Building with oi-userland

> ⚠ **Deprecation warning**
>
> The page has been migrated to http://docs.openindiana.org/dev/userland/ . Avoid touching this wiki page.

Using OpenIndiana's unified build system

OpenIndiana Hipster's primary build framework **oi-userland** is set to replace all existing software consolidations, vastly simplifying how we build the operating system.

It is also tied into Hipster's continuous integration platform.

When an update is committed to the oi-userland git repository:

- an automated build is kicked off,
- then automatically the binary package will be published to the `/hipster` repository,
- finally, the status of the build will be reported by oibot to #oi-dev freenode IRC channel.

Overview of oi-userland

Originally **oi-userland** is a fork of Oracle's userland-gate, which we have extended and added additional software to: the layout is very similar.

Inside oi-userland is a directory called "`components`", under which lives a directory for each software package. Inside each of these software package directories is a main `Makefile`, as well as one or more `.p5m` IPS manifest files, and there may also be license files and patches.

The `Makefile` essentially contains a "build recipe".

To build a piece of software, you simply `cd` into the directory of the software, and type "`make TARGET`", where **TARGET** can be one of:

`prep`	Download, extract and patch the software archive
`build`	Configure and build the software
`install`	Install the software into the prototype directory
`sample-manifest`	Create a sample manifest file in the build directory
pre-publish	Run pre-publication checks
`publish`	Publish the software to the local userland IPS repo
REQUIRED_PACKAGES	Guess and generate build dependencies for a package, manual edit might be needed
env-check	Check environment for missing packages
env-prep	Installs missing build dependencies (requires elevated privileges)

For more details about writing Makefiles for userland, see oi-userland Makefile targets and variables

> ⚠ **Before adding new packages to oi-userland...**
>
> Before considering adding a new package to oi-userland, please check first whether someone else is working on the package by checking the issue tracker, mailing oi-dev@openindiana.org or asking on the IRC (#oi-dev at irc.freenode.net)
>
> - If you don't find anyone already working on a port, please register your effort by opening an issue.
> - If you wish to update an existing port, look at the log for the component `Makefile` ("`git log Makefile`") and make sure you either contact the person who last updated the `Makefile` or include them on notifications for the issue by ticking their name.
>
> This will ensure efforts aren't duplicated and help to ensure sanity and comity amongst project members.

Setting up your build environment

We **strongly** recommend building packages inside a fresh local zone set up exclusively for building. A good example howto on creating a build-zone can be found in the CrossBow instructions, see Building in zones.

Some suggestions about build environment (dataset layouts, etc.) can also be found in the illumos wiki: How to build illumos (note however that compilers required for the core OS may be different than those acceptable for userland software).

Coming from /dev branch

oi-userland requires that your system be updated with the latest software from the http://pkg.openindiana.org/hipster repository.

The procedure to do this is as follows:

```
sudo /usr/bin/pkg unset-publisher openindiana.org
sudo /usr/bin/pkg set-publisher -p http://pkg.openindiana.org/hipster
sudo /usr/bin/pkg install -v package/pkg
sudo /usr/bin/pkg update -v
```

The first line unsets the /dev repo. the second line add Hipster repository to your package publisher list and the third line allows packages from the Hipster repo to replace those installed from the openindiana.org repo. The last line updates your system with the newer packages.

Note: If you are doing this in a zone and encounter an error from pkg about being unable to clone the current boot environment, you will need to update the zone from the global zone by doing:

```
ZONE=yourzonename
ZONEROOT=`zonecfg -z $ZONE info zonepath | awk '{print $NF}'`
sudo zoneadm -z $ZONE halt
sudo zoneadm -z $ZONE ready
export PYTHONPATH=${ZONEROOT}/root/usr/lib/python2.7/vendor-packages
sudo ${ZONEROOT}/root/usr/bin/pkg -R ${ZONEROOT}/root unset-publisher openindiana.org
sudo ${ZONEROOT}/root/usr/bin/pkg -R ${ZONEROOT}/root set-publisher -p http://pkg.openindiana.org/hipster
sudo ${ZONEROOT}/root/usr/bin/pkg -R ${ZONEROOT}/root install -v package/pkg
sudo ${ZONEROOT}/root/usr/bin/pkg -R ${ZONEROOT}/root update -v
unset PYTHONPATH ZONE ZONEROOT
```

/hipster

Note that the /hipster repo is under continuous development and may contain breakage. Thus we (as mentioned) strongly recommend building inside a dedicated development zone.

Installing from Hipster ISOs

If the system was installed from the Hipster installation media, it is required just to update the system with the latest packages.

```
sudo /usr/bin/pkg update -v
```

Installing pre-requisites

First, lets install the required packages. build-essential contains everything that is needed to build oi-userland and illumos-gate itself (compilers, linkers, various runtimes..).

```
sudo /usr/bin/pkg install build-essential
```

Changes on Sun Studio compilers

Sun Studio compiles are not used for userland builds, so contributors are discouraged from using them.

Getting started with the build system

This section assumes that you are logged into the build zone if you set up the build environment in a zone, the directory were oi-userland is cloned can otherwise be anywhere you like.

Downloading oi-userland and preparing for first use

*See the illumos wiki instructions How to build illumos for **optional** suggestions about build environment (dedicated ZFS dataset layouts, etc.)*

Start by forking oi-userland repository on Github and then check out the repository (subdirectory `oi-userland` must not pre-exist):

```
cd ~
git clone https://github.com/mylogin/oi-userland
cd oi-userland
```

Now we will run the setup stage, which will prepare some tools, and create an IPS `pkg5` repository for first use under the `i386/build` directory:

```
cd $HOME/oi-userland
gmake setup
```

Assigning RBAC profile to the build user

Installing software requires privileges, so your build user must have at minimum the software installation profile:

```
$ profiles
Software Installation
ZFS File System Management
Console User
Suspend To RAM
Suspend To Disk
Brightness
CPU Power Management
Network Autoconf User
Desktop Removable Media User
Basic Solaris User
All
```

If it is not the case, add this profile to your build user:

```
pfexec su -
usermod -P'Software Installation' <username>
```

This is not necessary if your user has already the `'Primary Administrator'` profile.

Adding the local repository to your publisher list

```
pfexec /usr/bin/pkg set-publisher -p file://$HOME/oi-userland/i386/repo
pfexec /usr/bin/pkg set-publisher -P userland
pfexec /usr/bin/pkg set-publisher --non-sticky openindiana.org
```

You will now be able to `pkg install` the software you have built and published via `oi-userland`.

Optional: Running a local pkg server for installation on other zones/hosts

If you would like to use your oi-userland repository on other zones or hosts, you can run a pkg server:

```
/usr/lib/pkg.depotd -d $HOME/oi-userland/i386/repo -p 10000
```

On other hosts, you can then specify http://hostname:10000 instead of the file:// address above.

If you only intend to install and test packages locally, this is not necessary as on-disk repository access suffices.

*See the illumos wiki instructions How to build illumos for **optional** suggestions about running a pkg server instance as an SMF service.*

Ready to build and install your first package

Simply descend into the `oi-userland/components/SOFTWARE` directory (where SOFTWARE is the name of the bit of software you wish to build):

```
cd $HOME/oi-userland/components/SOFTWARE
gmake env-prep
gmake publish
pfexec pkg install SOFTWARE
```

To speed up the compilation you can pass an optional argument to `gmake` using the `COMPONENT_BUILD_ARGS` variable, for instance with

COMPONENT_BUILD_ARGS

```
COMPONENT_BUILD_ARGS=-j24
```

to use 24 jobs for builds.

Contributing changes back to oi-userland

Committing changes

When you think you are ready with changes, you need to commit them locally and push those changes back to your Github repository.

```
cd ~/oi-userland/components/SOFTWARE
git add Makefile *.p5m <etc.>
git commit .
```

Commit messages should be simple and describe what you did, e.g "added XYZ", "fixed XYZ", "enabled/disabled XYZ" or "<bug id> <bug summary>", where <bug id> is issue number from issue tracker and <bug summary> is the issue name.

Asking for change integration

This is as simple as creating a Pull Request into the main oi-userland repository and asking developers to review your changeset. We should beware of possibly breaking packages as it adds additional work and can be unpleasant for other contributors (imagine a situation where gcc, perl or anything else needed for building packages is broken).

Changes can be reverted quite easily, but once the package is built and published additional steps are needed. So try taking Per-package testing and asking for wider testing into consideration.

If you contribute a package, which is known to work, but its functionality might be broken because of some issues, consider disabling it till the issue is not removed.

Checking Jenkins instance

Once the changes are merged into the main oi-userland repository, Jenkins instance will pick up those bits and build them. If the build was successful, the built packages will be pushed into http://pkg.openindiana.org/hipster repository. If the package build was unsucessful, check build logs and please try to come up with a solution and fix the problem, so you can have package published into the repository.