Interchange Installation

Table of Contents

1. Installation of Interchange

Interchange is the industry's most widely distributed and implemented open source e−commerce platform. This document provides detailed installation instructions for Interchange and the demonstration store included with the program.

2. Getting Ready

1. Verify that Perl 5.005 or newer is installed. At a shell prompt, enter the command `perl -v` to see what version you have.
2. Install the necessary Perl modules from CPAN (from `Bundle::Interchange`).
3. Make sure Apache (or some other web server) is installed and works.
    - Know where the Apache configuration file is located. This is not required, but if available, can help the installer answer the next two questions automatically. The file is usually called httpd.conf or apache.conf and is often located somewhere like:

    /etc/httpd/conf/httpd.conf
    /usr/local/apache/conf/httpd.conf
    - Know where the document root directory is located and have write access to it. It's usually something like:

    /var/www/html
    /home/httpd/html
    /usr/local/apache/htdocs
    - Know where the cgi−bin directory is located and have write access to it. It's usually something like:

    /var/www/cgi−bin
    /home/httpd/cgi−bin
    /usr/local/apache/cgi−bin
4. Install and test your database server (MySQL, PostgreSQL, Oracle, etc.) unless you plan to use Interchange's built−in database system. Know:
    - The DSN (data source name), which is what tells the Perl DBI how to connect to the database. If the database server is on the same host as the Web server, it will probably be something like the following:

    dbi:mysql:
    dbi:Pg:
    dbi:Oracle:
    - The database administrator user name and password
5. Create a new UNIX username and group or choose an existing one (default is `interch`).
6. Decide where to install the Interchange software, and make sure to have write access there. Common choices are:
   /usr/local/interchange
   /usr/lib/interchange
   /home/someuser/interchange
7. Decide where the Interchange catalogs are to be located, and make sure the `interch` user has write access there. Common choices are:
   /usr/local/interchange/catalogs
   /var/lib/interchange
   /home/someuser/interchange/catalogs
8. Choose a name for the first demo catalog, usually `construct`.

3. Installing Interchange

To install Interchange, there are two required steps: downloading the program and setting up a catalog. The following sections will explain this installation process in detail.

1. Obtain, decompress, and untar the distribution on a UNIX server:
   ```
   gzip -dc interchange-4.6.x.tar.gz | tar xvf -
   ```

Note: Check the Interchange developer site at http://developer.akopia.com/ for any patches that might have been issued since the last release.

---

1. Run the configuration script:
   ```
   cd interchange-4*
   ./configure
   ```

1. When asked where to install Interchange, indicate any directory with write privileges. The defaults are:
   ```
   Installed as root: /usr/local/interchange
   Installed as user: ~/interchange
   ```

Note: ~ is shorthand for the home directory. This directory is referred to later in the documentation as VendRoot or the Interchange software directory.

---

1. Installing the link program requires write permission on the CGI and HTML directories. The definition of these are typically part of the HTTP server configuration. On an Apache server, the CGI directory is set by the Apache configuration variable ScriptAlias; the HTML directory is set by Apache configuration variable DocumentRoot.

3.1. Setting Up a Catalog

After completing the above script, Interchange can be installed. Installation of the demo catalogs is strongly recommended as a starting point for a custom catalog. Interchange will not function properly until a catalog is created.

Begin by separating the directories that hold the catalog pages and databases. The `makecat` program supplied with Interchange does this.

Note: Any pages with Interchange elements/tags go in the directory set by the `PageDir` directive (the default is `~/catalogs/catalog_name/pages`). For the demos supplied with Interchange, this means that only one or a few pages will be copied to the HTML directory, with the remainder of the pages staying in the directory defined as PageDir.

If working with an ISP where all of the files are in HTML document space, disable all access to the Interchange catalog directory with the proper HTTP access restrictions. Normally that is done by creating a

.htaccess file like this:

```
<Limit GET POST>
order allow,deny
deny from all
</Limit>
```

If unable to do this, do not run Interchange unless file permissions can be set such that files will not be served. However, security will be a problem and customers' personal information could be placed at risk.

3.2. makecat − Setup a Catalog from a Template

The supplied `makecat` script, which is in the Interchange program directory `bin`, is designed to set up a catalog based on the user's server configuration. It interrogates the user for parameters like which directories to use, a URL to base the catalog in, HTTP server definitions, and file ownership. It is self−documenting, meaning it asks questions and gives relevant examples.

The `makecat` script requires a template catalog to operate properly. The "construct" demo template is distributed with Interchange. Other demo catalogs are available at `http://developer.akopia.com/`.

Note: A catalog can only be created once. All further configuration is done by editing the files within the catalog directory.

A catalog template contains an image of a configured catalog. The best way to see what the makecat program does is to configure the simple demo and then run a recursive `diff` on the template and configured catalog directories:

```
cd /usr/local/interchange
diff −r construct catalogs/construct
```

The files are mostly identical, except that certain macro strings have been replaced with the answers given to the script. For example, if `www.mydomain.com` was answered at the prompt for a server name, this difference would appear in the catalog.cfg file:

```
# template
Variable SERVER_NAME  __MVC_SERVERNAME__

# configured catalog
Variable SERVER_NAME  www.mydomain.com
```

The macro string __MVC_SERVERNAME__ was substituted with the answer to the question about server name. In the same way, other variables are substituted, and include:

```
MVC_BASEDIR      MVC_IMAGEDIR
MVC_CATROOT      MVC_IMAGEURL
MVC_CATUSER      MVC_MAILORDERTO
MVC_CGIBASE      MVC_MINIVENDGROUP
MVC_CGIDIR       MVC_MINIVENDUSER
MVC_CGIURL       MVC_SAMPLEHTML
MVC_DEMOTYPE     MVC_SAMPLEURL
MVC_DOCUMENTROOT MVC_VENDROOT
```

```
MVC_ENCRYPTOR
```

Note: Not all of these variables are present in the "construct" template, and more may be defined. In fact, any environment variable that is set and begins with MVC_ will be substituted for by the `makecat` script. For example, to set up a configurable parameter to customize the COMPANY variable in catalog.cfg, run a pre−qualifying script that set the environment variable MVC_COMPANY and then place in the catalog.cfg file:

Variable COMPANY __MVC_COMPANY__

All files within a template directory are substituted for macros, not just the catalog.cfg file. There are two special directories named `html` and `images`. These will be recursively copied to the directories defined as SampleHTML and ImageDir.

Note: The template directory is located in the Interchange software directory, i.e., where `interchange.cfg` resides. Avoid editing files in the template directory. To create a new template, it is recommended that it should be named something besides 'construct' and a copy of the `construct` demo directory be used as a starting point. Templates are normally placed in the Interchange base directory, but can be located anywhere. The script will prompt for the location if it cannot find a template.

In addition to the standard parameters prompted for by Interchange, and the standard catalog creation procedure, four other files in the `config` directory of the template may be defined:

```
additional_fields −− file with more parameters for macro substitution
additional_help   −− extended description for the additional_fields
precopy_commands  −− commands passed to the system prior to catalog copy
postcopy_commands −− commands passed to the system after catalog copy
```

All files are paragraph−based. In other words, a blank line (with no spaces) terminates the individual setting.

The additional_fields file contains:

```
PARAM
The prompt. Set PARAM to?
The default value of PARAM
```

This would cause a question during makecat:

```
The prompt. Set PARAM to?.....[The default value of PARAM]
```

If the additional_help file is present, additional instructions for PARAM may be provided.

```
PARAM

These are additional instructions for PARAM, and they
may span multiple lines up to the first blank line.
```

The prompt would now be:

```
These are additional instructions for PARAM, and they
may span multiple lines up to the first blank line.

The prompt. Set PARAM to?.....[The default value of PARAM]
```

If the file config/precopy_commands exists, it will be read as a command followed by the prompt/help value.

```
mysqladmin create __MVC_CATALOGNAME__
We need to create an SQL database for your Interchange
database tables.
```

This will cause the prompt:

```
We need to create an SQL database for your Interchange
database tables.

Run command "mysqladmin create simple"?
```

If the response is "y" or "yes," the command will be run by passing it through the Perl system() function. As with any of the additional configuration files, MVC_PARAM macro substitution is performed on the command and help. Proper permissions for the command are required.

The file config/postcopy_commands is exactly the same as <precopy_commands>, except the prompt occurs after the catalog files are copied and macro substitution is performed on all files.

There may also be SubCatalog directives:

```
SubCatalog easy simple /home/catalogs/simple /cgi-bin/easy
```

easy

The name of the subcatalog, which also controls the name of the subcatalog configuration file. In this case, it is `easy.cfg`.

simple

The name of the base configuration that will be the basis for the catalog. Parameters in the easy.cfg file that are different will override those in the catalog.cfg file for the base configuration.

The remaining parameters are similar to the Catalog directive.

Additional interchange.cfg parameters set up administrative parameters that are catalog wide. See the server configuration file for details on each of these.

Each catalog can be completely independent with different databases, or catalogs can share pages, databases, and session files. This means that several catalogs can share the same information, allowing "virtual malls."

3.3. Manual Installation of Catalogs

An Interchange installation is complex, and requires quite a few distinct steps. Please see the `iccattut` document for a full tutorial on build a catalog by hand.

Normally you will want to use the interactive configuration script `makecat` that is included with

Interchange. It merely does automatically what is described below. It makes the process much easier, and will install the demo catalog. It can also be used to install from a custom catalog template. See the supplied demo template `foundation` (`construct` for Interchange 4.6) for examples.

3.4. Needed Directories

The Interchange program, and its supporting libraries, should go into one directory as installed by the installation program. User catalog pages, user databases, and user configuration files should all go into their private directories. Because the catalog pages are served through the Interchange CGI program and contain nonstandard elements, they should not be put into a public Web directory, nor do they need to have world−readable file permissions.

Since catalogs are all run under one server, permissions are complex and very important.

A public Web directory for inline image graphic files is needed. Interchange does not serve the images, only the HTML tags calling them. A useful convention is to place all buttonbars, backgrounds, and icons in the /images directory, with the catalog items located in the /images/catalog directory. Regardless of the directory structure, an *absolute path* must be used. Relative paths are unacceptable. Use the *ImageDir* directive, which places that as the absolute path in front of all relative IMG and INPUT SRC, and BODY, TABLE, TR, TH, and TD BACKGROUND specifications.

A cgi−bin directory in which to put the vlink or tlink program is also needed.

4. Building the Demo Store

Note: Red Hat can set up a server with Interchange pre−installed. See the our Web site at `http://www.akopia.com/` for more details.

This detailed process describes the entire installation and setup process of the demo store. The text seen during this process will be displayed with <−− next to it. The text marked with −−> identifies the required response.

In order to begin, the following permissions on the server are needed:

- Write on the catalog install directory
- Write on the Interchange install directory
- Write on the cgi−bin directory
- Write on the httpd document root

1. Change to the directory in which Interchange is installed:
   ```
   cd /home/user/mvend
   ```

1. Run the makecat script
   ```
   bin/makecat
   ```

A series of questions will be asked at this point.
Note: The answers can be changed later by editing the configuration scripts.

1. ```
   <−−Select a short, mnemonic name for the catalog. This will be
   <−−used to set the defaults for naming the catalog, executable,
   <−−and directory, so you will have to type in this name frequently.
   <−−NOTE: This will be the name of 'vlink' or 'tlink,' the link CGI
   <−−program. Depending on your CGI setup, it may also have the
   <−−extension .cgi added.
   <−−If you are doing the demo for the first time, you might use "basic."
   ```

This will be the name of the catalog. The installer will use this name as the unique identifier for this catalog. This will be the name of the cgi script that intercepts interchange requests. It will also be used as the default for the directory in which to store catalog fields.

```
<−−Catalog name?

−−>example
```

"example" was chosen as the catalog name for this walkthrough.

1. ```
   <−−makecat −− Interchange catalog installation program.
   <−−*** We will be making a catalog named 'example'. ***
   <−−########## BEGINNING CATALOG CONFIGURATION ##########
   <−−# The server name, something like: www.company.com
   <−−#                                  www.company.com:8000
   <−−#                                  www.company.com/~yourname
   <−−#
   <−−Server name? [localhost]

   −−>localhost
   ```

It is the name at which the web server is receiving connections. Interchange will also be running on this server.

1.
```
<--# The type of demo catalog to use. Standard types
<--# distributed are:
<--#
<--#    simple   -- database-based catalog, not really simple
<--#    basic    -- simplified set of features
<--#    barry    -- Barry's Books version of "simple"
<--#
<--# If you have defined your own custom template catalog,
<--# you can enter it's name.
<--#
<--# If you are new to Interchange and not a sophisticated web designer,
<--# use "barry" to start with.
<--DemoType? [basic]

-->barry
```

There are currently three types of demo's. "Basic" is a basic site implementation of "The Art Store." "Simple" is a simple implementation of "The Art Store." "Barry" is the newest demo. It has the same feature set as Simple, but with the "Barry's Bikes, Books, and Birkenstocks" formerly of Tallyman fame. This is the one being used in this walkthrough.

1.
```
<--# The email address where orders for this catalog should go.
<--# To have a secure catalog, either this should be a local user name and
<--# not go over the Internet -- or use the PGP option.
<--#
<--MailOrderTo? [orders@barry.com]

-->user@localhost.com
```

The barry catalog is designed to email the orders, once they are completed, to the person responsible for handling orders.

1.
```
<--# Where the Interchange files for this catalog will go, pages,
<--# products, config and all. This should not be in HTML document
<--# space! Usually a 'catalogs' directory below your home directory
<--# works well. Remember, you will want a test catalog and an online
<--# catalog.
<--#
<--CatRoot? [/home/user/catalogs/example]

-->/home/user/catalogs/example
```

This is where the catalog config files and working data reside. The default location is a good place for this.

1.
```
<--# The location of the normal CGI directory. This is a
<--# file path, not a script alias.
<--#
<--# If all of your CGI programs must end in .cgi, this is
<--# should be the same as your HTML directory.
<--#
<--CgiDir? [/usr/local/httpd/cgi-bin]

-->/usr/local/httpd/cgi-bin
```

A file with the store's name will go into the cgi−bin dir. This program (vlink or tlink) will provide the bridge

between the Web server and the Interchange server.

```
1. <--# The URL location of the CGI program, without the
   <--# or server name.
   <--#
   <--#
   <--#                  ^^^^^^^^^^^^
   <--#
   <--#
   <--#                  ^^^^^^^^^^^
   <--#
   <--CgiUrl? [/cgi-bin/example]

   -->/cgi-bin/example
```

This is the URL to the cgi program for your catalog (example in this case). This would be the script−alias in the httpd.conf set up for your cgi−bin.

```
1. <--#
   <--# Additional URL locations for the CGI program, as with CgiUrl.
   <--# This is used when calling the catalog from more than one place,
   <--# perhaps because your secure server is not the same name as the
   <--# non-secure one.
   <--#
   <--#
   <--#                    ^^^^^^^^^^^^^^^^
   <--#
   <--# We set it to the name of the catalog by default to enable the
   <--# internal HTTTP server.
   <--#
   <--Aliases? [/example]

   -->/example
```

This one is "beyond the scope of this document." For purposes of getting started, this option can be safely ignored. For those who really want to know what it is for right now, see the documentation for minivend.cfg.

```
1. <--# The base directory for HTML for this (possibly virtual) domain.
   <--# This is a directory path name, not a URL -- it is your HTML
   <--# directory.
   <--#
   <--DocumentRoot? [/usr/local/httpd/htdocs]

   -->/usr/local/httpd/htdocs
```

This is the directory path to theHTML document root. A number of static elements of the catalog will be stored here.

```
1. <--# Where the sample HTML files (not Interchange pages) should be
   <--# installed. There is a difference.  Usually a subdirectory of
   <--# your HTML directory.
   <--#
   <--SampleHtml? [/usr/local/httpd/htdocs/example]

   -->/usr/local/httpd/htdocs/example
```

Interchange will create a directory under the HTML document root in order to store the static HTML pages for the demo catalog.

```
<--# Where the image files should be copied. A directory path
<--# name, not a URL.
<--#
<--ImageDir? [/home/sonny/public_html/example/images]

-->/home/sonny/public_html/example/images
```

Additionally, a place for the images.

```
<--# The URL base for the sample images. Sets the ImageDir
<--# directive in the catalog configuration file. This is a URL
<--# fragment, not a directory or file name.
<--#
<--#
<--#                   ^^^^^^^^^^^^^^
<--#
<--ImageUrl? [/example/images]
```

This is the URL for the images stored in the directory created above. Interchange needs this in order to create tags for the catalogs dynamic pages.

```
<--Interchange can use either UNIX- or internet-domain sockets.
<--Most ISPs would prefer UNIX mode, and it is more secure.
<--If you already have a program there, or use a common program
<--and the FullURL directive, select NONE. You will then need
<--to copy the program by hand or otherwise ensure its presence.
<--INET or UNIX mode? [UNIX]

-->UNIX
```

Again, this is outside this document's scope. UNIX sockets will work just fine.

```
<--Do you use CGIWRAP or SUEXEC? [n]

-->n
```

If the answer is unknown, "no" should be the right answer.

```
<--Checking directories.......mkdir /home/sonny/akopia/catalogs2/example
<--mkdir /home/sonny/public_html/example
<--mkdir /home/sonny/public_html/example/images
<--done.
<--Copying demo files.......................found more to ask.
```

Processing.

```
<--Your company name: .....................[Barry's Books]
<--Your company address: ..................[123 Any St.]
<--Your company city/state/zip: ...........[Anytown, USA 00000]
<--Your company phone: .....................[(555) 555-5555]
<--Your company fax number: ................[(555) 555-5556]
<--Your company tollfree number (if any):..[(888) 555-5555]
```

Here are several questions concerning the store's information. These are the defaults. Each can be modified.

```
<--Interchange can do order pages in any way; two examples are provided.
<--The default is single-page.
<--Set to 1 to enable multi-page order screens: ..
```

```
-->1
```

This demo catalog accommodates configuration of a single or multiple page order finalization. The multiple page setup will break the confirmation process into several pages asking for shipping and billing address, payment info, and so on. The single page setup groups this together on a single page.

1.
```
<--There are three color schemes available as an example of
<--how you might template catalogs. Select one of:
<--
<--  brown1  blue1  yellow1
<--
<--Select color scheme: ...................[green1]

-->blue1
```

Three color scheme templates are available. These particular ones are designed around the demo catalog.

1.
```
<--For the US, this is usually the state(s) your business is
<--located in. Non-US users will probably have to set this
<--differently depending on their tax laws.
<--Area(s) to tax in: .....................[VA UT]

-->VA UT
```

List the states for which to calculate sales tax.

1.
```
<--For US users, this looks like STATE=RATE, where the rate
<--is in percent. The default below taxes Ohio at 6% and
<--Illinois at 7.25 percent. More states can be added or
<--you can have only one. Should correspond to TAXAREA.
<--Percentage rate(s) for tax in different areas: ..[VA=7.5, UT=7.25]

-->VA=4.5 UT=6.35
```

The percentage of tax to calculate for each state.

1.
```
<--For the UPS lookup, the standard UPS tables are normally
<--used. If you want to add a handling charge, do it here.
<--Amount to add on to standard UPS costs: ..[3.00]

-->3.00
```

Add this amount to the shipping charges for each order when using the UPS shipping methods.

1.
```
<--Origin zip code for UPS lookups: .......[00000]

-->83617
```

UPS calculates shipping from one zip code to another zip code. This zip code is the origin of all shipments. If shipping from multiple locations, consult the documentation or contact us for paid support.

1.
```
<--Interchange has a workable internal database, but many things will
<--work better if you use a SQL database. Interchange
<--can configure MySQL and Postgres in a test configuration.
<--Set to 1 if you want to use MySQL or Postgres: ..
```

```
-->0
```

For the purposes of this procedure, use the Interchange internal database. Consult the documentation for information about using the SQL database of choice.

1.
```
<--You can use MiniMate, Interchange's companion configuration
<--interface, to do upload/download of files, manipulation of
<--the database, reconfiguration of the catalog, and much more.
<--To enable MiniMate, you will need a "super-user" account
<--name that has full access.
<--Account name that will control this catalog: ..[sonny]

-->user
```

In this and all future versions, Minimate has been replaced with the Interchange administrative interface. Tallyman users should find it very familiar since it's based on Tallyman's interface. (See the Interchange Back-Office document.) This gives the user with full administrative permissions complete control of the catalog.
The administrative password for [user] above is "pass." This is a very weak password. It should be changed with the Administrators->Change password function in the UI.

1.
```
<--Found system commands to run.
```

This prompt may occur if MySQL or Postgress was selected. Otherwise, it will be silent (and do nothing). If Mysql or Postgres were selected, and this doesn't work, the catalog will probably not work.

1.
```
<--done.
<--Moving link program to /usr/local/httpd/cgi-bin/example.....done.
<--Moving HTML files to /home/sonny/public_html/example........done.
<--Moving image files to /home/sonny/public_html/example/images..done.
```

Processing.

1.
```
<--Add catalog to minivend.cfg? [y]

-->y
```

This will add the catalog to the list that Interchange will use when it starts up.

1.
```
<--Done with installation. If my reading of your input is correct, you
<--should be able to access the demo catalog with the following URL:
<--
<--
<--
<--In any case, you should get direct access at:
<--
<--
<--
<--That is, after you START or RESTART the Interchange server.  8-)
<--It is best done with:
<--
<--        /home/user/mvend/bin/minivend -r
<--
<--For session expiration, you might want to place a line like this in your
<--crontab:
<--
<--44 4 * * * /home/user/mvend/bin/expireall -r
```

```
<--
<--It will prevent the session databases from getting too large.
```

Installation is complete.

1. Lastly, start the Interchange server: /home/user/mvend/bin/minivend −r
   Text similar to this should be displayed on the monitor screen:

```
Low traffic settings.
 Calling UI....
 ....UI is loaded....
 Interchange V4.6.x
 Configuring catalog example...done.
 Interchange server started in INET and UNIX mode(s) (process id 22200)
```

At this point the Interchange is running and waiting for connections.

4.1. Installation Troubleshooting

Interchange uses the services of other complex programs, such as Web servers and relational databases, to work. Therefore, when there is a problem, check these programs before checking Interchange. It may have to do with Perl or the HTTP server setup. In fact, over the past four years, many more basic installation problems have to do with those than with Interchange itself.

If an error message is received about not being able to find libraries, or a core dump has occurred, or a segment fault message, it is always improperly built or configured Perl. Contact the system administrator or install a new Perl.

The `makecat` program is intended to be used to create the starting point for the catalog. If the demo does not work the first time, keep trying. If it still does not work, try running in INET mode.

Check the two error log files: `error.log` in the Interchange home directory (where interchange.cfg resides) and `error.log` in the catalog directory (where catalog.cfg resides; there can be many of these). Many problems can be diagnosed quickly if these error logs are consulted.

Check the README file, the FAQ, and mail list archive at the official Interchange web site for information:

```
http://developer.akopia.com/
```

Double check the following items:

1. Using UNIX sockets?
   - ♦ Check that the `vlink` program is SUID, or the appropriate changes have been made in the SocketPerms directive. Unless the files are world−writable, the vlink program and the Interchange server must run as the same user ID! If running CGI−WRAP or SUEXEC, the `vlink` program must not be SUID.
   - ♦ If having trouble with the vlink program (named `construct` in the demo configuration), try re−running `makecat` and using INET mode instead. (Or copy the `tlink` INET mode link program over `vlink`). This should work unchanged for many systems.
   - ♦ If using an ISP or have a non−standard network configuration, some changes to interchange.cfg are necessary. For `tlink` to work, the proper host name(s) must be configured into the TcpHost directive in interchange.cfg. The program selects port 7786 by default (the ASCII codes for "M" and "V"). If another port is used, it must be set to the same

number in both the tlink program (by running `compile_link`) and the `minivend.cfg` file. The `tlink` program does not need to be SUID.

2. Proper file permissions?
   - The Interchange server should not run as the user `nobody`! The program files can be owned by anyone, but any databases, ASCII database source files, error logs, and the directory that holds them must be writable by the proper user ID, that is the one that is executing the MiniVend program.
   - The best way to operate in multi−user, multiple catalog setups is to create a special `interch` user, then put that user in the group that contains each catalog user. If a group is defined for each individual user, this provides the best security. All associated files can be in 660 or 770 mode. There should be no problems with permissions and no problems with security.
3. Is the `vlink` program being executed on a machine that has the socket file `etc/socket` on a directly attached disk?
   - UNIX−domain sockets will not work on NFS−mounted file systems! This means that the server `minivend` and the CGI program `vlink` must be executing on the same machine.
   - The `tlink` program does not have this problem, but it must have the proper host name(s) and TCP ports set in the TcpHost and TcpPort directives in `interchange.cfg`. Also, be careful of security if sensitive information, like customer credit card numbers, is being placed on a network wire.

5. Installing Perl Modules without Root Access

Installing Interchange without root access is no problem. However, installing Perl modules without root access is a little trickier.

You must build your makefile to work in your home dir. Something like:

```
PREFIX=~/usr/local \
INSTALLPRIVLIB=~/usr/local/lib/perl5 \
INSTALLSCRIPT=~/usr/local/bin \
INSTALLSITELIB=~/usr/local/lib/perl5/site_perl \
INSTALLBIN=~/usr/local/bin \
INSTALLMAN1DIR=~/usr/local/lib/perl5/man \
INSTALLMAN3DIR=~/usr/local/lib/perl5/man/man3
```

Put this in a file, say 'installopts', and use it for the Makefile.PL.

```
perl Makefile.PL `cat installopts`
```

Then, forget ./config. Just do:

```
make
make test
make install
```

Some of the tests may fail, but that's probably ok.

Also make sure to install Bundle::Interchange, which will need the same config data as you put into 'installopts'.

6. Frequently Asked Questions

6.1. Where are the pages?

Interchange pages are not kept in normal HTML space. Look in the catalog subdirectory pages. The pages are always filtered through the Interchange daemon before being delivered.

6.2. Where are the images?

Interchange is a CGI program, and if relative image paths are used, IMG tags like the following will occur:

<IMG SRC="/cgi−bin/simple/../whatever.jpg">

Interchange, by default, uses an `ImageDir` for a prefix. In the demo, image specs that have no absolute path information are prefixed with `/simple/images/`.

In an Interchange page, this tag:

```
<IMG SRC="ordernow.gif">
```

will become this:

```
<IMG SRC="/simple/images/ordernow.gif">
```

This tag:

```
<IMG SRC="items/00−0011.jpg">
```

will become this:

```
<IMG SRC="/simple/images/items/00−0011.jpg">
```

Absolute image paths are not affected. An image, such as </other/images/whatever.gif>, will not be changed.