cp Robust.ini.example Robust.ini
# vi Robust.ini
```

図6.2 Robust.iniのコピーと編集

わーい． サーバがたくさんあるー!

```
[DatabaseService]
  StorageProvider = "OpenSim.Data.MySQL.dll"
  ConnectionString = "Data Source=localhost;Database=opensim_db;User
 ID=opensim_user;Password=opensim_pass;"
```

図 6.3 Robust.ini の変更箇所

6.1 ROBUST サーバ

先にも述べましたが，OpenSim 0.7 ではグリッドモード用の標準サーバ群は ROBUST サーバとして完全統合されました．従って，スタンドアロンモードからの移行も以前よりはかなり楽に行うことができます（正確に言えば，ROBUST サーバはサーバシェルとも言えるもので，ROBUST サーバ内で各サーバが動作しています）．

6.1.1 Robust.ini

ROBUST サーバの設定ファイルは **bin/Robust.ini** ですので，Robust.ini.example からコピーして動作環境に合わせて内容を変更します（図 6.2）．最も重要な設定は MySQL への接続パラメータである **[DatabaseService]**セクションの **ConnectionString** です．「3.8.7 項」で設定した MySQL の管理ユーザの設定に従って **ConnectionString** の内容を変更します（図 6.3）．MySQL サーバがリモートマシンの場合は，ConnectionString の **Data Source** に localhost の代わりに MySQL サーバの IP アドレスか FQDN（正式なホスト名）を指定します．

また，**[GridInfoService]**セクションを設定すると，Imprudence Viewer のような GetGridInfo プロトコル（グリッドの情報を自動的に取得するプロトコル．Second Life にはない）に対応させることが可能になります．

[GridInfoService] の設定項目

login	**ログインURI**
gridname	**グリッドの名称**
gridnick	**グリッドのニックネーム**
welcome	**ビューアのログイン画面に表示されるページのURL**
economy	**ヘルパーURI**
about	**グリッドに関する情報ページへのURL**
register	**グリッドへの登録ページへのURL**
help	**ヘルプページのURL**
password	**パスワードを紛失した場合の対応ページへのURL**

さらに，**[LoginService]**セクションの **WelcomeMessage** には，ログイン時に画面に表示されるログインメッセージを指定する事が可能です（日本語：UTF-8 可）．

ボイスチャットを利用する場合は **FreeSwitch** の設定も行います．この設定は 0.7.2 で **OpenSim.ini** から **Robust.ini** に設定場所が変更になったものです．この設定の詳細については，9 章をご覧ください．

6.1.2 ROBUST サーバの起動とアバターの作成，サーバの停止．

Robust.ini 中の ConnectionString を修正後，図 6.4 のコマンドで ROBUST サーバを起動します．ただし，ROBUST サーバの起動前に，MySQL サーバが正常に起動している必要があります．MySQL サーバを手動で起動する方法は図 5.4 を参照してください．

ROBUST サーバが正常に起動すれば，**R.O.B.U.S.T.#** のコマンドプロンプトが表示されます．

もし，グリッドモードを稼動させる前に，MySQL サーバを使用してスタンドアロンモードを試した場合は，その時作成したリージョンとアバターの情報がグリッドモードにも引き継がれます（ただし，MySQL のデータベースを変更しなかった場合）．一方，SQLite3 を使用してスタンドアロンモードを試した場合や，グリッドモードからはじめた場合は，ここで必ずアバターを一名以上作成する必要があります．何故ならば，グリッドモードではリージョンサーバの起動時にアバターを作成することがで

```
# cd /usr/local/opensim/bin
# mono Robust.exe
```

図 6.4 ROBUST サーバの起動

きないからです. ここで管理用のアバターを作成しない場合, リージョンサーバを新規に起動することはできなくなります.

アバターの作成では, R.O.B.U.S.T.# のコマンドプロンプトに対して, **create user** のコマンドを入力します. すると, スタンドアロンモードでのアバター作成時 (5.2.3項) とほぼ同じメッセージが表示されますので, アバターのファーストネーム, セカンドネーム, ログインパスワード, ユーザのE-mailアドレス (任意) を入力します (図6.5). 2人目以降のアバターを追加する場合にも同様に行います.

なおパスワードは, 入力中は画面には表示されませんが, エンターキーを押すと画面に表示されてしまいます!? (スタンドアロンモードの時と同じ)

サーバプロセスを停止させるには, コマンドプロンプトに対して **quit** または **shutdown** コマンドを入力します.

6.2 リージョンサーバ

グリッドモードのリージョンサーバで使用する設定ファイルは **bin/OpenSim.ini**, **bin/config-include/GridCommon.ini** および **bin/config-include/FlotsamCache.ini** です. これらは, それぞれの exampleファイルからコピーして作成します (図6.6). リージョンの設定ファイル **bin/Regions/*.ini** も必要ですが, スタンドアロンモードと同様に, これはサーバプロセス起動時に対話的に作成されます (または手動で作成することも, サーバコマンドで作成することも可能です).

なお, FlotsamCache.ini は GridCommon.ini から読み込まれるキャッシュ設定用ファイルですが, 内容を変更する必要はありません.

6.2.1 OpenSim.ini

次にリージョンの動作モードを設定します. リージョンの動作モードは OpenSim.ini の **[Architecture]セクション**で指定します. **[Architecture]セクション**の **Include-Standalone** をコメントアウトし, **Include-Grid** を有効にすれば, リージョンの動作モードはグリッドモードとなります (図6.7).

```
R.O.B.U.S.T.# create user
First name [Default]: Test
Last name [User]: User
Password*******
Email []:
18:57:07 - [AUTHENTICATION DB]: Set password for principalID d6596617-260c-4e26-
a64e-208d58fe66b2
18:57:07 - [GRID SERVICE]: GetDefaultRegions returning 0 regions
18:57:07 - [USER ACCOUNT SERVICE]: Unable to set home for account Test User.
18:57:07 - [USER ACCOUNT SERVICE]: Account Test User created successfully
R.O.B.U.S.T.#
```

図6.5 create user コマンドによるアバターの作成

```
# cd /usr/local/opensim/bin
# cp OpenSim.ini.example OpenSim.ini
# vi OpenSim.ini          (図5.2, 図6.7参照)
# cd config-include
# cp GridCommon.ini.example GridCommon.ini
# cp FlotsamCache.ini.example FlotsamCache.ini
# vi GridCommon.ini       (図6.8参照)
```

図6.6 グリッドモードのリージョンサーバでの必要なファイルの準備と書き換え

```
[Architecture]
  ..............
  ;Include-Standalone    = "config-include/Standalone.ini"
  ;Include-HGStandalone = "config-include/StandaloneHypergrid.ini"
  Include-Grid           = "config-include/Grid.ini"
  ;Include-HGGrid       = "config-include/GridHypergrid.ini"
  ;Include-SimianGrid   = "config-include/SimianGrid.ini"
```

図 6.7 [Architecture]セクションで Include-Grid 行を有効にする

HGGrid? ってなんだろ？

HGGrid のHGはハイパーグリッドっていって，他のグリッドと繋げちゃうモードだよ．でも今回はパイパーグリッドの説明はなしだよ．
次回を期待してね．

Simianグリッド(おさるグリッド？)はOpenMetaverseが開発した，WEBアプリケーションでROBUSTサーバの代用をしようというものだよ．
Apacheがサーバの代わりになって，PHPで動くんだ!! とても軽いよ!

6.2.2 GridCommon.ini

bin/config-include/**GridCommon.ini** では，さまざまなサービスを提供するサーバのURI(URL)を指定します．実際にはほとんど全てのサービスをROBUSTサーバが提供するので，ROBUSTサーバのサービスURI(URL)を指定することになります．

図6.8に GridCommon.ini の設定例（一部）を示します．実際には13個のセクションについて，同様にしてURIを指定します．ROBUSTサーバがリモートマシンで稼動している場合には，localhostの代わりに ROBUSTサーバのFQDNかIPアドレスを指定します．

[DatabaseService]セクションのデータベース（MySQL）の設定では，MySQLサーバ使用時のスタンドアロンモードでの設定（5.1.2項）と同様に，MySQLサーバへの接続を記述します（図5.2）．

```
[AssetService]
    DefaultAssetLoader = "OpenSim.Framework.AssetLoader.Filesystem.dll"
    AssetLoaderArgs = "assets/AssetSets.xml"
    ;
    ; Change this to your grid-wide asset server. ......
    ;
    AssetServerURI = "http://localhost:8003"

[InventoryService]
    ;
    ; Change this to your grid-wide inventory server
    ;
    InventoryServerURI = "http://localhost:8003"
```

図 5.3 GridCommon.ini での設定例（一部）

6.2.3 リージョンサーバの起動と停止

OpenSim.iniとGridCommon.iniの設定が終了し，MySQLサーバとROBUSTマネーサーバが起動済みであることを確認したら，OpenSim.exeを monoを使用して起動します（図5.5）．なお，マネーサーバ（次節）を使用する場合には，マネーサーバも起動済みであることを確認します．

もし起動時に bin/Regions/*.ini が存在しなければ，Regions/Regions.ini を作成するために幾つかの質問が表示されるなど，その動作はスタンドアロンモードの時と同じです（5.2.1項を参照）．

bin/Regions/*.ini が存在すれば，その設定に従ってリージョンが起動し始めます．続いてこれもスタンドアロンモードと同様に，エステートの設定とエステートの管理者のアバター情報を入力するように促されます．ただし，グリッドモードでのエステート管理アバターの情報入力では，既に作成済みのアバターの名前を入力する必要があります（スタンドアロンモードでは，ここでアバターが作

```
# cd /usr/local/opensim/bin
# mono OpenSim.exe
```

図 5.5 OpenSim の起動　（再掲載）

```
[MySql]
hostname=localhost
database=opensim_db
username=opensim_user
password=opensim_pass
pooling=false
port=3306
;Max DB connections kept by money server.
MaxConnection = 10
```

図6.9 MoneyServer.ini ファイルの[MySQL]セクション （太字の部分が修正箇所）

成されました）．従って，これに先立ってROBUSTサーバでアバターを作成しておく必要があります（6.1.2項参照）．作成済みのアバターのファーストネームとセカンドネームを正確に入力すれば，リージョンの起動が完了します．

正常に起動が完了した場合，**Region（リージョン名）#** のコマンドプロンプトを表示して，コマンド入力待ちとなります（マルチリージョンの場合は **Region (root) #** のコマンドプロンプトとなります）．サーバプロセスを停止させるには，コマンドプロンプトに対して **quit** または **shutdown** コマンドを入力します．

6.2.4 エラーが出力されて起動できない場合

Linuxのディストリビューションによっては，稀ではありますが，ODEや libopenjpegのエラーが出て起動できない場合があります．その場合は，「3.11節」または「3.12節」を参照してください．

6.3 DTL/NSL Money Server （オプション）

本書で紹介してるマネーサーバは，DTL Currency Processing Project （http://forge.opensimulator.org/gf/project/currency/）のDTL Currency Serverのソースコードを元に，NSLによってLinux上のOpenSimで動作するように改造されています．

ただし，現バージョンでのDTL/NSLのマネーサーバはセキュリティ的には甘く，一般ユーザがこのマネーサーバを誤魔化すことも不可能ではありません．従って，仮想通貨を厳密に管理したい場合には，このマネーサーバは使用しないでください．

DTL/NSL マネーサーバを使用する場合，0.7.2ではOpenSimのソースにパッチを充てて再コンパイルする必要はありません．DTL/NSL マネーサーバの配布パッケージには，マネーサーバのバイナリも添付されています．

マネーサーバをコンパイルする場合は図6.10の **./build.sh** コマンドで行います．マネーサーバの設定ファイルは **bin/MoneyServer.ini** で，このファイルはマネーサーバをコンパイルしたときに自動的に binディレクトリにコピーされます．設定すべき項目は，[MySQL]セクションでのMySQLサーバにアクセスするための情報です．hostname，database，username，passwordに「3.8.7項」で設定したパラメータを指定します（図6.9）．今まで他の設定ファイルで行ってきたMySQLの設定とほとんど変わりありません．

MoneyServer.iniの設定が終了したら，図6.10の monoコマンドでマネーサーバを起動させます．ただし，マネーサーバの起動の前に MySQLサーバが正常に起動している必要があります．また リージョンサーバよりも前に起動させなければなりません．マネーサーバが正しく起動すれば，**Money#** のコマンドプロンプトが表示されます．サーバプロセスを停止させるには，プロンプトに対して **quit** または **shutdown** コマンドを入力します．

リージョンサーバをマネーサーバに対応させるには，OpenSim.iniの[Economy]セクションの設定を行います．まず，[Economy]セクションの **SellEnabled** を **true** に変更し， **UserServer**，**CurrencyServer** および **EconomyModule** を図6.11のように追加します．この例ではマネーサーバがROBUSTサーバ，リージョンサーバと同じマシン（202.26.159.211）で稼動していると想定しています．また，**CurrencyServer** に設定する通信プロトコルがHTTPSであることにも注意してください．

```
# cd /usr/local/opensim/opensim.currency-0.7
# ./build.sh
# cd ../bin
# vi MoneyServer.ini
# mono OpenSim.Grid.MoneyServer.exe
```

図6.10 マネーサーバのコンパイルと起動

```
[Economy]
  ..........
  ; Enables selling things for $0
  SellEnabled = "true"

  UserServer = "http://202.26.159.211:8003/"
  CurrencyServer = "https://localhost:8008/"
  EconomyModule = DTLMoneyModule
  ; 45000 is the highest value that the sim could possibly report because of
protocol constraints
  ObjectCapacity = 45000
```

図6.11 OpenSim.ini の Economy セクションの設定の例（太字の部分が変更箇所）

UserServer にはROBUSTサーバを指定しますが，ROBUSTサーバがローカルマシンで起動していても localhost（127.0.0.1）を指定しないで，正式なIPアドレスまたはFQDNを指定してください．これはマネーサーバがアバターの識別にこのアドレスを使用しているからです．このアドレスにlocalhost(127.0.01)を使用するとマネーサーバはアバターを正しく識別することができなくなる場合があります．

図6．11にも示しているように，**UserServer** のポート番号は8003（または8002）番，**CurrencyServer** のポート番号は標準で8008番です．

6.4　グリッドモードでのビューアの設定

グリッドモードでの接続URLには，スタンドアロンモードのときとは違って，ユーザサーバを指定します．従って，今回の例では接続URLは http://202.26.159.211:8002 となります．

Second Lifeのビューアを用いてのOpenSimに接続する場合は，ショートカットのプロパティを表示させ，「リンク先」にコマンドの引数として

-loginuri　http://**ユーザサーバのIPアドレスまたはFQDN**:8002/

を追加します．今回の例では -loginuri http://202.26.159.211:8002 です．

なお 「10.6.4項」のヘルパー機能を使用する場合は，-loginuriに加えて **-helperuri** の引数も追加する必要があります（詳細は 10.6.4項の ヘルパー機能を参照）．

また，表5.13（5.3節）にOpenSimで使用可能なサード・パーティ製のビューアを載せましたので，いろいろなビューアを試してみるのも良いでしょう．ただし，v2.x系のビューアは現段階ではOpenSimでは使用できませんので，v1.x系のビューアを使用するようにしてください．

7. OpenSim起動後の設定と拡張機能

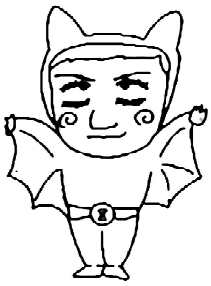
ここまでの作業で OpenSimのサーバは起動しましたが，設定はほとんどデフォルトのままでした．ここでは，OpenSim起動後に設定すべき項目や，OpenSimの便利な（面白い）拡張機能に紹介したいと思います．

全部終わったね！

ここまで長かったね～（泣

もうちょっと設定があるよ．
がんばろう！

Grid サーバーが有れば複数のサーバで
構築できるんだ！　NSL には一体何台の
OpenSim サーバがあるんだろう？



15 台くらいかな．普通の自作マシンで
Pentium4 ～ PhenomX4 まで様々だけどね．
中央の ROBUST サーバーは，少々高い
BladeCenterを使わせてもらってる．

そんなにあったんだ！

7.1 デフォルトアバター

7.1.1 ルース（Ruth）

OpenSim に最初に表示される赤いタイツを履いたちょっと怖い顔のお姉さんは，ルース（Ruth）と呼ばれるアバターです.

初期状態ではルースは何も装着していない状態ですので,このままでは容姿を変更することができません.「持ち物」メニューから「作成」を選択し，ボディパーツ（シェイプ，スキン，髪，目）を作成して，必ず装着してください.ボディパーツを装着することにより，その部分が変更可能となります.

図 7.1 ルース

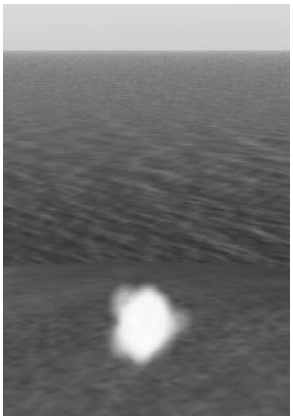


図7.2 煙状のアバター

7.1.2 煙状のアバター

時たま，アバターが煙状になってしますことがあります. この場合も,「持ち物」メニューから「作成」を選択し，ボディパーツ（シェイプ，スキン，髪，目）を作成して装着すれば,ルースが表示されます.ボディパーツを装着してもルースが表示されない場合は，ビューアのバージョンを確認します. もしビューアのバージョンが v2.x ならば，v1 系に戻してみてください. 現在の所，v2 系のビューアは OpenSim では使用することはできません.

ルースさん.
なんだか本当に怖いよ（泣

7.2 土地の標高の編集

7.2.1 土地の平坦化

初期状態のリージョン（SIM）には丸いたこ焼きのような島があるだけです. ここではSIMを平坦にするコマンドを紹介します.リージョンサーバのコマンドプロンプトから以下のコマンドを入力します（図 7.3）.

図 7.3 で 25 は土地の標高です（単位はm）. この標高でリージョン全体が平坦化されます. ちなみに，（海）水面の標高は 20m ですので，ここを 20 より小さくすると土地は全て水没します. その他の tree サブコマンドについては「付録 D」を参照してください.

```
Region (TEST_SIM)# terrain fill 25
```

図7.3 リージョンの平坦化コマンド

7.2.2 標高データのファイルフォーマットと読み込み

外部から土地の標高データを読み込んで,リージョンの標高を変化させることも可能です.データファイルのフォーマットはヘッダなしで，1 点の標高を 4Byte(32bit) float（32bit 実数）で表します（バイト順序はLittle Endian*）. 一つのリージョンの大きさは 256x256 ですのでファイルサイズは 4Bytex256x256 = 262,144Byte となります（図 7.4）. リージョンの座標は左下隅が原点で，Y 軸

注*) Little Endian： ガリバー旅行記の小人の国で卵を細い方から割る人々. 転じて，FFFF...FF がメモリの先頭で，0000...00 がメモリの終わりであるよなインテル系の CPU のメモリ配置のこと. 通常，ファイルにデータを保存したり，ネットワークにデータを流したりする場合には Big Endian という暗黙の了解があったはずなのに，最近では Wintel の勢力拡大に伴って，データファイルの保存に関する約束事は破棄されつつある. そのうちネットワークもそうなるのか？ そういえば SL や OpenSim はポート番号などを転送する場合，完全に Little Endian だよ. まあ Windows の場合は普通に読み書きして OK ということ.

全然説明になっていないぞ！

は上方が＋方向となります．この種のデータを作成するツールとしては L3DT（http://www.bundysoft.com/L3DT/）があります．

作成したデータファイルの拡張子は必ず **.r32** とします．拡張子でファイルフォーマットの識別を行いますので，拡張子を間違えると正しく読み込まれません．データファイルをリージョンサーバに読み込ませるには，リージョンサーバのコマンドプロンプトから図7.5のようなコマンドを入力します．ここで filename.r32 は標高データが格納されたデータファイルの名前です．必要ならファイルのパスも指定します．

なお，デフォルトの設定ではリージョン全体を表示しようとして，ビューアのカメラ位置（目の位置）を手前に引くと，地形が荒く表示されてしまいます．これはLOD（Level of Detail）と呼ばれる機能で，遠くにあるものを適当に描画することにより全体の描画速度を上げるためのものです．Second Lifeや OpenSimのビューアでは，LODを 0.0～4.0 まで指定可能ですることが可能です．このLODを4.0に設定すれば，リージョン全体を綺麗に表示することが可能です．

4Byte float	(0,0)
4Byte float	(1,0)
4Byte float	(2,0)
·····	
4Byte float	(255,0)
4Byte float	(0,1)
4Byte float	(2,1)
·····	
4Byte float	(255,1)
4Byte float	(0,2)
4Byte float	(1,2)
·····	
·····	
4Byte float	(254,255)
4Byte float	(255,255)

図7.4 データファイルの構造

地形のLODの設定の方法は，まず Ctl+Alt+DでAdvancedメニューを表示します．Advancedメニューから「Debug Settings...」を選択すると，「デバッグ設定」のダイアログが表示されます（図7.6）．「デバッグ設定」の上の入力フィールドに **RenderTerrainLODFactor** と入力すると，下方の入力フィールドに現在の数値が表示されますので，ここを4に修正すればOKです．

RenderTerrainLODFactorの他に，オブジェクト描画時のLODを決定する **RenderVolumeLODFactor** などの変数もあります．

```
Region (TEST_SIM)# terrain load filename.r32
```

図 7.5 標高データの読み込み

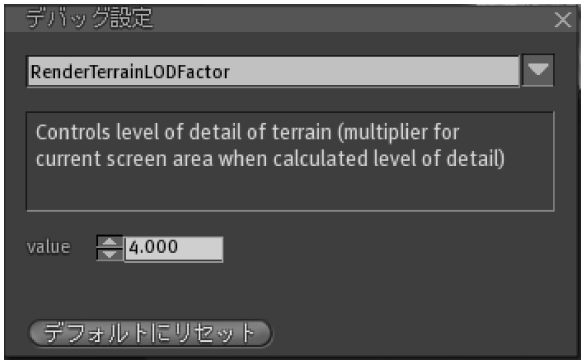


図 7.6 デバッグ設定ダイアログ

7.2.3 r32標高データへの変換

標高データのファイルフォーマット（図7.4）が分かっていますので，がんばればいろいろな画像データを標高データに変換して遊ぶこともできます（たとえば図7.7）．図7.7はFumi Haxが画像処理の授業で使用している自作ツール(CTView)に r32用の標高データ変換モジュールを追加して変換したものです（図7.8）．このツールに関しては，以下のページに簡単な解説があります．

CTView用 r32データ変換モジュール: http://www.nsl.tuis.ac.jp/xoops/modules/xpwiki/?OpenSim%2Fr32file

図 7.7　画像データから変換して作成したキャラ島

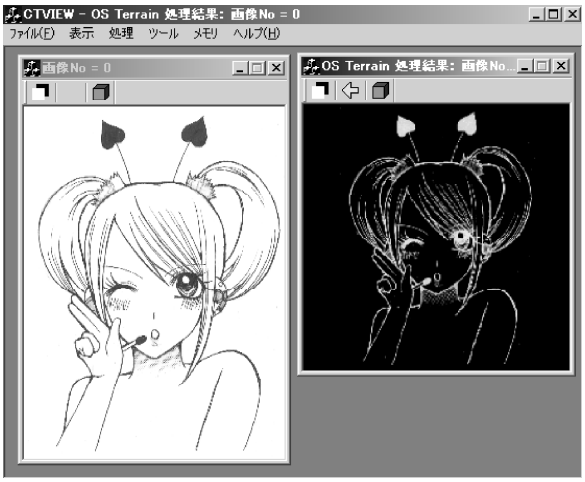


図 7.8　画像変換用ツール（CTView+ 変換モジュール）

わたしだ！わたしだ！わたし島だ！！
すごい！！

CTView用に簡単なモジュール
を作ったんだ.

プログラムを公開しないの？

たいしたプログラムじゃないし，何より
マニュアル書くのが面倒くさ～！

横着しないで，Wikiでも書きなさい！！

はっはい；

7.2.4 OpenSimジオラマシステム

NSLでは，OpenSimのソースコードを改造し，WEBサイトから標高データを取り込んで直接地形を変形させたり，スカルプテッドプリムを地表に貼り Dynamic Texture で地図データを表示させるなどの手法を用いて，OpenSim上にリアルなジオラマを作成する研究も行っています．

図7.9はこの手法を用いてOpenSim上に再現した富士山のジオラマです．ページ数の関係上，ここではシステムについて詳細に紹介することはできませんが，下記URLにWikiと動画がありますので，ご興味の有る方はご覧ください．

ジオラマシステムWiki: http://www.nsl.tuis.ac.jp/xoops/modules/xpwiki/?OpenSim%2FDiorama
ジオラマシステム動画: http://www.nsl.tuis.ac.jp/xoops/modules/x_movie/x_movie_view.php?cid=2&lid=28

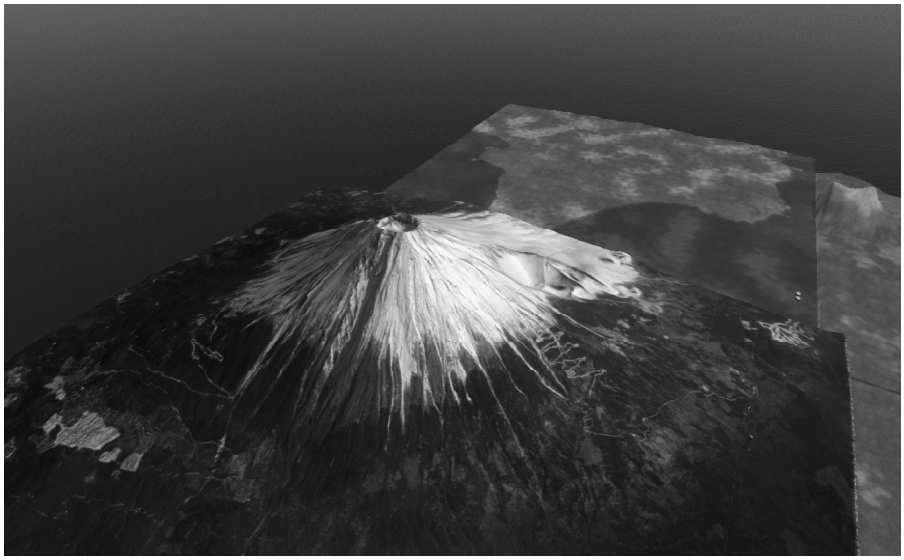


図7.9 OpenSimジオラマシステムでの富士山

7.3 OAR (OpenSim ARchive)

Second Lifeが OpenSimに比べて優位な点はその圧倒的な物量です．OpenSimでは地形を自由に変形できるとは言え，リージョン内で見栄えがする程度にオブジェクトを作成・配置するだけでも大変な労力と時間が必要です．Meerkat などのビューアや Second Inventory (http://www.secondinventory.com/)などのツールを使用すれば，Second Life内のオブジェクトをOpenSimに持ち込むことも可能ですが，それでもリージョンを一から作成するのは大変です．

OpenSimには，OAR(OpenSim Archive)ファイルと呼ばれるものがあります．OARファイルはリージョンの状態(地形やオブジェクトの配置など)を記録したファイルです．OARファイルがあればリージョンの状態を完全に復元することが可能です．OARファイルはリージョンの状態のバックアップとしても使用できますが，自分の作ったリージョンをOARファイルとして公開することも可能です．また公開されているOARファイルを読み込んで，簡単にリージョンを構成することもできます．

下記に，OARファイルを公開しているサイトへのリンクページ(OpensimWorlds)を示します．最初はこれらのサイトからOARファイルをダウンロードして，自分のサーバに読み込んでみるのも良いでしょう．

OpensimWorldsのサイトURL: http://www.opensimworlds.com/index.php
OpensimWorldsのダウンロードリンクURL: http://www.opensimworlds.com/index.php?part=worlds

ダウンロードした OAR ファイルをリージョンサーバに読み込ませるには，**load oar** コマンドを使用します (図 7.10).[ ]内の --merge は省略可能ですが，これを指定すると，現在のリージョンと読み込んだOARファイルのリージョンのオブジェクトがマージされます(地形データは読み込まれません)．

なお，現在のリージョンの状態をOARファイルに保存するには，**save oar** コマンドを使用します(図7.11)．

```
Region (TEST_SIM)# load oar [--merge] oar_filename
```

図7.10 OARファイルの読み込み

```
Region (TEST_SIM)# save oar oar_filename
```

図7.11 OARファイルの保存

7.4 オブジェクト（プリム）のパーミッション

OpenSimのデフォルトの設定では，誰でも他人の作成したオブジェクト（プリム）を自由に扱うことが可能です．しかしながら，この設定は OpenSimを複数のユーザが利用する場合などには不都合です．作成したオブジェクト（プリム）に権限を設定するには，[Startup]セクションのPERMISSIONSブロックの設定を行います（図7.12)．

図7.12の **permissionmodules** では，オブジェクトのパーミッションを処理するモジュールを指定し，さらに **serverside_object_permissions** でその処理をサーバ側で行うことを指定します．また，**region_owner_is_god**，**region_manager_is_god**，**parcel_owner_is_god** ではビューアのGoodモード*への移行の可否を指定しています．これは各グリッドの環境に合わせて変更してください．

```
permissionmodules = "DefaultPermissionsModule"
serverside_object_permissions = true
allow_grid_gods = false
region_owner_is_god = true
region_manager_is_god = false
parcel_owner_is_god = flase
```

図7.12 Permissionの設定

7.5 スクリプトエンジン

OpenSim.iniの[XEngine]セクションで，**AllowedCompilers** に言語を追加すると（図7.13)，LSL(Lenden Scripting Language)以外の言語でスクリプトを作成することが可能となります．OpenSimのWikiページ（下記）によれば，LSLの他にC#，VB.NET，JScript.NET，Yield Prologが使用可能のようです．我々が確認しているのは LSLの他は C#のみですが，他の言語も環境を整えれば(VB.NETのコンパイラを用意するなど)，たぶん動作するでしょう．

OpenSimのスクリプト言語に関するWikiページ: http://opensimulator.org/wiki/Scripting_Languages

図7.14にC#でのサンプルを示します．C#でスクリプトを作成する場合は，必ず第一行目に **//c#** と記述しなければなりません．

注*) Goodモード：ビューアで Ctrl+Alt+D を押すとAdvancedメニューが表示される． Advancedメニューの Request Admin Srarus (Ctrl+Alt+G) を選択すると，（許可されているならば）Goodモードに入ることができる． Goodモードでは通常許可されない様々な操作を行うことが可能．
Goodモードを抜けるには， Advancedメニューの Leave Admin Srarus (Ctrl+Alt+Shift+G) を選択する．

```
AllowedCompilers=lsl,cs
```

図 7.13 スクリプト言語として C#を追加

```
//c#
string message = "Hello World!";
string hellmsg = "ようこそ. 地獄の世界へ!!";

public void default_event_state_entry()
{
    llSay(0, message);
}

public void default_event_touch_start(LSL_Types.LSLInteger total_number)
{
    string hell = message.Replace("Hello", "Hell") + " " + hellmsg;
    llSay(0, hell);
}
```

図 7.14 C#によるスクリプトの例

また，**AllowOSFunctions** に true を設定すると，LSL で OpenSim の拡張関数（OS 関数: OSSL）が使用できるようになります.OSSLで使用可能になる関数の詳細については下記のページをご覧ください.

OpenSimのOSSLに関するWikiページ: http://opensimulator.org/wiki/OSSL_Implemented

7.6 Ninja Physics

OpenSim.ini の[ODEPhysicsSettings]セクションの Joint support ブロックにある **use_NINJA_physics_joints** を true にすると，物理プリムを使って，自由に曲がるジョイントをつくることができます．ジョイントには 2 種類あり，**ball-and-socket** ジョイントは 2 つのプリムの間にあって，両側のプリムに自由な回転を許可します．一方 **hinge** ジョイントはドアの蝶番のように，両側のプリムに対して一つの軸（X 軸）の周りの回転のみを許可します．

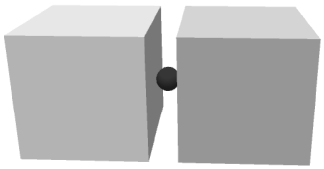


図7.15 ジョイントの構成

ジョイントを作成するには，まず 2 つのプリムを作成します．この 2 つのプリムにそれぞれユニークな名前を付けます（例えば box1, box2)．次にこの2つのプリムの間にジョイントとなる小型のプリムを配置します（図 7.15)．ジョイントとなるプリムの編集モードにおいて，「一般」タグでプリムの名前を **balljoint**[uniqname] と指定すると，このジョイントは ball-and-socket ジョイントとなります．また，プリムの名前を **hingejoint**[uniqname]とすると hinge ジョイントとなります．ここで [uniqname] には他のプリムと名前が被らないユニークな名前（文字）を指定します．

さらに，ジョイントとなるプリムの「**説明**」を入力するフィールド（名前を入力するフィールドの直ぐ下）に両端のプリムの名前を，スペースを 1 つ空けて記述します．今回の例では“box1 box2”となります（図 7.16)．ジョイントの片側を固定する場合には，そこにはプリムを配置せず，ジョイン

図 7.16 Ninja Physics の balljoint の設定

トの「**説明**」の箇所にはプリムの名前の代わりに **NULL** を指定します.

最後に，編集モードこれらのジョイントプリムと両端のプリム全体を選択し，「形状」タグの「物理プリム」のチェックボックスにチェックを入れれば，ジョイント機能が動作し始めます.

もし分かりにくければ，下記 URL に Ninja Physics のデモ用 OAR ファイルがあります（拡張子は tgz ですが中身は OAR ファイルです）.

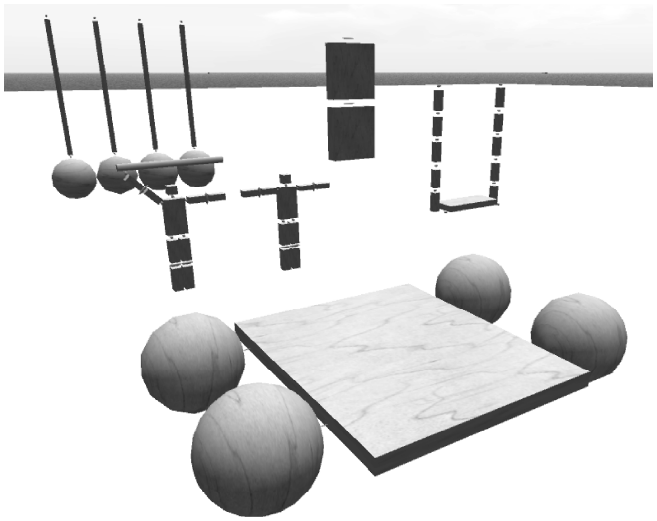
Ninja Physicsデモ用OAR: http://forge.opensimulator.org/gf/download/frsrelease/142/304/demo-playground.tgz

ダウンロードした OAR ファイルを図 7.17 のようにして読み込めば，図 7.18 のようなオブジェクトが（25m 地点に）読み込まれます. 編集モードで，読み込んだ全てオブジェクトを選択し「形状」タグの「物理プリム」のチェックボックスにチェックを入れれば，ジョイントが動き出します. ロボット型のものは頭部にスクリプトが内臓されており，チャットモードで **arise**, **kneel**, **die** などの命令を与えられます.

ロボットのスクリプトを参考にしたい場合は，Good モード*に入ってロボットを選択し，Admin メニューから「Object」→「Force Owner to Me」を選択すれば，ロボットのオーナになれますので，スクリプトも参照可能となります.

```
Region (TEST_SIM)# load oar --merge demo-playground.tgz
```

図 7.17 Ninja Physics のデモ用 OAR ファイルの読み込み



このロボット，酔っ払ってるみだいだよ!!

図 7.18 Ninja Physics デモ用オブジェクト

7.7 メガリージョン

OpenSim.iniの[Startup]セクションで，**CombineContiguousRegions**を trueにすると，メガリージョンを扱うことができます．メガリージョンのモードでは一つのプロセスで複数の隣接したリージョンを制御できます．従来の方法（マルチリージョンを含む）では，アバターがリージョンの境界を越える場合，ビューアと通信しているプロセス（スレッド）も切り替わるためタイムラグが発生します．また，スクリプトを含むオブジェクトがリージョンの境界を越える場合，スクリプトの再コンパイルが必要なため，OpenSimでは色々と問題が発生することがあります．

メガリージョンでは全体が一つのリージョンとして扱われるため，上記のような制限は発生しません．メガリージョンのモードに移行するには，CombineContiguousRegionsをtrueにし，複数のリージョンの設定ファイルの名前を，アルファベット順（読み込み順）に「ある順番」になるように変更します．複数のリージョン設定ファイルを作成する代わりに，一つのリージョン設定ファイルに複数のリージョンの設定を「ある順番」になるように記述することも可能です．

「ある順番」とは図7.19，7.20，7.21のような順番です．ここで，一番最初に設定が読み込まれるリージョンと最後に読み込まれるリージョンの位置は必ず決まっています．一番最初のリージョンの位置は必ず左下（南西:SW）で，最後のリージョンの位置は必ず右上（北東:NE）でなければなりません．

メガリージョンに関するさらに詳細な記述に関しては，下記URLを参照してください．

メガリージョンの設定URL: http://opensimulator.org/wiki/Setting_Up_Mega-Regions

NSLでは 5x5までのメガリージョンの作動を確認しています．また全体の形は正方形でなくとも矩形（四角形）であれば大丈夫のようです．ただし，メガリージョンの機能はまだ開発中であり，既存のリージョン(SIM)をメガリージョンにした場合には問題が発生する可能性があると，Wikiに注意書きがあります．

既存のリージョン(SIM)をメガリージョンの一部にする場合には，必ずリージョンの状態をOARファイルにバックアップしてから，メガリージョンのモードに移行してください．OARファイルの作成には，**save oar** コマンドを使用します（図7.11）．

7.8 ツリーモジュール

ツリーモジュールは成長する植物群（**copse**）を管理するモジュールです．OpenSim.iniの[Trees]セクションの **active_trees**を trueにすることにより，ツリーモジュールが有効になります（またはコマンドで **tree active true**を入力しても良い）．

copseを定義するXMLファイルの例を図7.22に示します．このようなXMLファイルを幾つか作成し，**tree load**コマンドでリージョンサーバに読み込むことにより，成長する植物群（seed，growth，dieサイクル）をOpenSim内でシミュレートすることが可能となります．XMLの各タグの意味については表7.23をご覧ください．なお，このときに指定する植物のタイプ（m_tree_type）は bin/OpenMetaverse.XML 内で定義されているものを使用しますが，それらを抜き出すと表7.24のようになります．

図7.22のようなcopseの定義XMLを **tree load**コマンドで読み込んだ後は，**tree plant**コマンドで実際に植物を植えます．このとき，植物を植える地点の標高が m_treeline_low ～ m_treeline_high の間にない場合には，植物を植えることはできません．

植物群の現在の状態は **tree statistics**コマンドで表示できます．また，植物群を削除する場合には **tree remove**コマンドを使用します（図7.25）．各treeサブコマンドの詳細については「付録D」を参照してください．

図7.26に3種類の copseを植えた場合の直後の様子と，しばらく時間が経過した場合の成長の様子を示します．

NW 3 (1000, 1001)	NE 4 (1001, 1001)
SW 1 (1000, 1000)	SE 2 (1001, 1000)

NW 2 (1000, 1001)	NE 4 (1001, 1001)
SW 1 (1000, 1000)	SE 3 (1001, 1000)

図7.19 2x2のメガリージョンでの読み込み順序

NW 7 (999, 1001)	N 8 (1000, 1001)	NE 9 (1001, 1001)
W 4 (999, 1000)	C 5 (1000, 1000)	E 6 (1001, 1000)
SW 1 (999, 999)	S 2 (1000, 999)	SE 3 (1001, 999)

NW 3 (999, 1001)	N 6 (1000, 1001)	NE 9 (1001, 1001)
W 2 (999, 1000)	C 5 (1000, 1000)	E 8 (1001, 1000)
SW 1 (999, 999)	S 4 (1000, 999)	SE 7 (1001, 999)

図 7.20 3x3 のメガリージョンでの読み込み順序

NW 21 (998, 1002)	NNW 22 (999, 1002)	N 23 (1000, 1002)	NNE 24 (1001, 1002)	NE 25 (1002, 1001)
WNW 16 (998, 1001)	NW 17 (999, 1001)	N 18 (1000, 1001)	NE 19 (1001, 1001)	ENE 20 (1002, 1001)
W 11 (998, 1000)	W 12 (999, 1000)	C 13 (1000, 1000)	E 14 (1001, 1000)	E 15 (1002, 1000)
WSW 6 (998, 999)	SW 7 (999, 999)	S 8 (1000, 999)	SE 9 (1001, 999)	ESE 10 (1002, 999)
SW 1 (998, 998)	SSW 2 (999, 998)	S 3 (1000, 998)	SSE 4 (1001, 998)	SE 5 (1002, 998)

図 7.21 5x5 のメガリージョンでの読み込み順序　その 1（その 2 は省略）

```
<Copse>
    <m_name>Copse1</m_name>
    <m_frozen>false</m_frozen>
    <m_tree_type>Palm1</m_tree_type>
    <m_tree_quantity>500</m_tree_quantity>
    <m_treeline_low>20</m_treeline_low>
    <m_treeline_high>50</m_treeline_high>
    <m_seed_point>
        <X>128</X>
        <Y>128</Y>
        <Z>0</Z>
    </m_seed_point>
    <m_range>100</m_range>
    <m_initial_scale>
        <X>5</X>
        <Y>5</Y>
        <Z>5</Z>
    </m_initial_scale>
    <m_maximum_scale>
        <X>20</X>
        <Y>20</Y>
        <Z>20</Z>
    </m_maximum_scale>
    <m_rate>
        <X>0.01</X>
        <Y>0.01</Y>
        <Z>0.01</Z>
    </m_rate>
</Copse>
```

図 7.22 成長する植物（copse）の定義 XML

m_name	この植物を定義・識別するためのCopse名を指定します.
m_frozen	デフォルト で植物の成長を凍結するかどうか指定します.
m_tree_type	この植物のタイプを指定します. デフォルト で指定可能なタイプは表7.23を参照してください.
m_tree_quantity	繁殖可能な個体の最大数を指定します.
m_treeline_low	繁殖可能な標高の最低値をメートルで指定します. 海水面の高さはデフォルトで20mであることに注意してください.
m_treeline_high	繁殖可能な標高の最高値をメートルで指定します.
m_seed_point	最初に植物を植える位置を指定します. 植物は地面に植えられますので，Z方向の位置は無視されます.
m_range	最初に植えられた位置から，どの距離まで繁殖可能かをメートルで指定します.
m_initial_scale	最初に植物を植えたときの，植物のサイズを指定します.
m_maximum_scale	植物の最大成長時のサイズを指定します.
m_rate	植物の成長係数.

図 7.23 copse 定義用の XML タグ

7.9 その他の機能

ボイスチャット機能については「9 章 FreeSwitch を利用したボイスチャット」を，ヘルパー機能については「10.6.4 項」を，グループ機能については「10.6.5 項」を，プロファイル機能については「10.6.6 項」を，エコノミー（マネーサーバ）機能については「6.3 節」をそれぞれご覧ください.

BeachGrass1	ビーチグラス	Palm1	ヤシ
Cypress1	糸杉(檜)	Palm2	ヤシ
Cypress2	糸杉(檜)	Pine1	パイン(松)
Dogwood	ハナミズキ	Pine2	パイン(松)
Eelgrass	甘藻(海草)	Plumeria	プルメリア
Eucalyptus	ユーカリ	SeaSword	？
Fern	シダ	TropicalBush1	トロピカルブッシュ
Kelp1	昆布	TropicalBush2	トロピカルブッシュ
Kelp2	昆布	WinterAspen	冬のポプラ
Oak	オーク(柏)	WinterPine1	冬の松
		WinterPine2	冬の松

表7.24 植物のタイプ

```
Region (TEST_SIM) # tree load copse.xml
11:24:04 - [TREES]: Loading copse definition....
11:24:04 - [TREES]: Loaded copse: ATPM: Copse1; 1000; 50.0; 20.0; 300.0; WinterAspen;
<140, 90, 0>; <4, 4, 4>; <15, 15, 15>; <0.01, 0.01, 0.01>;
Region (TEST_SIM) # tree plant Copse1
11:24:27 - [TREES]: New tree planting for copse Copse1
Region (TEST_SIM) # tree statistics
11:24:41 - [TREES]: Activity State: True;  Update Rate: 1000
11:24:41 - [TREES]: Copse Copse1; 1 trees; frozen False
Region (TEST_SIM) # tree remove Copse1
11:25:04 - [TREES]: Copse Copse1 has been removed
```

図7.25 treeコマンドの例

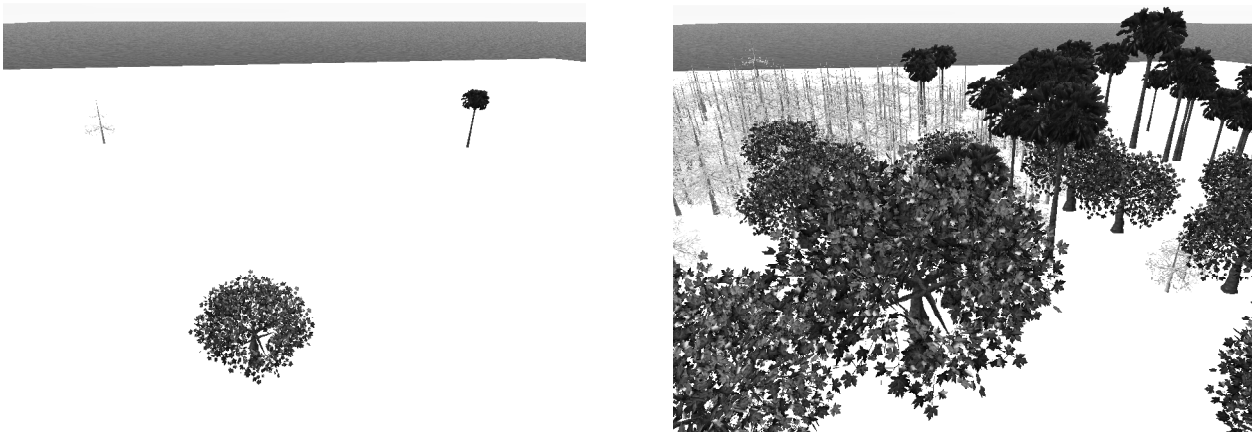


図7.26 ツリーモジュールによる植物の成長と繁殖の様

どんどん森が広がって行くよ！
でもremoveコマンドにまだバグがあるみたいだよ.

8. サーバプロセスのバックグラウンド起動

OpenSimのサーバ群は通常起動した場合，フォアグラウンドで起動します．コンソールからコマンドを入力する場合などはフォアグラウンドで起動した方が便利ですが,24時間稼動のサーバマシンで運用する場合などには扱いにくい場合があります．

例えばヘッドレス（ディスプレイの無い）のサーバマシンで OpenSimサーバを起動するには，sshなどでサーバマシンにリモートログインしてプロセスを起動することになりますが，この場合はリモートログインに使用した端末ソフトウェアが終了しただけで OpenSimサーバも落ちてしまいます．

この章では，OpenSimサーバをバックグラウンドで起動する方法として，Linuxの screenコマンドを使用する方法と，OpenSimのRestConsoleを使用する方法を紹介します．

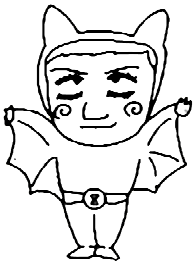
どうかな？ OpenSimには慣れてきたかな?

mono OpenSim.exe の入力とかが大変だけど
大丈夫だよー
Gridモードで複数の画面での管理とかも大変かな

じゃあ，そろそろ,起動スクリプトでも用意
してみようか

そんな便利なものがあるなら最初から教えてよーーー．

お前らばかりに楽させ
てなるものか！



フフフ,そうもいかないな．
何事も苦労が大事じゃ
若い内は苦労しろ！

なんか，オジンくさ～

8.1 screen コマンドによるバックグラウンド起動

8.1.1 ROBUSTサーバのバックグラウンド起動

24時間稼動のサーバマシンでOpenSimをバックグラウンド起動するためのスクリプトは，OpenSimのWikiやOpenSim Forgeプロジェクト（http://forge.opensimulator.org/gf/）にも幾つか見受けられますが，ここではNSLで作成した簡単なスクリプトを紹介します．

これらは screenコマンドを利用してOpenSimのサーバ群をバックグラウンド起動するためのスクリプトです．ブート時の自動起動やサーバへのコマンド入力には向きませんが，24時間稼動のシステムにおいて十分使用に耐えることができます．

ROBUSTサーバおよびマネーサーバのバックグラウンド起動用スクリプト **opensim_server**を図8.1に示します．例えばこのスクリプトを /etc/init.d に設置すれば，図8.2ようなコマンドでサーバの「起動」，「停止」および「再起動」を行うことができます．

なお，マネーサーバを使用しない場合は，図8.1のマネーサーバに関する部分はコメントアウトしなければなりません．

```
# /etc/init.d/opensim_server start          起動
# /etc/init.d/opensim_server stop           停止
# /etc/init.d opensim_server restart        再起動
```

図 8.2 ROBUST サーバのバックグラウンド制御

8.1.2 リージョンサーバのバックグラウンド起動

screenを使用したリージョンサーバのバックグラウンド起動用スクリプト **opensim_region** を図8.3に示します．このスクリプトは，30秒に一度，サーバが正常に稼動しているかどうかをチェックし，万が一リージョンサーバがダウンしている場合には自動で再立ち上げを行います．ただし，リージョンサーバがハングしているような場合には，正常な状態に戻すことはできません．

opensim_serverと同様にこのスクリプトを/etc/init.d に設置すれば，図8.4ようなコマンドで，サーバの「起動」，「停止」および「再起動」を行うことができます．ただし，リージョンサーバを再起動させる場合は，なるべくrestartオプションは使用せずに stopオプションでサーバを停止させ，確実に停止していることを確認してから startオプションでサーバを起動した方が良いでしょう．これは，リージョンサーバが停止するのに非常に長い時間がかかる場合や，途中でハングして完全停止しない場合があるからです．このような場合には restartオプションでのリスタートは失敗してしまいます．

ここで紹介した **opensim_server** および **opensim_region** のスクリプトは，「4.1.2項」でダウンロードした opensim.nsl.patch-0.7の configディレクトリにも格納されています．

```
# /etc/init.d/opensim_region start          起動
# /etc/init.d/opensim_region stop           停止
# /etc/init.d/opensim_region restart        再起動（非推奨）
```

図 8.4 リージョンサーバのバックグラウンド制御

```
#!/bin/bash

OSDIR=/usr/local/opensim          # OpenSim をインストールしたディレクトリを指定
MONO=/usr/local/bin/mono          # mono へのパス
SLEEPTM=5                         # リスタート時のスリープ時間


start() {
    cd $OSDIR/bin
    echo "OpenSim R.O.B.U.S.T. Server Start."
    screen -dmS opensim_robust $MONO $OSDIR/bin/Robust.exe
    echo "OpenSim Money Server Start."
    screen -dmS opensim_bank $MONO $OSDIR/bin/OpenSim.Grid.MoneyServer.exe
}


stop() {
    screen -S opensim_bank -p 0 -X stuff $'quit¥n'
    echo "OpenSim Money Server Stoped."
    screen -S opensim_robust -p 0 -X stuff $'quit¥n'
    echo "OpenSim R.O.B.U.S.T. Server Stoped."
}


restart() {
    stop
    sleep $SLEEPTM
    start
}


case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  restart|reload)
    restart
    ;;
  *)
    echo $"Usage: $0 {start|stop|restart}"
    exit 1
esac

exit $?
```

図 8.1 ROBUST, マネーサーバのバックグラウンド起動用スクリプト

```
#!/bin/bash

OSDIR=/usr/local/opensim              # OpenSim をインストールしたディレクトリを指定
PRGFL=/etc/init.d/opensim_region      # このスクリプトのフルパス
MONO=/usr/local/bin/mono              # mono へのパス
SLEEPTM=15                            # リスタート時のスリープ時間
CHKTM=30                              # 監視間隔
export MONO_THREADS_PER_CPU=256
ulimit -s 262144

SCRNID=opensim_region
PIDFL=/var/run/opensim_region_shell.pid
```

```
start() {
    cd $OSDIR/bin
    echo "OpenSim Region Server Start."
    screen -dmS $SCRNID $MONO $OSDIR/bin/OpenSim.exe
}


stop() {
    screen -S $SCRNID -p 0 -X stuff $'quit¥n' 1> /dev/null 2>&1
    echo "OpenSim Region Server Stoped."
}


kill_loop() {
    PID=`cat $PIDFL 2> /dev/null`
    if [ "$PID" != "" ]; then
       kill -9 $PID 2> /dev/null
       rm -f $PIDFL
    fi
}


loop_check() {
    kill_loop
    echo $$ >| $PIDFL
    while [ "" = "" ]; do
        sleep $CHKTM
        CHECK=`ps ax|grep SCREEN |grep $SCRNID`
        if [ "$CHECK" = "" ]; then
            start
        fi
    done
}


case "$1" in
  start)
    start
    /bin/bash $PRGFL check &
    ;;
  stop)
    kill_loop
    stop
    ;;
  restart|reload)
    kill_loop
    stop
    sleep $SLEEPTM
    start
    /bin/bash $PRGFL check &
    ;;
  check)
    loop_check
    ;;
  *)
    echo $"Usage: $0 {start|stop|restart}"
    exit 1
esac

exit $?
```

図8.3 リージョンサーバのバックグラウンド起動用スクリプト

8.2 RestConsole モードでのバックグラウンド起動

8.1節で紹介したscreenコマンドを使用したバックグラウンド起動では，サーバにコマンドを入力するのが非常に面倒です（やってやれないことはありませんが）．

OpenSim のサーバでは，起動オプションとして **console** オプションを指定することが可能です．console として **rest** を指定してサーバを起動すると，サーバは **RestConsole** モードになり，HTTP (REST) 経由でコマンドを受け付けるようになります．RestConsoleモードでは，キーボード（標準入力）からのコマンドを受け付けませんので，このままバックグラウンドに持っていっても何の問題も起こりません．

RestConsole モードのサーバを制御するために，**OpenSim.ConsoleClient** というユーザインターフェイスも用意されています．

8.2.1 RestConsole モード

ROBUST サーバを RestConsole モードで起動するには，OpenSim.ini の[Startup]セクション内に **console = rest** の一行を追加します．リージョンサーバの場合も同様に，Robust.ini の[Startup]セクション内に **console = rest** の一行を追加します．残念ながらマネーサーバを RestConsole モードで起動することはできません．

また，REST(REpresentational State Transfer)*で接続する場合の情報として，それぞれの設定ファイルの[Network]セクションに，ユーザ名（**ConsoleUser**），パスワード（**ConsolePass**），HTTP (REST) の接続ポート（**ConsolePort**，**console_port**）も記述しておきます（図8.5，8.6）．

ROBUSTサーバとリージョンサーバでは，接続ポートの設定用フィールドの名前が違いますので注意してください．なお，ROBUSTサーバとリージョンサーバが同一のマシンで動作している場合には，それぞれの接続ポート（ConsolePort，console_port）には違う値を指定しなければなりません．また，ポート番号（ConsolePort，console_port）に0を指定した場合は，サーバのHTTPのサービスポート（それぞれ，port またはhttp_listener_port で指定したポート）のポート番号が併用されます．

[Startup]セクションに **console = rest** を追加せずに **-console=rest** のオプションをつけて起動しても RestConsole モードにすることが可能です（図8.7）．

```
[Startup]
.................
console = rest
.................
[Network]
.................
ConsoleUser = console_user
ConsolePass = console_pass
ConsolePort = 8005
.................
```

図 8.5 Robust.ini の変更部分の例

```
[Startup]
.................
console = rest
.................
[Network]
.................
ConsoleUser = console_user
ConsolePass = console_pass
http_listener_port = 9000
console_port = 9005
.................
```

図 8.6 OpenSim.ini の変更部分の例

```
# mono Robust.exe -console=rest &
# mono OpenSim.exe -console=rest &
```

図 8.7 サーバのバックグラウンド起動の例

注 *) REST： REST に関する詳しい解説はこの本の趣旨ではありませんので，とりあえず下記 URL を参照してください．とても分かり易い解説サイトです．

http://yohei-y.blogspot.com/2005/04/rest_23.html

8.2.2 OpenSim.ConsoleClient

RestConsoleモードのサーバを制御するには,OpenSim.ConsoleClientを使用します.ただし,ここでの説明通りにOpenSim.ConsoleClientを動作させるためには,「4.1.2項」に従ってNSLのパッチが適用されている必要があります.

まず, OpenSim.ConsoleClientの設定ファイル（OpenSim.ConsoleClient.ini）をOpenSim.ConsoleClient.ini.exampleからコピーして作成し,内容を接続するサーバに合わせて書き換えます（図8.8）. 一つのOpenSim.ConsoleClientから複数のサーバを制御する場合には, -host=, -port= などのオプションを指定することも可能です. これらのオプションは設定ファイルより優先されます（図8.9）.

OpenSim.ConsoleClient が正常にサーバに接続できれば,接続したサーバのコマンドプロンプトがそのまま表示されます.ただし,接続に使用するターミナルによっては,サーバからの画面制御用のエスケープシーケンスを正常に処理できずに画面が崩れる場合があります.NSLの環境では,端末の種類（TERM環境変数の値）が vt100の場合に若干表示がおかしくなってしまいました.

サーバのコマンドプロンプトが表示されれば,そこから接続先のサーバにコマンドを与えることが可能となります. ただし, **quitコマンド**は OpenSim.ConsoleClientの終了コマンドとなります. 接続先のサーバを終了させるには **shutdownコマンド**を使用してください.

```
[Startup]
    ..................
     user = console_user
    ..................
     host = 202.26.159.211
    ..................
    port = 9005
    ..................
     pass = console_pass
```

図8.8 OpenSim.ConsoleClient.ini の例

```
# mono OpenSim.ConsoleClient.exe -host=localhost -port=8005
```

図8.9 オプションを指定した場合の OpenSim.ConsoleClient の起動例

8.2.3 RestConsoleモードでの起動スクリプト

RestConsoleモードを使用している場合の起動スクリプトの例（**opensim_server_rest**, **opensim_region_rest**）を図8.10, 8.11に示します. このスクリプトではサーバの設定ファイルを自動で読み取って,RestConsoleの設定が為されていない場合には screenコマンドを使用してサーバを起動します.

スクリプトの使用方法は図8.2, 8.4と全く同じです.このスクリプトでROBUSTサーバまたはリージョンサーバを起動した場合, それらがバックグランドで起動している状態であっても, OpenSim.ConsoleClientを使用すれば,いつでもインターラクティブにコマンドラインからの制御が可能となります.

このスクリプト本体もopensim.nsl.patch-0.7の configディレクトリ（4.1.2項）に格納されています. なお, このスクリプトは, 空白とタブに対する操作を行っていますが, 図8.10, 8.11を目で見た限りでは空白とタブの区別が付かないと思いますので,使用する場合は必ずopensim.nsl.patchのconfigディレクトリ内のものを使ってください.

```
#!/bin/bash

OSDIR=/usr/local/opensim          # OpenSim をインストールしたディレクトリを指定
MONO=/usr/local/bin/mono          # mono へのパス
SLEEPTM=5                         # リスタート時のスリープ時間
REST_CNSL="yes"                   # RestConsole を使用するかどうか


if [ "$REST_CNSL" = "yes" ]; then
    ROBUST_FL=$OSDIR/bin/Robust.ini
    if [ -f $ROBUST_FL ]; then   # [ ] 内は空白とタブ
        REST_USER=`grep "^[  ]*ConsoleUser[  ]*=[  ]*" $ROBUST_FL|¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        REST_PASS=`grep "^[  ]*ConsolePass[  ]*=[  ]*" $ROBUST_FL|¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        REST_PORT=`grep "^[  ]*ConsolePort[  ]*=[  ]*" $ROBUST_FL|¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        if [ "$REST_PORT" = "0" ]; then
            REST_PORT=`grep "^[  ]*port[  ]*=[  ]*" $ROBUST_FL |¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        fi
        if [ "$REST_USER" = "" -o "$REST_PASS" = "" -o "$REST_PORT" = "" ]; then
            REST_CNSL="no"
        else
            REST_OPT="-console=basic -host=localhost -port=$REST_PORT ¥
                                     -user=$REST_USER -pass=$REST_PASS"
        fi
    else
        REST_CNSL="no"
    fi
fi


start() {
    cd $OSDIR/bin
    echo "OpenSim R.O.B.U.S.T. Server Start."
    if [ "$REST_CNSL" = "yes" ]; then
        $MONO $OSDIR/bin/Robust.exe -console=rest 1>/dev/null 2>&1 &
    else
        screen -dmS opensim_robust $MONO $OSDIR/bin/Robust.exe
    fi

    echo "OpenSim Money Server Start."
    screen -dmS opensim_bank $MONO $OSDIR/bin/OpenSim.Grid.MoneyServer.exe
}


stop() {
    screen -S opensim_bank -p 0 -X stuff $'quit¥n'
    echo "OpenSim Money Server Stoped."

    if [ "$REST_CNSL" = "yes" ]; then
        cd $OSDIR/bin
        echo -e "shutdown" |¥
           $MONO $OSDIR/bin/OpenSim.ConsoleClient.exe $REST_OPT 1>/dev/null 2>&1
    else
        screen -S opensim_robust -p 0 -X stuff $'quit¥n'
    fi
    echo "OpenSim R.O.B.U.S.T. Server Stoped."
}


restart() {
    stop
    sleep $SLEEPTM
    start
}
```

```
case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  restart|reload)
    restart
    ;;
  *)
    echo $"Usage: $0 {start|stop|restart}"
    exit 1
esac

exit $?
```

図 8.10 RestConsole を利用した ROBUST, マネーサーバのバックグラウンド起動用スクリプト

```
#!/bin/bash

OSDIR=/usr/local/opensim                  # OpenSim をインストールしたディレクトリを指定
PRGFL=/etc/init.d/opensim_region_rest     # このスクリプトのフルパス
MONO=/usr/local/bin/mono                  # mono へのパス
SLEEPTM=15                                # リスタート時のスリープ時間
CHKTM=30                                  # 監視間隔
REST_CNSL="yes"                           # RestConsole を使用するかどうか
export MONO_THREADS_PER_CPU=256
ulimit -s 262144

SCRNID=opensim_region
PIDFL=/var/run/opensim_region_shell.pid


if [ "$REST_CNSL" = "yes" ]; then
    OPENSIM_FL=$OSDIR/bin/OpenSim.ini
    if [ -f $ROBUST_FL ]; then   # [ ] 内は空白とタブ
        REST_USER=`grep "^[  ]*ConsoleUser[  ]*=[  ]*" $OPENSIM_FL|¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        REST_PASS=`grep "^[  ]*ConsolePass[  ]*=[  ]*" $OPENSIM_FL|¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        REST_PORT=`grep "^[  ]*console_port[  ]*=[  ]*" $OPENSIM_FL|¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        if [ "$REST_PORT" = "0" ]; then
            REST_PORT=`grep "^[  ]*http_listener_port[  ]*=[  ]*" $OPENSIM_FL |¥
                   awk -F"[  ]*=[  ]*" '{print $2}'|awk -F"[  ]" '{print $1}'`
        fi
        if [ "$REST_USER" = "" -o "$REST_PASS" = "" -o "$REST_PORT" = "" ]; then
            REST_CNSL="no"
        else
            REST_OPT="-console=basic -host=localhost -port=$REST_PORT ¥
                                     -user=$REST_USER -pass=$REST_PASS"
        fi
    else
        REST_CNSL="no"
    fi
fi


start() {
    cd $OSDIR/bin
    echo "OpenSim Region Server Start."
    if [ "$REST_CNSL" = "yes" ]; then
        $MONO $OSDIR/bin/OpenSim.exe -console=rest 1>/dev/null 2>&1 &
    else
        screen -dmS $SCRNID $MONO $OSDIR/bin/OpenSim.exe
    fi
}
```

```
stop() {
    if [ "$REST_CNSL" = "yes" ]; then
        cd $OSDIR/bin
        echo -e "shutdown root" |¥
           $MONO $OSDIR/bin/OpenSim.ConsoleClient.exe $REST_OPT 1>/dev/null 2>&1
    else
        screen -S $SCRNID -p 0 -X stuff $'quit¥n' 1> /dev/null 2>&1
    fi
    echo "OpenSim Region Server Stoped."
}


kill_loop() {
    PID=`cat $PIDFL 2> /dev/null`
    if [ "$PID" != "" ]; then
       kill -9 $PID 2> /dev/null
       rm -f $PIDFL
    fi
}


loop_check() {
    kill_loop
    echo $$ >| $PIDFL
    while [ "" = "" ]; do
        sleep $CHKTM
        if [ "$REST_CNSL" = "yes" ]; then
            CHECK=`ps ax|grep OpenSim.exe | grep "console=rest"`
        else
            CHECK=`ps ax|grep SCREEN |grep $SCRNID`
        fi
        if [ "$CHECK" = "" ]; then
            start
        fi
    done
}


case "$1" in
  start)
    start
    /bin/bash $PRGFL check &
    ;;
  stop)
    kill_loop
    stop
    ;;
  restart|reload)
    kill_loop
    stop
    sleep $SLEEPTM
    start
    /bin/bash $PRGFL check &
    ;;
  check)
    loop_check
    ;;
  *)
    echo $"Usage: $0 {start|stop|restart}"
    exit 1
esac

exit $?
```

図 8.11 RestConsole を利用したリージョンサーバのバックグラウンド起動用スクリプト

9. FreeSwitchを利用したボイスチャット

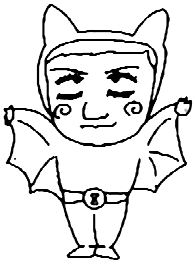
この章では，OpenSimでボイスチャットを行うための，SIPサーバに関する設定の解説を行います．

そろそろ,文字のチャットも飽きてきちゃったな～

音声チャットでもやってみようか？

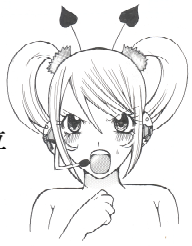
すごい，できるの？！

もちろん,SecondLife互換だからね～



残念ながらまたソースコードからコンパイルするけどね．
今回はSIP+VoIPという音声通信技術を外部のプログラムで行うよ．

ぎゃぁぁぁ,またかーーーーーー(泣



9.1 FreeSwitchのインストール

OpenSimでは**FreeSwitch**（SIPサーバ）を使用したボイスチャットをサポートしています. FreeSwitchは多機能なSIPサーバで,’10 7/20 現在の最新バージョンは v1.0.6ですので，これを使用します.

http://files.freeswitch.org/ よりfreeswitch-1.0.6.tar.gz をダウンロードし，図9.1に従ってインストールを行います． なお,FreeSwitchはOpenSimと同一のサーバで起動することも,別のリモートマシンで起動しすることも可能です.

FreeSwitch サイトURL:　　http://www.freeswitch.org/
FreeSwitch ダウンロードURL:　http://files.freeswitch.org/

図9.1での modules.confの設定では，**xml_int/mod_xml_curl**の一行を有効にします（図9.2).

make installが正常に終了すれば，FreeSwitchは /usr/local/freeswitch以下にインストールされます.

```
# tar zvfx freeswitch-1.0.6.tar.gz
# cd freeswitch-1.0.6
# vi modules.conf                    (図 9.2 参照)
# ./configure
# make
# make install
```

図 9.1 freeswitch のインストール （freeswitch-1.0.6 の場合）

9.2 FreeSwitchの設定

FreeSwitchの設定ファイルは /usr/local/freeswitch/conf 以下にインストールされます.OpenSimでFreeSwitchを使用するために,これのファイルについて若干の設定変更を行います.以下の項で変更する（/usr/local/freeswitch以下の）ファイル名とその変更箇所を示します.

```
........
xml_int/mod_xml_rpc
xml_int/mod_xml_curl
xml_int/mod_xml_cdr
........
```

図 9.2 modules.conf の変更点 （太字の部分が変更箇所）

9.2.1 conf/autoload_configs/modules.conf.xml

modules.conf.xmlにおいて,mod_xml_curlモジュールの読み込みを有効にするために図9.3のように **<load module="mod_xml_curl"/>** の部分を有効にします.

```
........
<!-- XML Interfaces -->
<!-- <load module="mod_xml_rpc"/> -->
<load module="mod_xml_curl"/>
<!-- <load module="mod_xml_cdr"/> -->
........
```

図 9.3 conf/autoload_configs/modules.conf.xml の変更箇所 （太字の部分が変更箇所）

9.2.2 conf/autoload_configs/xml_curl.conf.xml

mod_xml_curl.xmlファイルでは，既存のファイルの内容を変更して，図9.4のようなファイルを作成します．ここで，<bindings>.....</bindings> が一台のROBUSTサーバに対応します．<binding name=....>の部分は適当な名前（見出し）を付けて置き換えます（名前は被らないようにします)．

[ROBUST Serve URL]にはROBUSTサーバに接続するためのURLを記述します．例えば ROBUSTサーバが 202.26.159.211 の場合，このURL部分は**http://202.26.159.211:8002** となります．なお，name="gateway-url" のbindings= は最初が **directory** で2番目が**dialplan** であるので間違えないように注意してください．

また，**freeswitch:password** はFreeSwitchのユーザ名とパスワードで，リージョンサーバのRobust.iniの[FreeSwitchService]セクションに，ここで指定したものと同じものを指定しなければなりません．

```
<configuration name="xml_curl.conf" description="cURL XML Gateway">

  <bindings>
    <binding name="name1">
       <param name="gateway-url" value="[ROBUST Server URL]/api/freeswitch-config"
bindings="directory"/>
      <param name="gateway-credentials" value="freeswitch:password"/>
     <param name="disable-100-continue" value="true"/>
    </binding>
    <binding name="name2">
       <param name="gateway-url" value="[ROBUST Server URL]/api/freeswitch-config"
bindings="dialplan"/>
      <param name="gateway-credentials" value="freeswitch:password"/>
     <param name="disable-100-continue" value="true"/>
    </binding>
  </bindings>

</configuration>
```

図 9.4 conf/autoload_configs/xml_curl.conf.xml ファイル （太字の部分は環境に合わせて書き換える）

9.2.3 conf/autoload_configs/conference.conf.xml

OpenSimでは，保留のための音楽が有効になっていると，不都合が生じる場合がありますので，conference.conf.xmlファイルの中で，図9.5の太字の部分を全てコメントアウトします．このFreeSwitchをOpenSim以外で使用しないのならば，この部分は削除してしまっても構いません．

```
........
<!-- File to play if you are alone in the conference -->
<!--<param name="alone-sound" value="conference/conf-alone.wav"/>-->
<!-- File to play endlessly (nobody will ever be able to talk) -->
<!--<param name="perpetual-sound" value="perpetual.wav"/>-->
<!-- File to play when you're alone (music on hold)-->
<!--<param name="moh-sound" value="$${hold_music}"/>-->
<!-- File to play when you join the conference -->
<!--<param name="enter-sound" value="tone_stream://%(200,0,500,600,700)"/>-->
<!-- File to play when you leave the conference -->
<!--<param name="exit-sound" value="tone_stream://%(500,0,300,200,100,50,25)"/>-->
<!-- File to play when you ae ejected from the conference -->
<!--<param name="kicked-sound" value="conference/conf-kicked.wav"/>-->
........
```

図 9.5 conf/autoload_configs/conference.conf.xml の変更箇所 （太字の部分が変更箇所）

9.3 リージョンサーバ側の設定

0.7.2では FreeSwitchサーバの設定は Robust.ini に移行しました. OpenSim.ini では図9.6のようにして Robustサーバを指定します.

```
[FreeSwitchVoice]
    Enabled = true
    LocalServiceModule = OpenSim.Services.Connectors.dll:RemoteFreeswitchConnector

    FreeswitchServiceURL = http://[Robust Server URL]:8004/fsapi
```

図 9.6 bin/OpenSim.ini の[FreeSwitcVoice] セクション

9.4 Robust サーバ側の設定

Robust.ini では図9.7のようにして FreeSwitchサーバを指定します(ここで,202.26.159.196がFreeSwitchサーバのIPアドレス).

各項目の詳細については, 付録Cの「[FreeSwitchService] セクション」を参考にしてください.

```
[FreeswitchService]
    LocalServiceModule = "OpenSim.Services.FreeswitchService.dll:FreeswitchService"

    ;; The IP address of your FreeSWITCH server.
    ;; This address must be reachable by viewers.
     ServerAddress = 202.26.159.196

    ;; By default, this is the same as the ServerAddress
     Realm = 202.26.159.196

    ;; By default, this is the same as the ServerAddress on port 5060
     SIPProxy = 202.26.159.196:5060

    ;; Default is 5000ms
    DefaultTimeout = 5000

    ;; The dial plan context.  Default is "default"
    Context = default

    ;; Currently unused
    UserName = freeswitch

    ;; Currently unused
    Password = password

    ;; The following parameters are for STUN = Simple Traversal of UDP through NATs
    ;; See http://wiki.freeswitch.org/wiki/NAT_Traversal
    ;; stun.freeswitch.org is not guaranteed to be running so use it in
    ;; production at your own risk
     EchoServer = 202.26.159.196
    EchoPort = 50505
    AttemptSTUN = false
```

図 9.7 bin/Robust.ini の[FreeswitchService] セクション （太字の部分が変更箇所）

9.5 FreeSwitch サーバの起動

全ての設定が終了したら，図9.8のコマンドでFreeSwitchを起動させ，リージョンサーバを再起動します（順番はどちらが先でも構いません）．リージョンサーバより前に，FreeSwitchが起動した場合は **Received HTTP error 0 trying to fetch** のエラーメッセージが表示される場合があります．また リージョンサーバでボイスチャットが有効になっていない場合は，**Received HTTP error 404 trying to fetch** のエラーメッセージが表示される場合がありますが，これらのエラーメッセージは基本的に無視しても構いません（**root tag missing** のエラーメッセージも無視しても構いません）．

FreeSwitch をバックグラウンド起動するには **-nc** オプションを指定して起動します．また，バックグラウンド起動したFreeSwitchを停止させるには **-stop** オプションを指定して freeswitch コマンドを起動します（図9.9）．

なお，OpenSimと同様に FreeSwitchの /etc/init.d/ 用の起動スクリプトを「4.1.2項」のopensim.nsl.patch-0.7の configディレクトリに用意しています．もし，マシン起動時に自動的にFreeSwitchを起動したいのであれば，/etc/init.d/ にfreeswitchをコピー後，図9.10のようにしてシンボリックリンクを張ります．

9.6 ボイスチャットモード

OpenSim上でボイスチャットを行うには土地毎に「ボイス」の設定も行わなければなりません．しかしながら，SecondLife Viewerのv1.23では，リージョン（エステート）の管理者であっても，「世界」→「土地情報」→「メディア」でのボイスの設定が図9.11のようになり，有効にすることができません（OpenSimのバグなのか，Viewerのバグなのかは不明）．

SecondLife Viewerのv1.22やそれらを元にした Meerkat Viewerではこの設定が可能なので（図9.12），面倒でもこれらのViewerを立ち上げて，「不動産空間チャンネル」または「プライベート空間チャンネル」を有効にしなければなりません（設定は一度だけで良い）．なお，この場合，「世界」メニュー→「地域/不動産」→「不動産」の「ボイスチャットを許可」のチェックボックスは何の意味も持ちません（チェックを外してもボイスチャットは行えます）．

```
# /usr/local/freeswitch/bin/freeswitch
```

図9.8 FreeSwitchの起動コマンド

```
# /usr/local/freeswitch/bin/freeswitch -nc          （バックグラウンド起動）
# /usr/local/freeswitch/bin/freeswitch -stop        （バックグラウンドプロセスの停止）
```

図9.9 FreeSwitchのバックグラウンド起動とその停止

```
# ln -s ../init.d/freeswitch /etc/rc3.d/S99freeswitch
```

図9.10 FreeSwitchの自動起動スクリプトの準備例 （ただし， Run Level 3, 起動順序99の場

また, ボイスを有効にする手段として, OpenSimのデータベースを直接書き換えることもできます. ボイスチャットの設定は landテーブルの**LandFalgs**項目に設定されています (図9.13). ここで「不動産空間チャンネル」のフラグの値は 0x40000000,「プライベート空間チャンネル」のフラグの値は 0x20000000ですので, 既存の値にこれらの値を足し算して, レコードをアップデートします(図9.14). ただし, この場合はレコードのアップデート後にリージョンサーバの再起動が必要です.

なお, 最新バージョンのXoopenSim/ModlosにはLandFalgs項目の書き換えの機能があり (10.7.3項参照), リージョンサーバを再起動する必要はありますが, WEB上からボイスの上記のLandFlags設定を行うことが可能です.

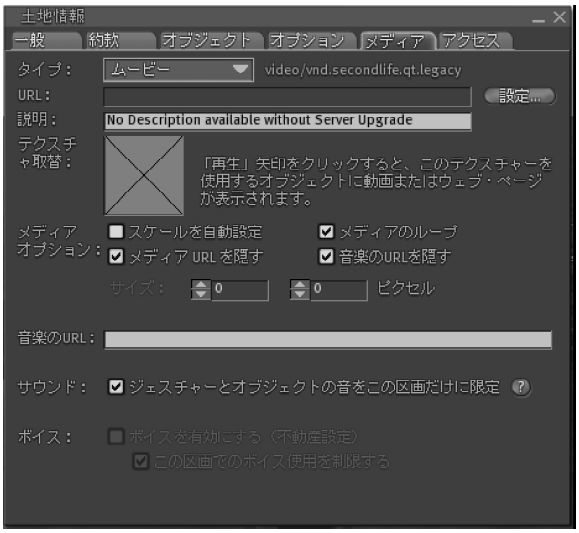


図 9.11 v1.23.5 の「土地情報」設定画面.
ボイス設定が有効にならない.

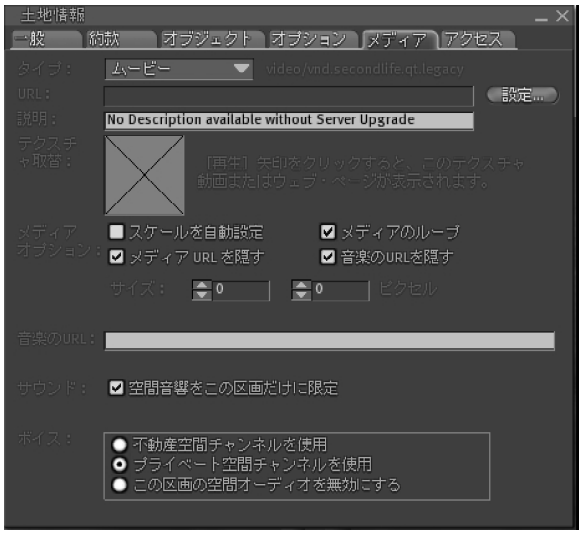


図 9.12 v1.22.11 の「土地情報」設定画面.
ボイス設定が可能.

```
$ /usr/local/mysql/bin/mysql -u root -p
Enter password: SQLPass    (MySQL の root のパスワード)

mysql> use opensim_db;
mysql> select Name,LandFlags from land;
+-----------------+------------+
| Name            | LandFlags  |
+-----------------+------------+
| Your Parcel     |  134258779 |
| Your Parcel     |  134258779 |
| Your Parcel     |  134258779 |
+-----------------+------------+
```

図 9.13 land テーブルの LandFlags の内容.

```
mysql> update land set LandFlags =  '671129691'
```

図 9.14 land テーブルの全ての LandFlags に 671129691 （=134258779 + 0x20000000）を設定.
リージョンサーバを再起動すれば, 全ての土地は「プライベート空間チャンネル」が使用可能となる.

10. WEBインターフェイス XoopenSim

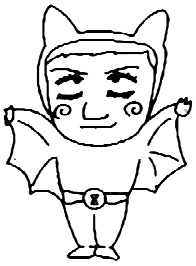
XoopenSimは opensimwi redux 0.32 を元に，NSLによって改造された Xoops Cube用OpenSim WEBインターフェイス・モジュールです．Xoops Cubeの多数のモジュールと併用することができ，多彩なOpenSim用のWEBサイトを構築することが可能です．また，Flotsamグループ機能や osprofileの機能もインクルードしているので，簡単な設定でOpenSimにグループ機能やユーザプロフィール機能を追加することも可能です．

XoopenSimはXoops Cubeのモジュールであるため，Xoops Cubeが既に稼動済みの環境では極めて簡単にインストールすることができます．一方 Xoops Cubeがインストールされていない環境ではXoops Cubeのインストールから始めなければならないため，多少手間がかかります．

この章では，Apache(WEBサーバ)，PHPのインストールから始めて，XoopenSimの使用開始までの手順の解説を行います．なお，XoopenSim（Xoops Cube）の稼動するマシンは，OpenSimと同じマシンでも良いし，別のマシンでも構いません．

XoopenSimってなに？

XoopsCubeにopensimwiを移植したやつよ～
XoopsとOpenSimをかけて作った名前ね



なかなかいい名前だろう！
もうひとつMoodleというCMSに移植したModlos
というのもあるぞ！

こいつがあると，ユーザー登録とかを簡単に行うことも出来るんだ～
その他にも色んなXoopsCubeのModuleを導入することによって色々な機能を提供できるんだよ！

なんかすごく便利なのはわかった！

10.1 Apache のインストール

10.1.1 Apache のコンパイル

Apache はLinuxで最も広く使用されているWEBサーバです. 現在はv2.3のαバージョンも存在しますが, ここでは安定版である v2.2の最新版を利用します.

サイトURL: http://httpd.apache.org/
ダウンロードURL: http://httpd.apache.org/download.cgi#apache22

ダウンロードサイトからソースコードをダウンロードしたら,図10.1に従ってインストールを行います. 正しくインストールされれば /usr/local/apacheディレクトリが作成されます.

```
# tar xfvz httpd-2.2.15.tar.gz
# cd httpd-2.2.15
# ./configure --enable-ssl --with-ssl=/usr/local/ssl --enable-suexec ¥
              --enable-auth-digest --enable-rewrite --enable-dav ¥
              --enable-so --prefix=/usr/local/apache
# make
# make install
```

図10.1 Apache2.2 のインストール手順 (httpd-2.2.15.tar.gz の場合)

10.1.2 起動スクリプトの準備

Apacheのインストールが成功したら, 図10.2に従って起動スクリプトを用意し, 自動起動するように適当な Run Level (2.4.1項で決めたRun Levelで図10.2の例では**3**) でシンボリックリンクを張っておきます.

図10.2 の例では, Apacheの起動順序を**99**に設定していますが, MySQLの起動優先順位より後であれば(数が大きければ)幾つでも良いでしょう. このように起動スクリプトを用意した場合,図10.3のコマンド群で Apacheの手動による起動, 停止, 再起動が可能となります.

```
# cp /usr/local/apache/bin/apachectl /etc/init.d/apache
# chmod a+rx /etc/init.d/apache
# ln -s ../init.d/apache /etc/rc3.d/S99apache
```

図10.2 起動スクリプトの準備

```
# /etc/init.d/apache start     起動
# /etc/init.d/apache stop      停止
# /etc/init.d/apache restart   再起動
```

図10.3 起動スクリプトを利用したApacheのコントロール

10.1.3 ドキュメントルートの準備

CentOSには **apache**ユーザとグループが予め登録されていますので, これをApacheの実効ユーザ, 実効グループとします.他のディストリビューションで適当な実行ユーザやグループが存在しない場合には, adduserコマンドなどで実効ユーザとグループの作成を行います(図10.4). また, CentOSでの apacheユーザのホームディレクトリは **/var/www** ですので, **/var/www/htdocs** をApacheのドキュメントルートとします.

ドキュメントルート /var/www/htdocs を作成し, 適切なパーミッションを設定する手順を図10.5に示します. なお, セキュリティについて細心の注意を払うなら, ドキュメントルートのオーナは root に戻し, かつ root にしかドキュメントを書き込めないようにして置くべきでしょう. ただし,

そのようにした場合，その後の設定が非常に難しくなるため，ここではその設定は行わないこととします．一方で，よくWEBサイトの入門ページなどで，ディレクトリやファイルのパーミッションを 777 (rwxrwxrwx) にする例を目にしますが，非常に危険ですので絶対にやめましょう！

```
# adduser apache -d /var/www -s /sbin/nologin
```

図 10.4 apache ユーザとグループの作成例 （ユーザ・グループ番号は自動割り振り）

```
# cd /var/www
# ¥rm -rf *     （不要な既存ファイルの削除．ドットファイル以外の全てのファイルが削除されるので注意）
# mkdir htdocs
# chown -R apache.apache .
# chmod g+rwxs,o-rwx htdocs
# chmod g+rwxs,o-rwx .
```

図 10.5 /var/www/htdcs の作成とパーミッションの設定

10.1.4 conf，logsディレクトリのパーミッション

Apacheサーバの実効ユーザである apacheが設定ファイルやログファイルを読めるように設定しなければなりません．また必要なら，一般ユーザにはそれらのファイルは読めないように設定します(図10.6)．なお，chmodコマンドではパーミッションを数字で指定することも可能ですが，初心者の場合は間違い易いので，キャラクタで指定した方が良いでしょう．

```
# cd /usr/local/apache
# chgrp -R apache conf logs
# chmod g+rx-w,o-rwx conf logs
# cd conf
# chmod g+r-w,o-rwx * */*
# chmod g+x extra original
```

図 10.6 conf, logsディレクトリに適切なパーミッションを設定

10.1.5 設定ファイル

/usr/local/apache/conf 以下の図10.7に示された主要な設定ファイルの編集を行います．通常の運用での標準的な変更点をWEB上に記述してありますので，各ファイルについてWEB上のファイルを参考にしながら，自分の環境に合わせて変更を行ってください．

WEB上のファイルでは，変更すべき箇所が色の付いた文字で表されています．WEB上のファイルで青字の部分は使用する環境に合わせて変更します．また赤字の部分はその通りに変更します(各自の環境や運用ポリシーによっては変更してもかまいません)．なお，緑字の部分は，今回の設定では使用しませんので何も変更しないでください (オプション部分)．

今回変更した項目の意味については，後で，Apacheの専門書などで確認しておくと良いでしょう．

```
httpd.conf
  http://www.nsl.tuis.ac.jp/etc/setting/apache2.2/httpd.conf.html
extra/httpd-userdir.conf
  http://www.nsl.tuis.ac.jp/etc/setting/apache2.2/extra/httpd-userdir.conf.html
extra/httpd-languages.conf
  http://www.nsl.tuis.ac.jp/etc/setting/apache2.2/extra/httpd-languages.conf.html
```

図 10.7 設定を行うべきファイルと設定箇所が記述されたWEB ページ

10.1.6 Apacheの起動と動作確認

全ての設定が終了したら，ドキュメントルートにテスト用のWEBページを作成し(図10.8)，Apacheを手動で起動して，ブラウザでテスト用ページが参照できるかどうか確認します．

なお，Apacheを手動起動したときに図10.9のような，「OpenSSLの共有ライブラリが見つからない」というエラーが出た場合は，「3.3節」の図3.5，3.6の共有ライブラリの設定を忘れている可能性があります．図3.5，3.6を参考に共有ライブラリの設定を行ってください．

```
# cd /var/www/htdocs
# vi test.html
# /etc/init.d/apache start
```

図 10.8 テスト用ページの作成とApache の手動起動

```
/usr/local/apache/bin/httpd: error while loading shared libraries:
libssl.so.0.9.8: cannot open shared object file: No such file or directory
```

図10.9 共有ライブラリの読み込みエラー

10.2 PHP のインストール

10.2.1 PHPのコンパイル

Xoops CubeはPHPで記述されているため，PHPをインストールする必要があります．なお，PHP5.3系ではXoops Cubeは作動しませんので，ここではPHP5.2系を使用します．

PHP サイトURL: http://jp.php.net/
PHP ダウンロードURL: http://jp.php.net/downloads.php

PHP5.2.xのソースコードをダウンロードし，図10.10のようにしてコンパイル，インストールを行います（図10.10）．

```
# tar zxfv php-5.2.13.tar.gz
# cd php-5.2.13
# ./configure --with-mysql=/usr/local/mysql --with-pdo-mysql=/usr/local/mysql ¥
              --with-mysqli=/usr/local/mysql/bin/mysql_config ¥
              --with-apxs2=/usr/local/apache/bin/apxs ¥
              --enable-mbstring --with-zlib=/usr --with-iconv ¥
              --with-openssl=/usr/local/ssl --with-curl --with-xmlrpc ¥
              --with-gd --with-jpeg-dir --with-png-dir
# make
# make test
# make install
```

図 10.10 php5.2 のインストール手順 (php-5.2.13.tar.gz の場

10.2.2 PHPの設定

PHPの設定ファイル(php.ini)を /usr/local/libに設置するために，PHPをコンパイルしたディレクトリにおいて図10.11のコマンドを実行します．

次に，下記ディレクトリのパーミッションをチェックし，必要なら一般ユーザが読めるようにしておきます．図10.12 に/usr/local/lib/php 以下のファイルパーミッションのを設定する場合の操作手順の例を示します．

- /usr/local/lib/php
- /usr/local/man/man1
- /usr/local/include/php

```
# cp php.ini-dist /usr/local/lib/php.ini
# chmod o+r /usr/local/lib/php.ini
```

図 10.11 php.ini の設置 （PHP をコンパイルしたディレクトリで実行すること）

```
# cd /usr/local/lib
# chmod -R o+r php
# find php -type d | xargs chmod o+x
```

図 10.12 /usr/local/lib/php のファイルパーミッションの設定手順例

10.2.3 Apache との連携の確認

PHPが正常にインストールされ,Apacheとの連携が可能かどうかを以下の手順により確認します.

1. /usr/local/apache/modules/libphp5.so ができていることを確認する.
2. /usr/local/apache/conf/httpd.confの L53あたりに php5のモジュール読み込みの命令(LoadModule php5_module modules/libphp5.so) が一行追加されているのを確認する.
3. /etc/init.d/apache restart で Apache を再起動し，ps ax コマンドで正常に作動していることを確認する.

10.3 MySQL のインストールと設定

Xoops Cube のために MySQL をインストールし，設定を行います．既にインストール済みである OpenSim 用の MySQL サーバを Xoops Cube の DB として併用してもかまいません（新規インストールについては 3.8 節を参照）.

MySQL の準備ができたら，図 10.13 のコマンドで Xoops Cube 用のユーザを登録します．図 10.13 では Xoops Cube のデータベース名を **xoops_db**，ユーザ ID を **xoops_user**，パスワードを **xoops_pass** としています.

10.4 Xoops Cube のインストール

10.4.1 Xoops Cube の展開

Xoops Cube のサイトから Xoops Cube Legacy の最新版をダウンロードします.’10 7/20 の時点での最新バージョンは 2.1.8 で，ソフトウェアのファイル名は **Package_Legacy_2_1_8.zip** です．これを /var/www で展開し，ファイルパーミッションの設定とシンボリックリンクの作成を行います（図 10.14）.

Xoops Cube サイト URL: http://xoopscube.jp/

```
$ /usr/local/mysql/bin/mysql -u root -p
Enter password: SQLPass     (MySQL の root のパスワード)

mysql> create database xoops_db default character set utf8;
mysql> grant all on xoops_db.* to xoops_user identified by 'xoops_pass';
mysql> flush privileges;
mysql> exit
```

図 10.13 Xoops Cube 用データベースの作成と権限の設定

次に Xoops Cube を UTF-8 で使用するための言語ファイルをコピーします（図 10.15）．現バージョンの Xoops Cube の標準の日本語コードは EUC であるため，UTF-8 の日本語コードは拡張機能となっています．一方，OpenSim では標準のコード系が UTF-8 となっており，OpenSim で日本語を扱う場合を考えて，Xoops Cube のコード系も UTF-8 とします.

```
# cd /var/www
# unzip [DL]/Package_Legacy_2_1_8.zip
# chown -R apache.apache Package_Legacy
# chmod -R o-rwx Package_Legacy
# ln -s ../Package_Legacy/html htdocs/xoops
```

図 10.14 Xoops Cube Legacy の展開 （Package_Legacy_2_1_8.zip の場合）
ただし [DL] は Package_Legacy_2_1_8.zip をダウンロードしたディレクトリを表す

```
# cd /var/www/Package_Legacy
# cp -Rdp extras/extra_languages/ja_utf8/html/* html
```

図 10.15 UTF-8 用の言語ファイルのコピー

10.4.2 Xoops Cubeのインストール

Xoops Cube をインストールするには，WEB ブラウザで http://[サーバの IP または FQDN]/xoops/ にアクセスします．なお，この http://[サーバの IP または FQDN]/xoops/ は以降 XOOPS_URL と記述します．

XOOPS_URL にアクセスすると Xoops Cube のインストール画面が表示されますので，言語として ja_utf8 を選択し（日本語のコード系を EUC にする場合には japanese を選択），「Next」ボタンをクリックします（図 10.16）．

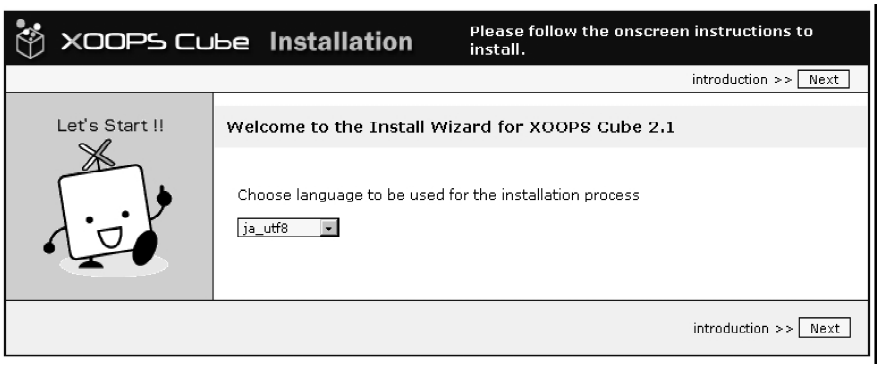


図 10.16 使用言語の選択 （ja_utf8 を選択）

その後，画面を確認しながら，「次へ」のボタンをクリックして行きます．データベースの設定画面が表示されたら，データベースのホスト名，データベースユーザ名，データベースパスワード，データベース名を設定します．データベースユーザ名，データベースパスワード，データベース名は「10.3 MySQL のインストールと設定」で設定したもの（例ではそれぞれ，xoops_user， xoops_pass，xoops_db）を指定します（図 10.17）．

入力が終わったら「次へ」のボタンをクリックします．入力に間違いが無ければ，特にエラーは発生しないはずですので，管理ユーザの設定画面が表示されるまで，画面を確認しながら「次へ」のボタンをクリックして行きます．

管理ユーザの設定画面（図 10.18）が表示されたら，管理者ユーザ名（何でも良いが通常は admin などを指定），管理者メールアドレス，管理者パスワードを入力し，「次へ」のボタンをクリックします．「次へ」のボタンを数回クリックすると，ログイン画面（図 10.20）が表示さます．

ここで管理者としてログインする前に，一旦Linuxのコンソール画面に戻って，パーミッションの再設定を行うと良いでしょう（図 10.19）．図 10.19 では，第 3 者に install スクリプトを実行されないように installディレクトリを削除していますが，別の名前に変更して Apacheサーバが実行でき

図10.17 データベースの設定

図 10.18 管理ユーザの設定

```
# cd /var/www/htdocs/xoops/
# ¥rm -r install
# chmod -R o-rwx *
# chmod a-w mainfile.php
```

図10.19 パーミッションの再設定

図 10.20 ログイン画面

図10.21 標準モジュールのインストール画面

ないようなパーミッションに変更しても良いでしょう(こうすれば必要な時,後でもう一度実行できます).

パーミッションの再設定が終わったら,WEBブラウザに戻って,図10.20のログイン画面から,図10.18で設定した管理ユーザのアカウントを使用してログインします.ログインが成功すると図10.21のような標準モジュールのインストール画面になりますので,そのまま「インストール」ボタンをクリックし,全てのモジュールをインストールします.

全て旨く行けば,図10.22のような Xoops Cubeの初期画面が表示されます. Xoops Cubeではテーマの設定によりサイトのデザインを変更することができ,また様々なモジュールを組み込むことにより機能の拡張を行うことも可能です.ただし,それらの詳細についてはこのマニュアルの範囲を超えますので,ここでは解説は行いません.専用の解説書か解説WEBサイトをご覧ください.

Xoops Cube解説 参考URL(うさぎにもできるXoops Cube入門): http://usadeki.jp/modules/pico/

図10.22 Xoops Cubeの初期画面

10.4.3 Xoops Cubeの最低限の設定と必須モジュール

図10.22で「管理者メニュー」をクリックすると,管理モードに移行します.管理モードで「互換モジュール」メニューの「全般設定」(図10.23)をクリックすると全体の設定画面に移動します.初期状態ではデバッグモードが「PHPデバッグ」になっていますが,デバッグモードがONになっていると,デバッグメッセージのために OpenSimからのXML RPCが正常に作動しなくなるため,デバッグモードは必ず「オフ」にしておいてください(図10.24).

また,ALTSYSモジュールは外部モジュールではありますが,システムを管理する上で非常に便利なモジュールですし,他のモジュールをインストールするときに必須となる場合もありますので,必ずインストールしておいた方が良いでしょう.

ALTSYS ダウンロードURL: http://xoops.peak.ne.jp/md/mydownloads/singlefile.php?lid=76

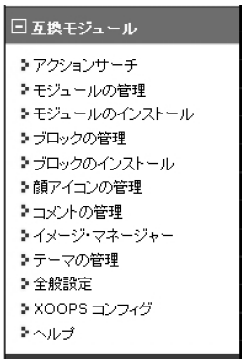


図10.23 互換モジュールメニュー

上記URLからダウンロードしたファイル(altsys-0.7.zip)を図10.25のようにして,展開と設定を行います. なお ALTSYSモジュールのインストールでは main.phpの編集も必要となりますが,その中でXOOPS_TRUST_PATHを図10.26のように設定します.

この状態でWEBブラウザを使用してXoops Cubeにアクセスし,管理モードの「互換モジュール」メニューから「モジュールのインストー

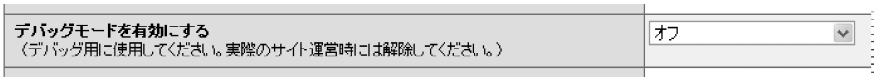


図10.24 全般設定のデバッグモード設定

ル」選択すると，図10.27のような ALTSYSのインストール画面になります．操作欄にある -ボタンをクリックすると確認が画面が表示されますので，そのまま「インストール」ボタンをクリックするとインストールが行われます．

```
# cd /var/www/Package_Legacy
# unzip [DL]/altsys-0.7.zip
# chown -R apache.apache html xoops_trust_path
# chmod -R o-rwx html xoops_trust_path
# cd html
# chmod u+w mainfile.php
# vi mainfile.php          （変更内容は図 10.26 を参照）
# chmod u-w mainfile.php
```

図 10.25 altsys の展開と設定 （altsys-0.7.zip の場合）
ただし [DL] は altsys-0.7.zip をダウンロードしたディレクトリを表す）

```
define('XOOPS_TRUST_PATH', '/var/www/Package_Legacy/xoops_trust_path');
```

図 10.26 図 10.25 での main.php の変更点 （太字の部分が変更箇所）

図10.27 ALTSYSのインストール画面

10.5 XoopenSim のインストール

NSLのサイトからXoopenSimの最新版をダウンロードし，Xoops Cubeの modulesディレクトリで展開します．展開後，必ずパーミッションを適切に設定します（図10.28）．

XoopenSim ダウンロードURL: http://www.nsl.tuis.ac.jp/xoops/modules/d3downloads/index.php?cid=5

展開後にWEBブラウザでXoops Cubeにアクセスし，ALTSYSの時と同様にXoops Cubeの管理モードの「互換モジュール」メニューから「モジュールのインストール」選択すると，図10.29のようなインストール画面になりますので，そのままインストールを行います．

```
# cd /var/www/htdocs/xoops/modules
# tar zxfv [DL]/xoopensim-1.20.tgz
# chown -R apache.apache xoopensim
# chmod -R o-rwx xoopensim
```

図 10.28 XoopenSim の展開とパーミッションの設定 （xoopensim-1.41 の場合）
ただし [DL] は xoopensim-1.41.tgz をダウンロードしたディレクトリを表す）

図 10.29 XoopenSim のインストール画面

10.6 XoopenSim の設定

XoopenSimが正しくインストールされると，管理モードのメニューに XoopenSimのメニューが追加されます．XoopenSimの管理メニューの項目は，「一般設定」，「ラストネーム管理」および「データベース更新」です．なお，これらについての簡単な解説が，下記の NSL の Wiki にもありますので．参考にしてください．

XoopenSim Wiki URL: http://www.nsl.tuis.ac.jp/xoops/modules/xpwiki/?XoopenSim

注意事項として，このモジュールはゲストでもアクセス可能にする必要があります．OpenSimからの XML RPC（主にOpenSimのグループ機能と土地売買の機能などのヘルパー機能）がゲストとしてアクセスしてくるので，ゲストアクセスが許可されていないと OpenSim からのこれらの XML RPC が正常に動作しなくなります．

モジュールへのアクセス管理は，管理モードでALTSYSモジュールの「ブロック管理」メニューから行います．即ち，ALTSYS のブロック管理画面で，XoopemSim を選択し，ゲストの「XoopenSim（x.xx）モジュールアクセス権限」と「Xoops OpenSim」にチェックを入れて「送信」ボタンをクリックすれば OK です（図 10.30）．

図10.30 ALTSYSのXoopenSim用ブロック管理画面

グリッド名	OpenSim のグリッド名.
SQLサーバ名	OpenSim の SQLサーバの FQDNか IPアドレス.
SQLデータベース名	OpenSim の SQLデータベースの名前.
SQLデータベースのユーザ名	OpenSim の SQLデータベースのユーザ名.
SQLデータベースのパスワード	OpenSim の SQLデータベースのパスワード.
Money サーバのURI	DTL Money サーバを使用する場合は，その URI(URL). (未使用)
ワールドマップのスタート位置 (X)	ワールドマップを表示する時の 中心 X座標の初期値.
ワールドマップのスタート位置 (Y)	ワールドマップを表示する時の 中心 Y座標の初期値.
マップ上のSIMのサイズ	ワールドマップ上のSIMのサイズ(px)のデフォルト値.
日付データのフォーマット	XoopenSimで使用する日付表示のフォーマットを指定する.
アバターの最大数	一人の Xoopsユーザが所有できるアバターの最大人数. 負数の場合は無制限.
ラストネーム管理	ラストネームを，予めデータベースに用意していた物に制限するかどうかの指定.
デフォルトのホームリージョン	アバター作成時のデフォルトのホームリージョン.
POST時に HTTPSを使用	パスワードなどの重要なデータの POST時に HTTPSを使用するかどうかを選択する. 既にサイト全体が HTTPSの場合は，選択する必要はない.
HTTPSの URL	HTTPSを使用する場合は，Xoops のモジュールの HTTPSでの URLを指定する. 省略した場合は，XOOPS_MODULE_URL の http: を https: に変換したものを使用する.
Flotsam グループデータベース読み込みアクセスキー	Flotsam Group Function を使用する場合に指定する読み込み用キー. OpnenSim.iniの [Groups]セクションの XmlRpcServiceReadKeyの値と一致させる必要がある.
Flotsam グループデータベースの書き込みアクセスキー	Flotsam Group Function を使用する場合に指定する読み込み用キー. OpnenSim.iniの [Groups]セクションの XmlRpcServiceWriteKeyの値と一致させる必要がある.
Xoopsユーザの名前からアカウント情報のページへのリンク	ページ上に Xoopsユーザの名前が表示された場合に，そのユーザのアカウント情報画面へのリンクを張るかどうかを指定する. なお，セキュリティ上の問題から，UserInfoProtectorなどを使用して，他のユーザのアカウント画面の表示を禁止している場合などは，ここを 「はい」 にすると煩わしいだけになる.
トップ (DB) ページのコンテンツ	XoopenSim のトップ (DB) ページのコンテンツ. HTMLタグ使用可能.
リージョンリストページのコンテンツ	XoopenSim のリージョンリストページのコンテンツ. HTMLタグ使用可能.
アバターリストページのコンテンツ	XoopenSim のアバターリストページのコンテンツ. HTMLタグ使用可能.
アバター編集ページのコンテンツ	XoopenSim のアバター編集ページのコンテンツ. HTMLタグ使用可能.
使用許諾の表示	アバター作成時に OpenSimの使用許諾を表示するかどうかの指定.
使用許諾の内容	OpenSim使用許諾の内容を記入する. HTMLタグ，BB Codeタグは使用不可.

表 10.31 XoopenSim の「一般設定」の項目の解説

また，OpenSim のリージョンサーバと Xoops Cube が同じマシンで動作している場合，PC の WEB ブラウザから両者に**同じ URL でアクセスする**と，OpenSim のリージョンサーバからの情報取得時に Xoops Cube のクッキーが自動的に付加および転送され，WorldMap などの一部の機能が正常に動作しなくなる場合があります. たとえ同じマシンであってもリージョンサーバとXoops Cubeは違うURLで接続するように設定を行わなければなりません (具体的にはDNSなどに違う名前で登録を行うと良いでしょう).

10.6.1 一般設定

XoopenSimの「一般設定」の画面は，XoopenSimを使用する場合に，必ず一番最初に設定しなければならない画面です（特にOpenSimのMySQLサーバの情報）．表10.31 にXoopenSimの「一般設定」の項目の一覧とその解説を示します．

10.6.2 ラストネーム管理

一般設定の「ラストネーム管理」で「はい」を選択した場合，この画面でアバターの作成（10.7.6項を参照）時に使用できるラストネームの「登録」,「一時使用停止」,「削除」が可能となります（図10.32）．

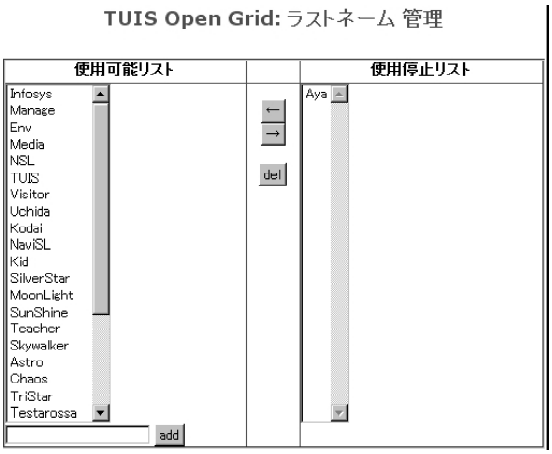


図 10.32 ラストネーム管理機能

10.6.3 データベース更新

データベース更新の機能は，OpenSim v0.6.xから v0.7に移行する場合にデータベースを更新する機能です．移行する場合は，0,7のROBUSTサーバを立ち上げて，アバターがログインする前にこの機能を実行します．v0.7から初めてXoopenSimを使用する場合には，この機能を実行する必要はありません．

10.7 XoopenSim の基本機能

10.7.1 データベースの状態表示

通常モードで XoopenSimのトップページにアクセスした場合，データベースの情報が表示されます（図10.33）．これはMySQLデータベースからの情報であり，状態が「オンライン中」であってもOpenSimグリッドがオンライン状態であるとは限りませんので注意してください．

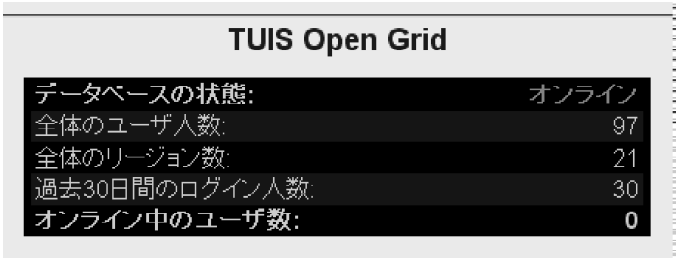


図 10.33 データベースの状態表示

10.7.2 ワールドマップ

XOOPS_URL/modules/xoopensim/?action=map にアクセスした場合,グリッドのワールドマップが表示されます（図 10.34）. また，各リージョン（SIM）の右下の *i* の部分をクリックすると，簡単なリージョン（SIM）の情報が表示されます.

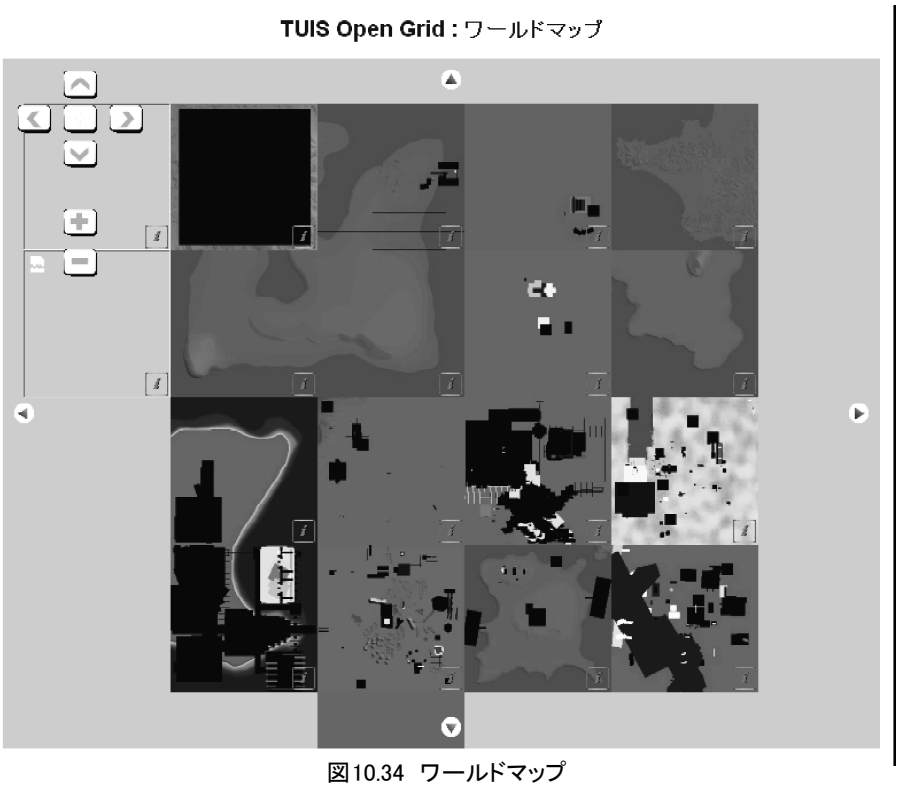


図10.34 ワールドマップ

10.7.3 リージョンリスト

XOOPS_URL/modules/xoopensim/?action=regions にアクセスした場合,グリッドのリージョンリストが表示されます(図10.35). またリスト中の管理者名をクリックするとそのアバターの情報ウィンドウが表示され（図10.36), リージョン名をクリックすると，そのリージョンの情報ウィンドウが表示されます（図10.37). ただし，ゲストユーザに対しては，リージョンが稼動しているサーバのIPア

TUIS Open Grid : リージョンリスト

No.	リージョン名	座標: X	座標: Y	管理者	ボイスチャネル	IP アドレス
21リージョン			ページ 1 / 1			
1	Chishirodai	1000	1001	Fumi Hax	プライベートチャネル	202.26.148.242
2	SandBox3	1001	1000	Fumi Hax	プライベートチャネル	202.26.148.246
3	Candy	1000	997	_	ボイス無効	202.26.159.201
4	Rin_1	997	1002	Nowsky NSL	プライベートチャネル	202.26.159.203
5	SandBox	1001	999	Fumi Hax	プライベートチャネル	202.26.148.247
6	Rin_2	997	1001	Nowsky NSL	プライベートチャネル	202.26.159.203
7	Welcome Sim	1000	999	Fumi Hax	プライベートチャネル	202.26.148.241
8	sirius_b4	998	1002	Fumi Hax	プライベートチャネル	202.26.148.245
9	Miku_2	998	999	Nowsky NSL	プライベートチャネル	202.26.159.204
10	Rigel_b3	1001	1001	Fumi Hax	プライベートチャネル	202.26.148.242
11	Kashiwanoha	1000	1002	Fumi Hax	プライベートチャネル	202.26.148.242
12	Tama_Univ	999	1002	Fumi Hax	プライベートチャネル	202.26.159.210
13	TUIS	1000	1000	Fumi Hax	プライベートチャネル	202.26.159.197

図10.35 リージョンリスト

ドレス情報およびリージョン情報とアバター情報のウィンドウへのリンクは表示されません.

リージョンの情報ウィンドウの中には, リージョン (正確にはエステート) のオーナと共にボイスチャットのモードも表示されます (図10.37). 本来, ボイスチャットのモードはパーセル毎に表示すべきものですが, パーセル単位での表示は非常に難しいため, ここで表示されるものはリージョン内の全パーセルのボイスモードの論理積 (つまり複数のパーセルの中で一番下のモード) となります.

リージョンのオーナとボイスチャットのモードはこのウィンドウで変更することも可能ですが, ボイスチャットのモードを変更した場合は, リージョン内の全てのパーセルのボイスチャットのモードが指定されたモードに変更されます. また, 変更されたボイスチャットのモードを有効にするためには, リージョンサーバを再起動する必要があります.

図 10.36 アバター情報

図 10.37 リージョン情報

10.7.4 アバターリスト

XOOPS_URL/modules/xoopensim/?action=avatars にアクセスするとアバターのリストが表示されます (図 10.38). ただし, このリストはゲストユーザには表示されません.

図 10.38 は Xoops Cube ユーザ **alice** でログインした場合に表示されるアバターリストです. リスト中の「オーナ」はそのアバタを作成した Xoops Cube ユーザを示しており, 自分の作成したアバタに限って, 「EDIT」リンクで情報を編集することが可能です (管理ユーザは全てのアバタを編集可能).

また, 「オーナ」が空白のアバタは, OpenSim の User サーバ側で作成したアバターであり, 現在 Xoops Cube ユーザと対応が付けられていないことを表します. ただし, 「OWNER」リンクをクリックして, OpenSim でのアバターのパスワードを入力すれば, そのアバターを自分に対応付ける (自分の管理下

におく）ことが可能です（図 10.39）.

なお,  リンクをクリックすると，図 10.36 のアバタ情報ウィンドウが表示されます.

図 10.38 アバターリスト

図 10.39 オーナ変更画面

10.7.5 アバター編集

アバターリストの画面（図 10.39）で,「EDIT」リンクをクリックすると，そのアバターの情報編集画面に移動します（図 10.40）.

アバターの情報編集画面では, パスワード, ホームリージョン, ステータスを変更可能です. ステータスを「使用禁止」にすると，そのアバターでは OpenSim にログインできなくなります. ただし，これは一時的にログインできなくなるだけなので, インベントリ等が削除されることはなく, アクティブにすれば再びログイン可能に戻ります.

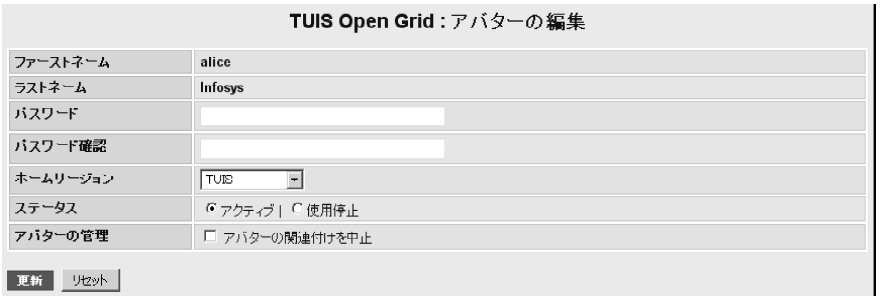


図 10.40 アバタの情報編集画面

アバターを削除する場合には，一旦アバターをこの「使用禁止」状態にします.「使用禁止」状態になると「削除」ボタンが表示されるようになりますので，それをクリックすることによりアバタを削除することができます（インベントリ等も削除されます).

また,「アバターの関連付けを中止」にチェックを入れると，そのアバターを自分の管理下から外す（Xoops Cubeのユーザとの対応を外す）ことができます. 一度管理から外したアバターをもう一度管理下におく（Xoops Cubeのユーザと対応付ける）には,前項の図10.39のオーナ変更画面でアバターのパスワードを入力します.

10.7.6 アバター作成

XOOPS_URL/modules/xoopensim/?action=create にアクセスすると，アバタの作成画面に移動します（図10.41). 当然このページにはゲストユーザはアクセスできませんし，アバタの保有数が「一般設定」で設定した上限に達しているユーザもアクセスすることはできません.

それ以外のユーザは図10.41の画面に必要な情報を入力して,「作成」ボタンをクリックすることによりOpenSimのアバターを作成することができます.

図10.41 アバタの新規作成画面 （ラストネーム管理が有効な場合）

10.8 XoopenSim の拡張機能

10.8.1 ヘルパー機能

ビューアの起動時の引数として **-helperuri** を XOOPS_URL/modules/xoopensim/helper/ と指定すれば，**currency**， **landtool** の**ヘルパー機能**が働きます（ただしXOOPS_URLはXoops Cubeのトップ URL).また,サードパーティ製のビューアを使用している場合は,Robust.iniの [GridInfoService] セクションの economy フィールドにこのURLを記述すれば，ビューアからの get_grid_info コールにより自動的にビューアに値が設定されます.この手法の詳しい設定方法はビューア毎に違いますので，各ビューアのマニュアルで確認してください（Hippo OpenSim Viewerの場合はグリッドへの接続設定画面に “Get Grid Info”ボタンがあります).

landtool helper の機能では，以下の機能が追加となります.

- **土地の分割，統合が可能となる.**
- **土地の販売設定が可能となる.**

currency helper の機能では，以下の機能が追加となります. この機能はマネーサーバが無くても働きます.

- **L$0 でのオブジェクト，土地の販売が可能になる.**

10.8.2 オフラインメッセージ機能

OpenSim でオフラインメッセージ機能を使用するには，OpenSim.ini の[Messages]セクションで，OfflineMessageModule， OfflineMessageURL を図 10.42 のように指定します．ここで，OfflineMessageModuleはオフラインメッセージを処理するモジュール名であり，OfflineMessageURLはデータベース更新のために呼び出される XML PRCのファイル名です.また,XoopenSim自体はミュートリスト機能をサポートしていませんが,ミュートリスト機能も記述しないとオフラインメッセージ機能が正しく動作しないため，図 10.42 のように MuteListModule，MuteListURL の設定も行っています．従って MuteListURL が指す mute.php の中身は，実はほぼ空の状態です．

なお,「グループへの招待」のメッセージは，この機能を使用してもオフラインでは使用することはできません.

```
[Messaging]
    .........
    InstantMessageModule = InstantMessageModule
    ; MessageTransferModule = MessageTransferModule
    OfflineMessageModule = OfflineMessageModule
    OfflineMessageURL = XOOPS_URL/modules/xoopensim/helper/offline.php
    MuteListModule = MuteListModule
    MuteListURL = XOOPS_URL/modules/xoopensim/helper/mute.php
```

図 10.42 OpenSim.ini でのオフラインメッセージ機能用の設定

10.8.3 Flotsam グループ機能

XoopenSim では Flotsamのグループ機能を内包しており,そのインストール時に自動的にFloatsamのグループ機能用のデータベーステーブルが作成されますので,改めてデータベースを作成する必要がありません.

```
[Groups]
    Enabled = true
    ........
    ;Module  = Default
    ........
    Module   = GroupsModule

    ; Enable Group Notices
    NoticesEnabled    = true
    ........
    ; Specify which messaging module to use for groups messaging and if it's enabled
    MessagingModule = GroupsMessagingModule
    MessagingEnabled = true
    ........
    ; SimianGrid Service for Groups
    ;ServicesConnectorModule = SimianGroupsServicesConnector
    ;GroupsServerURI = http://mygridserver.com:82/Grid/

    ; Flotsam XmlRpc Service for Groups
    ServicesConnectorModule = XmlRpcGroupsServicesConnector
    GroupsServerURI = XOOPS_URL/modules/xoopensim/helper/xmlgroups.php

    ; XmlRpc Security settings.  These must match those set on your backend groups service.
    XmlRpcServiceReadKey     = 68000
    XmlRpcServiceWriteKey    = 68030
    ........
```

図 10.43 グループ機能を使用する場合の OpenSim.ini の[Group]セクション （太字の部分が変更箇所）
ただし， XmlRpcServiceURL の XOOPS_URL は Xoops Cube のトップの URL

Flotsamのグループ機能を使用するには，OpenSim.ini の [Group]セクションを図10.43のように記述します.ここでデータベースへのアクセスキー(XmlRpcServiceReadKey, XmlRpcServiceWriteKey)は，XoopenSimの「一般設定」(表10.31) で設定した，Flotsamグループデータベースのアクセスキーと同じにしなければなりません.

図10.43のように設定後，リージョンサーバを再起動するだけで，OpenSimのグループ機能が利用可能になります.なお,このとき XoopenSimにゲストのアクセス許可がない場合,アバターはOpenSimに**ログインできなくなります**ので十分に注意してください（10.6節参照）.

Flotsamのグループ機能の詳細については下記URLを参照してください.

Floatsam Group Function: http://code.google.com/p/flotsam/wiki/XmlRpcGroups

10.8.4 osprofile機能

XoopenSimでは osprofile機能も内包しています. osprofile機能は元々は OpenSimのForgeプロジェクトの一つですが,不完全な状態のままでしたので,NSLで正常に動作するように改造を行い,ついでにXoopenSimへの組み込みを行いました.

Forge osprofileプロジェクト: http://forge.opensimulator.org/gf/project/osprofile

osprofile機能を利用するには,新しくリージョンサーバ用モジュールをコンパイルしなければなりません. XoopenSimのディレクトリにある osprofileのディレクトリを OpenSimのディレクトリにコピーして，そこで build.shを利用してコンパイルを行います（図10.44).

次にリージョンサーバの設定ファイル OpenSim.iniに [Profile]セクションを追加します（図10.45). 設定が終了したらリージョンサーバを再起動させます.

```
# cd "XoopenSimのインストールディレクトリ"
# cp -Rpd xoopensim/osprofile "OpenSimのインストールディレクトリ"
# cd "OpenSimのインストールディレクトリ"
# cd osprofile
# ./build.sh
```

図 10.44 osprofile モジュールのコンパイル

```
[Profile]
   ProfileURL = XOOPS_URL/modules/xoopensim/helper/profile.php
```

図 10.45 OpenSim.ini への osprofile 用の追加部分

10.9 Modlos

XoopenSimには姉妹ソフトウエアとして，Moodle上で動作する**Modlos**もいうWEBインターフェイスもあります. Modlosには XoopenSimの機能に加えて，**Sloodle***との連携が可能などの特徴があります. なお，Modlosについての詳細は下記URLを参照してください.

Modlos Wiki URL: http://www.nsl.tuis.ac.jp/xoops/modules/xpwiki/?Modlos

---

注*) Sloodle：Simulation Linked Object Oriented Dynamic Learning Environmentは， 学習用CMS(LMS)であるMoodleとSecond Life/Opensimを統合するためのシステムです． Second Life/OpenSimの同期性とMoodleの非同期性をうまく融合し， Second Life/OpenSimにMoodleの学習環境を提供します． Sloodleの詳細については下記URLを参照してください.
http://www.sloodle.org/

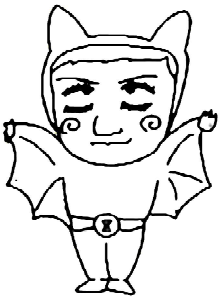
11. サーバのNAT(NAPT)越えの問題

自宅でOpenSimのサーバを立てて外部に公開する場合，もっとも問題となるのが「NAT越えの問題」です．この章では，NAT越えの問題とその解決方法について解説します．

そろそろ，外部にOpenSimサーバーを
公開したいな～

それだったら，NATを超えるための
措置を行わないといけないね．

いわゆる「オーバー・ザ・レインボー」だね．



それを言うなら「オーバー・ザ・ファイアウォール」
でしょう！ 「オーバー・ザ・レインボー」じゃオズ
の魔法使いだよ．

まあ，私は魔法使い級のハッカーを目指している
ということだよ．

でもオズの魔法使い級だったら，ぜんぜん
駄目ってことだよね

も～

11.1 NAT 越えの問題

OpenSimは基本的にNAT(正確にはNAPT)には対応していないので,NAT内部でサーバを運用しようとすると幾つかの問題を引き起こします.リージョンの設定ファイルである bin/Regions/Regions.ini を見ると,InternalAddressとExternalHostNameという項目がありますので,InternalAddressにNAT内部のプライベートIPアドレス,ExternalHostNameにグローバルIPアドレスを指定すれば良いように思われがちですが,ソースコードを見る限りではこれらの変数の間にはそのような関係は見当たりません.

InternalAddressとExternalHostNameの取り扱われ方の違いを言えば,ExternalHostNameはデータベースに保存されますが,InternalAddressは保存されません.また,InternalAddressはUDP通信時のIPアドレス通知のみ使用され,ExternalHostNameは主にHTTP(S)のRESTサービス(OpenSimではCAPS:CAPability System と呼ばれている)のURLの指定で使用されています(ExternalHostNameからIPアドレスを正引きしてUDPデータ転送時のアドレス通知に用いる場合もあります).

これらのアドレスはOpenSimのリージョンサーバからビューアに通知される訳ですが,リージョンサーバはNATの内側と外側に対して同じアドレスを通知するため,リージョンサーバに内側のアドレス(プライベートアドレス)を設定すればNAT外部のPCからは接続できず(勿論NAT上で必要なポートは開けてあるものとします),外側のアドレス(グローバルアドレス)を設定すればNAT内部のPCから接続できないと言った状況が生まれます(図11.1, 11.2).

ExternalHostNameについては,FQDNを指定してNATの内側と外側にそれぞれ別のDNSサーバを立て(一台のDNSサーバを使用して,内部で選り分けても良い),NATの内部に存在するPCに対してはプラ

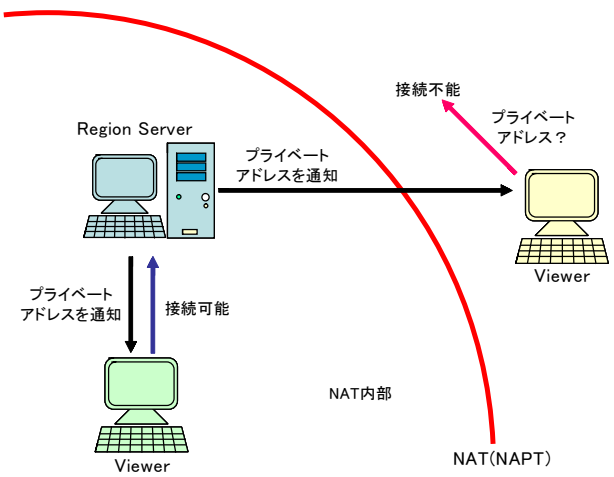


図 11.1 プライベートアドレスを設定した場合

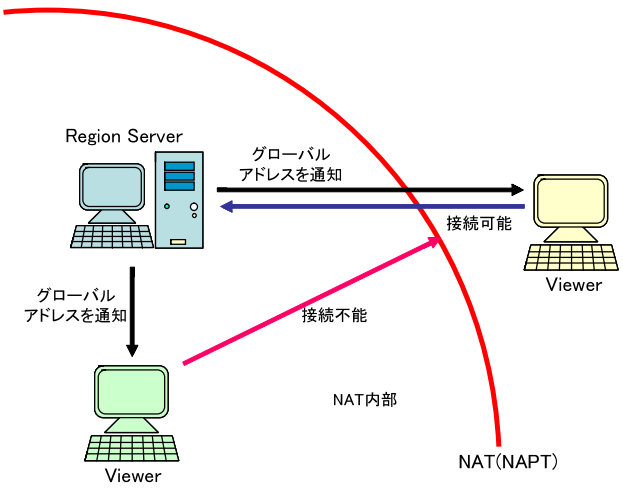


図 11.2 グローバルアドレスを設定した場合

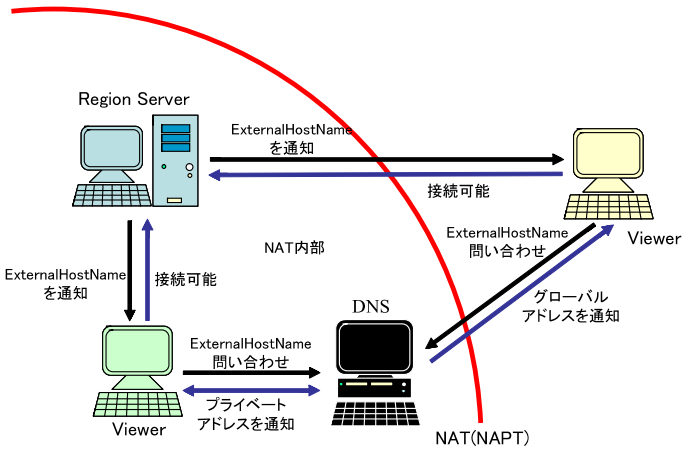


図 11.3 DNSを用いたExternalHostNameに対する解決方法

イベートアドレスを,NAT外部に存在するPCに対してはグローバルアドレスを返すようにすれば,HTTP(S)通信については解決します（図 11.3）．一方，UDP のデータ通信についてはは直接 IP アドレスをビューアに通知するので，このような手法でも解決できません．（実はNSL では 0.6.8/9に対しては，UDP 通信を NAT 内部と外部で自動識別して，それぞれに違うアドレスを通知するパッチを持っていますが，0.7 に対する検証は行っていません）

なんで 0.7 で検証しないのかな？

次に紹介するNATループバックの方が全然簡単だからやる気が失せたみたいだよ

11.2 NAT ループバック

このような問題を解決する最も単純で効果的な手法は,NAT(NAPT)用ルータとして,**NATループバック機能**付の Broad Band ルータ（以後 BB ルータ）を使用することです．NAT ループバック機能とは図 11.4 のように内部からの BB ルータのグローバルアドレスへの接続リクエストに対してもアドレス（ポート）変換を行う機能のことです．通常の BB ルータには NAT ループバック機能は搭載されておらず,内部からのBBルータのグローバルアドレスへの接続リクエストはBBルータ自身へ転送されます．

従って NAT ループバック機能を持つ BB ルータを使用した場合，リージョンサーバのアドレス（InternalAddress と ExternalHostName）としてグローバルな IP アドレスを指定すれば，NAT 内部からでも問題なくリージョンサーバにアクセスすることが可能となります（図 11.4）．

注）ただし当然のことですが，外部からのアクセスに対してはBBルータ上で必要なポート（リージョンサーバが使用する TCP と UDP のポート）を開けておく必要があります．

現在，NAT ループバック機能を搭載している主な BB ルータとしては，以下のものが挙げられます．

- **バッファロー BBR-4MG**
- **バッファロー BBR-4HG**
- Yamaha RT-57i
- Yamaha RT-58i　　**（このうち我々が実際に検証済みであるのは，バッファロー BBR-4HG のみです．）**

もしBBルータとしてNATループバック機能付のものが使用できるなら,それがこの問題に対する最も優れた解決方法です．

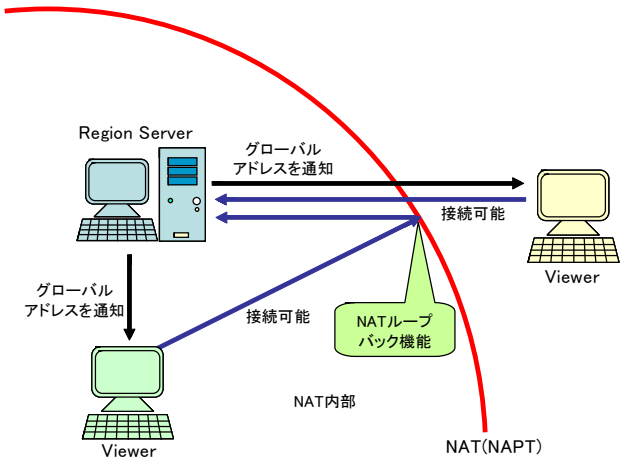


図 11.4 NAT ループバック機能

11.3 VPN

NATループバック機能付のBBルータが使用できない場合の解決方法の一つとして，VPNを用いる方法があります．OpenSimを利用しようする相手のPCも実はNATの内部にいるケースが多く，このような場合，相手のプライベートネットワークと自己のプライベートネットワークをVPNで繋ぐことが可能となります．

お互いのプライベートネットワークをVPNで繋ぎ，リージョンサーバにはプライベートIPアドレスを設定しておけば，それぞれのPCからリージョンサーバに接続することが可能となります(図11.5)．

VPNソフトとしては，HamachiやOpenVPN等が使用される事が多いようです．VPNソフトの設定には若干のスキルが要求され，接続相手にそのスキルを要求することになりますので，一般のユーザには少し敷居が高いと思われます．

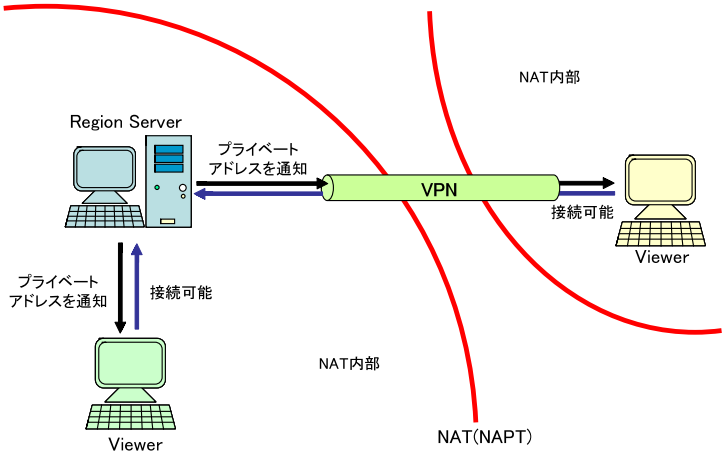


図 11.5 VPNを使用した接続

11.4 sl_proxy

今までの話はサーバがファイアウォールを越える話でしたが，大学や会社などではその組織のファイアウォールのために，逆にビューアが外部のOpenSimやSecond Lifeサーバに接続できないといった問題が発生します．NSLではこのような問題の解決のために，Second Life/OpenSim専用のプロキシシステムも開発しています．それが **sl_proxy** です．

sl_proxyはSecond Life/OpenSim専用のアプリケーションゲートウェイです．sl_proxyを使用すれば組織内のプライベートアドレスを持ったPCからでも，外部の Second LifeやOpenSimに接続することが可能となります（図11.6）．またパケット中継の他に，テクスチャデータのキャッシュや，リージョンへのアクセス制御機能も持っています．無論音声チャット（SIP）の中継も可能です．

sl_proxyについて興味のある方は下記URLをご覧ください．

sl_proxy Wiki URL: http://www.nsl.tuis.ac.jp/xoops/modules/xpwiki/?sl_proxy

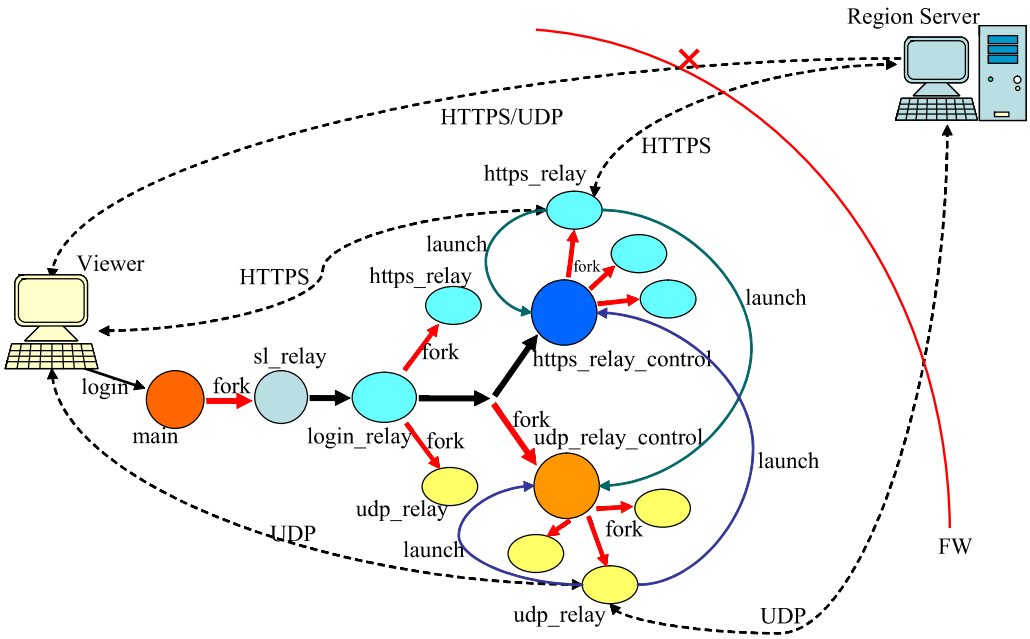


図 11.6 sl_proxy のパケット中継機能におけるプロセス間通信

12. あとがき

こんにちは,今回このテクニカルレポートと言う名の自費出版マニュアル(俗に同人誌！)を計画した「佐倉佐紀」と申します．私は，OpenSimの研究を大学で始めてもう１年半となります．

OpenSimは，すでに説明があるとおり有名なSecond Lifeと互換性のある３次元仮想空間サーバです.まだまだ未完成部分も多いOpenSimですが非常に興味深い機能がいっぱいで,海外のコミュニティを中心に精力的に開発が行われています．私たちNSLの中にあるメタバース研究会でも基本的なOpenSimの開発バージョンの追跡，各種ツールの開発，Moduleのテストや改造などの研究をメインに行っています．

現在特に力を入れようとしている研究は,Open Dynamics Engineなどの物理エンジンをハードウェア上で実現させようというものです．最近はGPGPUやOpenCL，AMD(旧ATI)のBulletへの参加など，私たちのサーバーでも手軽にハードウェア物理演算を実現できる時代となりました．それをOpenSimで実現したいと思っています．

この本を手に取ってくれた方々,購入していただいた方,OpenSimに興味を持っていただいた方,全員に感謝申し上げます．

うぁぁぁ メールサーバについて書くのを忘れた!

メールサーバなんて， 何に使うの？

Xoops Cube のユーザ登録だよ

そんなこと言ってたらDNSだってやってないよ

じゃあ．メールサーバの Tips を一件だけ
後は次回ということで.....

CentOS では PHP でメールを自動送信すると，**X-Authentication-Warning ヘッダ**が自動的に付加されるよ．ISPによってはこのヘッダが付いていると迷惑メールと判断されてしまうんだ．
これを防ぐには /etc/mail/trust-users に Apacheの実効ユーザ(この本では apache)を追加すればいいよ．

それじゃあ．みんな頑張ってね！

バイバイ!

またね～

次回予定は Hyperグリッド，Simianグリッド，拡張モジュール詳細だよ！

付録A　TUIS Open Grid

NSLメタバース研究会では，3次元仮想空間の有効利用を研究テーマに，TUIS Open Grid というOpenSimグリッドを運営しています．この研究はグリッドの名称の通り完全にオープンに行われており，どなたでも参加することができます．

参加に対する責務は，「使用許諾の遵守」のみであり，料金等も一切発生しません．個人を対象として，若干ではありますが土地の移譲も行っています．また，グループ単位で面白そうなプロジェクト案があれば，リージョン（SIM）ごと無料で貸し出すこともできます（数に限りがあります）．

ご興味のある方は是非サポートWEBサイト **http://www.opensim.tuis.ac.jp/** （図A.1）をご覧ください．サポートWEBサイトおよびTUIS Open Gridには本書で紹介した技術を余すところなく使用しています．

TUIS Open Gridのマスターアバターであるグリーン・ゴブリン（図A.2）が皆様のご利用をお待ちしております．

図A.1 http://www.opensim.tuis.ac.jp

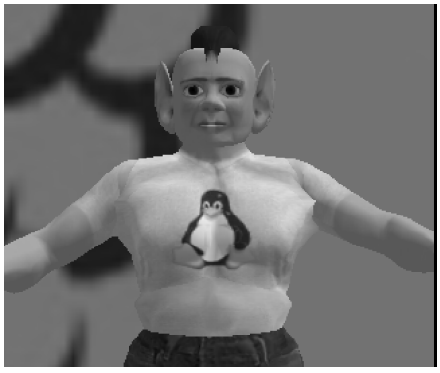


図 A.2 マスターアバター グリーン・ゴブリン
モノクロ印刷では色が分からないのが残念！

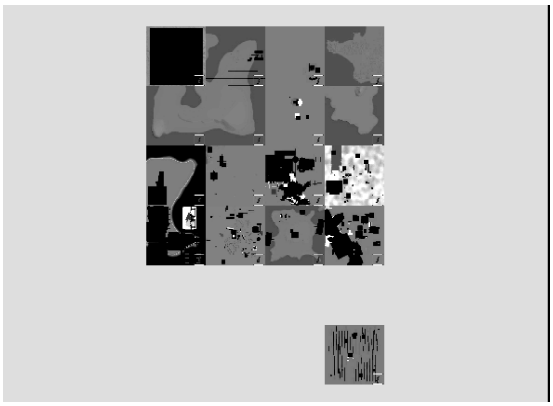


図 A.3 TUIS Open Gridのワールドマップ

付録B　MS Windows上での OpenSimの起動

ここでは，MicroSoft Windowsで OpenSimを起動させる場合の例を説明します．

B.1　ファイルのダウンロードとコンパイル

B.1.1 ファイルのダウンロード

プログラムのダウンロードページは http://opensimulator.org/wiki/Main_Pageです．手っ取り早くダウンロードしたい人のために，直リンクを以下に示します．

バイナリコード: http://dist.opensimulator.org/opensim-0.7-bin.zip
ソースコード:　http://dist.opensimulator.org/opensim-0.7-source.zip

ダウンロードしたファイルを適当なディレクトリ（何処でも可）で解凍します．プログラムをコンパイルするつもりがない場合は，バイナリコードをダウンロードして，このままB.2へ進んでください，

B.1.2 ソースコードのコンパイル

ソースコードをコンパイルするには MicroSoft の **Visual Studio .NET** が必要です．インストールされている　.NETのバージョンが　2010の場合は，解凍されたフォルダーの中にある**runprebuild2010.bat**ファイルをダブルクリックしてこれを起動します．.NETのバージョンが2010以外の場合は，**runprebuild.bat**を起動します．これらが正常に実行されれば **compile.bat** というコンパイル用バッチファイルが生成されますので，これをダブルクリックしてコンパイルを行います．

コンパイルが正常に終了すれば，binフォルダの中に OpenSim.exeや Robust.exeなどの実行ファイルが生成されます．

B.2　スタンドアロンモード （SQLite3を使用する場合）

バイナリコードをダウンロードした場合，OpenSimを起動する場合の設定ファイルは，ほとんど既に用意されています．ただし，bin¥config-includeディレクトリにあるキャッシュ設定用のファイル CenomeCache.ini.example　には　.example　がついたままですので，.exampleを外して，**CenomeCache.ini**に変更します．CenomeCache.iniはキャッシュ用の設定ファイルです．

ソースコードからコンパイルした場合は CenomeCache.iniの他に，bin¥OpenSim.ini.exampleと bin¥config-include¥StandaloneCommoni.ini.exampleの .exampleを取り除きます．

後は，bin¥**OpenSim.exe** をダブルクリックするだけで，OpenSimがスタンドアロンモードで起動します．ただし，64bitOSの場合は OpenSim.exeの代わりに **OpenSim.32BitLaunch.exe**をダブルクリックします．起動時に入力する項目については，Linuxの場合と同じですので，本編の「5.2.1項」をご覧ください．

B.3 WampServer

MS Windows上で Apache（WEBサーバ），MySQL（データベースサーバ），PHPを動かすためのシステムとして WampServerがあります．WampServerの **Wamp**には **W**... Windows，**a**... Apache，**m**... MySQL，**p**... PHPの意味があります．因みに，Linux上での同様の環境を Lampと呼びます．

WampServer は残念ながら英語版のみですが，非常に簡単に Windows上に Apache，MySQL，PHPの環境を作り出すことが可能です．WampServerをインストールしておけば，phpMyAdminなどの管理ツールを使用することもできます．また，後でOpenSimのWEBインターフェイスをインストールすることも簡単ですので，ここでは MySQLを WampServerを使って起動することにします．

B.3.1 WampServer のダウンロード

WampServer を下記URLよりダウンロードします.’10 7/20 現在のWampServerの最新バージョンは 2.0i です. 各サーバのデフォルトのバージョンは, Apacheは2.2.11, MySQLは5.1.36, PHPは5.3.0ですが, アドオンとして下位バージョンのものを組み込むこともできます.

英語サイトURL: http://www.wampserver.com/en/

サイト画面の “1”の下のリンクをクリックし, 本体のダウンロード画面に飛び, WampServer本体をダウンロードします.

次に最初の画面に戻って,“2”の下のリンクをクリックしてアドオン画面に移動します. ここで, PHP 5.3は互換性の面で問題を起こす場合がありますので, PHP 5.2をアドオンとして組み込むことにします. “-PHP”のリンクからPHPのダウンロード画面に飛び, PHP 5.2の最も新しいものをダウンロードします.

B.3.2 WampServer のインストール

ダウンロードした WampServer本体のインストーラをダブルクリックしてインストールを行います. インストーラから WampServerをインストールする場所を尋ねられますので,″適当なドライブとフォルダを指定します. またデフォルトのWEBブラウザの確認画面では, そのまま “OK”または“はい”で良いでしょぅ. 最後のメールサーバ (SMTPサーバ) とメールアドレスの設定でも, ″適当なメールサーバがなければ (分からなければ) デフォルトのままでOKです.

次に, PHP 5.2のアドオンのインストーラを起動します. こちらでも最後にメールの設定画面が表示されますが,WampServer本体での設定の内容と同じにします.

’11 12/8現在では以下のバージョンの Apache と PHP で正常に動作することを確認しています.

Apache 2.2.8 (unable access restrictions) + PHP 5.2.9-2 (with php_curl and php_xmlrpc extensions)

図 B.1 WampServer のメニュー

B.3.3 WampServer の実行

メニューから WampServerを起動すると, アイコントレイに半円形のスピードメータのよぅなアイコン が現れます. この状態で, 既にApacheとMySQLが起動しているはずです. このアイコンをクリックすると, 図B.1のよぅなメニューが表示されます. ここで**phpMyAdmin**を選択すると, WEBブラウザが起動し, 図B.2のよぅなMySQLの制御画面が表示されます.

図 B.2 phpMyAdmin のトップページ

B.3.4 MySQLの管理パスワードの設定 （オプション）

初期状態ではMySQLに管理パスワードは設定されていません．ここではphpMyAdminを使用して管理パスワードを設定する方法を説明します．この設定は，間違えると MySQLサーバに接続できなくなりますので，十分に注意してください．テストでWampServerをインストールした場合や，管理者からのアクセスをローカルマシンに限定（デフォルトの設定）するのであれば，パスワードは無しのままで良いかもしれません（各自で判断してください）．

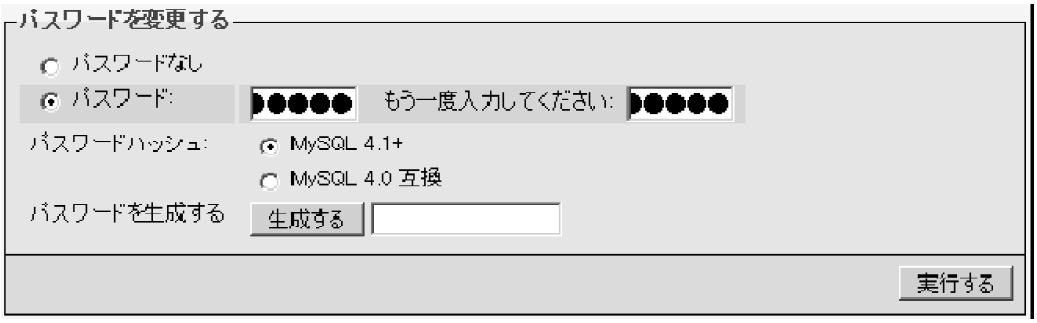
管理パスワードを変更する場合は，phpMyAdminページの上方のタブから，「特権」を選択します．図B.3のような特権設定ページが現れますので，rootユーザ（図では2名存在）の  アイコンをクリックします．特権設定ページの「パスワードを変更する」のブロック（図B.4）にパスワードを入力して「実行する」をスリックします．rootユーザが 2名存在する場合には両方に対してパスワード設定の操作を行います．

管理者(root)のパスワードを変更した場合，接続用のパスワードが変化したことになりますので，phpMyAdminのこれまでの設定では，MySQLに接続することができなくなります（図B.5）．この場合は[WampServerのインストールディレクトリ]¥apps¥phpmyadmin3.2.0.1¥**config.inc.php** に変更したパスワードを記述します．ここで，3.2.0.1は phpMyAdminのバージョン番号ですので，phpMyAdminのバージョンによって変化します．config.inc.phpの変更では，**$cfg['Servers'][$i]['password']** に新しいパスワードを設定します．

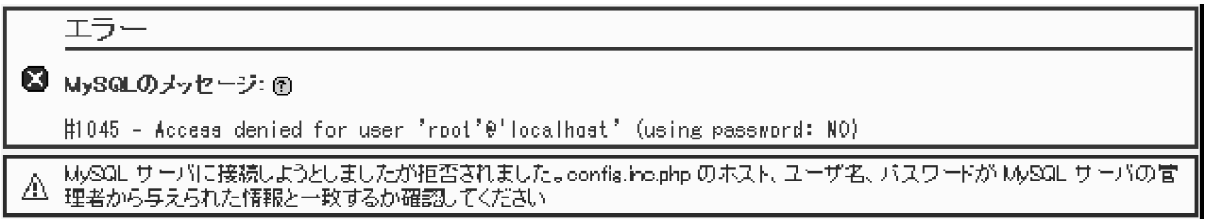
config.inc.phpに新しいパスワードを設定したら，WampServerのメニューから[Apache]→[Service]→[Restart Service]を選択して，Apacheを再起動させます．

	ユーザ	ホスト	パスワード	グローバル特権 [1]	権限委譲	
☐	すべて	%	--	USAGE	いいえ	
☐	root	127.0.0.1	いいえ	ALL PRIVILEGES	はい	
☐	root	localhost	いいえ	ALL PRIVILEGES	はい	

図B.3 phpMyAdminの特権設定画面



図B.4 phpMyAdminのパスワード設定画面



図B.5 phpMyAdminでのデータベース接続エラー

B.3.5 データベースの作成と権限の設定

新しくデータベースを作成するには，  アイコンでトップページに移動します．トップページ(図B.2)の「新規データベースを作成する」でデータベース名を入力して「作成」ボタンをクリックします．図B.6は opensim_db というう名前のデータベースを作成している図です．なおデータベース作成時の設定はデフォルトのままでかまいません．

「データベース」タグをクリックし，データベースが作成されていることを確認します．次に該当データベース名が表示されている行の右端の  アイコンをクリックして，特権設定ページに移動します．ここで，さらに「新しいユーザを追加する」をクリックします．「新しいユーザを追加する」画面（図B.7）では，「ログイン情報」ブロックにユーザ名とパスワードを入力し，ページの一番下にある「**実行する**」をクリックします．なお，ホストについては環境に合わせて指定します．もしよく分からなければ空欄のままにしておいてください．図B.7では，ユーザ名 opensim_user，パスワード opensim_pass のユーザを作成しています（パスワードは伏字になっています)．

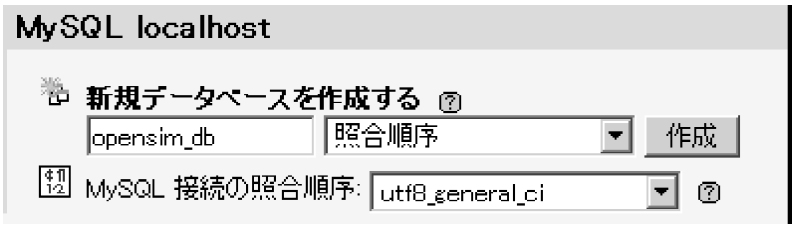


図 B.6 phpMyAdmin のデータベース作成画面

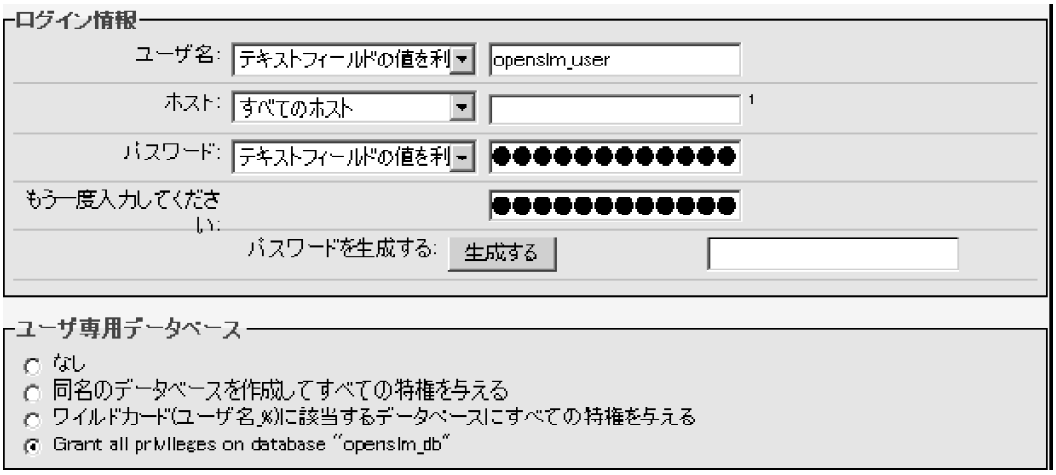


図 B.7 phpMyAdmin の特権ユーザ追加画面

B.4 スタンドアロンモード （MySQL サーバを使用する場合）

データベースにMySQLサーバを使用して，スタンドアロンモードでOpenSimを起動する場合には，WampServerを使用します．OpenSimを起動する前に，WampServerを起動させ，前節の内容に従って，OpenSim用のデータベースと管理用のユーザを作成しておく必要があります．

また設定ファイルについては，本編の「5.1.2項」,「5.1.3項」を参考にして，前節で作成したデータベースへの接続情報を bin¥OpenSim.iniと bin¥config-include¥StandaloneCommon.iniに記述します．

後は，ＳＱＬｉｔｅ3 を使用する場合と同様に，ｂｉｎ¥**ＯｐｅｎＳｉｍ．ｅｘｅ**（６４ｂｉｔＯＳ の場合は **OpenSim.32BitLaunch.exe**）をダブルクリックするだけで，OpenSimがスタンドアロンモードで起動します．起動時に入力する項目については，Linuxの場合と同じですので，本編の「5.2.1項」をご覧ください．

B.5 グリッドモード

OpenSim をグリッドモードで起動する場合は，MysQL サーバの使用が必須ですので，この場合も WampServer を使用します．OpenSim の起動に先立って，B.3.5 項の内容に従って，OpenSim 用のデータベースと管理用のユーザを作成しておきます．

B.5.1 ROBUST サーバの設定と起動

グリッドモードでは，リージョンサーバ（OpenSim.exe）の他に ROBUST サーバを起動する必要があります．設定ファイルは，bin¥**Robust.ini** ですので，bin¥Robust.ini.example からリネーム（またはコピー）して作成します．Robust.ini では MySQL サーバへの接続を記述します．Robust.ini の **ConnectionString** を例えば図 B.8 のように変更します（本編「6.1.1 項」参照）．

全て設定し終えたら，Robust.exe（64bitOS の場合は **Robust.32BitLaunch.exe**）をダブルクリックして，ROBUST サーバを起動します．ROBUST サーバが正常に起動すれば，**R.O.B.U.S.T.#** のコマンドプロンプトが表示されます．

```
[DatabaseService]
  StorageProvider = "OpenSim.Data.MySQL.dll"
  ConnectionString = "Data Source=localhost;Database=opensim_db;User
 ID=opensim_user;Password=opensim_pass;"
```

図 B.8 Robust.ini の変更箇所

B.5.2 アバターの作成

グリッドモードではリージョンサーバ（OpenSim.exe）を起動する前に，最低限一名のアバター（エステートの管理者となるアバター）を作成しておかなければなりません．

作成方法は，ROBUST サーバのコマンドプロンプトから **create user** コマンドを入力します．詳細については，本編「6.1.2 項」および「図 6.5」を参照してください．

B.5.3 リージョンサーバ(OpenSim.exe)の設定と起動

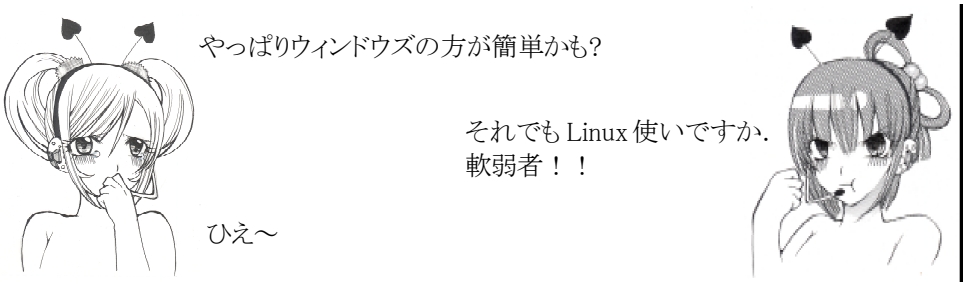
グリッドモードでのリージョンサーバの設定ファイルは bin¥OpenSim.ini， bin¥config-include¥GridCommon.ini および bin¥config-include¥FlotsamCache.ini です．これらの設定も Linux の場合と全く同じですので，本編「6.2.1 項」，「6.2.2 項」を参照してください．

リージョンサーバを起動するには OpenSim.exe（64bitOS の場合は OpenSim.32BitLaunch.exe）をダブルクリックします．起動時に入力する項目についても，本編の「5.2.1 項」をご覧ください．

正常に起動完了した場合，**Resion（リージョン名）#** のコマンドプロンプトを表示して，コマンド入力待ちとなります（マルチリージョンの場合は **Resion（root）#** のコマンドプロンプトとなります）．

B.6 ビューアの設定

ビューアの設定については，「5.3 節」および「6.4 節」をご覧ください．



付録C　ini ファイルの概要

ここでは各 ini ファイルの主要な設定項目（フィールド）について解説を行う．

C.1 Robust.ini ファイル

Robust.ini はグリッドモード時の ROBUST サーバの設定を行うファイルである．

C.1.1 [Startup] セクション

ROBUST サーバのコネクタモジュールを指定する．

C.1.2 [Network] セクション

ネットワーク接続に関する設定を行う．

port

ROBUST サーバがサービスを提供する HTTP ポートを指定する．デフォルトは 8003.

ConsoleUser

RestConsole モードを使用する場合の，接続用 ID． RestConsole については，本編 8.2 節を参照．

ConsolePass

RestConsole モードを使用する場合の，接続用パスワード．

ConsolePort

RestConsole モードを使用する場合の，接続用 HTTP ポート．0 を指定した場合は RestConsole のポートとして port で指定したものが併用される． デフォルトは 8003.

C.1.3 [DatabaseService] セクション

データベースへの接続のための設定を行う．

StorageProvider

データベースのモジュールを指定する．MySQL を使用する場合は OpenSim.Data.MySQL.dll を指定する．

ConnectionString

データベースへの接続のためのパラメータを指定する． 通常は **Data Source= マシン名;Database= データベース名;User ID= ユーザ ID;Password= パスワード;** とする．

C.1.4 [FreeswitchService] セクション

ボイスチャットのための FreeSwitch の設定を行う．

ServerAddress

FreeSwitch サーバのアドレスを指定する．

Realm

SIP の領域名（固有名）を指定する． 他と被らなければ良いので， FreeSwitch サーバの IP アドレスで良い．

SIPProxy

SIP Proxy のアドレスとポート番号を指定する．通常は FreeSwitch サーバの IP アドレス:5060

Password

FreeSwitch サーバに接続するためのパスワード． conf/autoload_configs/xml_curl.conf.xml の パスワード（gateway-credentials の値）と一致させる．

EchoServer

SIP のエコーサーバを指定する． 通常は FreeSwitch サーバ．

C.1.5 [LoginService] セクション

WelcomeMessage

ログイン時に表示されるメッセージを指定する．UTF-8の日本語もOK.

C.1.6 [GridInfoService] セクション

get_grid_info機能をサポートするビューアへ通知するための，サーバ情報を記述する．

login

OpenSimにログインするためのURLを指定する． ビューアが get_grid_info 機能を持つ場合 **loginuri** オプションの値として設定される．

gridname

グリッドの名前を指定する．

gridnick

グリッドの短い名(ショートネーム)またはニックネームを指定する．

welcome

ビューア起動時の中央画面に表示するためのWebページを指定する．ビューアが get_grid_info機能を持つ場合，**loginpage** プションの値として設定される．

economy

ヘルパー機能用のXML RPCのURLを指定する． ビューアが get_grid_info 機能を持つ場合， **helperuri** プションの値として設定される．

about

グリッドの説明を記述したWebページのURLを指定する．ページが存在しないならば，特に指定する必要はない．

register

アカウント（アバター）の作成方法を記述したWebページのURLを指定する． ページが存在しないならば， 特に指定する必要はない

help

グリッドに関するヘルプページのURLを指定する． ページが存在しないならば，特に指定する必要はない．

password

パスワードに関するヘルプページのURLを指定する． ページが存在しないならば，特に指定する必要はない．

C.2 MoneyServer.ini ファイル

MoneyServer.iniはDTL/NSLマネーサーバの設定を行うファイルである．

C.2.1 [MySql] セクション

OpenSim用のMySQLサーバへの接続設定を行う．

hostname

MySQLサーバが稼動するホストのFQDNまたはIPアドレスを指定する．

database

MySQLデータベースの名前を指定する．

username

MySQLデータベースへの接続用のユーザ名を指定する．

password

MySQLデータベースへの接続用のパスワードを指定する．

port

MySQLサーバのポート番号を指定する． 通常は 3306を指定する．

MaxConnection
MySQLサーバとのコネクションの最大数

C.2.2 [MoneyServe] セクション

DefaultBalance
アバター作成時の所持金を指定.

BankerAvatar
バンカーアバターのUUIDを指定する．ここに指定されたアバターはシステムから，マネーをコスト無しに入手できる．00000000-0000-0000-0000-000000000000 を指定した場合は，全てのアバターがバンカーアバターとなる.

EnableForceTransfer
アバターがログインしていない場合でも送金を可能にする．ベンディングマシンで支払いスクリプトを動かす場合には必須.

EnableScriptSendMoney
送金スクリプトで，お金の送金を可能にする．アバターにサラリーやボーナスを自動的に支払う場合は必須.

MoneyScriptAccessKey
送金スクリプトへのアクセスキー.

MoneyScriptIPaddress
送金スクリプトを実行するマシンのIPアドレス

BalanceMessage....
お金のやり取りが発生した場合のメッセージを指定できる.

ServerCertFilename
マネーサーバとHTTPS通信を行う場合に使用するpfx(pkcs12)ファイルを指定する．ヌル文字の場合はHTTPSを使用しない.

ServerCertPassword
pfx(pkcs12)ファイルのパスワードを指定する.

C.3 OpenSim.iniファイル

OpenSim.iniはリージョンサーバの設定を行うファイルである.

C.3.1 [Startup] セクション

リージョンサーバの基本的な動作の設定を行う.このセクションで最も重要な設定項目は,プリム用のデータベースを定義するための storage_plugin と storage_connection_string である.

DrawPrimOnMapTile
World Mapなどの表示で，リージョン上のオブジェクトを描画するかどうかを指定する.

TextureOnMapTile
World Mapなどの表示で，リージョンの地面のテクスチャを描画を指定する．デフォルトは false.

NonPhysicalPrimMax
（物理プリムでない）通常のプリムの最大サイズ．単位はメートル.

PhysicalPrimMax
物理プリムの最大サイズ．単位はメートル.

ClampPrimSize
プリムのサイズを制限する．デフォルトは false.

AllowScriptCrossing
リージョンの境界を越えた時に，スクリプトを再コンパイルする．デフォルトは true.

TrustBinaries

リージョンの境界を越えた時に，コンパイル済みのバイナリコードをそのまま使用する．デフォルトは false．このフィールドを true にした場合，移動元のリージョンから危険なバイナリコードを流し込まれるというリスクが発生する．

CombineContiguousRegions

連続したリージョンを一つの大きなリージョン（メガリージョン）として扱う．メガリージョンの構成方法については，本編 7.7 節を参照．なお，このモードはまだ実験段階であり，通常のリージョンをメガリージョンにした場合，既にあるオブジェクトは壊れる可能性がある．

physical_prim

物理プリムを許可するかどうかを指定する．通常はデフォルトの true をそのまま使用する．

meshing

オブジェクト（プリム）のメッシュの有無を指定する．通常はデフォルトである true をそのまま使用する．

physics

物理エンジンの指定．現時点ではデフォルトの OpenDynamicsEngine(ODE)が最も高機能である．

permissionmodules

オブジェクトのパーミッション機能を提供するモジュールを指定する．オブジェクトのパーミッションを有効にする場合にはコメントを外して有効にし，かつ serverside_object_permissions も true にする．

serverside_object_permissions

オブジェクトのパーミッションのチェックをサーバで行うかどうかを指定する．オブジェクトのパーミッションを有効にするには，これを true にし，permissionmodules でパーミッションモジュールを指定しなければならない．

region_owner_is_god

リージョンのオーナが God になれるか（God モードに入れるか）どうかを指定する．

region_manager_is_god

リージョンのマネージャが God になれるかどうかを指定する．

parcel_owner_is_god

パーセル（土地）のオーナが God になれるかどうかを指定する．

DefaultScriptEngin

スクリプトエンジンを指定．デフォルトは XEngine．

HttpProxy

LSL の llHTTPRequest()関数または Dynamic Texture 使用時の WEB プロキシを指定する．

HttpProxyExceptions

Httproxy の例外サイトを指定する．

startup_console_commands_file, startup_console_commands_file

リージョンサーバが起動，またはシャットダウンする場合に自動的に実行されるサーバコマンドを記述したファイルを指定する．

C.3.2 [SMTP]セクション

リージョンサーバの SMTP（メール）サーバ機能の設定を行う．なお，OpenSim で Sloodle Set を動作させるには，SMTP モジュールを有効にする必要がある．

enabled

SMTP（メール）モジュールを有効にするかどうかを指定する．デフォルトは false．

internal_object_host

リージョンサーバのメールのドメイン名を記述する.

host_domain_header_from

メールヘッダのFrom行に記述される名前を指定する

SMTP_SERVER_HOSTNAME

SMTPサーバを指定する.

SMTP_SERVER_PORT

SMTPサーバのポート番号を指定する.

SMTP_SERVER_LOGIN

認証が必要なSMTPサーバの場合，ログイン名を指定する.

SMTP_SERVER_PASSWORD

認証が必要なSMTPサーバの場合，パスワードを指定する.

C.3.3 [Network] セクション

リージョンサーバのネットワーク設定について記述する.

ConsoleUser

RestConsole モードを使用する場合の， 接続用 ID． RestConsole については， 本編 8.2 節を参照.

ConsolePass

RestConsoleモードを使用する場合の，接続用パスワード.

console_port

RestConsoleモードを使用する場合の， 接続用 HTTPポート．0を指定した場合は RestConsoleのポートとして http_listnet_port で指定したものが併用される.

ExternalHostNameForLSL

LSLのllRequestURL/llRequestSecureURL関数を指定した場合に通知されるリージョンサーバの名前．FQDNかIPアドレスを指定する.

C.3.4 [Messaging] セクション

OfflineMessageModule

オフラインメッセージを処理するモジュールを指定する．通常は OfflineMessageModuleを使用する.

OfflineMessageURL

オフラインメッセージをデータベースに格納するための，XML PRCの呼び出しURLを指定する．使用方法については， 本編 10.8.2 項を参照.

MuteListModule

ミュートリスト（無視リスト）を処理するモジュールを指定する．通常は MuteListModuleを使用する.

MuteListURL

ミュートリスト（無視リスト）をデータベースに格納するための，XML PRCの呼び出しの外部URLを指定する（外部プログラムを必要とする）.

C.3.5 [ODEPhysicsSettings] セクション

物理エンジンに ODE を使用している場合，ODE の設定を行う.

use_NINJA_physics_joints

プリム同士の関節接続（ジョイント）機能を有効にしたい場合に true にする．デフォルトは false.

C.3.6 [Wind], [Cloud], [LightShare], [Trees]セクション

風，雲，光り（ライト），樹木に関する設定を行う.

enable_windlight

Wind Light 機能（光の効果）を有効にする．ただし，MySQL データベースを使用している状態でければ，この機能を有効にすることはできない．デフォルトは false.

active_trees

ツリーモジュールを有効にする．デフォルトは false. ツリーモジュールについては，本編「7.8 節」を参照.

C.3.7 [Economy] セクション

L$ やマネーサーバについての設定を行う.

SellEnabled

オブジェクトを販売可能にするかどうかを指定する．デフォルトは false.

EconomyModule

マネーサーバへの接続モジュールを指定する. DTLマネーサーバを使用する場合は，DTLMoneyModule を指定する． DTL マネーサーバについては，

CurrencyServer

マネーサーバへの接続 URL を指定する． DTL マネーサーバを使用する場合は，"https://[マネーサーバの FQDN または IP アドレス]:8008/" となる.

UserServer

DTL マネーサーバを使用する場合は，ユーザサーバの URL を指定する．ただし v0.7 からは ROBUST サーバの URL を指定する． URL の指定に localhost または 127.0.0.1 は使用してはいけない.

PriceUpload

テクスチャやアニメーションをリージョンサーバにアップロードする場合の費用．デフォルトは 0L$.

PriceGroupCreate

グループを作成する場合の費用．デフォルトは 0L$.

C.3.8 [XEngine] セクション

スクリプトエンジンの設定を行う.

MaxThreads

起動可能なスクリプトのスレッドの最大数を指定する.

Priority

スクリプト（スレッド）の実行優先度を指定する．デフォルトは BelowNormal.

MaxScriptEventQueue

スクリプトのイベントキューのサイズを指定する.

ThreadStackSize

スクリプトのスタックサイズを指定する. 大規模なスクリプトを実行する場合，MaxThreads，MaxScriptEventQueue，ThreadStackSizeの値を大幅に増加させる必要がある.

DefaultCompileLanguage

デフォルトで使用するスクリプト言語を指定する.

AllowedCompilers

使用可能なスクリプト言語を指定する．lsl(LSL), cs(C#), js(J#), vb(Visual Basic)が指定可能．ただしLinuxで作動を確認しているのは LSL と C# のみ.

AllowOSFunctions

OS 関数を使用可能にするかどうかを指定する． OS 関数については， 本編 7.4 節を参照.

OSFunctionThreatLevel

OS 関数の実行優先度を指定する．デフォルトは VeryLow.

C.3.9 [FreeSwitchVoice] セクション

FreeSwitch（SIPサーバ）を利用して，ボイスチャットを行う場合にしている．FreeSwitchによるボイスチャットの設定の詳細については 「7. FreeSwitchを利用したボイスチャット」を参照して頂きたい．

enabled

FreeSwitchサーバを使用してボイスチャットを行う場合はtrueにする．

LocalServiceModule

StandAloneモードの場合は **FreeswitchServiceを選択．Gridモードの場合は，RemoteFreeswitchConnectorを選択する．**

FreeswitchServiceURL

FreeSwitchサービスへの接続URL（ROBUSTサーバ:8004)を指定する．

C.3.10 [Groups] セクション

現時点では，グループ機能を使用するには Flotsamグループ機能を使用するか，Simianグリッドを使用なければならない．Flotsamを使用する場合は，これを改造したXoopenSimの拡張機能を使用するという選択肢もある．XoppenSimの詳細については本編「10章 Webインターフェイス XoopenSim」を参照すること．

Enabled

グループ機能を使用する場合はtrueにする．デフォルトは false．

Module

グループ機能を使用する場合は GroupsModuleを指定する．

NoticesEnabled

グループ通知の有無を指定する．グループ機能を使用する場合はtrueにする．なお，現バージョンのOpenSimでは，オフラインアバターにはグループ通知は届かない．

MessagingModule

グループ内メッセージを処理するモジュールを指定する．通常は GroupsMessagingModuleを指定する．

MessagingEnablede

グループ内メッセージを有効にするかどうかを指定する．デフォルトは true．

ServicesConnectorModule

サーバとのコネクタモジュールを指定する．Simianグリッドを使用する場合は SimianGroupsServicesConnectorを，Flotsamグループ機能を使用する場合は XmlRpcGroupsServicesConnectorを指定する．

GroupsServerURI

Simian Gridのグループ機能を使用する場合は Simianグリッド側で設定したサービス提供用のURLを指定する．Flotsamグループ機能を使用する場合は，そのXML RPCのURLを指定する． XoopenSim でFlotsamグループ機能を使用する場合は，XOOPS_URL/modules/xoopensim/helper/xmlgroups.php を指定する．

XmlRpcServiceReadKey

XFlotsamグループ機能を使用する場合に XML RPCの読み込みキーを指定する． XML RPC実行側の設定と合わせる必要がある．

XmlRpcServiceWriteKey

XFlotsamグループ機能を使用する場合に XML RPCの書き込みキーを指定する．XML RPC実行側の設定と合わせる必要がある．

C.3.11 [WebStats] セクション

enabled

これを trueにした場合，http://[リージョンサーバのFQDNまたはIPアドレス]:9000/SStats により，そのリージョ

ンサーバで稼動してるリージョンの状態を取得することができる. なお, このページはAjaxの機能により自動更新される. またURL中の 9000は [Network]セクションの http_listner_portフィールドに設定した値を指定する

C.3.12 [Architecture] セクション

OpenSimの動作モードに応じて，これらのInclude設定のうち何れか一つを選択する.

Include-Architecture

動作モードを指定する. スタンドアロンモード，グリッドモード，Simianグリッドモード (ROBUSTサーバの代わりに Simianグリッドを使用する)の基本モードと それぞれの基本モード＋ハイパーグリッドモードの組み合わせを選択可能.

C.3.13 [Profile] セクション

osprofile機能を使用する場合に,OpenSim.iniへ追加する.osprofile機能については,本編10.8.4項を参照.

ProfileURL

osprofile 用の XML RPC の URL を指定する. XoopenSim を使用している場合は XOOPS_URL/modules/xoopensim/helper/profile.php とする.

C.3.14 [Search] セクション

ossearch機能を使用する場合に，OpenSim.ini へ追加する.

SearchURL

ossearch 用の XML RPC の URL を指定する. XoopenSim を使用している場合は XOOPS_URL/modules/xoopensim/helper/query.php とする.

C.4 config-include/GridCommon.ini ファイル

GridCommon.iniはグリッドモード時に，OpenSim.ini → config-include/Grid.ini → config-include/GridCommon.ini の順に呼び出されるファイルで，主にROBUSTサーバシェル内の各サーバの設定を行う.

C.4.1 [～Service] セクション

～ ServerURL

グリッドモード時の各サーバのHTTPサービスのURLを指定する. ROBUSTサーバのみを使用している場合は各フィールドにデフォルトで http://[ROBUSTサーバのFQDNかIPアドレス]:8003/ を設定する. URL中の 8003は [Network]セクションの portフィールドに設定した値を指定する.

C.5 Regions/Regions.ini ファイル

Regions.iniはリージョンの設定を行うファイルである. もっとも Regionsディレクトリ内にあって，拡張子が .ini ならば，ファイル名は何でも良い.

RegionUUID

リージョンのUUIDを指定する. 通常はシステムが生成した値を使用する. 手動で UUIDを生成するには uuidgen コマンドを使用する.

Location

リージョンのX,Y座標位置をカンマ（,）区切りで指定する. 各座標は 0～65535の間で指定する.

InternalAddress

UDP通信に使用するIPアドレスを指定する. NAT(NAPT)を使用している場合は，NATルータのグローバルア

ドレスを指定しても良いが，この場合，NATルータがNATループバックの機能をサポートしてないと，NAT内部のビューアとの通信ができなくなる（本編 11 章参照）．0.0.0.0を指定するとシステムが適当なIPアドレスを自動設定する．

InternalPort

UDP通信に使用するポート番号を指定する．一台のサーバマシンでマルチリージョンやメガリージョンなど複数のリージョンを稼動させる場合は，各リージョンでこのポート番号が被らないようにしなければならない．

AllowAlternatePorts

実験的な機能．デフォルトの Falseのままにしておく．

ExternalHostName

サーバマシンのFQDNかIPアドレスを指定する． NAT(NAPT)を使用している場合は， NATルータのグローバルアドレスを指定しても良いが，この場合，NATルータがNATループバックの機能をサポートしてないと，NAT内部のビューアとの通信ができなくなる（本編 11 章参照）．SYSTEMIPを指定するとシステムが適当なIPアドレスを自動設定する．

付録D サーバコマンド一覧

ここでは，OpenSimサーバ群のコマンドの概要について説明を行う．コマンドの引数の表記仕方は以下の通りとする．

コマンド表記：
< > は変数（他の文字に置き換わる），[ ] は省略可能，(|) はどちらかを選択．

D.1 ROBUST サーバ

command-script <file>
コマンドの記述されたファイル<file> を実行する

create user [<first_name> [<last_name> [<password> [<email>]]]]
新しいアバターを作成する．

delete asset <uuid>
データベースからID <uuid>のアセットを削除する

help [<command>]
ヘルプを表示する．<command> を指定した場合は，そのコマンドのヘルプを表示する．

login level <level>
ログイン可能な最小ログインレベルを設定．レベルはアバター毎に割り当てられている．グリッドモード時のみ有効．

login reset
ログインレベルをリセット(0に)する．グリッドモード時のみ有効．

login text <text>
ログイン時に表示されるテキストを設定．グリッドモード時のみ有効．

quit
サーバプログラムを終了する．

reset user password [<first_name> [<last_name> [<new_password>]]]
アバターのパスワードを変更する．

set regions flags <region_name> <flags>
リージョンフラグ（データベースの regions.flagsレコード）を設定する．フラグは数字（整数）かフラグ名を用いる．フラグ名には， DefaultRegion(1), FallbackRegion(2), RegionOnline(4), NoDirectLogin(8), Persistent(16), LockedOut(32), NoMove(64), Reservation(128), Authenticate(256), Hyperlink(1024)がある． 負の整数を与えた場合はフラグを外すことができる．デフォルトは RegionOnline(4).フラグの意味については詳細不明．

show digest <uuid> - Show asset digest
指定されたID <uuid>のアセットのダイジェスト情報を表示する．

show hyperlinks
ハイパーグリッドのリージョンのリストを表示する．

show region <region_name>
リージョンの情報を表示する．

shutdown
サーバプログラムを終了する．

D.2 DTL マネーサーバ

set log level <level>
　コンソールのロギングレベルを設定する.

show info
　サーバプロセスの一般情報を表示する.

show stats
　サーバプロセスの統計情報を表示する.

show threads
　スレッドの状態を表示する.

show uptime
　サーバプロセスの稼動時間を表示する.

show version
　サーバプロセスのバージョンを表示する.

D.3　リージョンサーバ

D.3.1 メインコマンド

alert <first_name> <last_name> <message>
　名前が <first_name> <last_name>のアバターにメッセージ<message>を送る.

alert general <message>
　全てのアバターにメッセージ<message>を送る.

backup
　データベースへ現在のオブジェクトのデータを送る. Unix の sync コマンドのようなもの？

bypass permissions (true | false)
　オブジェクトに対してパーミッションのチェックを回避するうかどうかを指定する. trueなら回避を行う. falseなら回避を行わない. つまりtrue にすると, 他人の作成したオブジェクトが編集可能になる. デフォルトはfalse. 設定が有効になるまでに若干のタイムラグ (？) 有り. このコマンドはOpenSim.iniでオブジェクトのパーミッションの設定が行われている場合にのみ意味を持つ.

change region <region_name>
　マルチリージョンの場合に, 操作 (コマンド入力) 対象のリージョンを変更する. リージョン名に rootを指定した場合は, リージョン全体が操作対象となる (デフォルト).

clear assets
　アセットのキャッシュをクリアする.

command-script <file>
　コマンドの記述されたファイル<file> を実行する

config get <section> <field>
　現在の設定の <section>セクションの <filed>の値を表示する.

config save <file>
　現在の設定を <file> に保存する.

config set <section> <field> <value>
　設定のうち, <section> セクションの <field> の値を <val> に変更する. 通常, 設定は OpenSim.ini から読み込まれるが, このコマンドで変更することが可能である.

create region <region_name> <region_file.ini>
　新しいリージョンを作成する. Regionsディレクトリに設定ファイル <region_file.ini>も作成する. なお, リージョン名は対話的に入力するため, ここで入力された <region_name>は無視される.

debug packet <level>
　パケットのデバッグのレベルを指定する.

debug permissions (true | false)

パーミッションのデバッグを有効にするか？

debug scene <scripting> <collisions> <physics>

Scene のデバッグを行うかどうか？ < > にはそれぞれ， true か false を指定する．

delete-region <region_name>

リージョン <region_name>を削除する．リージョン設定ファイルは残っているので，サーバププロセスを立ち上げ直せば， 土地は元に戻るが， オブジェクトは削除される． remove-region と同じ？

dump asset <uuid> <file> - dump one cached asset

キャッシュさているID <uuid>のアセットをファイル <file>にダンプする．

edit scale <name> <x> <y> <z>

<name>の名前のプリムの大きさを変更する．

export <command>

export サブコマンドを実行する．

export-map [<file>]

現在のリージョンのマップ画像をJPEGデータとして <file>に出力する．<file>を省略した場合，exportmap.jpgに出力される．マルチリージョンの場合は，change regionで特定のリージョンに変更しないとこのコマンドは実行できない．

fcache assets

全てのアセットのキャッシュを試みる．

fcache clear (file | memory)

ファイルおよびメモリのキャッシュ上の全てのアセットデータをクリアする．

fcache expire <datetime>

定された日時よりも古いアセットのキャッシュを削除する．

fcache status

アセットのキャッシュの状態を表示する．

fix phantom objects

メガリージョンにファントムプリムをインポートした場合にファントムプリムの不具合を修正する．

force permissions (true | false)

falseにするとオブジェクトの作成，編集およびRezができなくなる．デフォルトはtrue．bypass permissions が false の場合は設定できない．

force update

全てのプリムデータをクライアント (ビューア) に送信し直す．

help [<command>]

ヘルプを表示する． コマンドを指定した場合は， そのコマンドのヘルプを表示する．

help export

export サブコマンドのヘルプを表示する．

help terrain

terrain サブコマンドのヘルプを表示する．

help tree

tree サブコマンドのヘルプを表示する．

help windlight

windlight サブコマンドのヘルプを表示する．

kick user <first_name> <last_name> [message]

<first_name> <last_name>のアバターをリージョンから追い出す．その際メッセージ[<message>]を表示する(メッセージは省略可)．

kill uuid <uuid>

ID が <uuid> であるオブジェクトを削除する.

link-region <Xloc> <Yloc> <HostName>:<HttpPort>[:<RemoteRegionName>]

リモートのリージョン(<HostName>:<HttpPort>[:<RemoteRegionName>])を(<Xloc>,<Yloc>)座標のローカルなリージョンとして割り当てる(ハイパーグリッドリージョン).

load iar <first_name> <last_name> <inventory_path> <password> [<IAR_file>]

save iar で保存したアバターのインベントリを IAR(inventory archive)ファイル から inventory_path に読み込む. inventory_path はアバターの「持ち物」の My Inventoryフォルダー以下のフォルダを / で区切って指定する. 例えば, inventory_path を Objects/load とすれば, My Inventor/Objectsフォルダの下に loadフォルダが作成され, それ以下にデータが読み込まれる. またMy Inventoryフォルダー自体を指定するには inventory_pathを / とする. なおこのコマンドはアバターがリージョンにログインしていないと使用できない.

load oar [--merge] [--skip-assets] [<OAR_file>]

OARフォーマットのリージョンデータファイル(リージョン全体の状態を保存したファイル)を読み込んで, 保存された状態を復元する. --merge を指定した場合は, 地形情報は読み込まれず, 現リージョンのオブジェクトも削除されない. また, --skip-assetsを指定した場合はアセット情報は読み込まれない. OAR_fileを省略した場合のファイル名は region.oar となる.

load xml [-newIDs [<x> <y> <z>]] <file>

XMLフォーマットのファイル<file>からリージョンデータを読み込む. このコマンドは近々廃止される予定なので, load xml2の方を使用すべきである.

load xml2 [<file>]

XML2フォーマットのリージョンデータを読み込む. ファイル名を省略した場合は prim-backup.xmlになる.

login disable

ログインを一時的に不能にする.

login enable

ログインを有効にする.

login status

ログイン情報を表示する.

modules list

読み込んでいるモジュールのリストを表示する.

modules load <name>

モジュール<name>を読み込む.

modules unload <name>

読み込んでいるモジュール<name>を削除する.

monitor

リージョンに関する様々な統計情報を表示する.

quit

サーバプログラムを終了する.

reload estate

エステートデータをデータベースより再読み込みする.

remove-region <region_name>

<region_name>リージョンを削除する. リージョン設定ファイルは残っているので, サーバププロセスを立ち上げ直せば, 土地は元に戻るが, オブジェクトは削除される. delete-regionと同じ?

restart

そのマシンで稼動している全てのリージョンを再起動する.

save iar <first_name> <last_name> <inventory_path> <password> [<IAR_file>]

アバターのinventory_pathにあるインベントリをファイルに保存する．inventory_path はアバターの「持ち物」の My Inventoryフォルダー以下のフォルダを / で区切って指定する．例えば，アバターのMy Inventory/Objects/ Save にあるインベントリを保存したい場合は Objects/Save と指定する．My Inventory自体を保存したい場合は / を指定する．ただし，現バージョンではインベントリ単体を保存することはできないようである．〈IAR_file〉を省略するとuser-inventory.iarとなる．また既にあるファイルを指定するとエラーになる．

save oar [〈OAR_file〉]

リージョン全体の状態を OARファイルとして保存する．OAR_fileを省略した場合のファイル名は region.oar となる．

save prims xml2

リージョン上のプリムデータを prim-backup.xmlへXML2フォーマットで保存する．

save xml 〈file〉

リージョンデータを〈file〉へXMLフォーマットで保存する．このコマンドは近々廃止される予定であるので，save xml2の方を使用すべきである．

save xml2 [〈file〉]

リージョンデータをXML2フォーマットで保存する．ファイル名を省略した場合は prim-backup.xml に保存される．

set log level 〈level〉

コンソールのロギングレベルを設定する．

set region flags 〈region_name〉 〈flags〉

現在のバージョンでは，リージョンサーバからのこのコマンドは正常に動作しない模様．

set terrain heights 〈corner〉 〈min〉 〈max〉 [〈x〉 〈y〉]

土地のテクスチャでの設定において，テクスチャの標高範囲を指定する．〈corner〉が0の場合は南西を，1の場合は北西を，2の場合は南東を，3の場合は北東を表す．〈x〉と〈y〉が指定されている場合は座標が一致している箇所にのみ設定が行われる．〈x〉または〈y〉に-1を設定した場合はワイルドカードとして取り扱われる．

set terrain texture 〈number〉 〈uuid〉 [〈x〉 〈y〉]

土地の標高テクスチャでの設定を行う．〈number〉は1が低，4が高を表す．また〈uuid〉はその標高に設定するテクスチャのUUIDである．〈x〉と〈y〉が指定されている場合は座標が一致している箇所にのみ設定が行われる．〈x〉または〈y〉に-1を設定した場合はワイルドカードとして取り扱われる．

show connections

アバターの接続状態を表示する．

show hyperlinks

ハイパーグリッドのリージョンのリストを表示する．

show info

サーバプロセスの一般情報を表示する．

show modules - Show module data

読み込んでいるモジュールの情報を表示する． modules listの結果とほぼ同じ？

show neighbours

隣接リージョンの情報を表示する．

show queues

それぞれのキューの状態を表示する．

show ratings

リージョンの種類（PGやMATUREなど）を表示する．

show region 〈region_name〉

指定されたリージョンの詳細な情報を表示する．

show regions

サーバで作動しているリージョンのリストを表示する．

show stats

サーバプロセスの統計情報を表示する．

show threads
　スレッドの状態を表示する.

show uptime
　サーバプロセスの稼動時間を表示する.

show users
　ログインしているアバターの情報を表示する.

show version
　サーバのバージョンを表示する.

shutdown
　サーバプログラムを終了する.

sun [param] [value]
　Sunモジュールのパラメータを指定する.

terrain <command>
　terrain サブコマンドを実行する.

tree <command>
　tree サブコマンドを実行する.

unlink-region (<local_region_name> | <HostName>:<HttpPort>)
　ハイパーグリッドリージョンを切り離す.

windlight <command>
　windlight サブコマンドを実行する.

D.3.2 exportサブコマンド

export save <region_name>
　<region_name>で指定したリージョンのデータを exportsディレクトリに保存する.

export save-all
　管理する全てのリージョンのデータを exportsディレクトリに保存する. 動作時に大量のメモリを消費する.

D.3.3 terrainサブコマンド

terrain load <file>
　指定したファイルから土地の標高データを読み込む.

terrain load-tile <file> <width> <height> <minX> <minY>
　256x256 より大きなデータからその一部を標高データとして読み出す. 読み出す範囲は (<minx>,<minY>)-(<minX>+<width>,<minY>+<height>) である.

terrain save <file>
　現在の標高データを<file>に書き出す.

terrain fill <height>
　リージョン全体を一定の高さ<height>メートルの平坦な土地にする. 実数も指定可能だが, 小数点以下は切り上げになる模様.

terrain elevate <amount>
　現在の土地の標高を全体的に <amount>メートル上昇させる. 実数で指定する.

terrain lower <amount>
　現在の土地の標高を全体的に <amount>メートル下降させる. 実数で指定する.

terrain multiply <amount>
　現在の土地の標高に <amoun>を乗算する.

terrain bake

現在の標高データを復帰用のデータとする.

terrain revert

復帰用の標高データを読み込む.

terrain newbrushes (true | false)

実験用の土地ブラシ（ブルドーザーのこと？）を使用するかどうか指定する．なお，この設定はデバッグ用であり，予告なしに削除される可能性がある.

terrain stats

土地の標高に関する情報を表示する.

terrain effect list

現在使用可能なプラグイン・エフェクトを表示する．プラグイン・エフェクトは土地の造成に色々な効果を与えるフィルターである.

terrain effect reload

プラグイン・エフェクトを再読み込みする.

terrain flip (x | y)

現在の土地のＸ座標またはＹ座標を反転させる.

terrain rescale <min> <max>

現在の土地の標高が〈min〉～〈max〉の間になるように，リスケールする．〈min〉,〈max〉は実数を指定する.

D.3.4 treeサブコマンド

tree active (true | false)

ツリーモジュールの有効・無効，つまり植物の成長（seed, growth, dieサイクル）の有無を指定する．デフォルトはfalse.

tree freeze <copse> (true | false)

〈copse〉で定義された植物の成長（seed, growth, dieサイクル）を凍結するかどうかを指定する.

tree load <file>

〈copse〉の定義をXMLファイルから読み込む.

tree plant <copse>

〈copse〉で定義された植物を一つ植える.

tree rate <updateRate>

植物の成長感覚を msec で指定する.

tree reload

リージョン内に植えられている直物(copse)のデータを再読み込みする.

tree remove <copse>

〈copse〉で定義された植物を全て削除する.

tree statistics

植物（copse）についての統計データを表示する.

D.3.5 windlightサブコマンド

windlight load

データベースから WindLight用プロファイルデータを読み込む.

windlight enable

WindLightのモジュールを読み込む.

windlight disable

読み込み済みの WindLightモジュールを削除する.

D.4 コマンド例

D.4.1 新規アバターの作成

D.4.2 新規リージョンの作成

```
R.O.B.U.S.T.# create user
First name [Default]: Test
Last name [User]: Avatar
Password*******                    (入力中はパスワードは表示されない)
Email []:
16:37:10 - [AUTHENTICATION DB]: Set password for principalID a9cfc722-1deb-46f1-8b00-
37f8cf284c2f
16:37:10 - [GRID SERVICE]: GetDefaultRegions returning 0 regions
16:37:10 - [USER ACCOUNT SERVICE]: Unable to set home for account Test Avatar.
16:37:10 - [USER ACCOUNT SERVICE]: Account Test Avatar created successfully
R.O.B.U.S.T.#
```

図 D.1 create user コマンド

```
Region (root) # create region temp_name Test.ini
====================================

We are now going to ask a couple of questions about your region.

You can press 'enter' without typing anything to use the default

the default is displayed between [ ]: brackets.
====================================

New region name []: TEST_SIM_02
Region UUID [3a24c676-4fa4-401c-bfea-4e4945b48c2c]:
Region Location [1000,1000]: 1000,1001
Internal IP address [0.0.0.0]:
Internal port [9000]: 9001
Allow alternate ports [False]:
External host name [SYSTEMIP]: 202.26.159.211
16:51:46 - [LLUDPSERVER]: Average Environment.TickCount resolution: 1ms
Your region is not part of an estate.
Do you wish to join an existing estate? [no]:
New estate name [My Estate]:
.................
.................
The current estate has no owner set.
Estate owner first name [Test]: Fumi
Estate owner last name [User]: Hax
.................
.................
16:57:22 - [GRID SERVICE]: Region TEST_SIM_02 (3a24c676-4fa4-401c-bfea-4e4945b48c2c)
registered successfully at 256000-256256
.................
Region (root) #
```

図 D.2 create region コマンド

リージョンが自動起動され， bin/Regions/Test.ini も生成される.

索引

記号

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

OpenSimサーバシステム構築入門

-- & ネットワークサーバ構築入門 --

2011年12月26日　第1.1.0版

東京情報大学
総合情報学部　情報システム学科
NSL（ネットワークシステム研究室）メタバース研究会

連絡先
http://www.nsl.tuis.ac.jp
http://www.opensim.tuis.ac.jp