ARM シリーズボード上
Qtopia 組込開発マニュアル

株式会社日昇テクノロジー

http://www.csun.co.jp

info@csun.co.jp

2010/7/18

- 修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2010/07/17

※ 使用されたソースコードはhttp://www.csun.co.jp/からダウンロードできます。
※ この文書の情報は、事前の通知なく変更されることがあります。

第一章　Qtopia 開発概要

1.1 概要

Qt（キュート）は C++言語で書かれたアプリケーション・ユーザインタフェース（UI）フレームワークである。GUI ツールキットとして広く知られている Qt であるが、コン ソールツールやサーバのような非 GUI プログラムでも広く使用されている。ノキアの一部門 Qt デベロップメントフレームワークス社に よって開発されている。
ライセンスには商用版とオープンソース版があり、現在のオープンソース版のライセンスは LGPL（Qt4.5 より）および GPL である。商用版を購入すると Qt 商用ライセンス（Qt Commercial Developer License）でソフトウェアを開発することができる。LGPL 版は、2009 年 3 月にリリースされた Qt 4.5 から提供され始めた。これにより Qt は営利企業にとってもより使いやすいライブラリーとなった。

日本では SRA が Qt デベロップメントフレームワークス社の パートナーとなり、関連サービスの販売を行っている。その他に、アイ・エス・ビーも パートナー契約を結んでいる。

Qt は C++で開発されており、単独のソースコードにより X Window System（Linux, UNIX 等）、Windows、Mac OS X、組み込みシステムといった様々なプラットフォーム上で稼働するアプリケーションの開発が可能である。 またコミュニティーにより多言語のバインディングが開発されており、Java から Qt を利用できる ようにした Qt Jambi、さらに Qt を Ruby、Python、Perl、C#な どから利用できるようにしたオープンソースの API が存在する。

このような開発の容易さに加えて高速、スタイリッシュな Qt は、オープンソース版、商用版を合わせて、世界中に 35 万人の開発者がいると 言われている。
Qt は GTK+や MFC 等、他の標準的なグラフィックツールキットに比べて、もっとも後発であることもあり、以前から存在するライブラリーのよいと ころを集めたアーキテクチャーとなっている。そのため、商業アプリケーションでの採用例が多い他、オープンソース版も用意されているおかげで、KDE という高品質なデスクトップ 環境も開発された。OpenGL や SVG、XML といった最新技術にも対応している他、日本語を含む多バイト文字入力フレームワークへも対応している。

1.2 商用版の概要

商用版の Qt には Console Edition、Light Edition、Desktop Edition の三つの形態があり、以下の違いがある。

機能	Console Edition	Light Edition	Desktop Edition
Qt コアクラス	○	○	○
Qt GUI クラス		○	○
ネットワーキング	○		○
OpenGL			○
データベース/SQL	○		○
SVG			○
XML	○		○
Qt3 サポート		部分的	○
Qt Designer 拡張クラス			○
単体テストフレームワーク	○	○	○
ActiveQt			○

1.3 オープンソース版

GPL または LGPL が適用される。LGPL は、バージョン 4.5 から適用できる。Windows や多くの UNIX 系 OS、Mac OS X 向け、あるいは Embedded Linux、Windows CE、Symbian（Qt4.6 より）向けにパッケージが配布されている。

1.4 開発環境・デザインツール

クロスプラットフォームの統合開発環境 Qt Creator、GUI エディタの Qt Designer、翻訳支援ツールの Qt Linguist、リファレンスドキュメントビューアの Qt Assistant 等の開発支援ツールが付属しており、これらを使用することで高速な開発が可能となっている。その他のものとして Windows の Visual Studio での開発を可能にするプラグイン Visual Studio Add-in が用意されている。また Java で作られているクロスプラットフォームの開発環境 Eclipse（統合開発環境）上で開発を可能にする Qt Eclipse Integration も用意されている。また、Unix/X11（Linux など）では、KDevelop が使用できる。

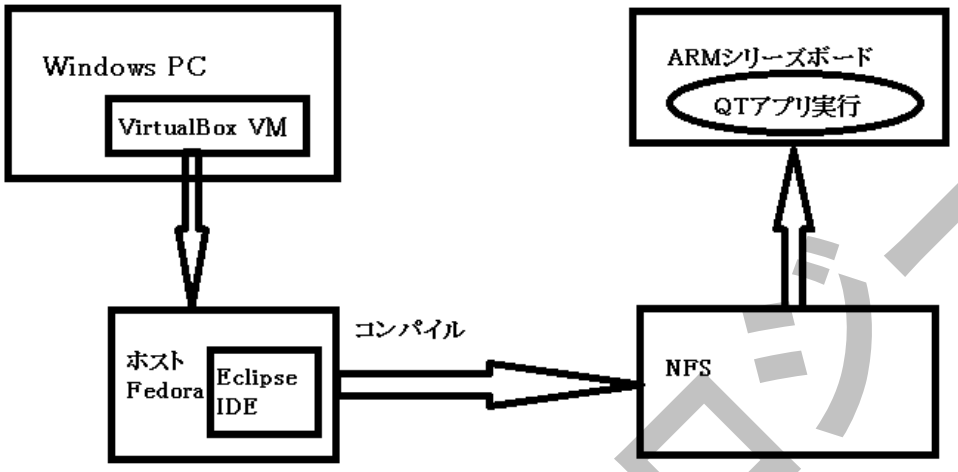
Qt/UNIX 上では GCC、Qt/Windows では Microsoft Visual Studio 上のコンパイラが使える他、MinGW 等のコンパイラでの開発も可能である。

1.5 本マニュアル説明対象

本マニュアルは弊社の ARM シリーズボード（Mini2440、Micro2440、Mini6410）に搭載されている Qtopia2.2 に基づき、開発環境構築から ARM ボード上 QT アプリの実行まで説明する。

第二章　開発環境構築に必要もの

2.1 開発環境イメージ



まず、開発用 PC に無料 VM ソフト「VirtualBOX」をインストールする。次に、VM マシンに「Fedora」ホストをインストールし、ホスト上に開発環境「Eclipse」と NFS を設定する。Eclipse は QT アプリ開発、コンパイル用の IDE となり、NFS は ARM シリーズボードからホストにコンパイルされたアプリをアクセスするためのネットワークとなる。コンパイル完了後、ARM シリーズボードから QT アプリを実行する。

2.2 必要もの取得

１．VM ツール「Virtual BOX」

ダウンロード URL:

http://www.virtualbox.org/wiki/Downloads

２．ホスト OS：Fedora

ダウンロード URL:

http://fedoraproject.org/ja/get-fedora

３．JRE：

ダウンロード：

http://java.sun.com/javase/downloads/index.jsp

４．Eclipse

ダウンロード URL：

http://www.eclipse.org/downloads/?osType=linux

＊名前：Eclipse IDE for C/C++ Developers

５．CDT プラグイン：

http://www.eclipse.org/cdt/downloads.php

６．ARM 用 Qtpoia：(最終 QT PDA バージョン：Qtopia2.2.0)

ダウンロード URL：

http://www.dragonwake.com/download/arm9-download/qt/arm-qtopia-2.2.0.tar.gz

７．Qt Eclipse Integration for C++

http://qt.nokia.com/developer/eclipse-integration

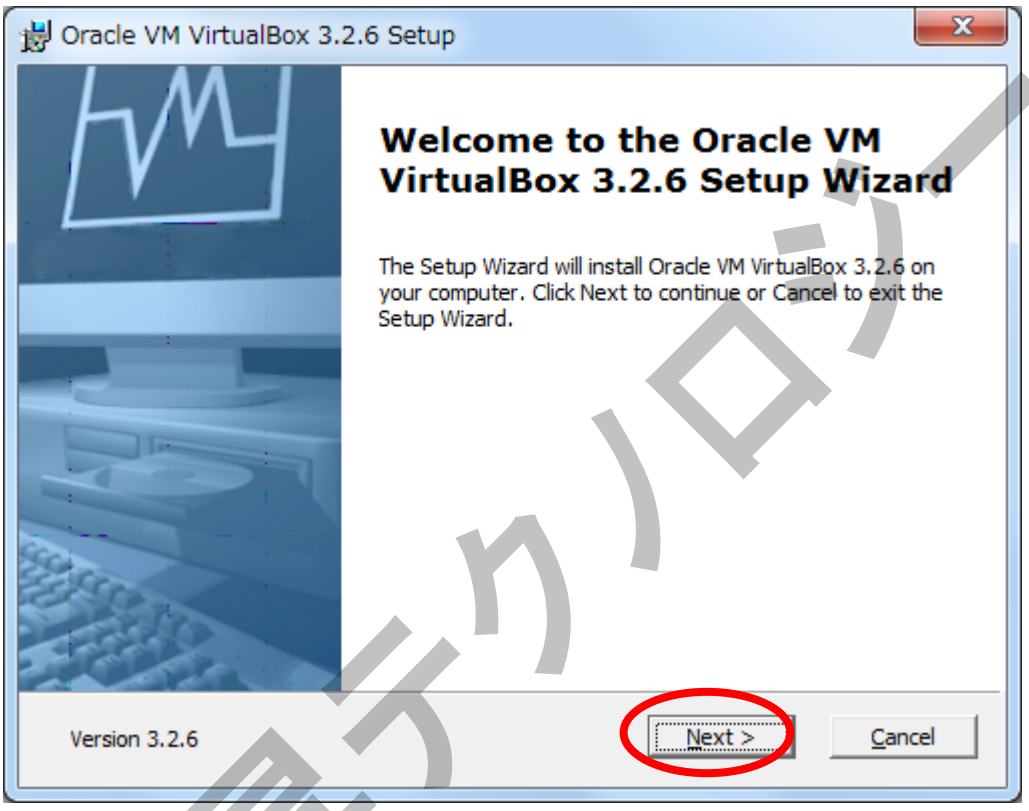
８．ARM コンパイラ

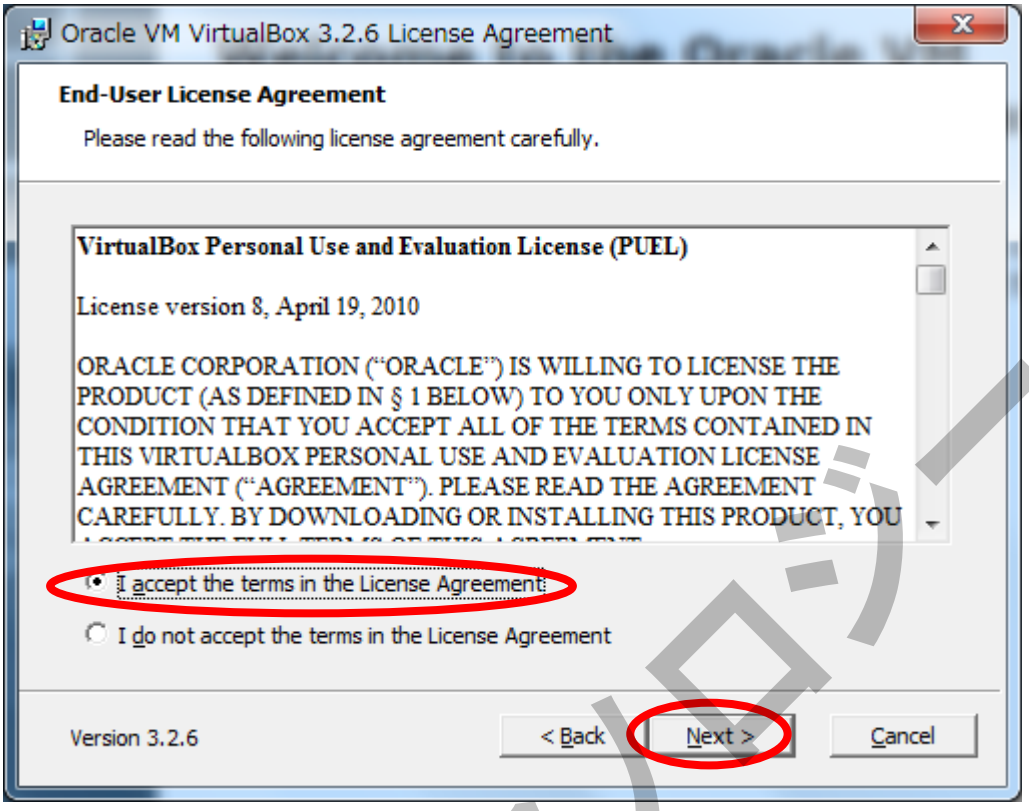
http://www.dragonwake.com/download/arm9-download/linux-toolchain/arm-linux-gcc-4.3.2.tgz

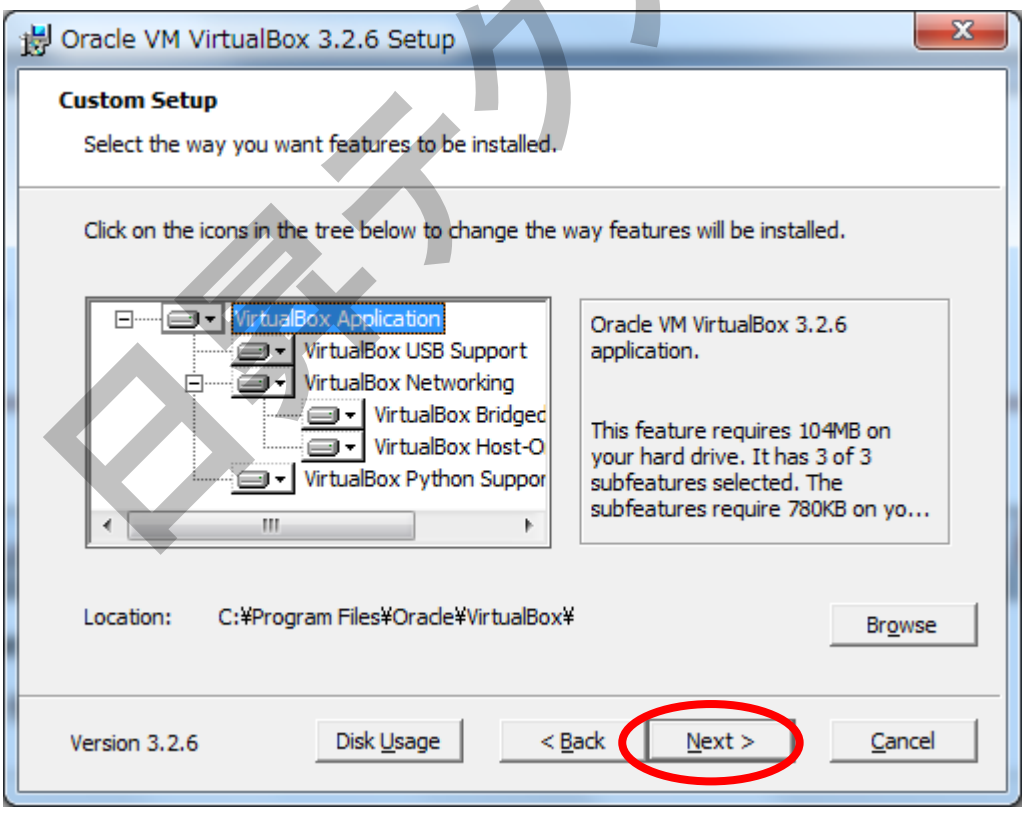
第三章　開発環境構築

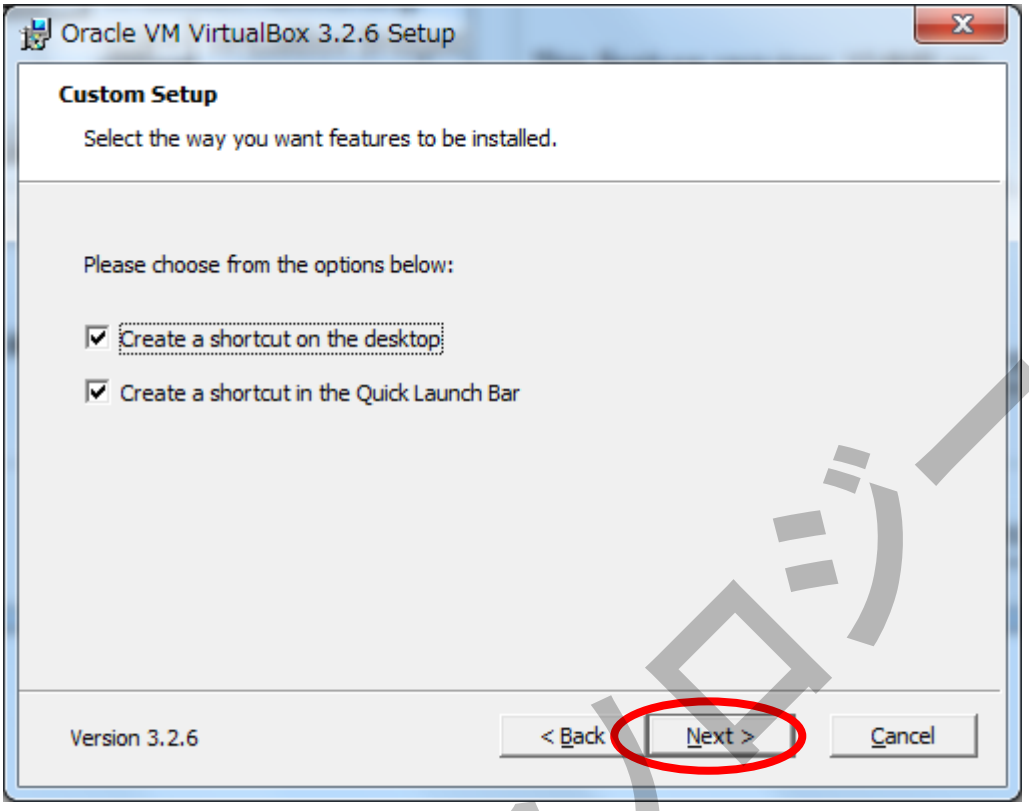
3.1 VM ツール VirtualBox インストール

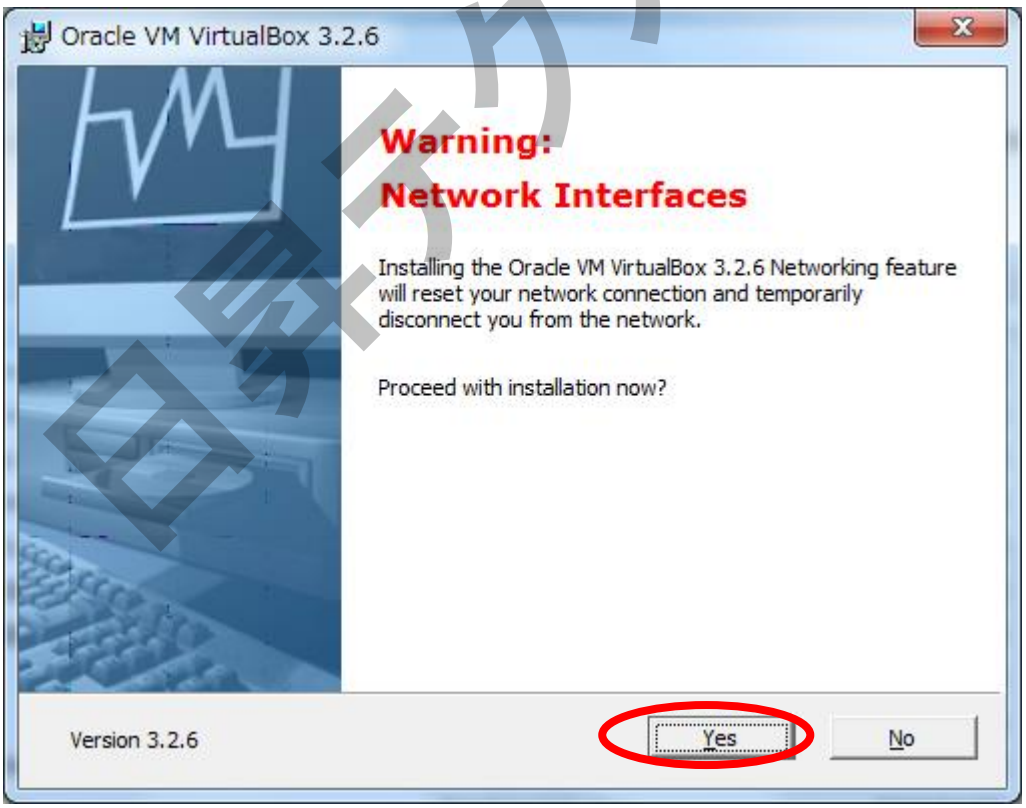
http://www.virtualbox.org/wiki/Downloads から VirtualBox をダウンロードする。
ダウンロードしたインストールファイルをクリックすると、下記画面に従って進めてください。

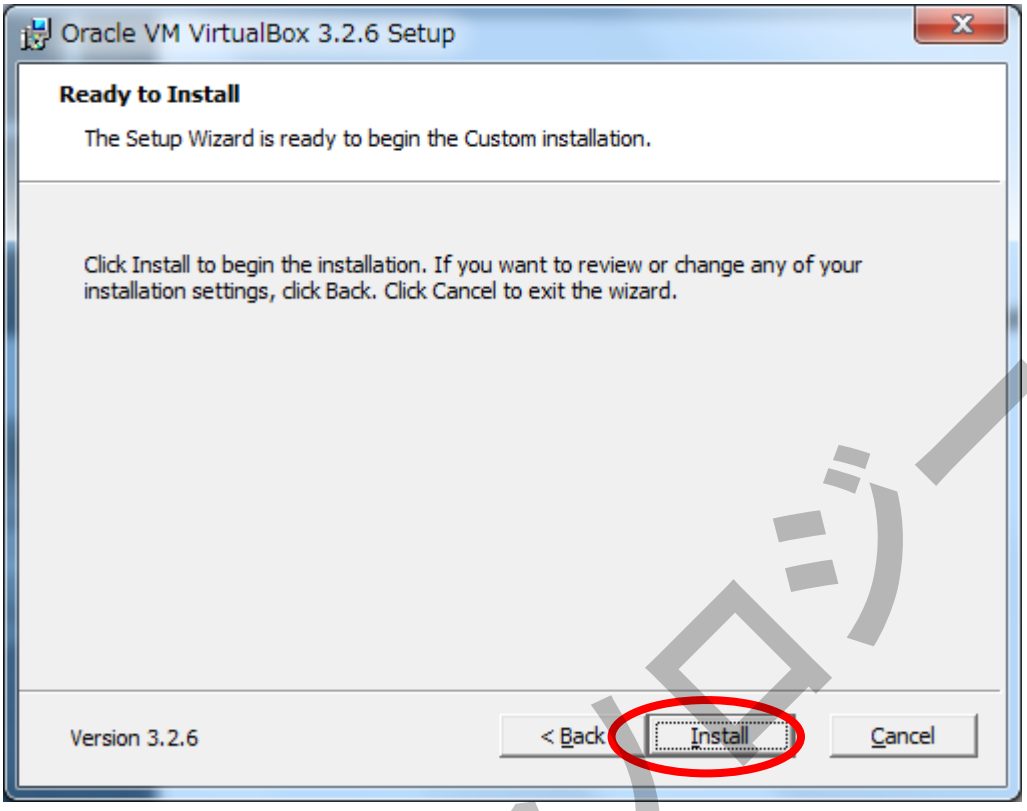



Oracle VM VirtualBox 3.2.6 License Agreement
End-User License Agreement
Please read the following license agreement carefully.
VirtualBox Personal Use and Evaluation License (PUEL)
License version 8, April 19, 2010
ORACLE CORPORATION ("ORACLE") IS WILLING TO LICENSE THE PRODUCT (AS DEFINED IN § 1 BELOW) TO YOU ONLY UPON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS VIRTUALBOX PERSONAL USE AND EVALUATION LICENSE AGREEMENT ("AGREEMENT"). PLEASE READ THE AGREEMENT CAREFULLY. BY DOWNLOADING OR INSTALLING THIS PRODUCT, YOU
I accept the terms in the License Agreement
I do not accept the terms in the License Agreement
Version 3.2.6
< Back
Next >
Cancel

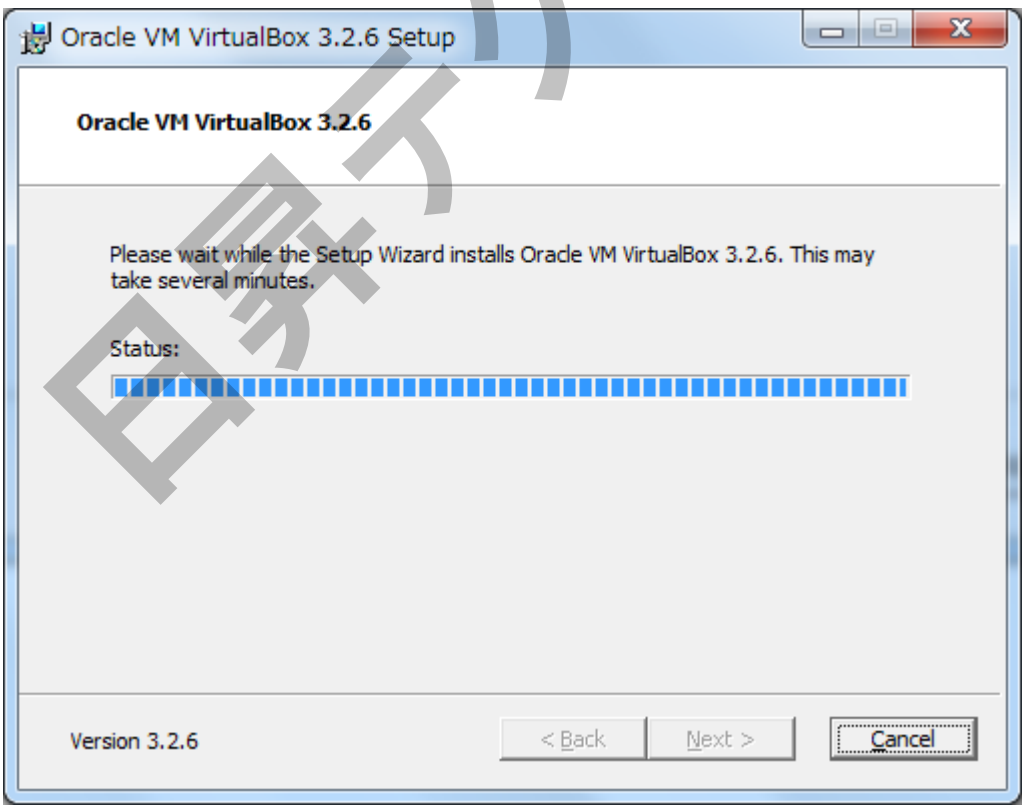

Oracle VM VirtualBox 3.2.6 Setup
Custom Setup
Select the way you want features to be installed.
Click on the icons in the tree below to change the way features will be installed.
VirtualBox Application
VirtualBox USB Support
VirtualBox Networking
VirtualBox Bridged
VirtualBox Host-O
VirtualBox Python Suppor
Oracle VM VirtualBox 3.2.6 application.
This feature requires 104MB on your hard drive. It has 3 of 3 subfeatures selected. The subfeatures require 780KB on yo...
Location: C:¥Program Files¥Oracle¥VirtualBox¥
Browse
Version 3.2.6
Disk Usage
< Back
Next >
Cancel

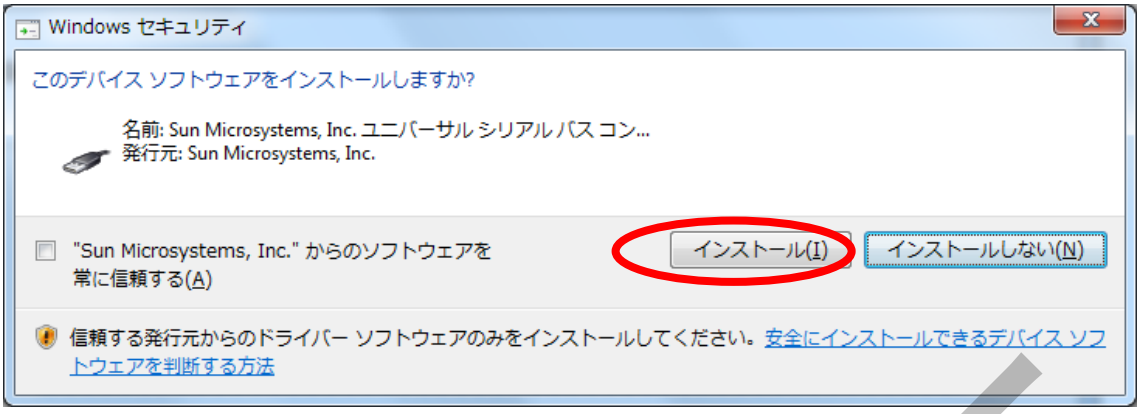

Oracle VM VirtualBox 3.2.6 Setup
Custom Setup
Select the way you want features to be installed.
Please choose from the options below:
Create a shortcut on the desktop
Create a shortcut in the Quick Launch Bar
Version 3.2.6
< Back
Next >
Cancel

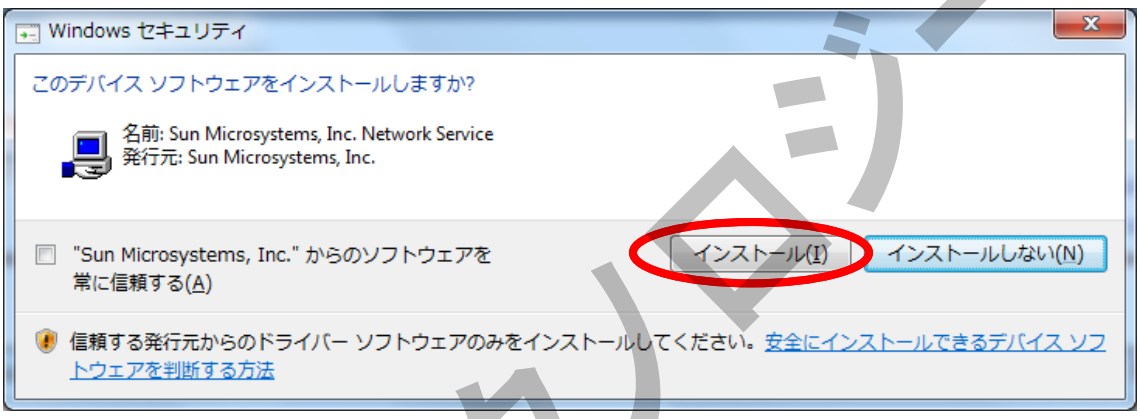

Oracle VM VirtualBox 3.2.6
Warning:
Network Interfaces
Installing the Oracle VM VirtualBox 3.2.6 Networking feature will reset your network connection and temporarily disconnect you from the network.
Proceed with installation now?
Version 3.2.6
Yes
No

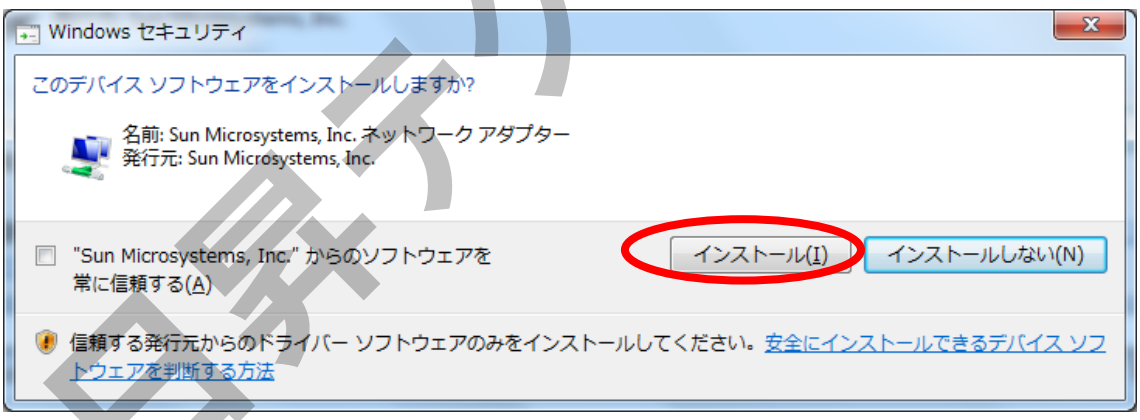


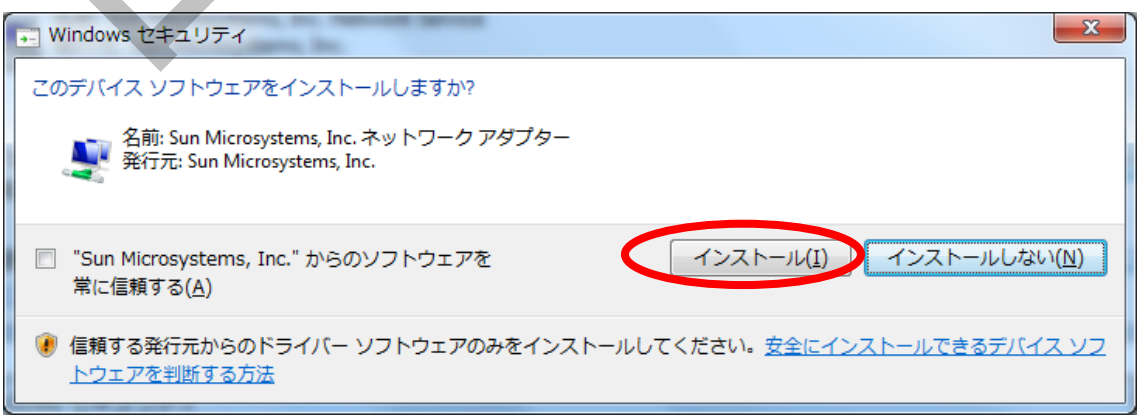
インストール中

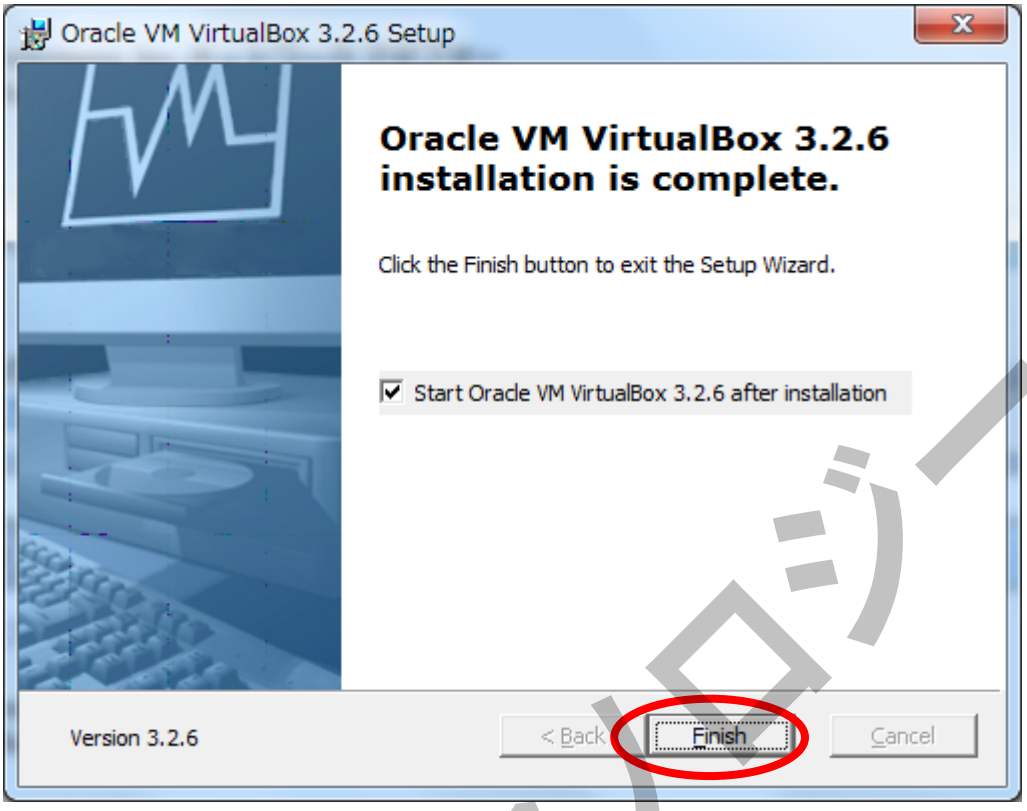



Windows セキュリティ
このデバイス ソフトウェアをインストールしますか?
名前: Sun Microsystems, Inc. ユニバーサル シリアル バス コン...
発行元: Sun Microsystems, Inc.
"Sun Microsystems, Inc." からのソフトウェアを常に信頼する(A)
インストール(I)
インストールしない(N)
信頼する発行元からのドライバー ソフトウェアのみをインストールしてください。安全にインストールできるデバイス ソフトウェアを判断する方法

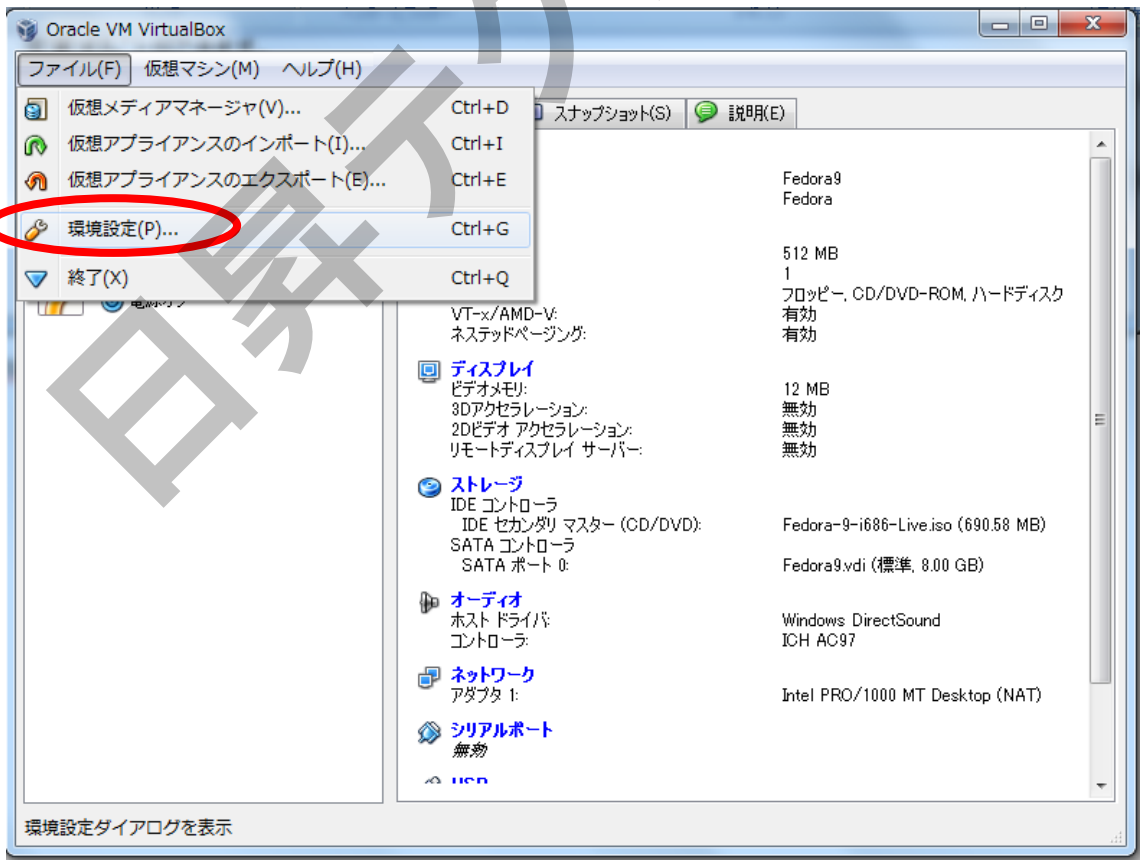

Windows セキュリティ
このデバイス ソフトウェアをインストールしますか?
名前: Sun Microsystems, Inc. Network Service
発行元: Sun Microsystems, Inc.
"Sun Microsystems, Inc." からのソフトウェアを常に信頼する(A)
インストール(I)
インストールしない(N)
信頼する発行元からのドライバー ソフトウェアのみをインストールしてください。安全にインストールできるデバイス ソフトウェアを判断する方法

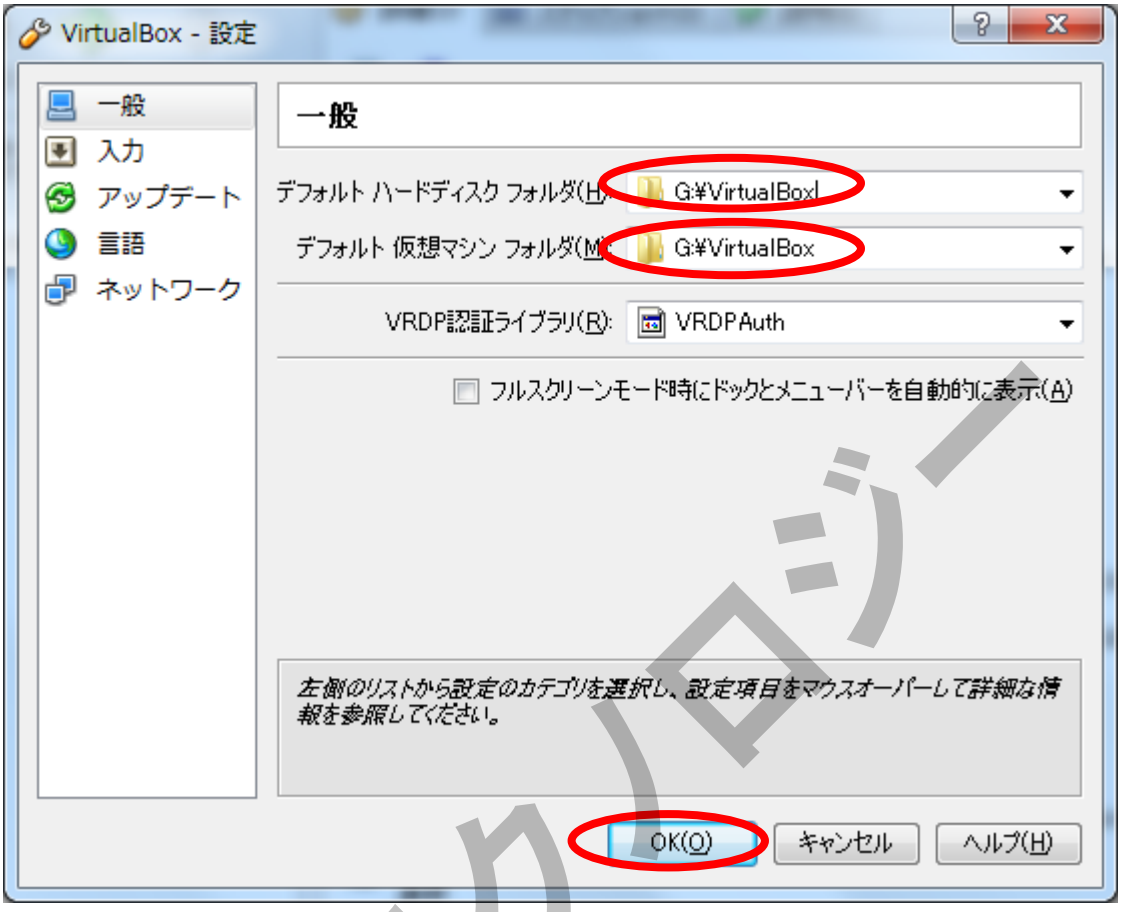

Windows セキュリティ
このデバイス ソフトウェアをインストールしますか?
名前: Sun Microsystems, Inc. ネットワーク アダプター
発行元: Sun Microsystems, Inc.
"Sun Microsystems, Inc." からのソフトウェアを常に信頼する(A)
インストール(I)
インストールしない(N)
信頼する発行元からのドライバー ソフトウェアのみをインストールしてください。安全にインストールできるデバイス ソフトウェアを判断する方法


Windows セキュリティ
このデバイス ソフトウェアをインストールしますか?
名前: Sun Microsystems, Inc. ネットワーク アダプター
発行元: Sun Microsystems, Inc.
"Sun Microsystems, Inc." からのソフトウェアを常に信頼する(A)
インストール(I)
インストールしない(N)
信頼する発行元からのドライバー ソフトウェアのみをインストールしてください。安全にインストールできるデバイス ソフトウェアを判断する方法



Virtual 環境設定

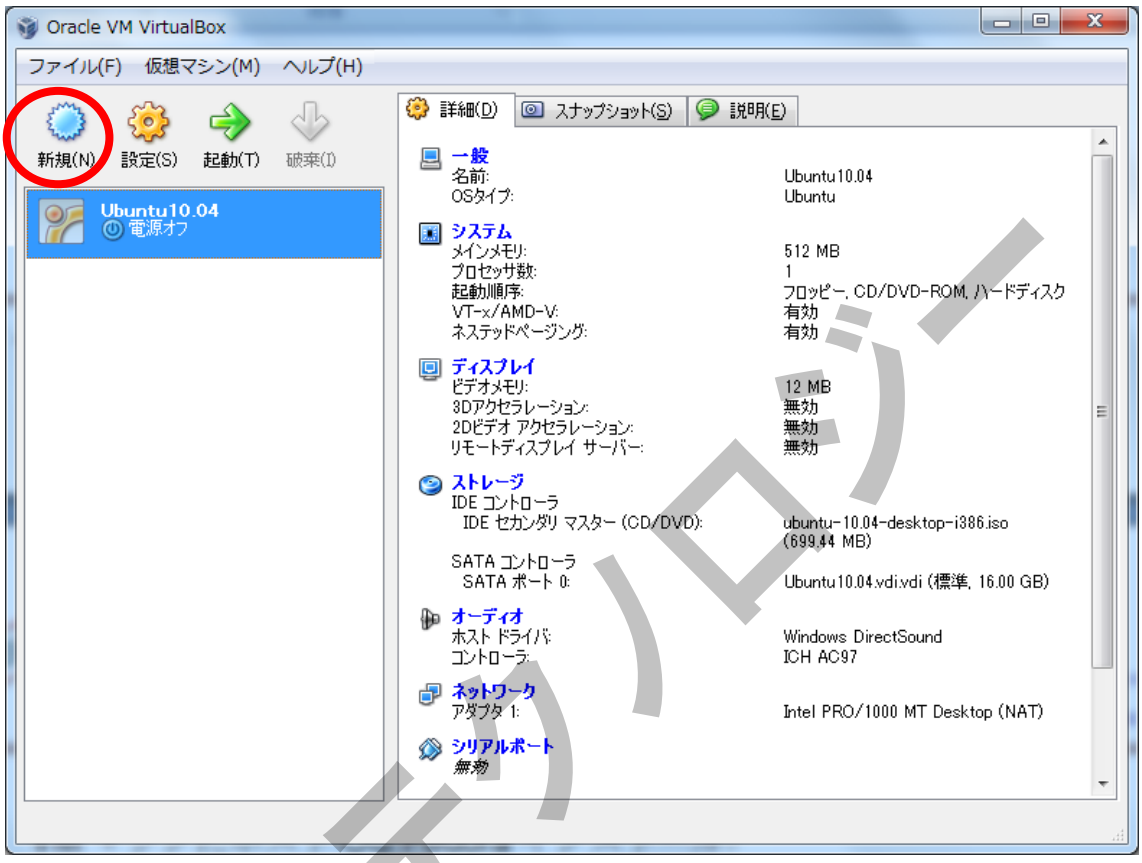



VirtualBox - 設定
一般
入力
アップデート
言語
ネットワーク
一般
デフォルト ハードディスク フォルダ(H):
G:¥VirtualBox¥
デフォルト 仮想マシン フォルダ(M):
G:¥VirtualBox
VRDP認証ライブラリ(R):
VRDPAuth
フルスクリーンモード時にドックとメニューバーを自動的に表示(A)
左側のリストから設定のカテゴリを選択し、設定項目をマウスオーバーして詳細な情報を参照してください。
OK(O)
キャンセル
ヘルプ(H)

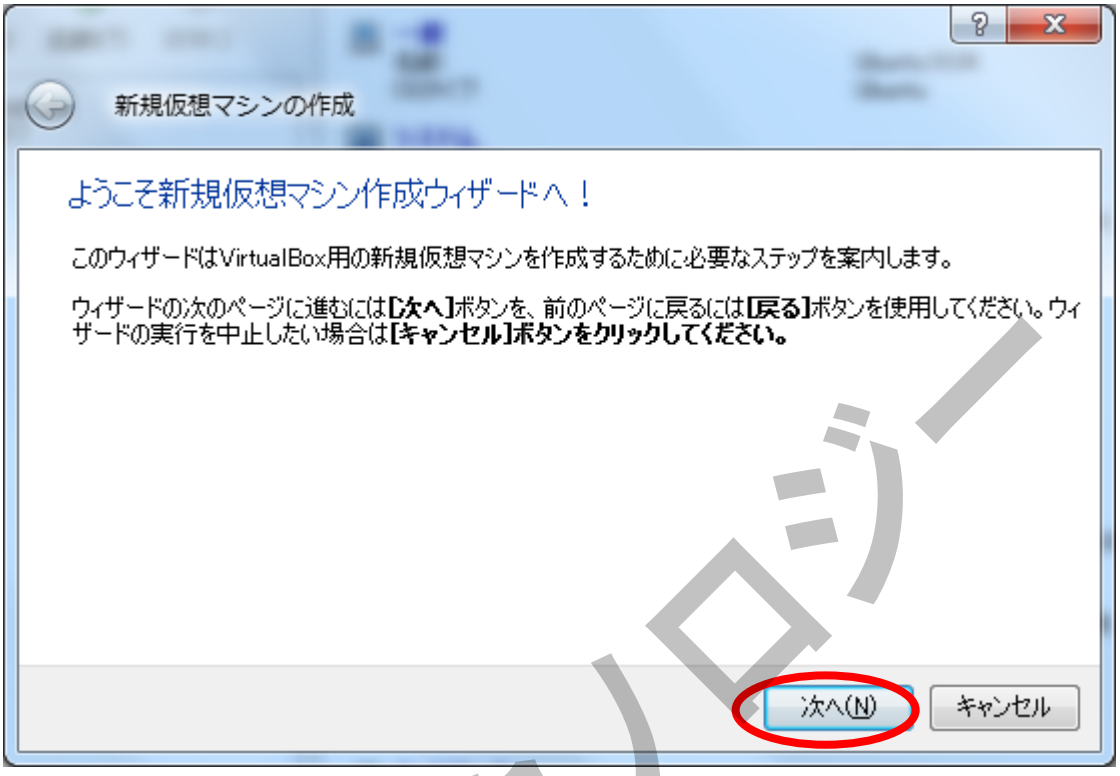
3.2 VM マシンにホスト OS Fedora インストール

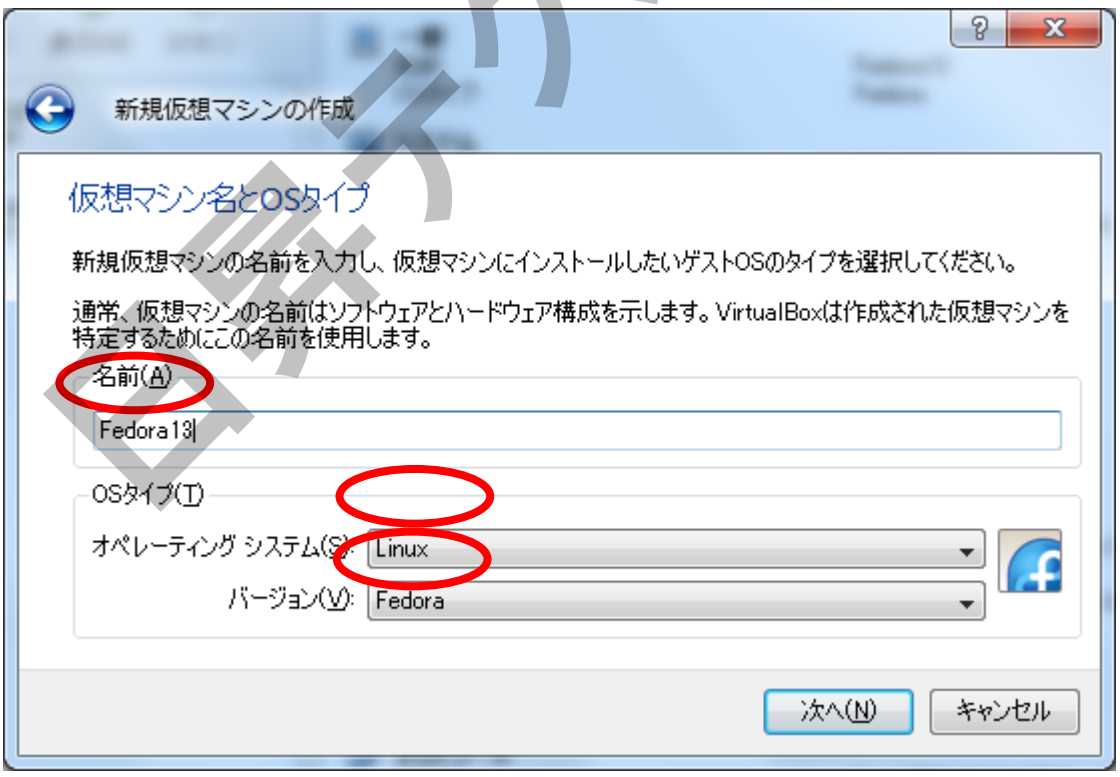
http://fedoraproject.org/ja/get-fedora から Fedora13（最新版）をダウンロードする。

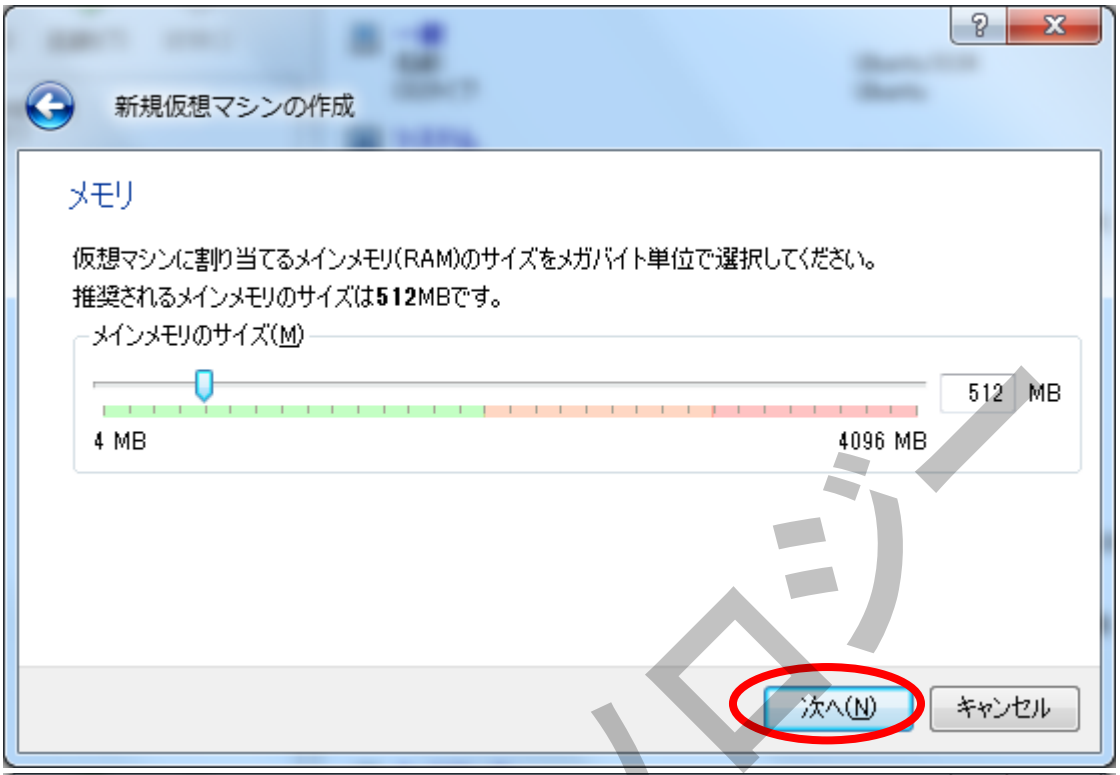
1．VirtualBox を起動

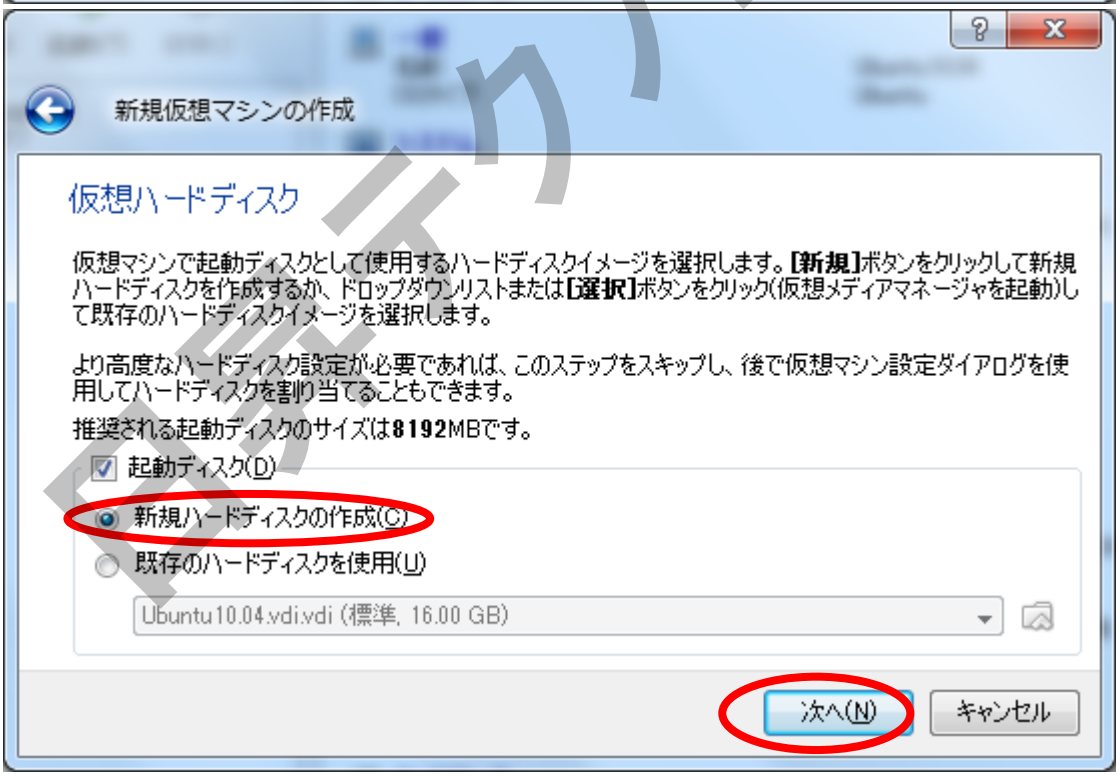


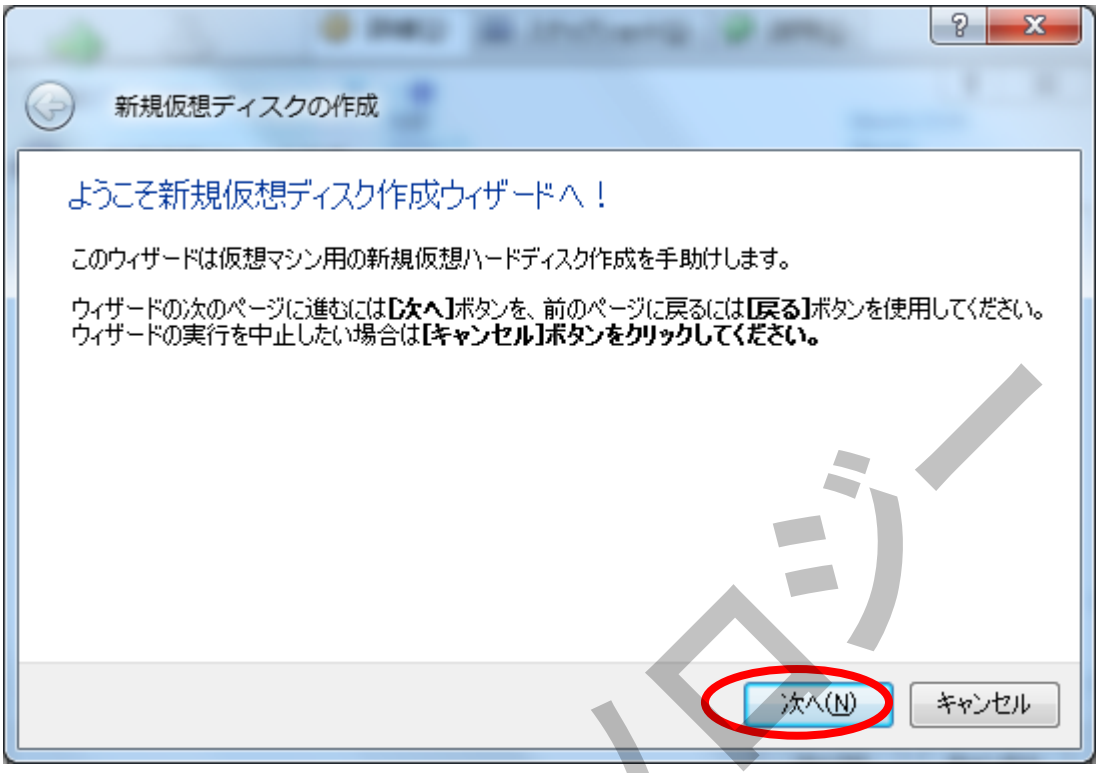
２．「新規」ボタンをクリック

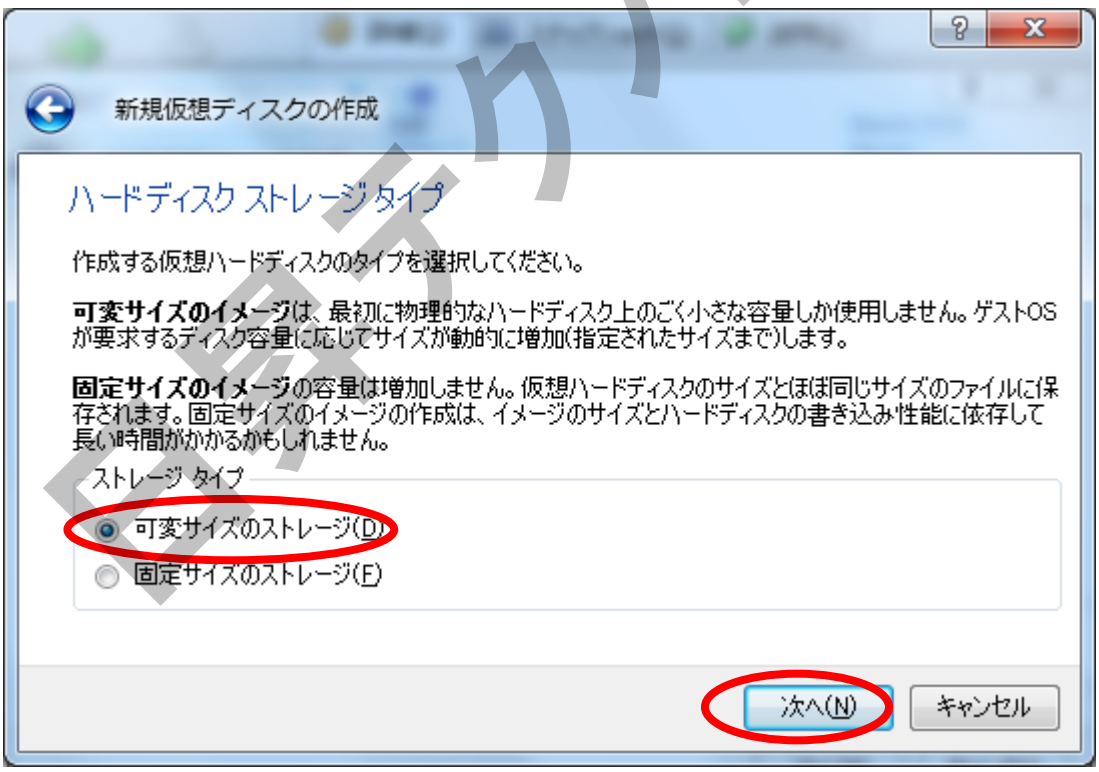


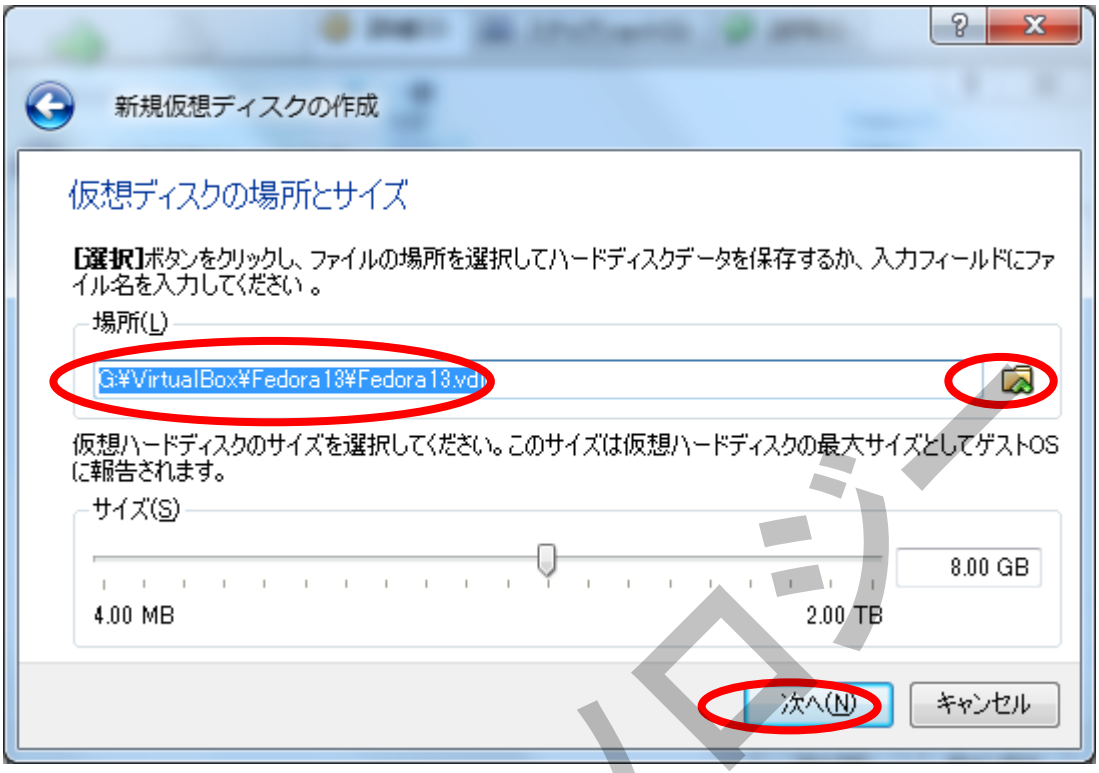


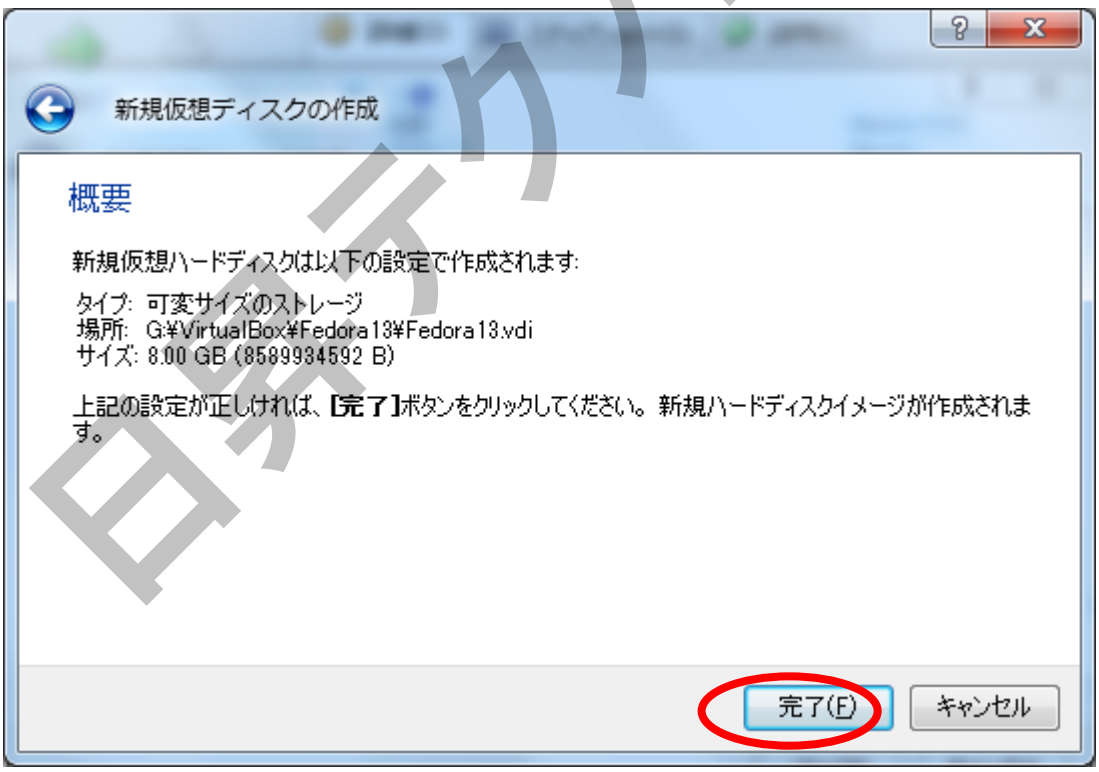

新規仮想マシンの作成
メモリ
仮想マシンに割り当てるメインメモリ(RAM)のサイズをメガバイト単位で選択してください。
推奨されるメインメモリのサイズは512MBです。
メインメモリのサイズ(M)
512 MB
4 MB
4096 MB
次へ(N)
キャンセル

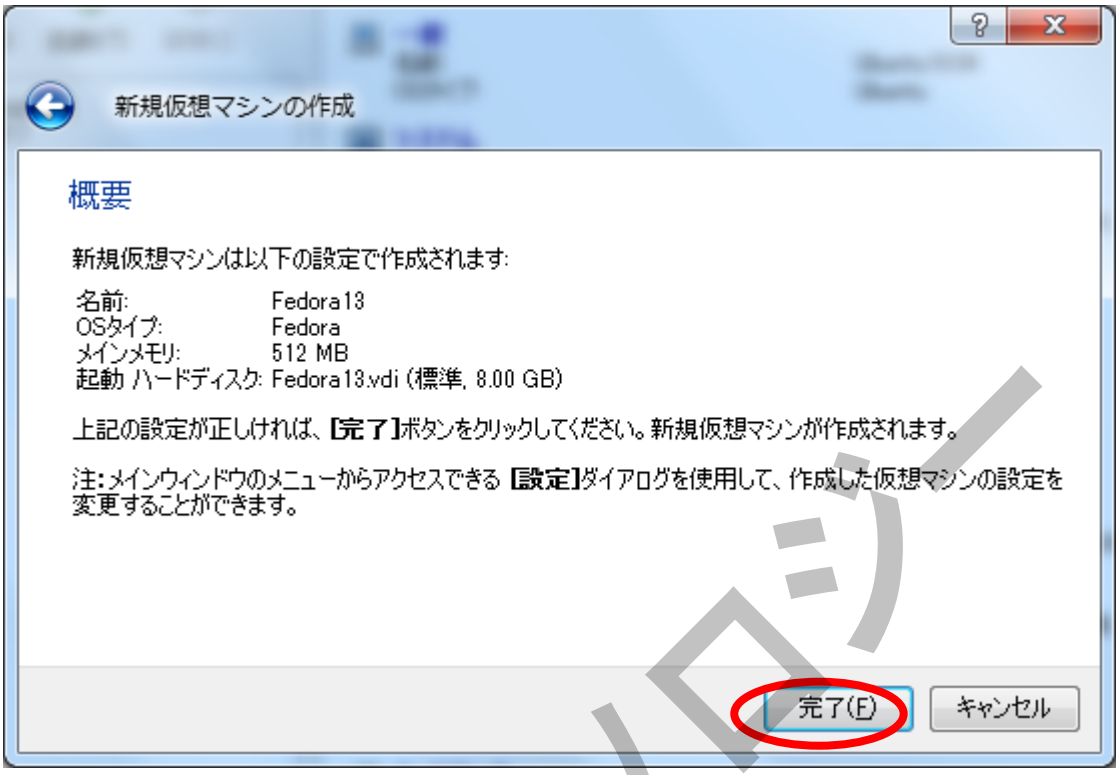

新規仮想マシンの作成
仮想ハードディスク
仮想マシンで起動ディスクとして使用するハードディスクイメージを選択します。[新規]ボタンをクリックして新規ハードディスクを作成するか、ドロップダウンリストまたは[選択]ボタンをクリック(仮想メディアマネージャを起動)して既存のハードディスクイメージを選択します。
より高度なハードディスク設定が必要であれば、このステップをスキップし、後で仮想マシン設定ダイアログを使用してハードディスクを割り当てることもできます。
推奨される起動ディスクのサイズは8192MBです。
起動ディスク(D)
新規ハードディスクの作成(C)
既存のハードディスクを使用(U)
Ubuntu10.04.vdi.vdi (標準, 16.00 GB)
次へ(N)
キャンセル


新規仮想ディスクの作成
ようこそ新規仮想ディスク作成ウィザードへ！
このウィザードは仮想マシン用の新規仮想ハードディスク作成を手助けします。
ウィザードの次のページに進むには[次へ]ボタンを、前のページに戻るには[戻る]ボタンを使用してください。
ウィザードの実行を中止したい場合は[キャンセル]ボタンをクリックしてください。
次へ(N)
キャンセル

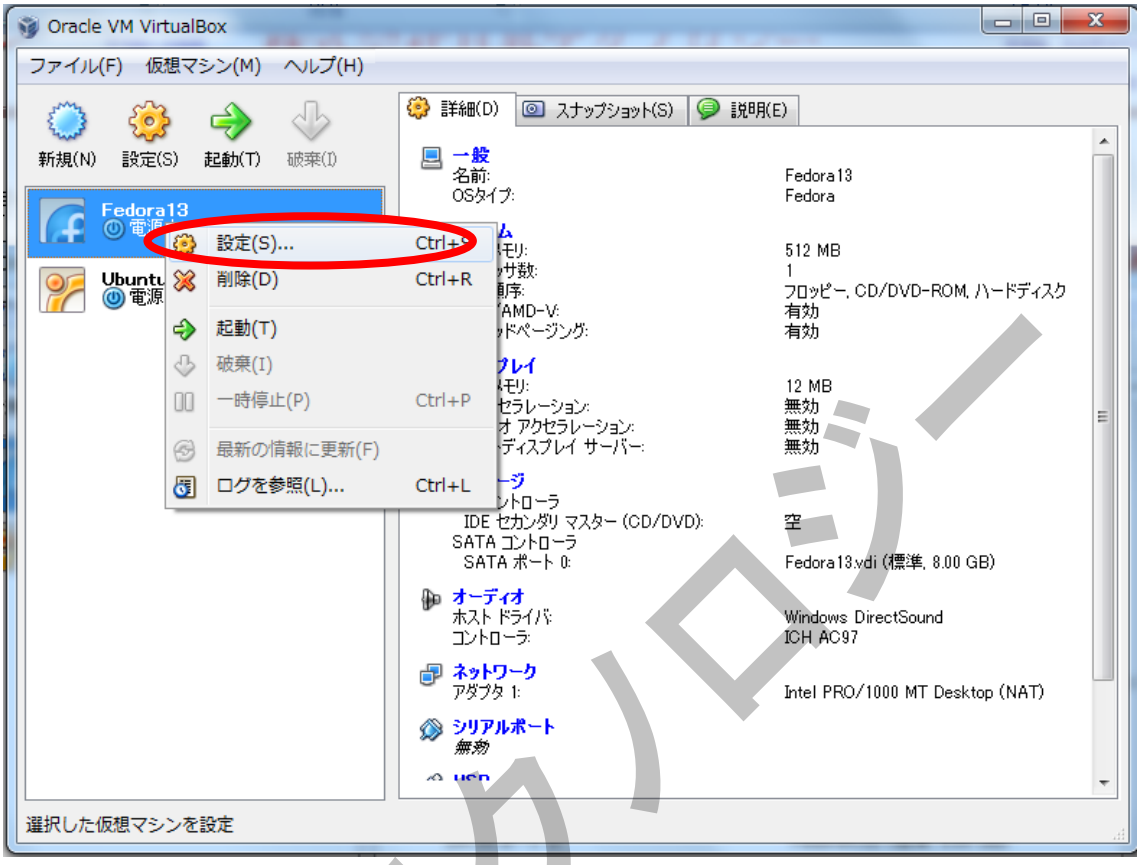

新規仮想ディスクの作成
ハードディスク ストレージ タイプ
作成する仮想ハードディスクのタイプを選択してください。
可変サイズのイメージは、最初に物理的なハードディスク上のごく小さな容量しか使用しません。ゲストOSが要求するディスク容量に応じてサイズが動的に増加(指定されたサイズまで)します。
固定サイズのイメージの容量は増加しません。仮想ハードディスクのサイズとほぼ同じサイズのファイルに保存されます。固定サイズのイメージの作成は、イメージのサイズとハードディスクの書き込み性能に依存して長い時間がかかるかもしれません。
ストレージ タイプ
可変サイズのストレージ(D)
固定サイズのストレージ(F)
次へ(N)
キャンセル

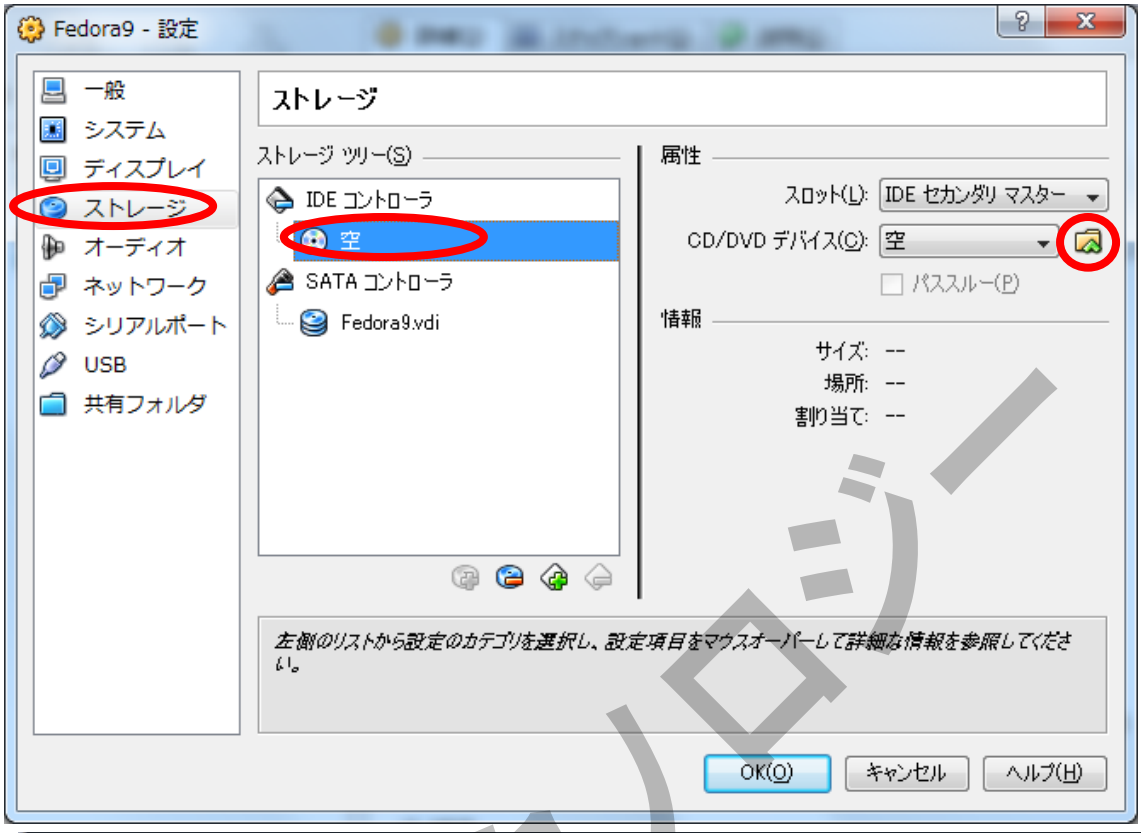

新規仮想ディスクの作成
仮想ディスクの場所とサイズ
[選択]ボタンをクリックし、ファイルの場所を選択してハードディスクデータを保存するか、入力フィールドにファイル名を入力してください。
場所(L)
G:¥VirtualBox¥Fedora13¥Fedora13.vdi
仮想ハードディスクのサイズを選択してください。このサイズは仮想ハードディスクの最大サイズとしてゲストOSに報告されます。
サイズ(S)
8.00 GB
4.00 MB
2.00 TB
次へ(N)
キャンセル

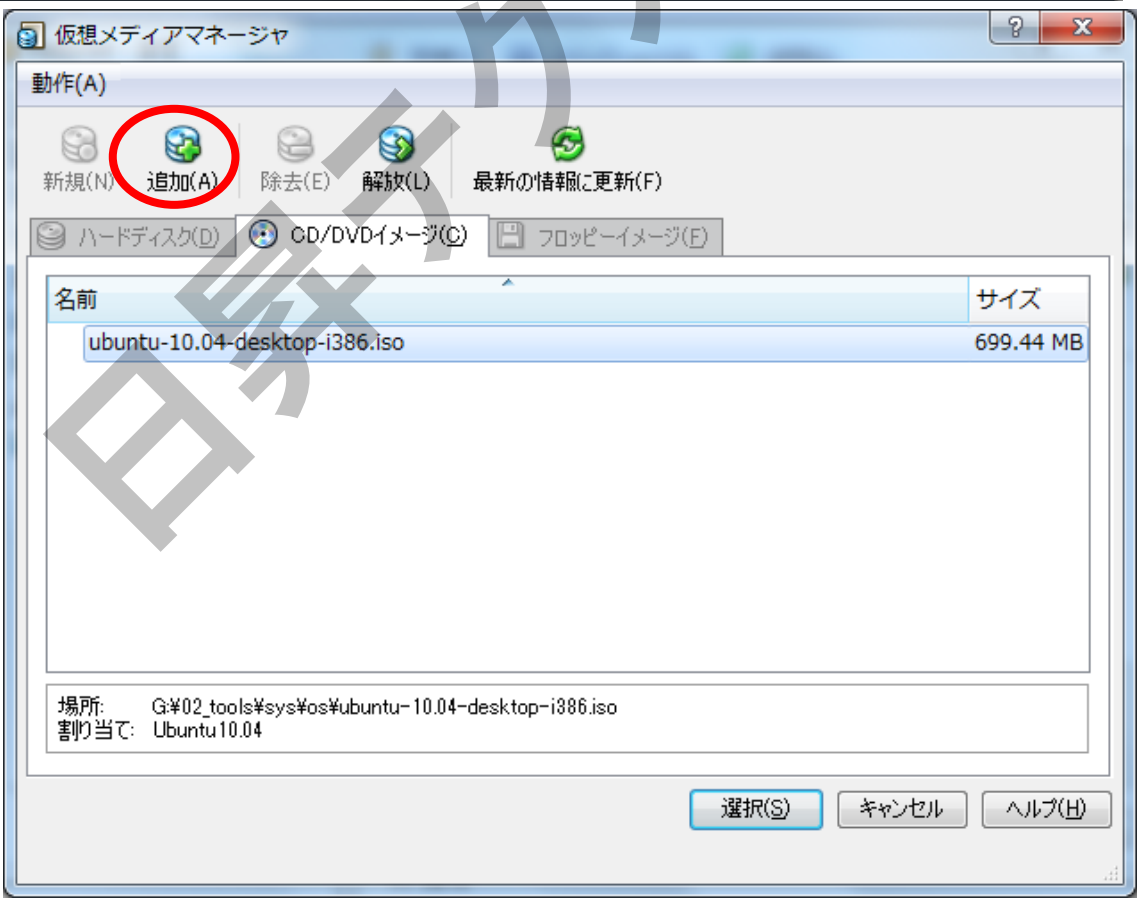

新規仮想ディスクの作成
概要
新規仮想ハードディスクは以下の設定で作成されます:
タイプ: 可変サイズのストレージ
場所: G:¥VirtualBox¥Fedora13¥Fedora13.vdi
サイズ: 8.00 GB (8589934592 B)
上記の設定が正しければ、[完了]ボタンをクリックしてください。新規ハードディスクイメージが作成されます。
完了(F)
キャンセル


新規仮想マシンの作成
概要
新規仮想マシンは以下の設定で作成されます:
名前: Fedora13
OSタイプ: Fedora
メインメモリ: 512 MB
起動 ハードディスク: Fedora13.vdi (標準, 8.00 GB)
上記の設定が正しければ、[完了]ボタンをクリックしてください。新規仮想マシンが作成されます。
注:メインウィンドウのメニューからアクセスできる [設定]ダイアログを使用して、作成した仮想マシンの設定を変更することができます。
完了(F)
キャンセル

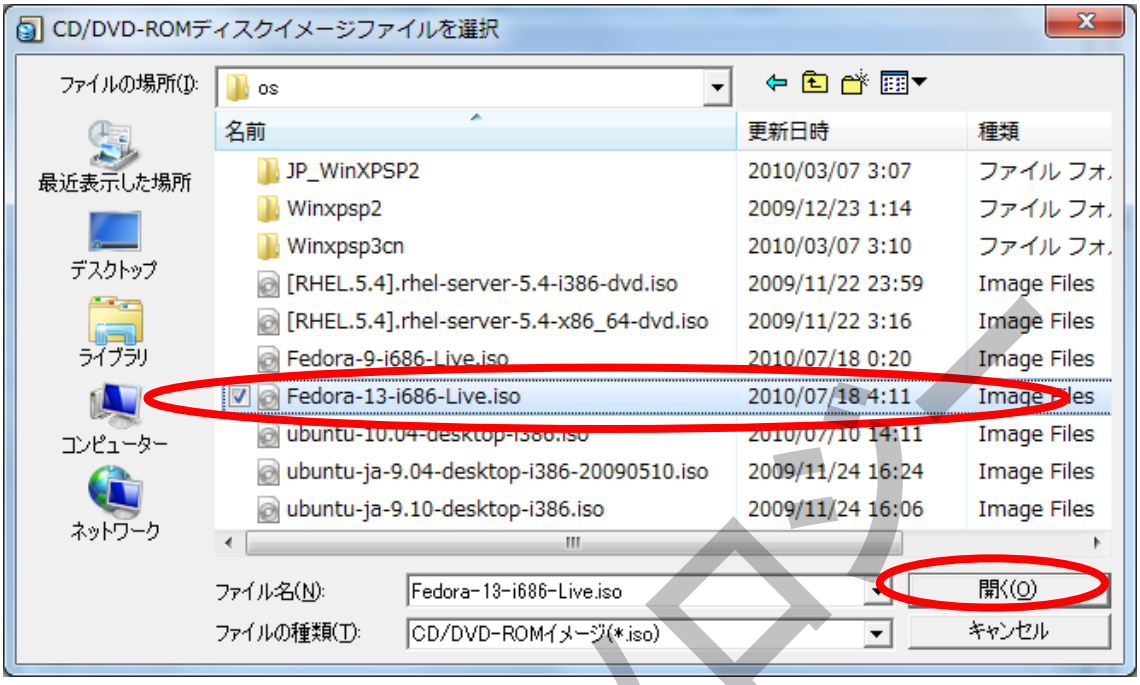
仮想マシン一覧から Fedora13 を選択して右クリックし「設定」メニューを押下

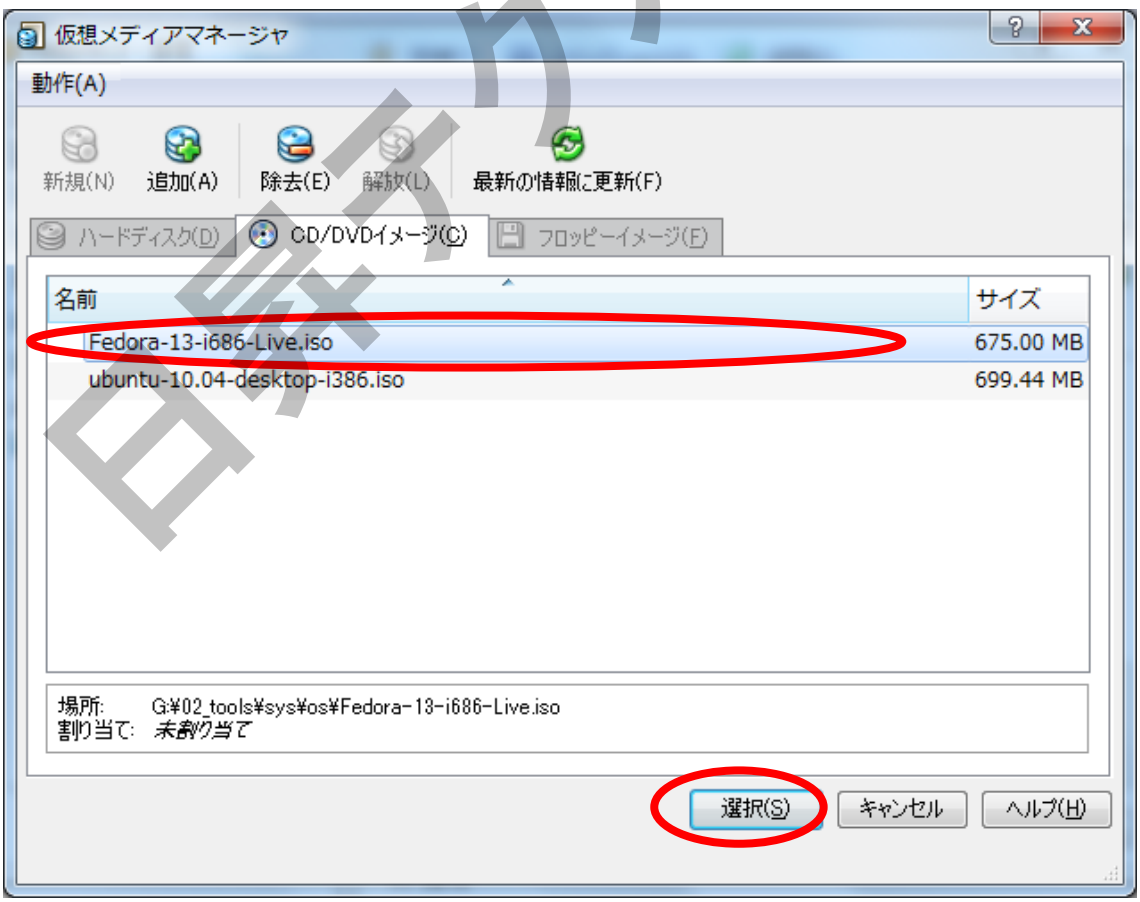


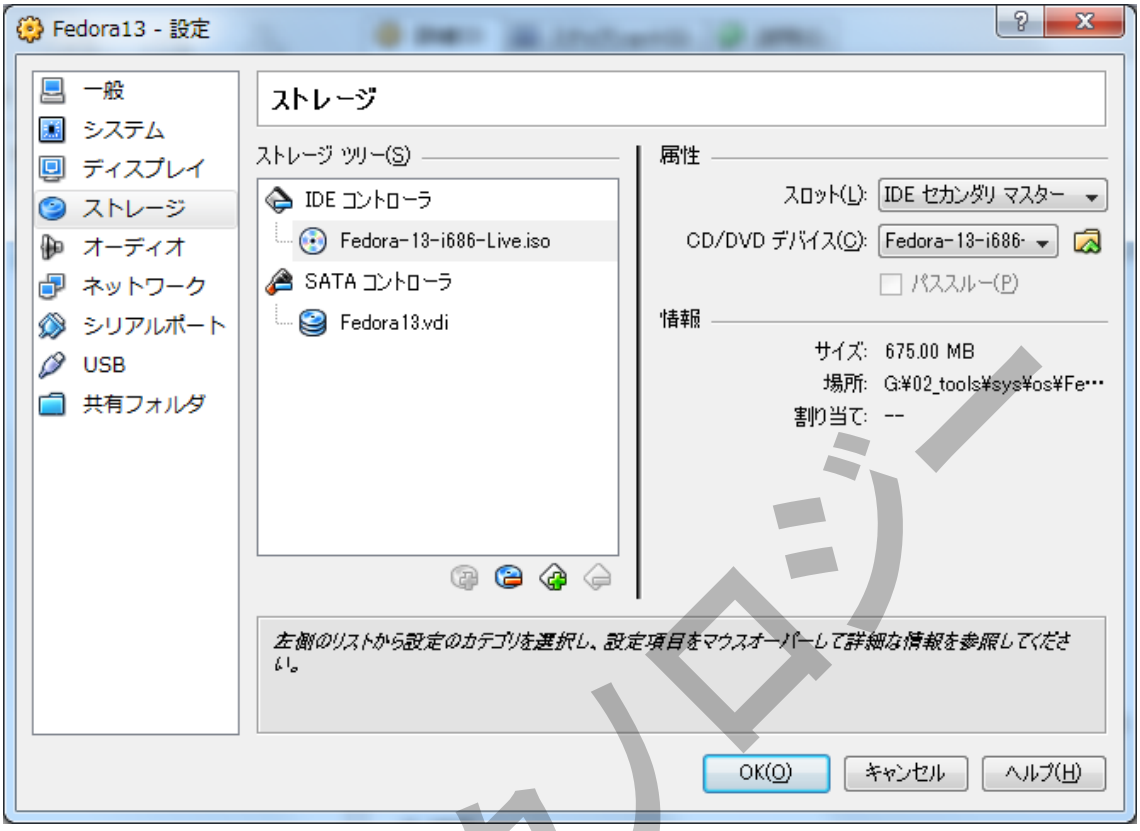

Fedora9 - 設定
一般
システム
ディスプレイ
ストレージ
オーディオ
ネットワーク
シリアルポート
USB
共有フォルダ
ストレージ
ストレージ ツリー(S)
IDE コントローラ
空
SATA コントローラ
Fedora9.vdi
属性
スロット(L): IDE セカンダリ マスター
CD/DVD デバイス(C): 空
パススルー(P)
情報
サイズ: --
場所: --
割り当て: --
左側のリストから設定のカテゴリを選択し、設定項目をマウスオーバーして詳細な情報を参照してください。
OK(O)
キャンセル
ヘルプ(H)

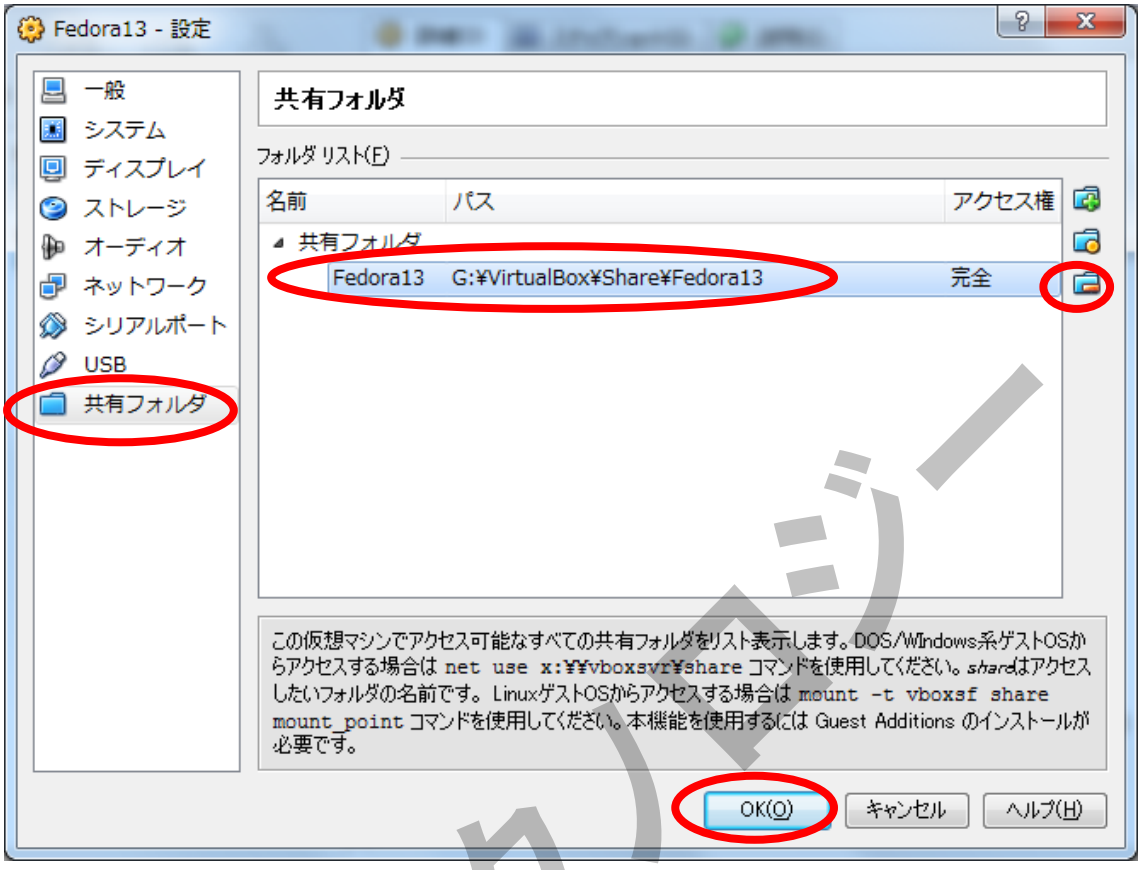

仮想メディアマネージャ
動作(A)
新規(N)
追加(A)
除去(E)
解放(L)
最新の情報に更新(F)
ハードディスク(D)
CD/DVDイメージ(C)
フロッピーイメージ(F)
名前
サイズ
ubuntu-10.04-desktop-i386.iso
699.44 MB
場所: G:¥02_tools¥sys¥os¥ubuntu-10.04-desktop-i386.iso
割り当て: Ubuntu10.04
選択(S)
キャンセル
ヘルプ(H)

ダウンロードした Fedora ISO ファイルを選択

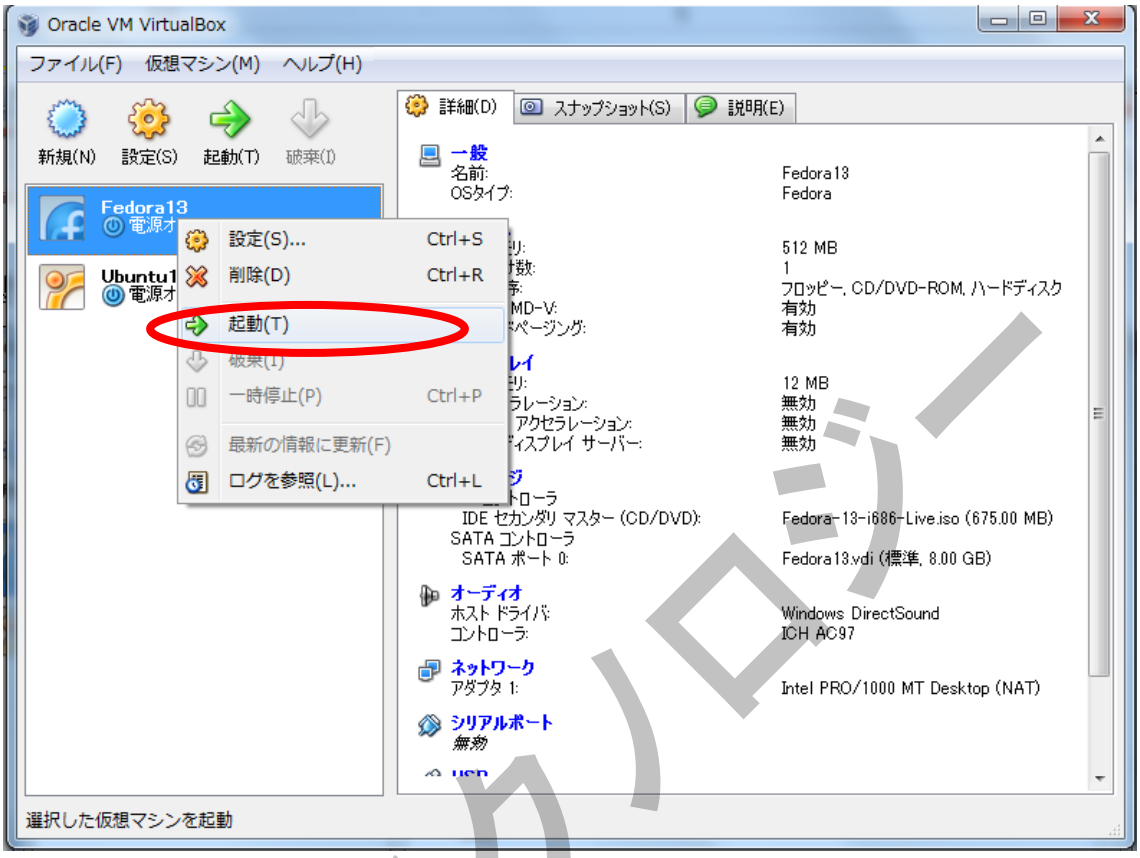





Fedora13 - 設定
一般
システム
ディスプレイ
ストレージ
オーディオ
ネットワーク
シリアルポート
USB
共有フォルダ
ストレージ
ストレージ ツリー(S)
IDE コントローラ
Fedora-13-i686-Live.iso
SATA コントローラ
Fedora13.vdi
属性
スロット(L): IDE セカンダリ マスター
CD/DVD デバイス(C): Fedora-13-i686-
パススルー(P)
情報
サイズ: 675.00 MB
場所: G:¥02_tools¥sys¥os¥Fe…
割り当て: --
左側のリストから設定のカテゴリを選択し、設定項目をマウスオーバーして詳細な情報を参照してください。
OK(O)
キャンセル
ヘルプ(H)

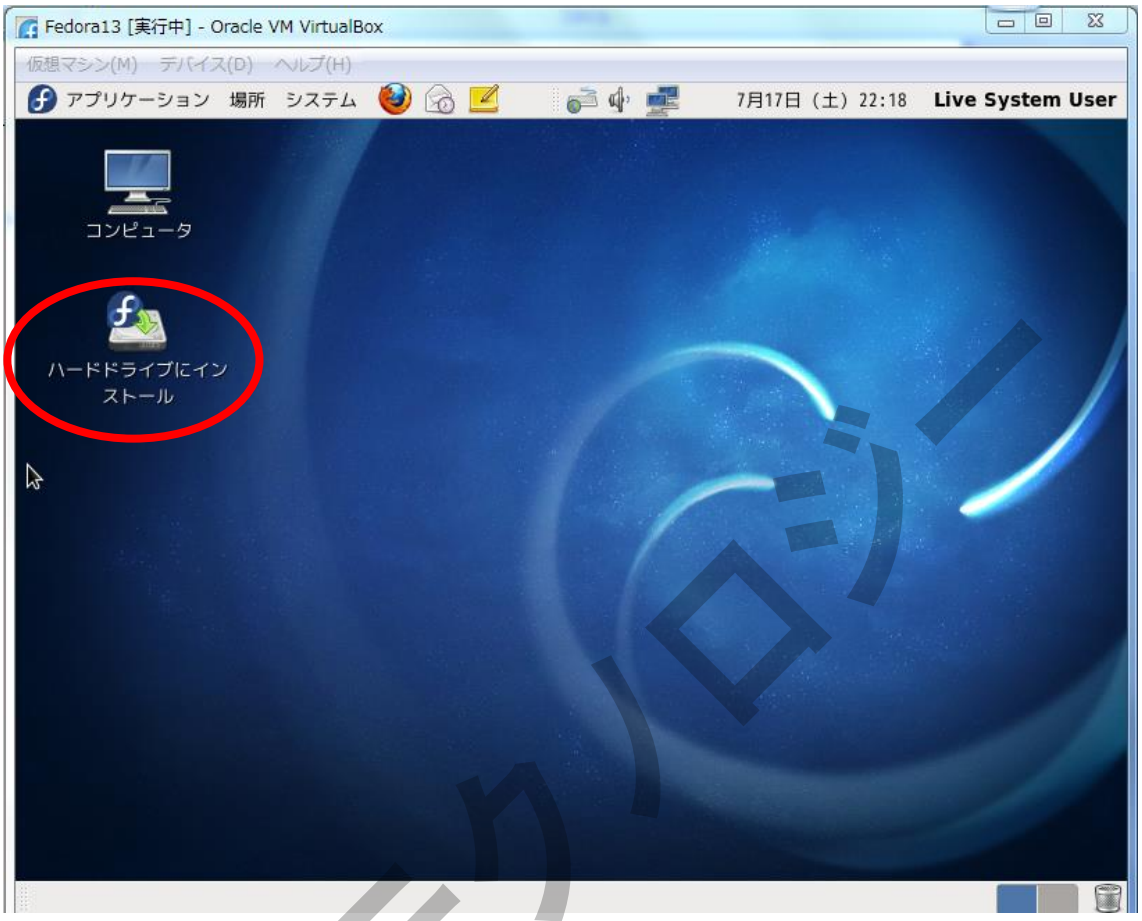

Fedora13 - 設定
一般
システム
ディスプレイ
ストレージ
オーディオ
ネットワーク
シリアルポート
USB
共有フォルダ
共有フォルダ
フォルダ リスト(F)
名前
パス
アクセス権
共有フォルダ
Fedora13
G:¥VirtualBox¥Share¥Fedora13
完全
この仮想マシンでアクセス可能なすべての共有フォルダをリスト表示します。DOS/Windows系ゲストOSからアクセスする場合は net use x:¥¥vboxsvr¥share コマンドを使用してください。shareはアクセスしたいフォルダの名前です。Linuxゲスト OSからアクセスする場合は mount -t vboxsf share mount_point コマンドを使用してください。本機能を使用するには Guest Additions のインストールが必要です。
OK(O)
キャンセル
ヘルプ(H)

仮想マシン一覧から Fedora13 を選択して右クリックし「起動」メニューを押下



インストール言語を選択

ログイン後、ハードドライブにインストール：



残り作業は普通の Fedora9 インストールと同じです、ここに省略。

3.3 Guest Additions インストール準備

Guest Additions をインストールするため、下記作業の準備が必要。

1. 現状カーネルバージョンを確認

```
# uname -r
2.6.33.3-85.fc13.i686
```

2.パッケージダウンロード時に最適なミラーサイトを選択できるようにする

```
# yum install -y yum-fastestmirror
```

3. gcc,kernel-devel 等のパッケージ更新

```
# yum install -y wget gcc kernel-devel libjpeg libjpeg-devel
```

4.カーネルソース取得

```
# yumdownloader --source kernel
```

5.カーネルソースインストール

```
#ls
2.6.33.6-147.fc13.src.rpm
#yum-builddep 2.6.33.6-147.fc13.src.rpm
#cd ~/rpmbuild/SPECS
#rpmbuild -bp --target=`uname -m` kernel.spec
#cd /root/rpmbuild/BUILD/kernel-2.6.33/linux-2.6.33.i686
#make oldconfig && make prepare
#make
*30 秒ぐらい「Ctrl+C」を中断、後ろ vboxadd をインストールする必要物を作成
#cp -rf /root/rpmbuild/BUILD/kernel-2.6.33/linux-2.6.33.i686 /usr/src/kernels
```

6. １番と５番でのバージョンが異なる場合、バージョンアップを行う

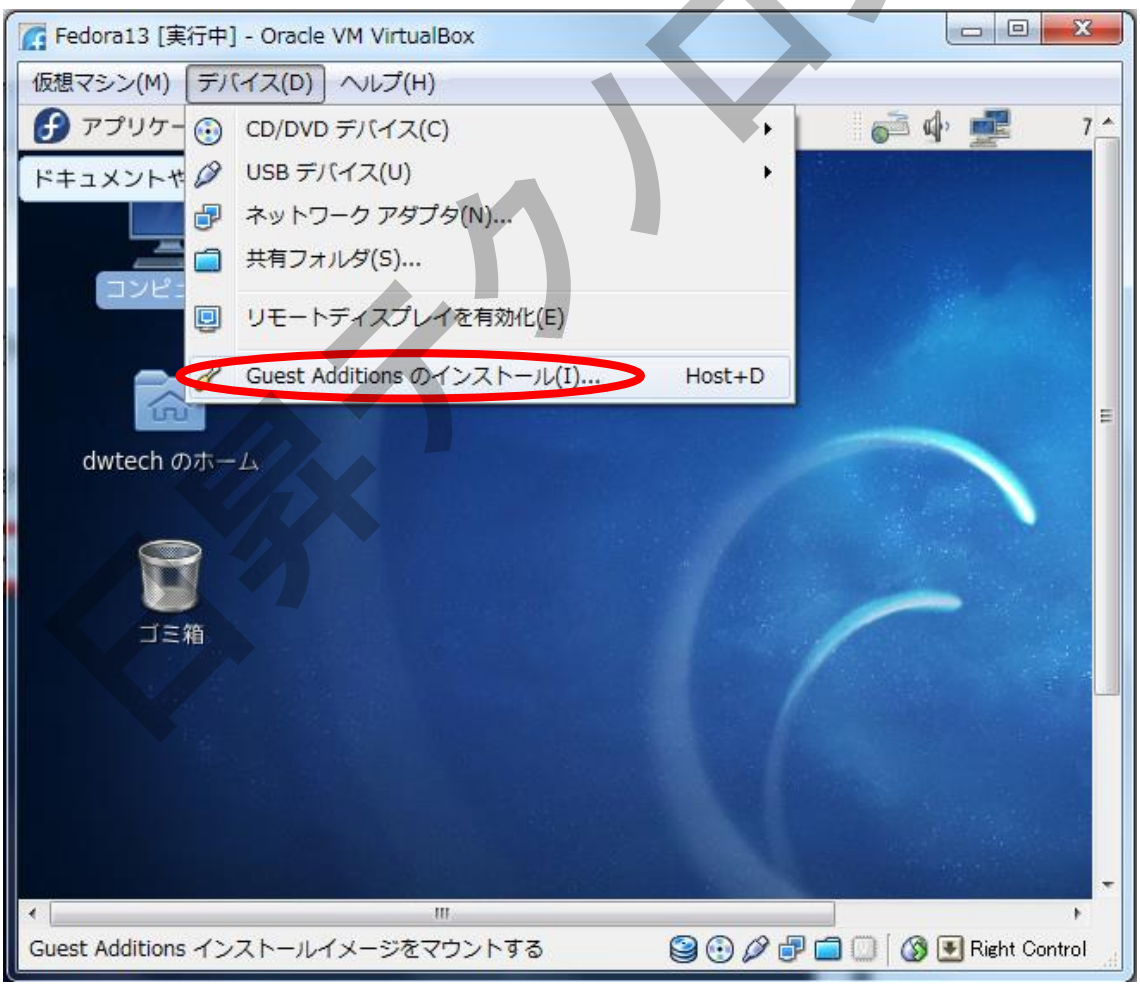
```
# yum install kernel.i686
```

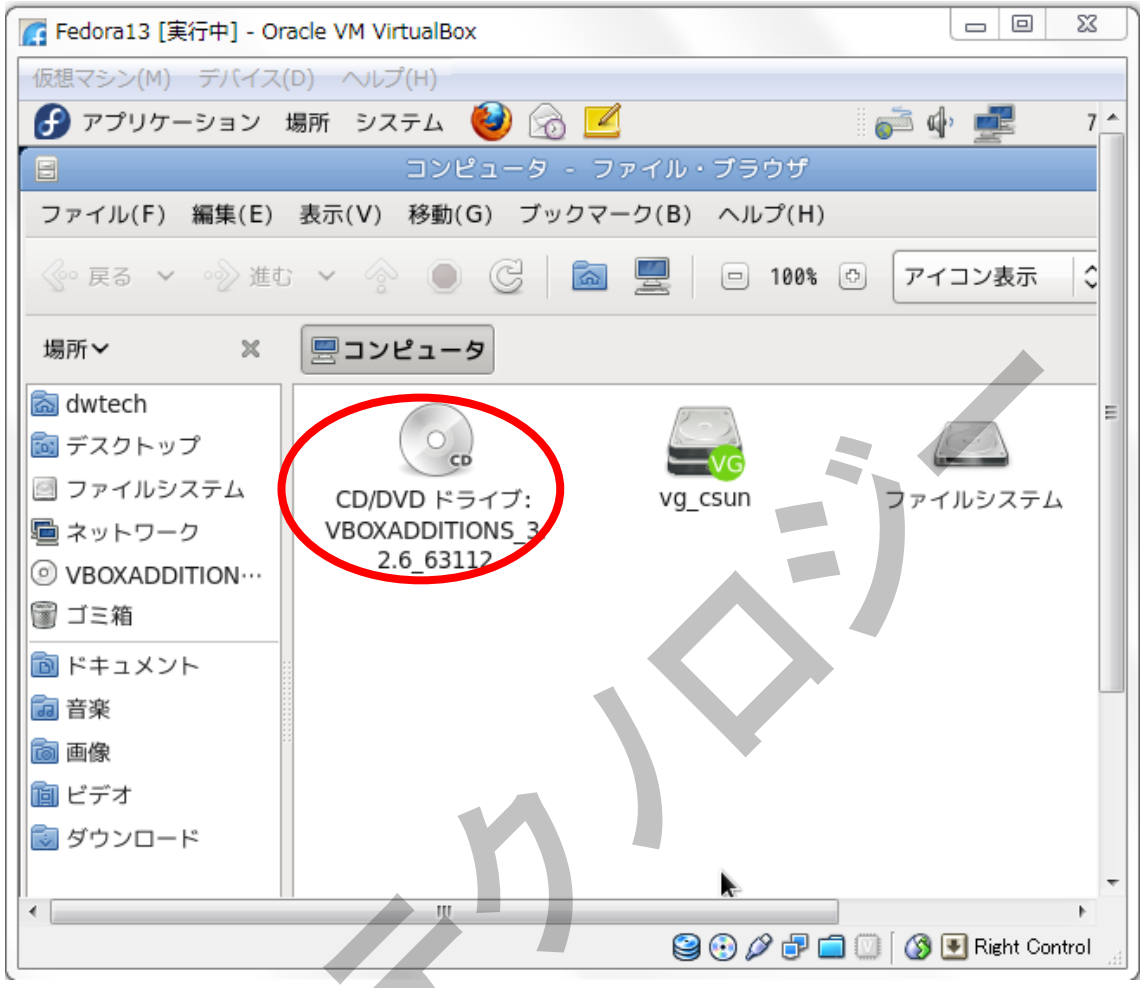
＊バージョンアップ完了後、再起動必要

7. 環境変数を設定

```
#vi ~/.bashrc(下記設定は一番後ろに追加)
export KERN_DIR=/usr/src/kernels/linux-2.6.33.i686
#source ~/.bashrc
```

Guest Additions のインストール：



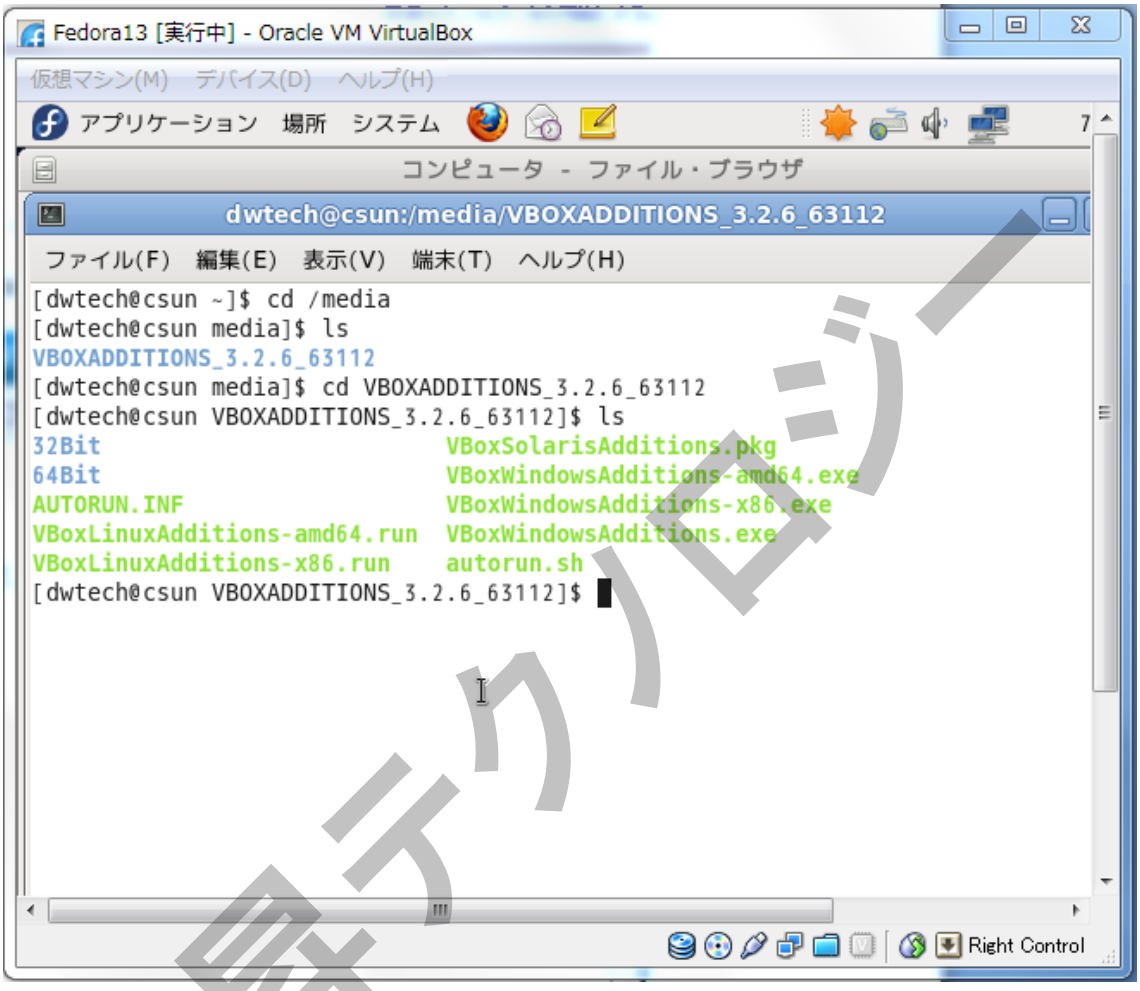


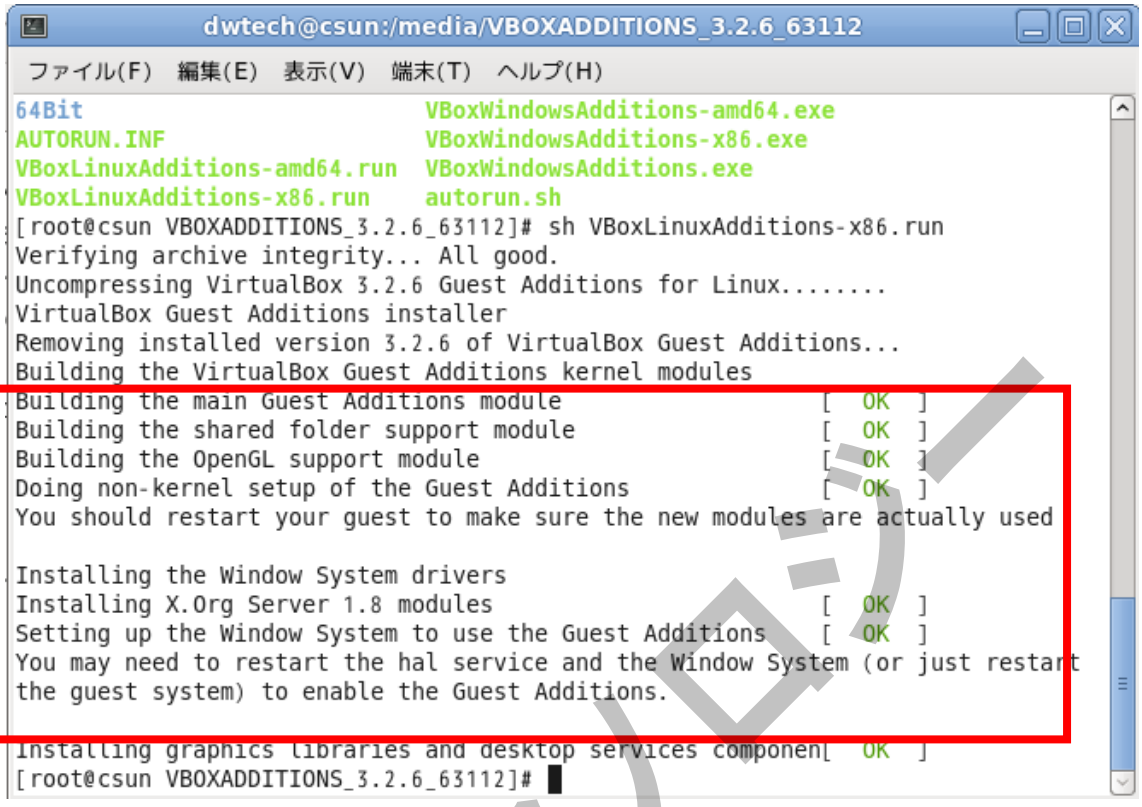
赤丸に右クリック、「マウント」を選択

インストール

ゲスト OS のターミナルで，次のように操作する

ゲスト OS が 32 ビット版の Linux のときは，amd64 の部分を読み替える．





Fedora を再起動してください。

＊ここにエラーが発生した場合、「/var/log/vboxadd-install.log」内容を確認の上、

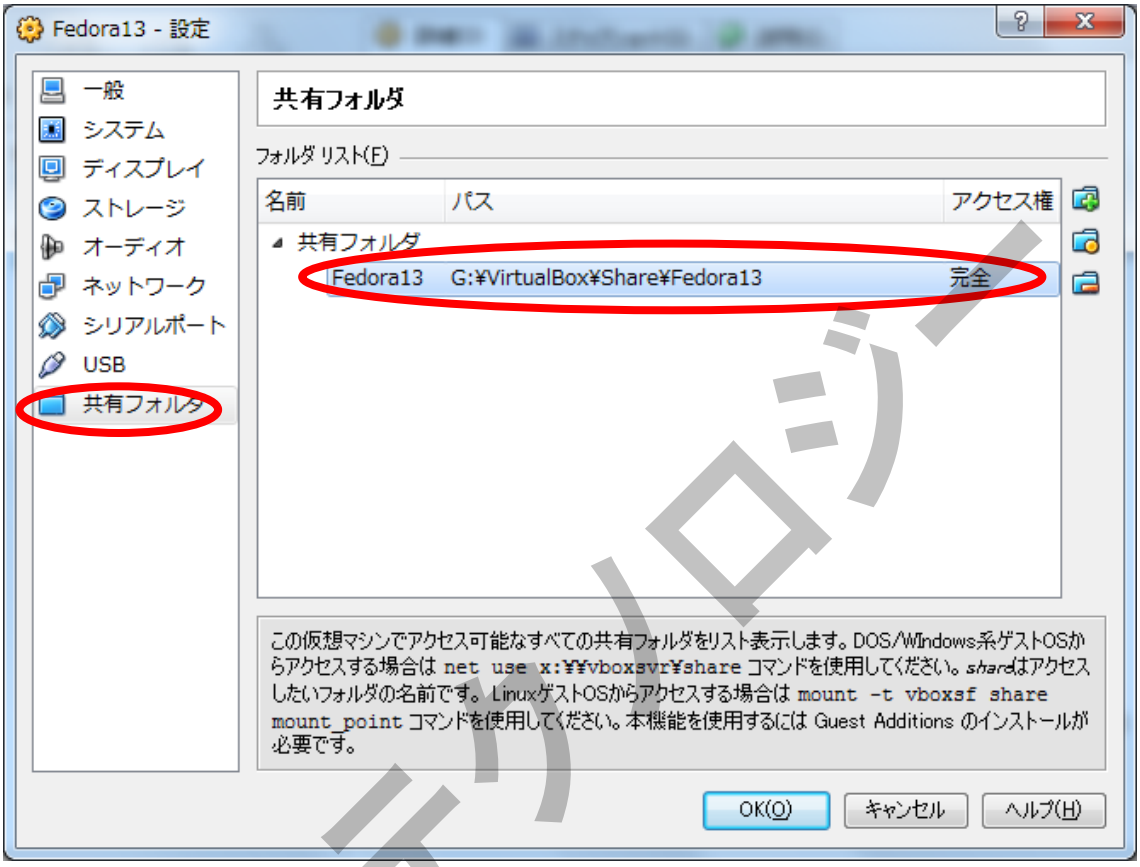
カーネルソース上に何にか足りないので、少し長く make を再度実施してください。

```
#cd /usr/src/kernels/ linux-2.6.33.i686
#make
*少し長く実施してから中断
```

3.4 共有フォルダアクセス

前の節に VirtualBox で下記のように共有フォルダーを設定された。

VirtulaBox 上共有フォルダー名：「Fedora13」



Fedora13 に共有フォルダーをアクセルするため、下記マウント必要。

#mkdir /mnt/share

#mount –t vboxsf Fedora13 /mnt/share

3.5 クロスコンパイラインストール

「http://www.dragonwake.com/download/arm9-download/linux-toolchain/arm-linux-gcc-4.3.2.tgz」からクロスコンパイラを共有フォルダーにダウンロードする

1.コンパイラインストール

```
[root@csun arm-qtopia]# tar -zxvf /mnt/share/arm-linux-gcc-4.3.2.tgz -C /
```

2.環境設定（ARM コンパイラパス設定）

```
[root@csun arm-qtopia]#vi ~/.bashrc
 ファイルの末尾に追加：「export PATH=$PATH:/usr/local/arm/4.3.2/bin」
 設定を更新
[root@csun arm-qtopia]# source ~/.bashrc
```

3.6 Qt SDK インストール

3.6.1 ARM 用の Qt SDK インストール

「http://www.dragonwake.com/download/arm9-download/qt/arm-qtopia-2.2.0.tar.gz」から共有フォルダーにダウンロードする

1.Qt SDK 格納フォルダーを作成

```
[root@csun dwtech]# mkdir /opt/FriendlyARM
[root@csun dwtech]# mkdir /opt/FriendlyARM/mini2440
```

2.Qtpoia-2.2.0 パッケージ展開

```
[root@csun dwtech]# tar -zxvf /mnt/share/arm-qtopia-2.2.0.tar.gz -C /opt/FriendlyARM/mini2440
```

3.g++コンパイラインストール(インストール済であれば飛ばす)

```
[root@csun dwtech]# yum install gcc-c++
```

4.他に qtopia をコンパイルする必要ものをインストール

```
[root@csun arm-qtopia]# yum install libX11-devel
[root@csun arm-qtopia]# yum install libXext-devel
```

◆Fedora でウインドウシステムにはモジュール化された X.org X11R7 を採用，インスト

ールパスが従来の /usr/*X11R6* から /usr に変更された。
但し、qt のコンパイルパスは「/usr/X11R6/include」に設定されているため、下記シンボリックリンクを作成しましょう。

```
[root@csun arm-qtopia]#mkdir /usr/X11R6
[root@csun arm-qtopia]#ln -s /usr/include/X11 /usr/X11R6/include
```

＊上記作業を行わない場合、「**kernel/qt_x11_p.h:66:22: error: X11/Xlib.h: そのようなファイルやディレクトリはありません**」というようなエラーはコンパイル時に出て来る。

◆libuuid（e2fsprogs)をサーポートするため、e2fsprogs をコンパイルする
「http://www.dragonwake.com/download/arm9-download/qt/e2fsprogs-1.41.12.tar.gz」
から共有フォルダーにダウンロードする

```
[root@csun arm-qtopia]# tar -zxvf /mnt/share/download/e2fsprogs-1.41.12.tar.gz -C /home/dwtech/download
[root@csun arm-qtopia]#./configure --enable-elf-shlibs --enable-dynamic-e2fsck
--disable-nls --mandir=/usr/share/man ￥
--infodir=/usr/share/info --enable-compression ￥
CFLAGS='-O2 -g -fsigned-char -D__NO_STRING_INLINES' --host=arm-linux
[root@csun arm-qtopia]#make -C util CFLAGS='-O2 -g -fsigned-char -D__NO_STRING_INLINES'
[root@csun arm-qtopia]#make
[root@csun arm-qtopia]#make install
```

＊上記作業を行わない場合、「**Failed to make pngscale**」というようなエラーはコンパイル時に出て来る。

◆「linux-arm-g++-shared」修正（arm-qtopia/qtopia-2.2.0-FriendlyARM/qt2/configs)

```
[root@csun arm-qtopia]# vi qtopia-2.2.0-FriendlyARM/qt2/configs/linux-arm-g++-shared
```

SYSCONF_LINK = arm-linux-gcc
SYSCONF_LINK_SHLIB = arm-linux-gcc
修正後：
SYSCONF_LINK = arm-linux-g++
SYSCONF_LINK_SHLIB = arm-linux-g++
＊上記修正を行わない場合、「**make[5]: arm-linux-g++: コマンドが見つかりませんでした**」というようなエラーはコンパイル時に出て来る。

5.qtopia コンパイル(凡そ 30 分)

```
[root@csun arm-qtopia]# ./build-all 2>&1 | tee /mnt/share/arm-qt-compile.log > /dev/null
```

コンパイル作業を分析するため、コンパイルログは共有フォルダーに出力するようにする。
コンソールに何にも出力されない。

Note： 2>&1：意味：1;標準出力、2：標準エラー出力、標準エラー出力内容と合わせて
標準出力内容をファイル「/mnt/share/arm-qt-compile.log」に出力。
tee：標準入力を標準出力とファイルに出力する

3.6.2 x86 用の Qt SDK インストール

PC 上に擬似で QT アプリを実行できるため、x86 用の Qt SDK も必要です。
PC 上に実行しない場合、飛ばしても OK。

「http://www.dragonwake.com/download/arm9-download/qt/x86-qtopia-2.2.0.tar.gz」から共有フォルダーにダウンロードする

1.Qtpoia-2.2.0 パッケージ展開

```
[root@csun dwtech]# tar -zxvf /mnt/share/x86-qtopia-2.2.0.tar.gz -C /opt/FriendlyARM/mini2440
```

2.qtopia コンパイル(凡そ 30 分)

```
[root@csun x86-qtopia]# ./build-all 2>&1 | tee /mnt/share/x86-qt-compile.log > /dev/null
```

第四章　QT アプリ作成

4.1 環境設定

①フォルダー「/opt/FriendlyARM/mini2440/arm-qtopia」に入る

②環境変数設定：

```
[root@csun arm-qtopia]# source qtopia-2.2.0-FriendlyARM/setQpeEnv
```

修正前：(初回のみ)

export PATH=$QPEDIR/bin:$QTDIR/bin:$DQTDIR/bin: $PATH

修正後：(tmake のパスを追加)

export PATH=$QPEDIR/bin:$QTDIR/bin:$DQTDIR/bin:$TMAKEDIR/bin:$PATH

変更を反映：(2 回以降直接実行)

```
[root@csun arm-qtopia]# source qtopia-2.2.0-FriendlyARM/setQpeEnv
```

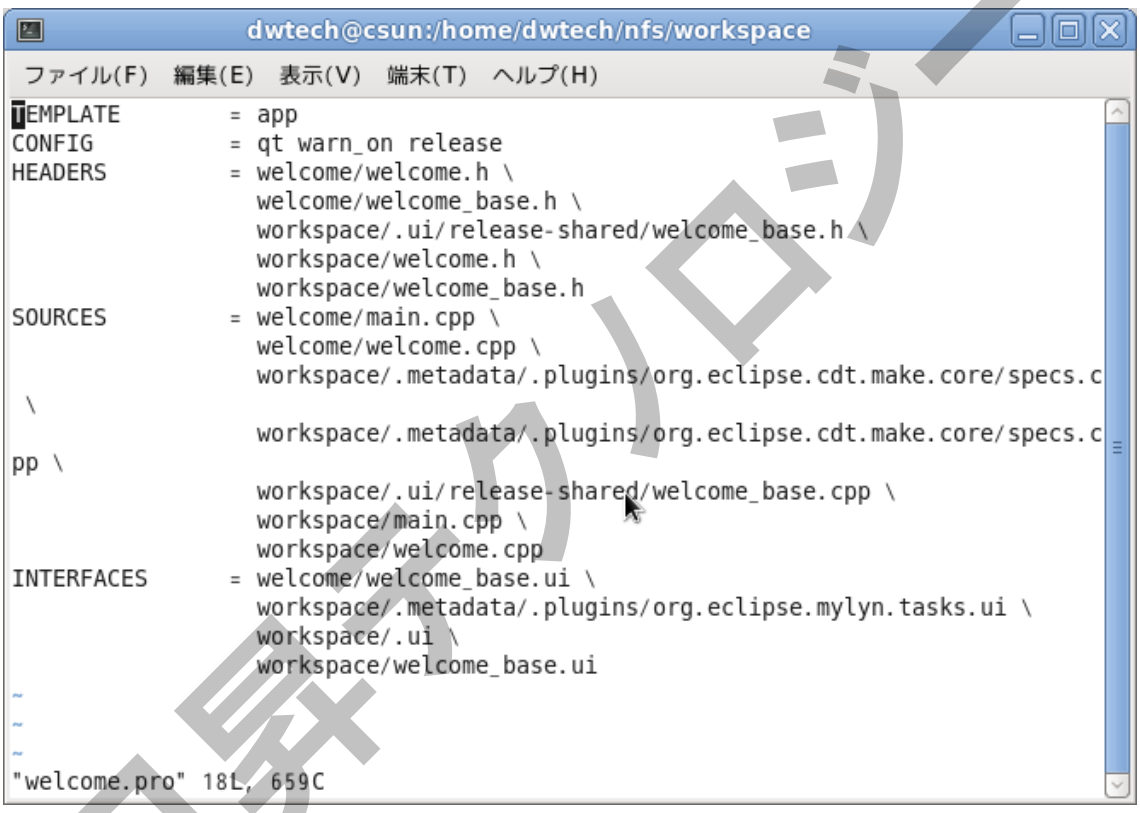
4.2 Qt アプリプロジェクト作成

マニュアル用サンプルソース：

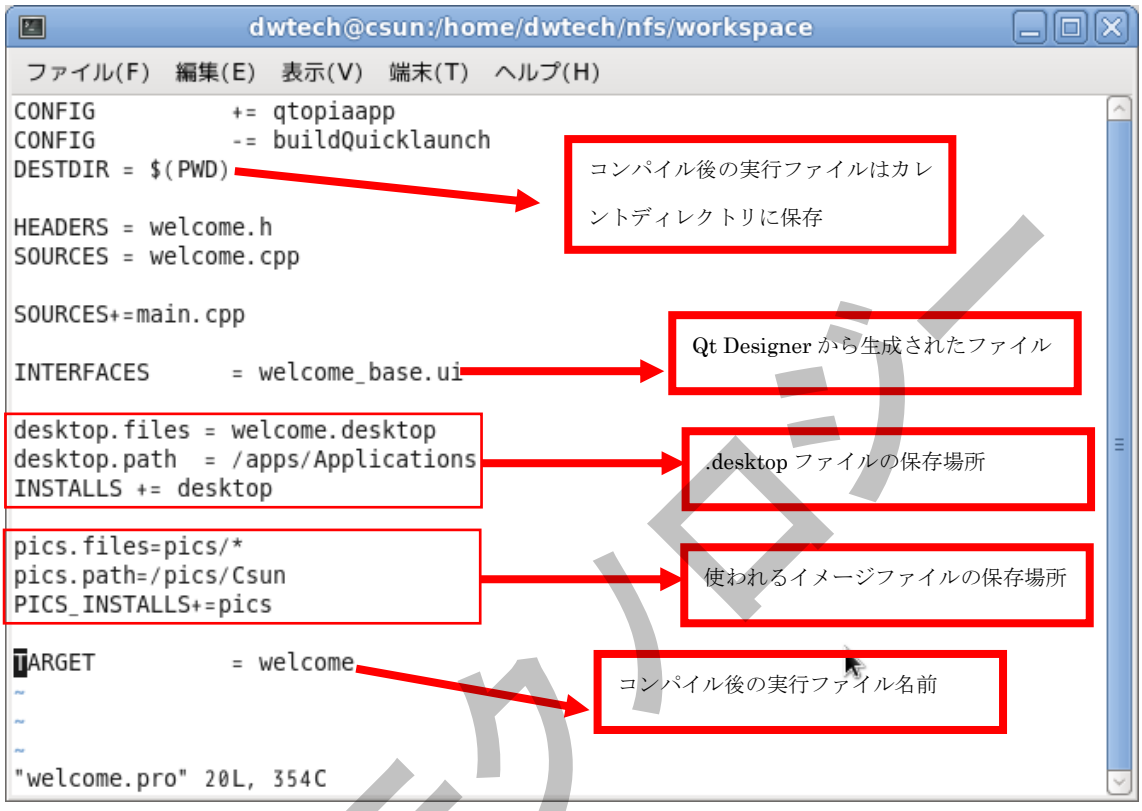
http://www.dragonwake.com/download/arm9-download/qt/welcome.tar.gz

＊プロジェクト格納先：NFS 共有フォルダーの中「/home/dwtech/nfs/workspace」

```
[root@csun arm-qtopia]# progen -t app -o /home/dwtech/nfs/workspace/welcome.pro
[root@csun arm-qtopia]#ls
welcome.pro
[root@csun arm-qtopia]#vi welcome.pro
```



プロジェクトファイルは下記のように修正（自動生成コードを殆ど削除）
ツール生成より、皆様から今後弊社の作成したプロジェクトファイルに基づき作成したがもっと簡単！

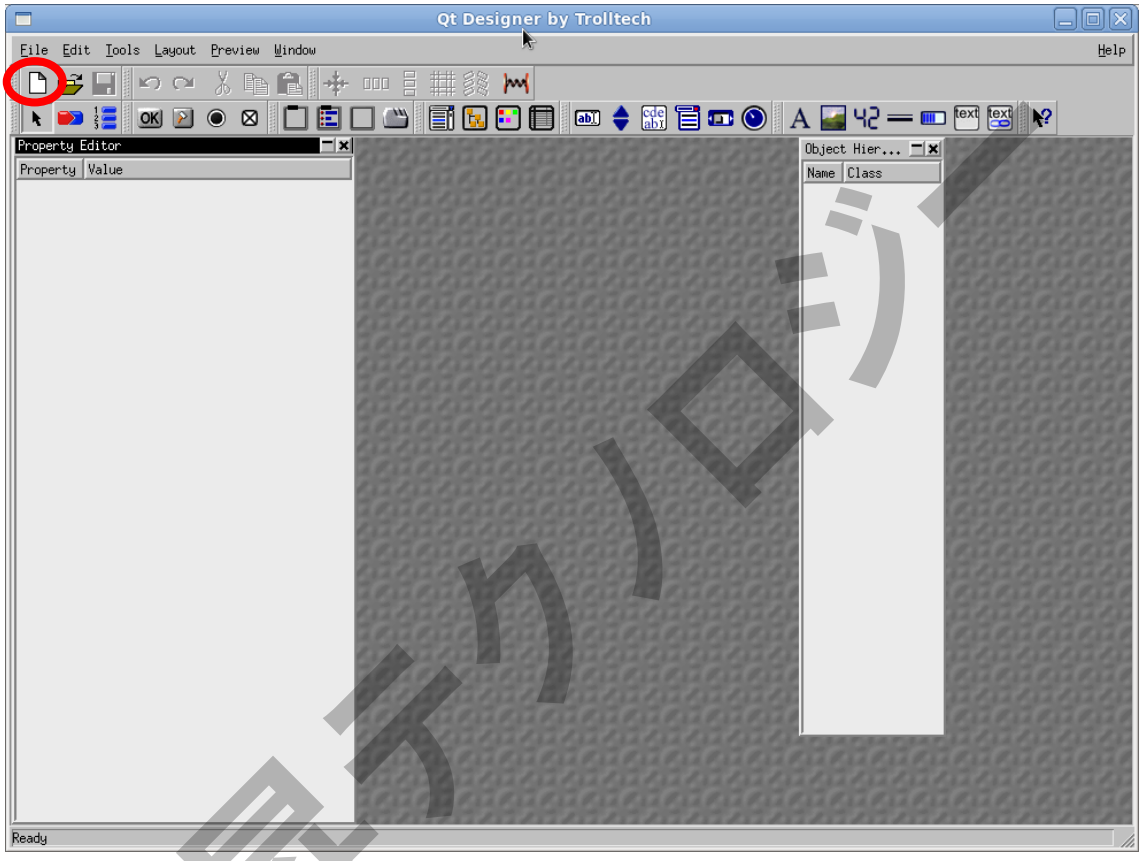


4.3 UI 画面をデザイナする

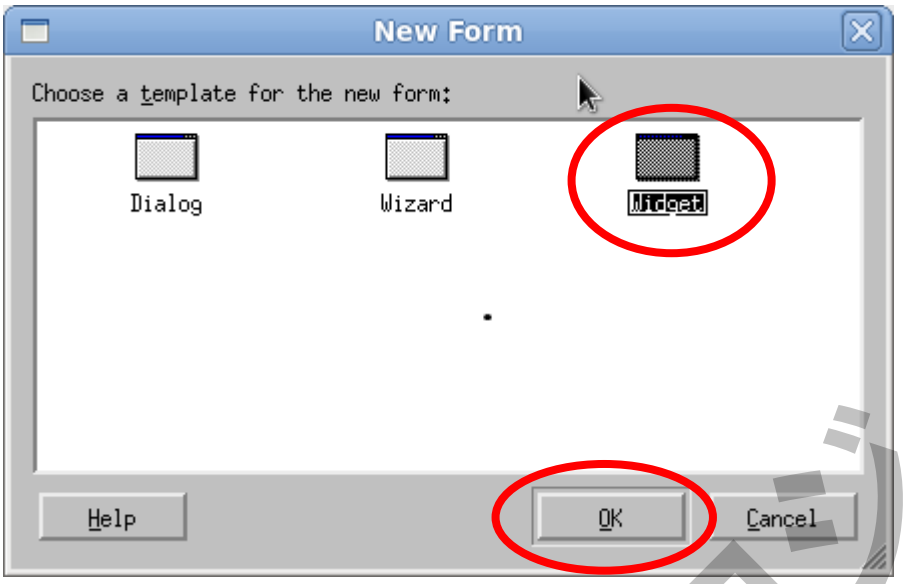
①Qt Designer を起動

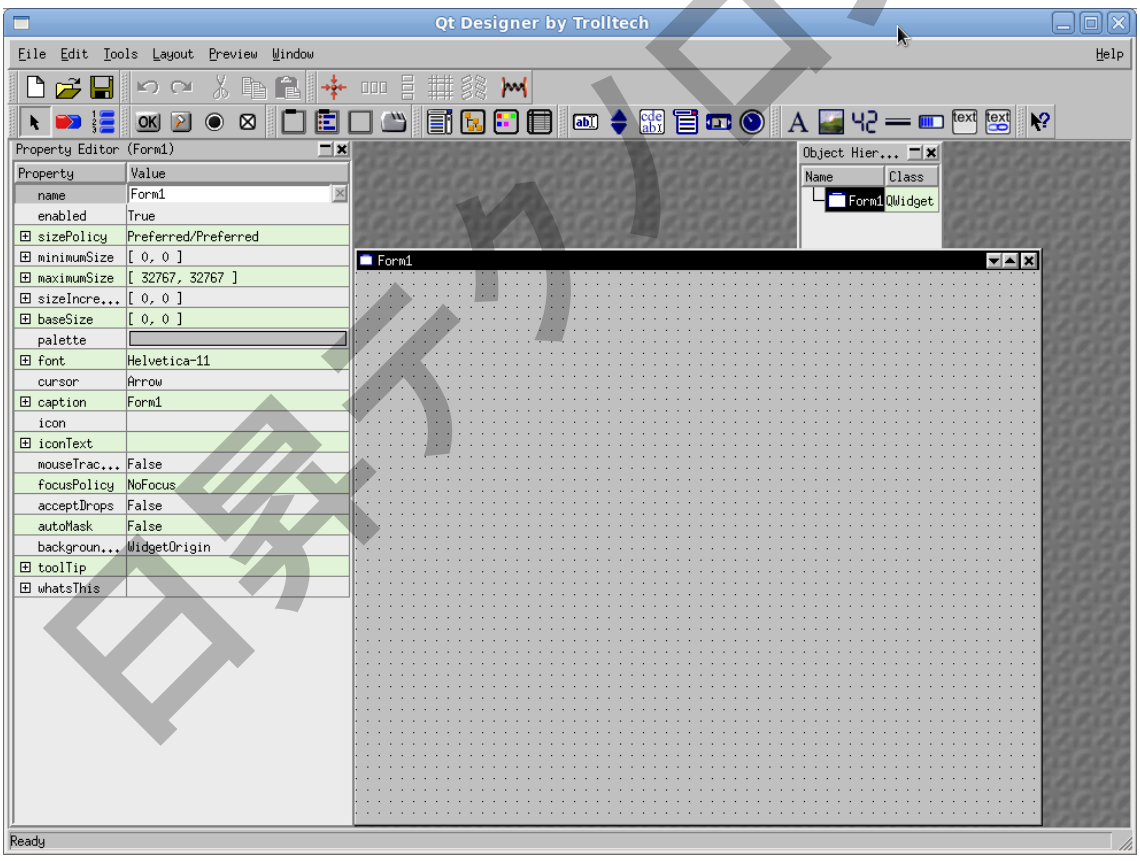
```
[root@csun arm-qtopia]# qtopia-2.2.0-FriendlyARM/qt2/bin/designer
```

②新しいフォームを新規作成：



「Widget」を選択、「OK」ボタンを押下

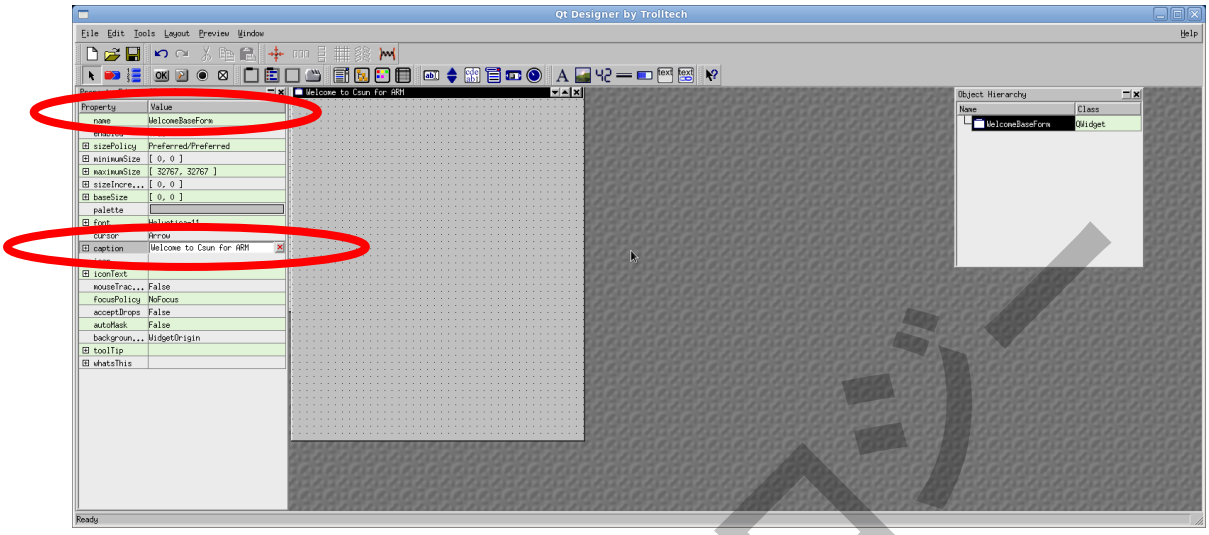




③新規作成フォーム属性を設定

◆フォーム名前（Qt 内部用 ID）：name→「WelcomeBaseForm」

◆フォームタイトル：caption→「Welcome to Csun for ARM」



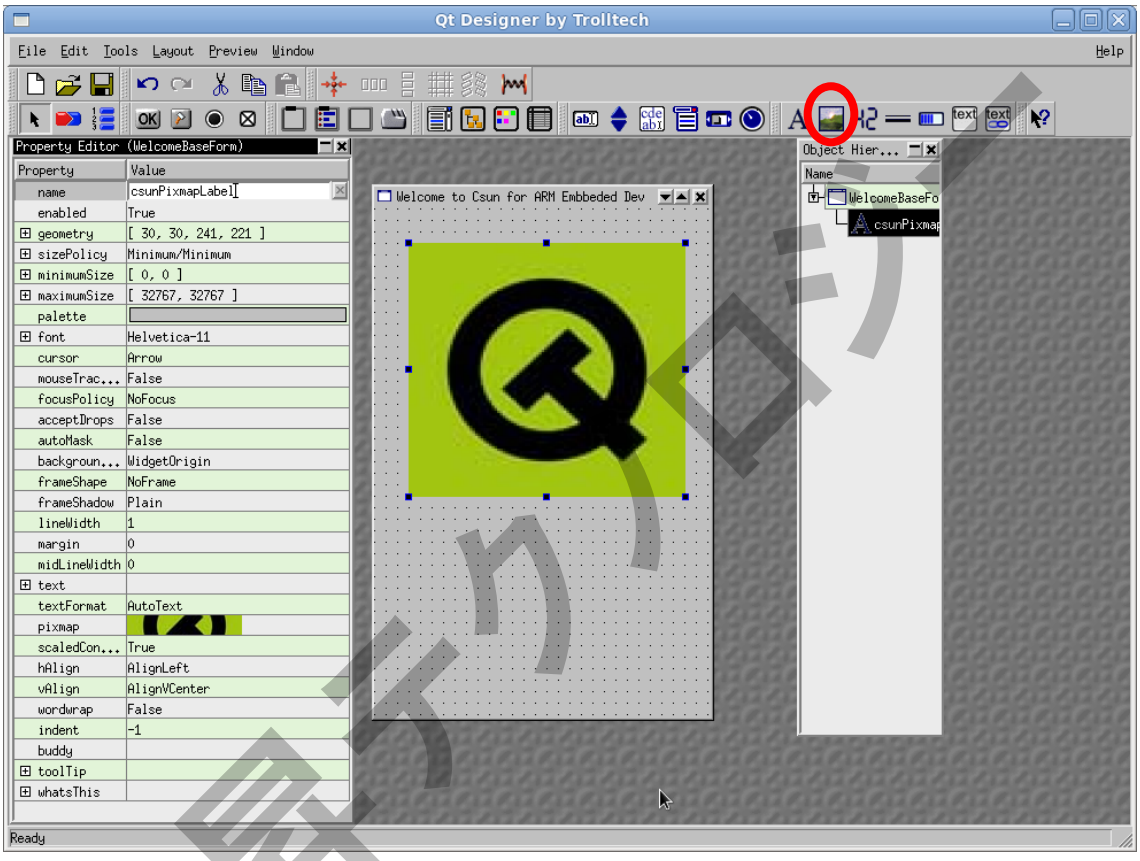
④Pixmap ラベルをフォームに追加（メッセージ等を表示するため）

◆「Pixmap Label」ボタンをクリックし、

フォームにドロップして下記のようにサイズを調整

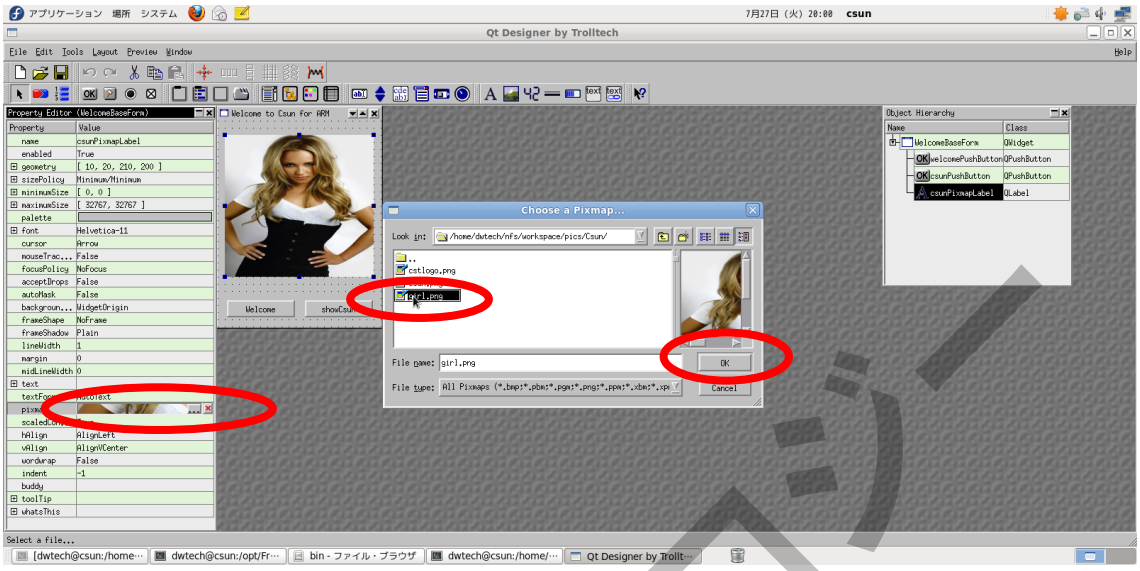
◆Pixmap の属性設定：

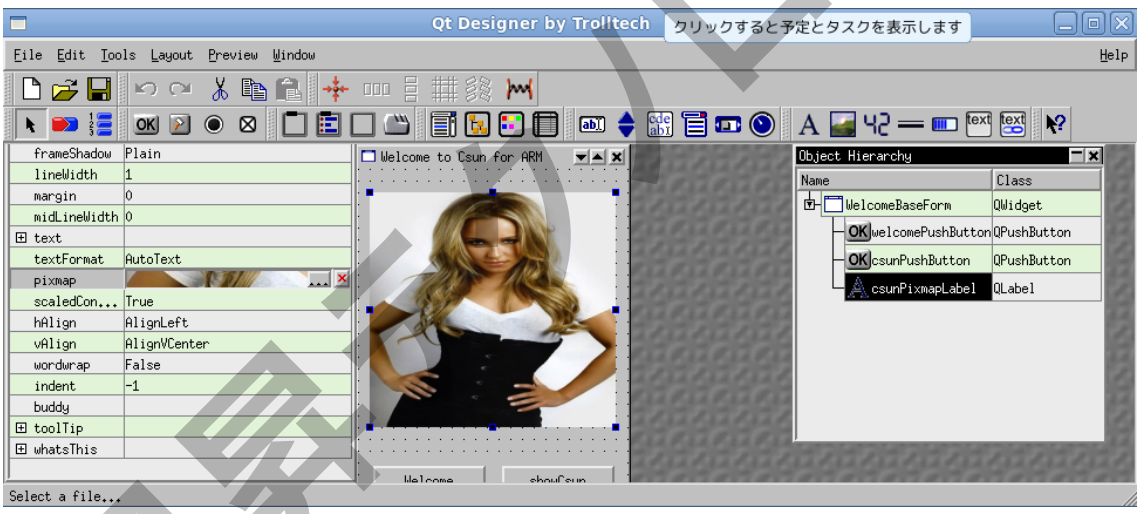
name→「csunPixmapLabel」



◆アプリ起動後の初期化イメージを設定

好きな画像を選択し設定





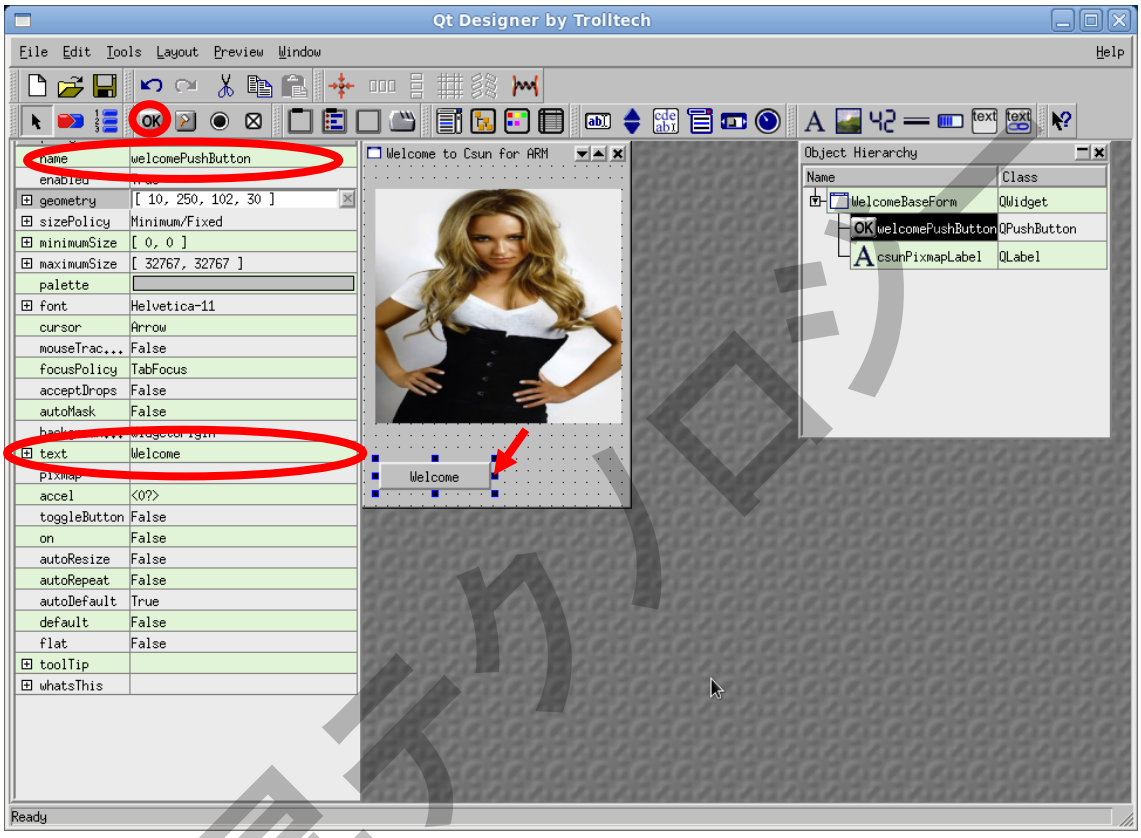
⑤操作ボタンを追加

◆Qt Designer のメニューバー上の「Ok」ボタンをクリックしたままフォームにドロップ

ボタン属性設定：

Name→「welcomePushButton」

text→「Welcome」

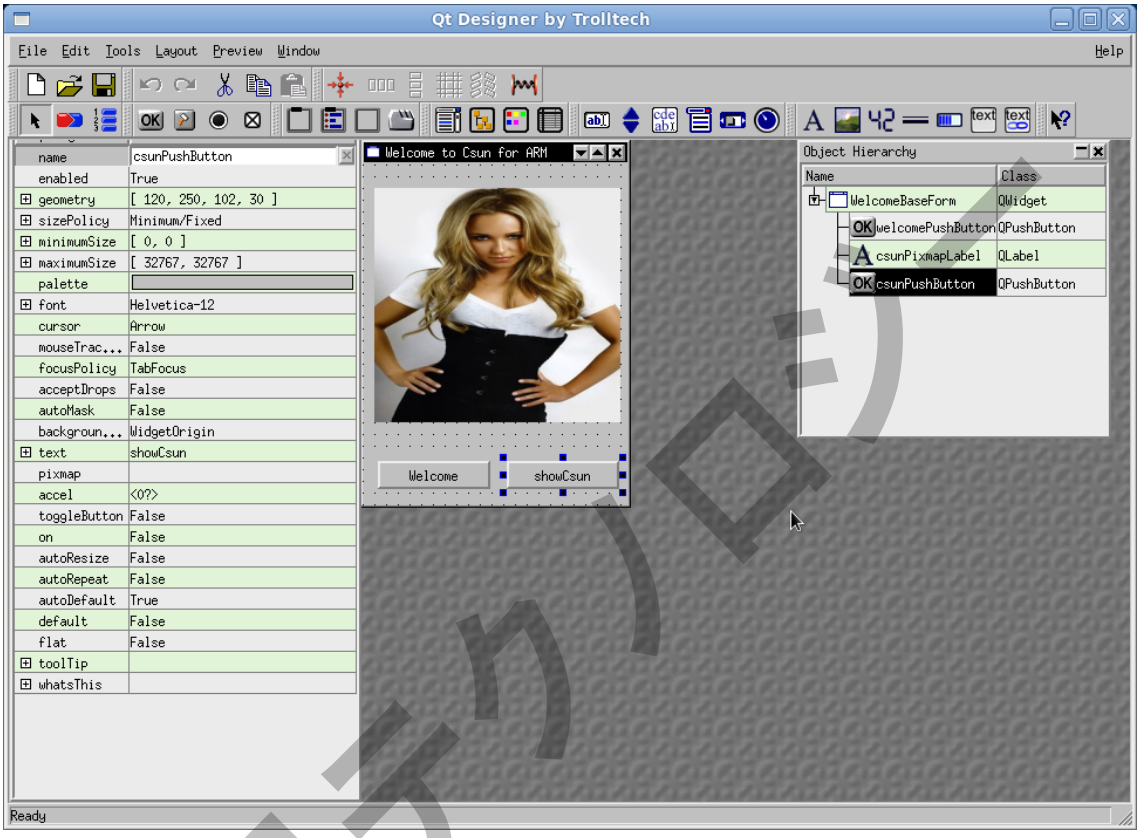


もう一つボタンは同じように追加

ボタン属性設定：

Name→「welcomePushButton」

text→「Welcome」

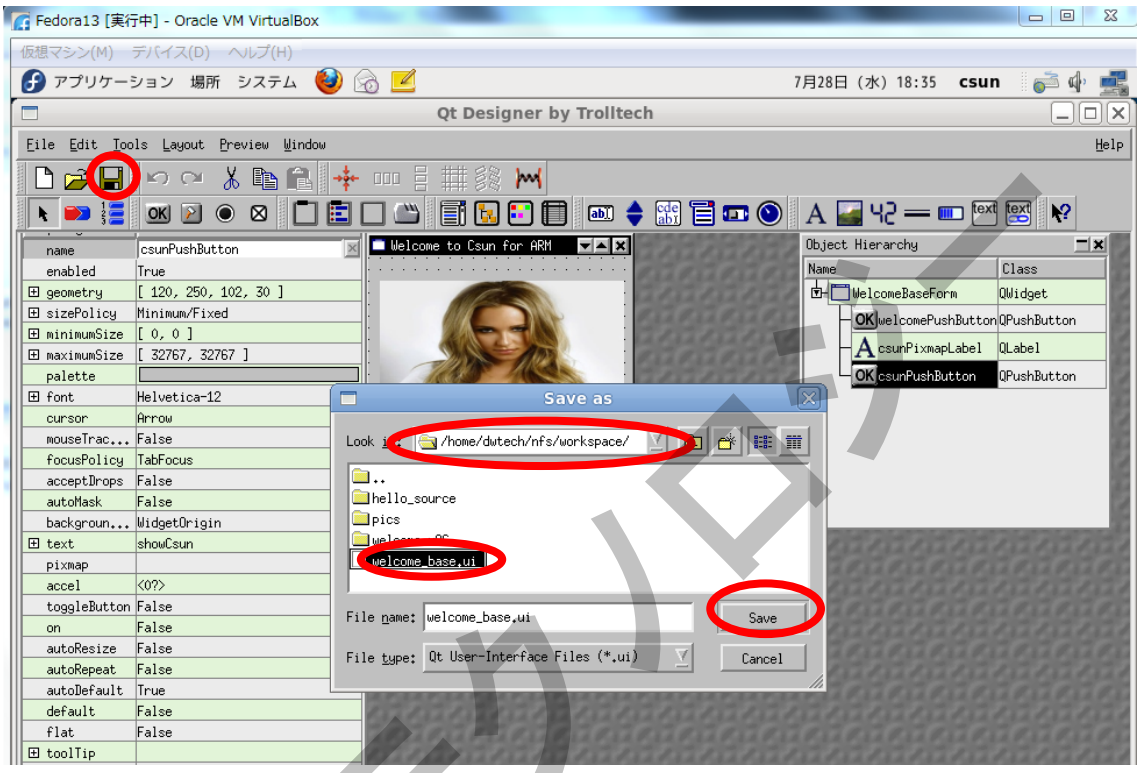


⑥作成した UI 画面を保存

保存場所：「/home/dwtech/nfs/workspace」

保存名前：「welcome_base.ui」

「Save」ボタンを押し保存



4.4 Qt アプリコーディング

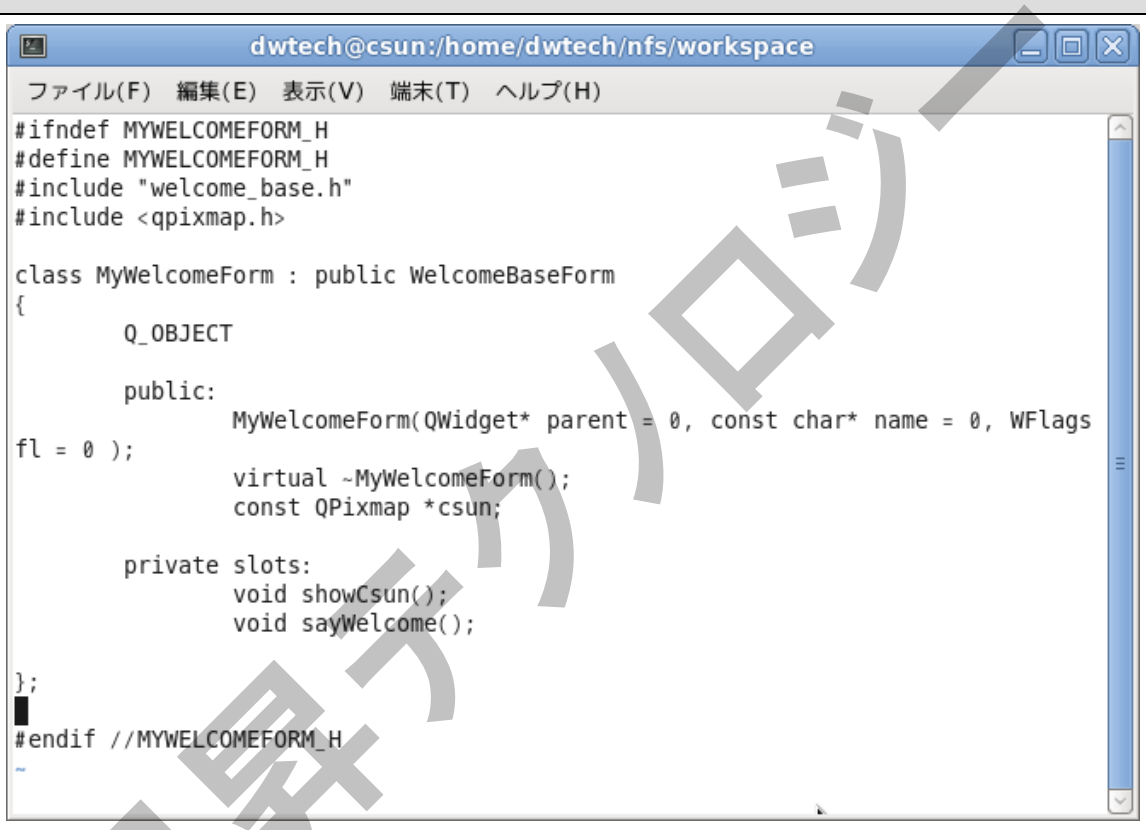
①.h ファイルを作成

◆ワーク用フォルダーに移動

```
[root@csun arm-qtopia]# cd /home/dwtech/nfs/workspace
```

◆本サンプル用の独自.h ファイルを新規作成（名前：「welcome.h」）

```
[root@csun workspace]# vi welcome.h
```



②.cpp ファイル作成（名前：「welcome.cpp」）

```
[root@csun workspace]# vi  welcome.cpp
```

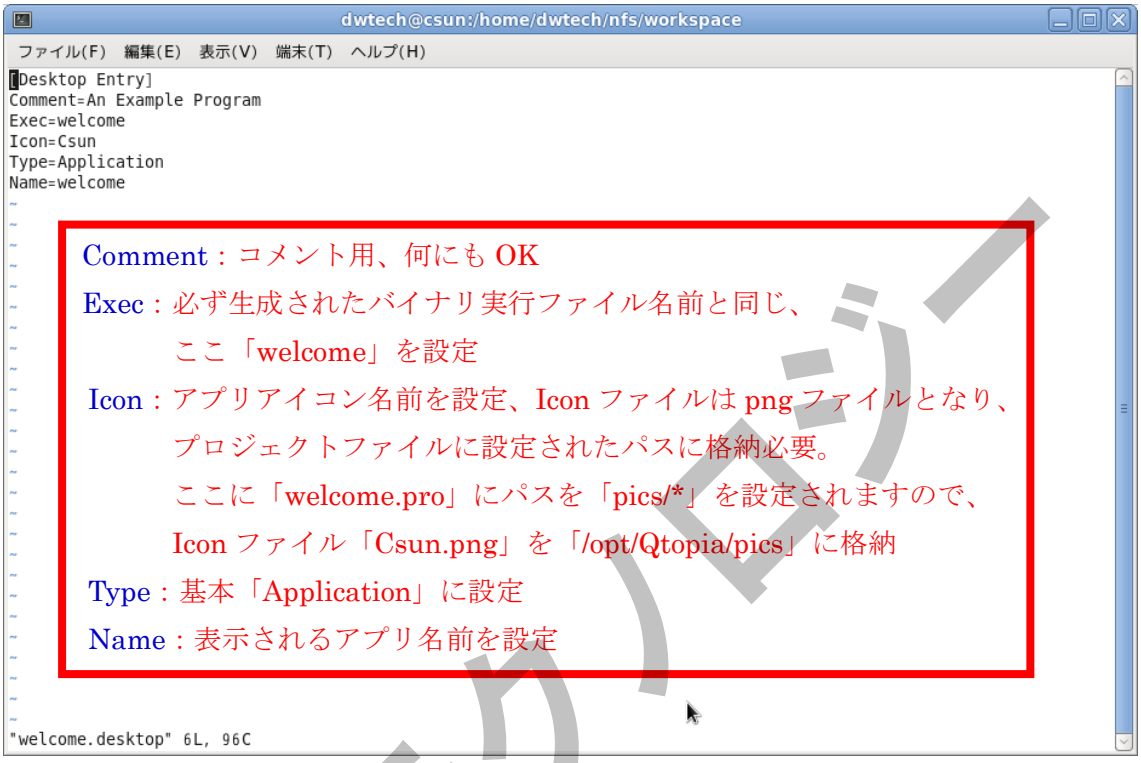
dwtech@csun:/home/dwtech/nfs/welcome_en

ファイル(F)　編集(E)　表示(V)　端末(T)　ヘルプ(H)

```
#include "welcome.h"
#include <qlabel.h>
#include <qpushbutton.h>

MyWelcomeForm::MyWelcomeForm( QWidget* parent, const char* name, WFlags fl)
            :WelcomeBaseForm(parent, name, fl)
{
        csun = new QPixmap("/opt/Qtopia/pics/Csun/showCsun.png");
        connect(welcomePushButton,SIGNAL(clicked()),this,SLOT(sayWelcome()));
        connect(csunPushButton,SIGNAL(clicked()),this,SLOT(showCsun()));
}

MyWelcomeForm::~MyWelcomeForm()
{
}

void MyWelcomeForm::sayWelcome()
{
        csunPixmapLabel->clear();
        csunPixmapLabel->setFrameStyle( QFrame::Panel | QFrame::Sunken );
        csunPixmapLabel->setText( "Hello\n\nWelcome to Csun\n\nfor ARM\n\nEmbedded Development." );
        csunPixmapLabel->setAlignment( AlignCenter );
}

void MyWelcomeForm::showCsun()
{
        csunPixmapLabel->clear();
        csunPixmapLabel->setPixmap(*csun);
}
~
```

③QT アプリ設定ファイル（desktop）を作成（名前：「welcome.desktop」）

```
[root@csun workspace]# vi  welcome.desktop
```



Comment：コメント用、何にも OK

Exec：必ず生成されたバイナリ実行ファイル名前と同じ、
ここ「welcome」を設定

Icon：アプリアイコン名前を設定、Icon ファイルは png ファイルとなり、
プロジェクトファイルに設定されたパスに格納必要。
ここに「welcome.pro」にパスを「pics/*」を設定されますので、
Icon ファイル「Csun.png」を「/opt/Qtopia/pics」に格納

Type：基本「Application」に設定

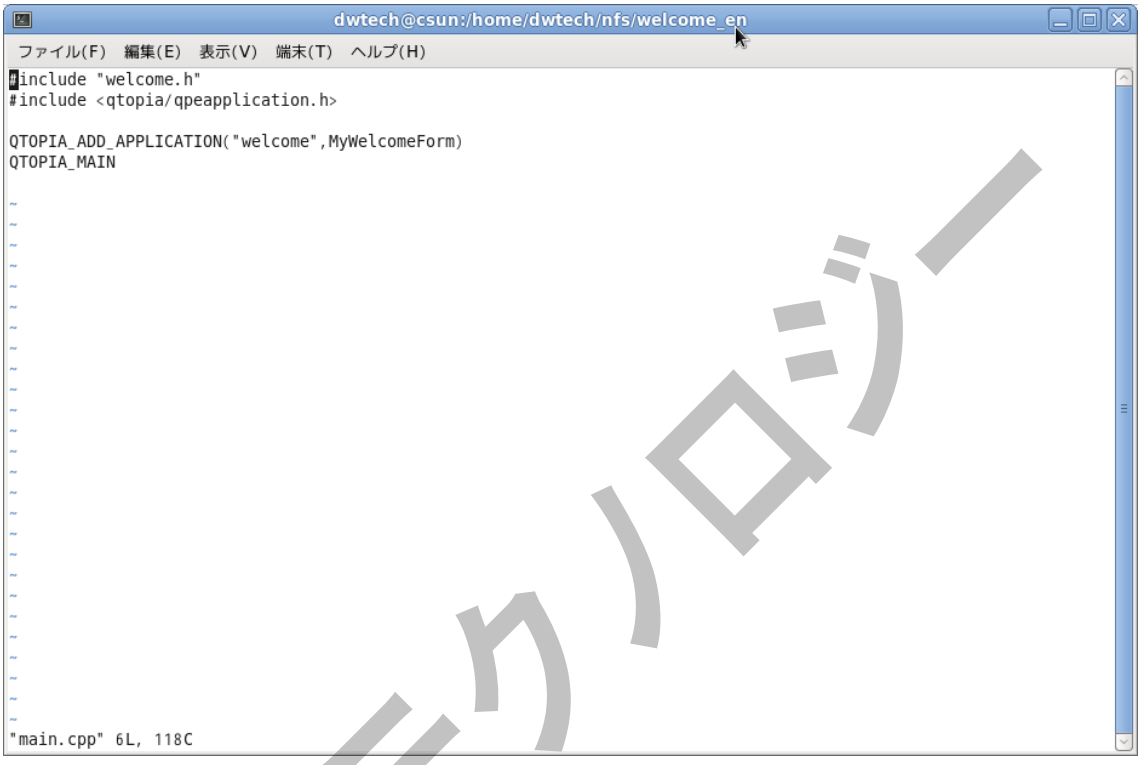
Name：表示されるアプリ名前を設定

④main.cpp 作成

◆vi で新規作成

```
[root@csun workspace]# vi main.cpp
```

下記のような内容を入力



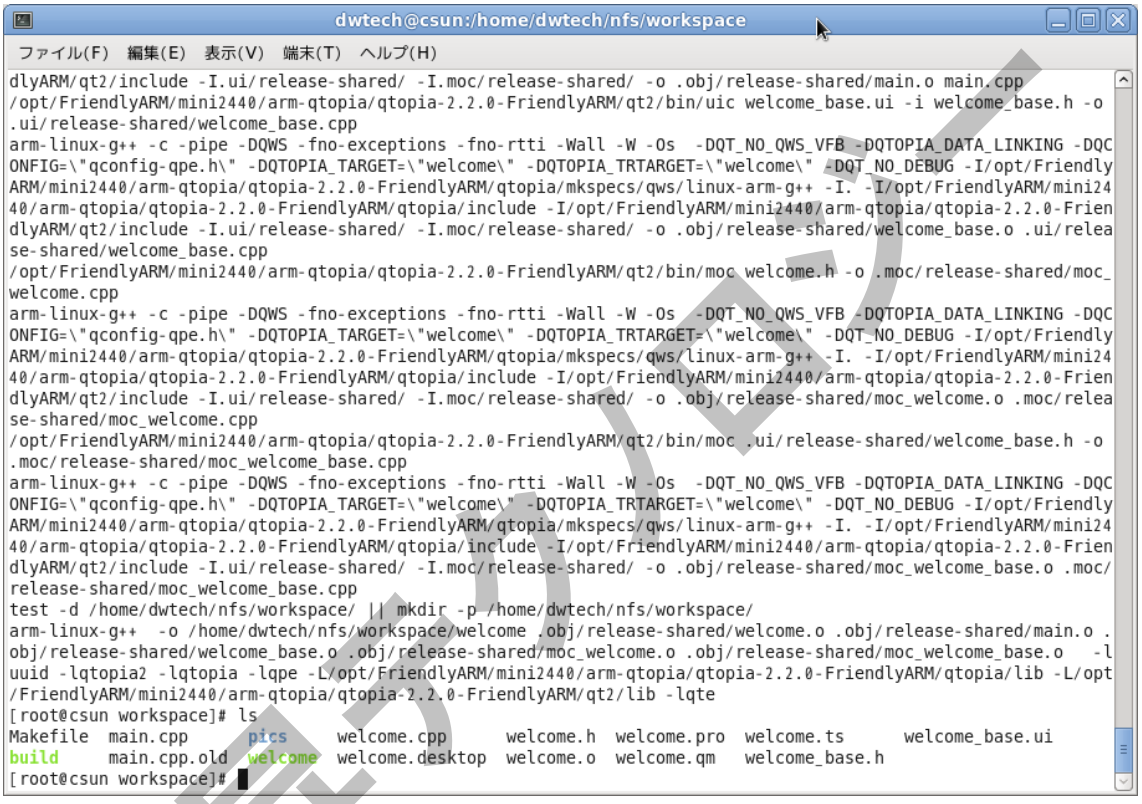
4.5 プロジェクトビルド

＊サンプル hello の build ファイルをそのまま利用

```
[root@csun workspace]# cp /opt/FriendlyARM/mini2440/arm-qtopia/hello/build .
[root@csun workspace]# ./build
```

成功にコンパイルすれば、同じフォルダーに実行ファイル「welcome」を生成された。

ここまでサンプル welcome の作成は完了



第五章　ARM ボードに QT アプリを動かす

5.1 Qt アプリを一つファイルで圧縮

第四章に作った Qt アプリは「/home/dwtech/nfs/workspace」のファイルを圧縮し、Window ホスト共有フォルダー「G:¥VirtualBox¥Share¥Fedora13」にコピー

```
[root@csun workspace]# tar -zcvf welcome.tar.gz ./*
[root@csun workspace]# cp welcome.tar.gz /mnt/share
```

5.2 ARM ボードにダウンロード方法

PC から ARM ボードに QT アプリをダウンロードする方法は色んな方法があるが、例えば、①USB メモリカード、②SD カード、③シリアルポート、④NFS、ここに重点としてシリアルポートという簡単な方法を紹介する。

＊上記方法の説明は Mini2440 マニュアルに全て記載されている。

「MINI2440-linux-2.6.32.2.pdf」の「第四章 初体験(コンソール)」を参照（P39）

「4.1 パソコン側のハイパーターミナルの設定」：③番方法の第一部

「4.3 USB メモリと外付けハードデスク」：①の方法

「4.4 SD/MMC カード」：②の方法

「4.5 シリアルポートでファイルを ARM9 にダウンロード」：③番方法の第二部

「4.15.5 ネットワーク・ファイルシステム(NFS)のマウント」（P52）：④番方法

5.3 シリアルポートでダウンロード

ARM ボードは PC シリアルポートを利用して PC と接続できる。

例：Mini2440（Micro2440）の場合、

DB9 メス・メス型ストレートケーブルを使って PC と接続できる。

製品例：http://www.csun.co.jp/SHOP/200804024.html

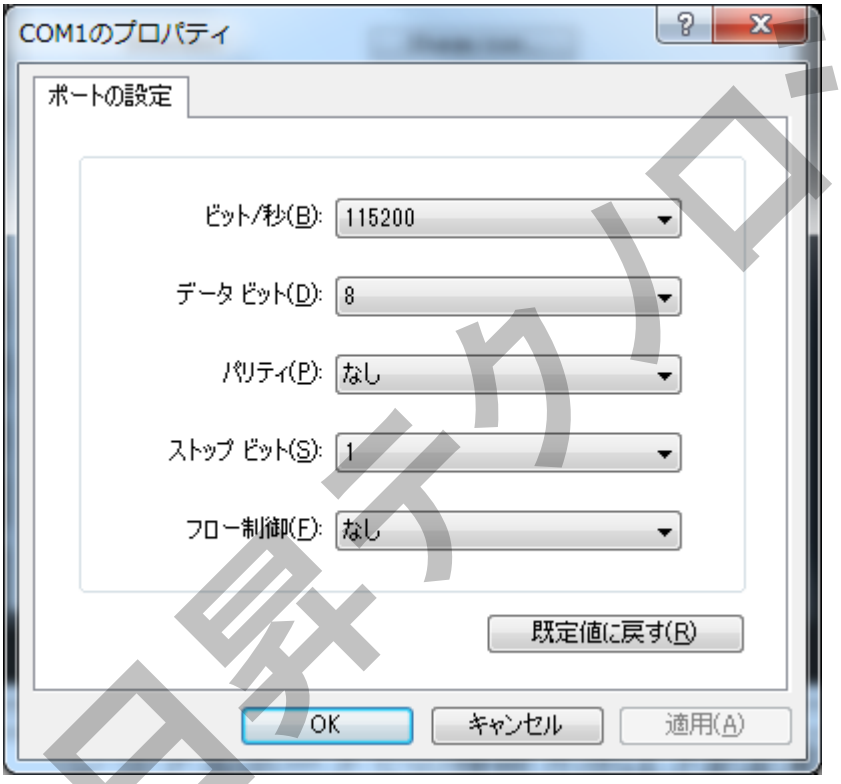
PC シリアルポートはない場合、RS232→USB ケーブルを利用すれば、

PC の USB として利用できる。

製品例：http://www.csun.co.jp/SHOP/200905191.html

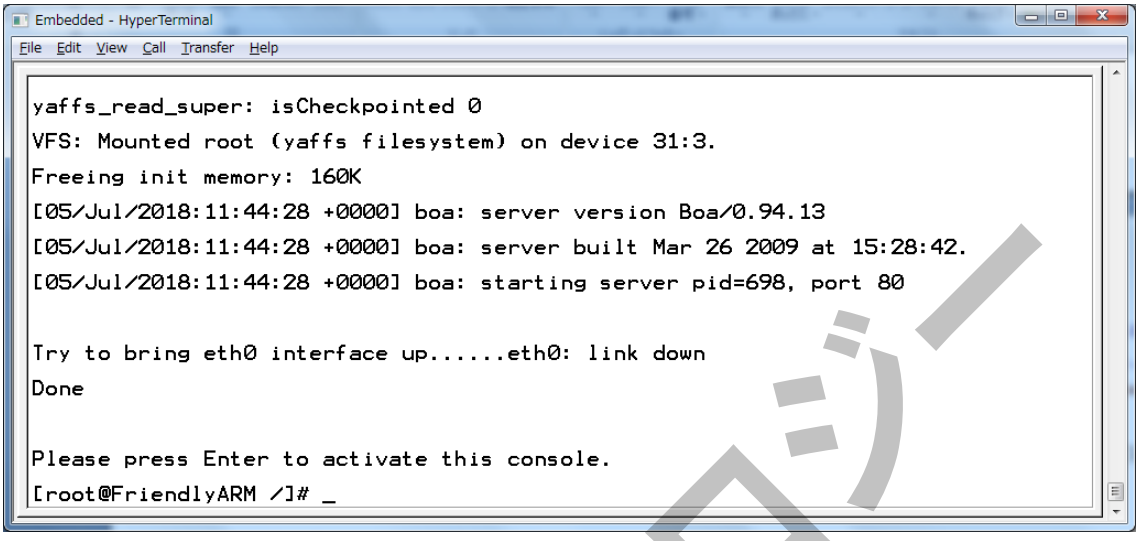
シリアルポートでの接続方法は詳しく上記記載されている Mini2440 のマニュアルを参照

1．ハイパーターミナルの設定

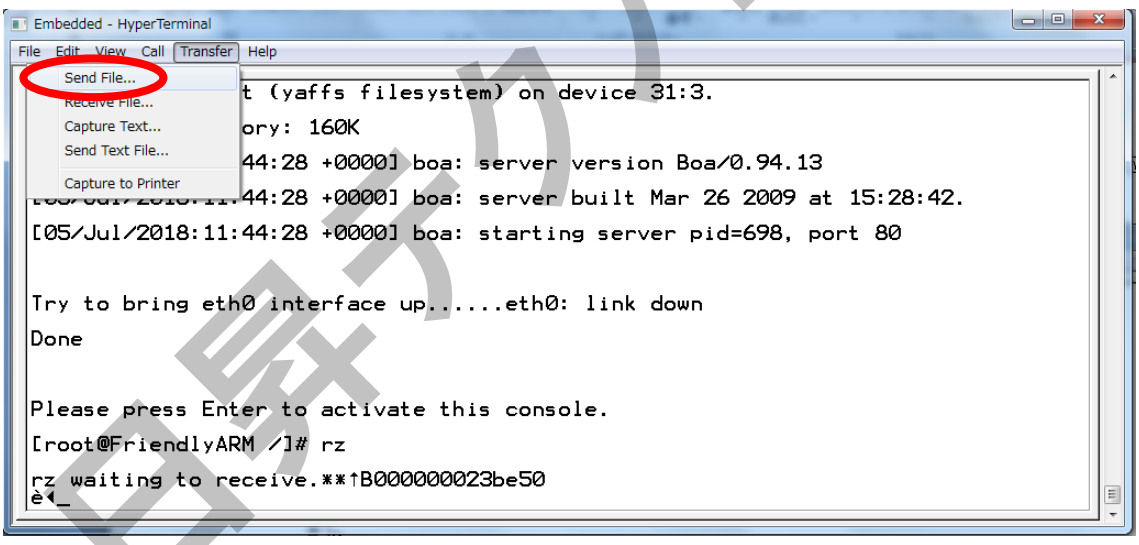


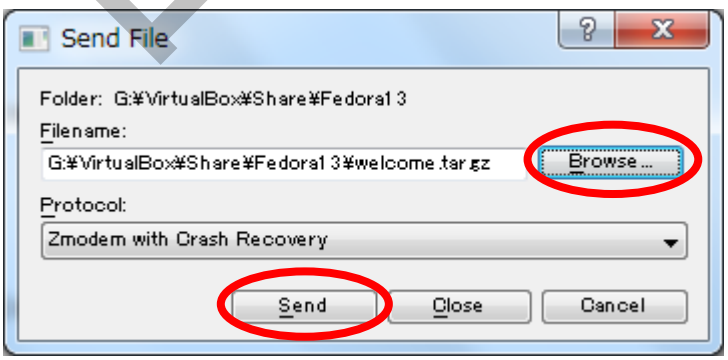
２．ARM ボードへ転送

ARM ボード（Mini2440 と PC は Open-JTAG を経由で接続済）に電源を入れ立ち上げ、起動後画面は下記通り

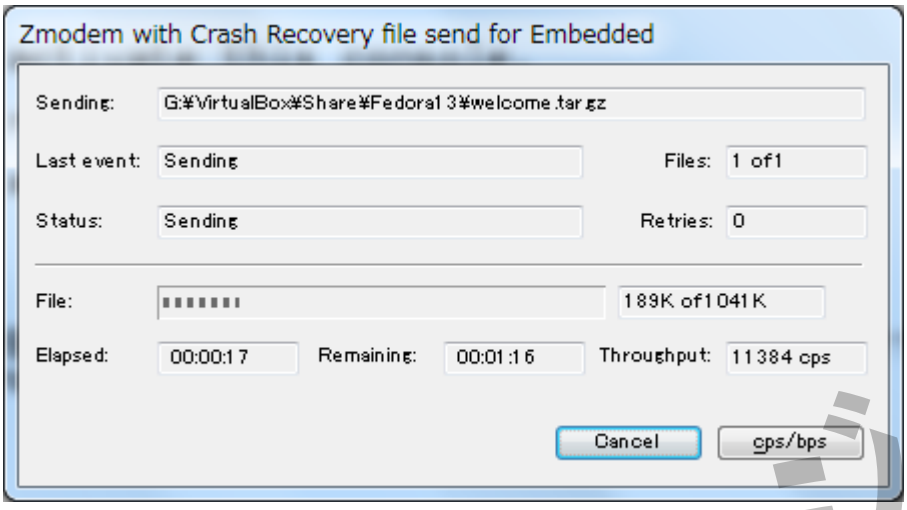


ターミナルでコマンド「rz」を入力し「Transfer」→「Send File」メニューをクリック





転送中の様子



転送完了後、ARM ボードの内容を確認

「/」の中、「welcome.tar.gz」がある事を確認した。

```
[root@FriendlyARM /]# ls
bin             linuxrc         root            var
dev             lost+found      sbin            welcome.tar.gz
etc             mnt             sys             www
home            opt             tmp
lib             proc            usr
```

「/tmp」に移動してから解凍

```
[root@FriendlyARM /]# mv welcome.tar.gz /tmp
[root@FriendlyARM /]# cd tmp
[root@FriendlyARM /tmp]# ls
led-control     qtembedded-0    qtopia-0        welcome.tar.gz
[root@FriendlyARM /tmp]# tar -zxvf welcome.tar.gz
./welcome/Makefile
./welcome/build
./welcome/main.cpp
./welcome/pics/
./welcome/pics/cstlogo.png
./welcome/pics/csun.png
./welcome/pics/Csun/
./welcome/pics/Csun/cstlogo.png
```

5.4 ARM ボードへの配布

①実行ファイル「welcome」を「/opt/Qtopia/bin」にコピー

```
[root@FriendlyARM /tmp]# cd welcome
[root@FriendlyARM welcome]# cp welcome /opt/Qtopia/bin
[root@FriendlyARM welcome]# ls -la /opt/Qtopia/bin/welcome
-rwxr-xr-x    1 root     root       627213 Jul  5 19:56 /opt/Qtopia/bin/welcome
[root@FriendlyARM welcome]#
```

②desktop ファイル「welcome.desktop」を「/opt/Qtopia/apps/Applications」にコピー

```
[root@FriendlyARM welcome]# cp welcome.desktop /opt/Qtopia/apps/Applications
[root@FriendlyARM welcome]# ls -la /opt/Qtopia/apps/Applications/welcome.desktop
-rw-r--r--    1 root     root           96 Jul  5 20:01 /opt/Qtopia/apps/Applications/welcome.desktop
[root@FriendlyARM welcome]#
```

③サンプル必要のイメージファイルを「/opt/Qtopia/pics」にコピー

```
[root@FriendlyARM welcome]# cp -r ./pics/* /opt/Qtopia/pics
[root@FriendlyARM welcome]# ls -la /opt/Qtopia/pics/Csun
drwxr-xr-x    1 root     root          2048 Jul  5 20:04 .
drwxr-xr-x    1 root     root          2048 Jul  5 20:04 ..
-rwxr-xr-x    1 root     root        116692 Jul  5 20:04 girl.png
-rwxr-xr-x    1 root     root         41474 Jul  5 20:04 showCsun.png
[root@FriendlyARM welcome]#
[root@FriendlyARM welcome]# ls -la /opt/Qtopia/pics/csun.png
-rwxr-xr-x    1 root     root         41474 Jul  5 20:04 /opt/Qtopia/pics/Csun.png
[root@FriendlyARM welcome]#
```

ここまでサンプルに必要の全てファイルは ARM ボードにコピーしました。

Mini2440 の画面「アプリケーション」から実行するため、ARM ボードに再起動を掛けしましょう。

5.5 ARM ボードに QT アプリ実行

再起動は完了したら、「アプリケーション」タブ画面に「welcome」という名前のアプリも出て来るべきです。タッチして welcome アプリを動かしましょう。