Contributing to FreeBSD

<!-- vale FreeBSD.Pronouns = NO -->

Abstract

This article describes the different ways in which an individual or organization may contribute to the FreeBSD Project.

Table of Contents

1. What Is Needed 1
2. How to Contribute 4
3. Contributing to ports 7
4. Getting Started in Other Areas 15

So you want to contribute to FreeBSD? That is great! FreeBSD *relies* on the contributions of its user base to survive. Your contributions are not only appreciated, they are vital to FreeBSD's continued growth.

A large and growing number of international contributors, of greatly varying ages and areas of technical expertise, develop FreeBSD. There is always more work to be done than there are people available to do it, and more help is always appreciated.

As a volunteer, what you do is limited only by what you want to do. However, we do ask that you are aware of what other members of the FreeBSD community will expect of you. You may want to take this into account before deciding to volunteer.

The FreeBSD project is responsible for an entire operating system environment, rather than just a kernel or a few scattered utilities. As such, our TODO lists span a very wide range of tasks: from documentation, beta testing and presentation, to the system installer and highly specialized types of kernel development. People of any skill level, in almost any area, can almost certainly help the project.

Commercial entities engaged in FreeBSD-related enterprises are also encouraged to contact us. Do you need a special extension to make your product work? You will find us receptive to your requests, given that they are not too outlandish. Are you working on a value-added product? Please let us know! We may be able to work cooperatively on some aspect of it. The free software world is challenging many existing assumptions about how software is developed, sold, and maintained, and we urge you to at least give it a second look.

1. What Is Needed

The following list of tasks and sub-projects represents something of an amalgam of various TODO

lists and user requests.

1.1. Ongoing Non-Programmer Tasks

Many people who are involved in FreeBSD are not programmers. The Project includes documentation writers, Web designers, and support people. All that these people need to contribute is an investment of time and a willingness to learn.

1. Read through the FAQ and Handbook periodically. If anything is poorly explained, ambiguous, out of date or incorrect, let us know. Even better, send us a fix (AsciiDoc is not difficult to learn, but there is no objection to plain text submissions).
2. Help translate FreeBSD documentation into your native language. If documentation already exists for your language, you can help translate additional documents or verify that the translations are up-to-date and correct. First take a look at the Translations FAQ in the FreeBSD Documentation Project Primer. You are not committing yourself to translating every single FreeBSD document by doing this - as a volunteer, you can do as much or as little translation as you desire. Once someone begins translating, others almost always join the effort. If you only have the time or energy to translate one part of the documentation, please translate the installation instructions.
3. Read the FreeBSD general questions mailing list occasionally (or even regularly). It can be very satisfying to share your expertise and help people solve their problems; sometimes you may even learn something new yourself! These forums can also be a source of ideas for things to improve upon.

1.2. Ongoing Programmer Tasks

Most of the tasks listed here may require a considerable investment of time, an in-depth knowledge of the FreeBSD kernel, or both. However, there are also many useful tasks which are suitable for "weekend hackers".

1. If you run FreeBSD-CURRENT and have a good Internet connection, there is a machine `current.FreeBSD.org` which builds a full release once a day-every now and again, try to install the latest release from it and report any failures in the process.
2. Learn about changes or problems that have already been proposed. There may be a problem you can comment constructively on or with patches you can test. Or you could even try to fix one of the problems yourself.
3. One way is to read the FreeBSD problem reports mailing list (for issues submitted through Bugzilla).
4. If you know of any bug fixes which have been successfully applied to -CURRENT but have not been merged into -STABLE after a decent interval (normally a couple of weeks), send the committer a polite reminder.
5. Move contributed software to src/contrib in the source tree.
6. Make sure code in src/contrib is up to date.
7. Build the source tree (or just part of it) with extra warnings enabled and clean up the warnings.

A list of build warnings can also be found from our CI by selecting a build and checking "LLVM/Clang Warnings".

8. Fix warnings for ports which do deprecated things like using `gets()` or including malloc.h.
9. If you have contributed any ports and you had to make FreeBSD-specific changes, send your patches back to the original authors (this will make your life easier when they bring out the next version).
10. Get copies of formal standards like POSIX®. Compare FreeBSD's behavior to that required by the standard. If the behavior differs, particularly in subtle or obscure corners of the specification, send in a PR about it. If you are able, figure out how to fix it and include a patch in the PR. If you think the standard is wrong, ask the standards body to consider the question.
11. Suggest further tasks for this list!

1.3. Work through the Bugzilla Problem Report Database

The FreeBSD Problem Report list shows all the current active problem reports and requests for enhancement that have been submitted by FreeBSD users via Bugzilla. This database includes both programmer and non-programmer tasks. Look through the open PRs, and see if anything there takes your interest. Some of these might be very simple tasks that just need an extra pair of eyes to look over them and confirm that the fix is a good one. Others might be much more complex, or might not even have a fix included at all.

Start with the PRs that have not been assigned to anyone else. If a PR is assigned to someone else, but it looks like something you can handle, email the person it is assigned to and ask if you can work on it-they might already have a patch ready to be tested, or further ideas that you can discuss with them.

1.4. Ongoing Ports Tasks

The Ports Collection is a perpetual work in progress. We want to provide our users with an easy to use, up to date, high quality repository of third party software. We need people to donate some of their time and effort to help us achieve this goal.

Anyone can get involved, and there are lots of different ways to do so. Contributing to ports is an excellent way to help "give back" something to the project. Whether you are looking for an ongoing role, or a fun challenge for a rainy day, we would love to have your help!

There are a number of easy ways you can contribute to keeping the ports tree up to date and in good working order:

- Find some cool or useful software and create a port for it.
- There are a large number of ports that have no maintainer. Become a maintainer and Adopting an unmaintained port.
- If you have created or adopted a port, be aware of The challenge for port maintainers.
- When you are looking for a quick challenge you could Finding and fixing a broken port.

1.5. Pick one of the items from the Ideas page

The FreeBSD list of projects and ideas for volunteers is also available for people willing to contribute to the FreeBSD project. The list is being regularly updated and contains items for both programmers and non-programmers with information about each project.

2. How to Contribute

Contributions to the system generally fall into one or more of the following 5 categories:

2.1. Bug Reports and General Commentary

An idea or suggestion of *general* technical interest should be mailed to the FreeBSD technical discussions mailing list. Likewise, people with an interest in such things (and a tolerance for a *high* volume of mail!) may subscribe to the FreeBSD technical discussions mailing list. See The FreeBSD Handbook for more information about this and other mailing lists.

If you are submitting a simple patch to the src repo, please consider submitting it to the project's GitHub mirror as a pull request. Suitable submissions should:

- It is ready or nearly ready to be committed. A committer should be able to land this patch with less than 10 minutes of additional work.
- It passes all the GitHub CI jobs.
- You can respond to feedback quickly.
- It touches fewer than about 10 files and the changes are less than about 200 lines. Changes larger than this may be OK, or you may be asked to submit multiple pull requests of a more manageable size.
- Each logical change is a separate commit within the pull request. Commit messages for each change should follow commit log guide.
- All commits have your name and valid email address as you'd like to see them in the FreeBSD repository as the author. Fake github.com addresses cannot be used.
- The scope of the pull request should not change during review. If the review suggests changes that expand the scope, please create an independent pull request.
- Fixup commits should be squashed with the commit they are fixing. Each commit in your branch should be suitable for FreeBSD's repository.
- Commits should include one or more `Signed-off-by:` lines with full name and email address certifying Developer Certificate of Origin.

When updating pull request, please rebase with a forced push rather than a merge commit. More complex changes may be submitted as pull requests, but they may be closed if they are too large, too unwieldy, become inactive, need further discussion in the community, or need extensive revision. Please avoid creating large, wide-ranging cleanup patches: they are too large and lack the focus needed for a good review. Misdirected patches may be redirected to a more appropriate forum for the patch to be resolved.

Pull requests submitted to the ports repository may or may not see action, based on the whims of developers. For now, you will have a better experience if you follow the ports submission process Contributing to ports.

The docs team also accepts pull requests via GitHub, but has not established any policy for them yet.

If you find a bug or are submitting a specific change, please report it using the Bugzilla bug submission form. Try to fill-in each field of the bug report. Unless they exceed 65KB, include any patches directly in the report. If the patch is suitable to be applied to the source tree put `[PATCH]` in the synopsis of the report. When including patches, *do not* use cut-and-paste because cut-and-paste turns tabs into spaces and makes them unusable. When patches are a lot larger than 20KB, consider compressing them (for example with gzip(1) or bzip2(1)) prior to uploading them.

After filing a report, you should receive confirmation along with a tracking number. Keep this tracking number so that you can update us with details about the problem.

See also this article on how to write good problem reports.

2.2. Changes to the Documentation

Changes to the documentation are overseen by the FreeBSD documentation project mailing list. Please look at the FreeBSD Documentation Project Primer for complete instructions. Send submissions and changes (even small ones are welcome!) using the same method as any other bug report.

2.3. Changes to Existing Source Code

An addition or change to the existing source code is a somewhat trickier affair and depends a lot on how far out of date you are with the current state of FreeBSD development. There is a special on-going release of FreeBSD known as "FreeBSD-CURRENT" which is made available in a variety of ways for the convenience of developers working actively on the system. See The FreeBSD Handbook for more information about getting and using FreeBSD-CURRENT.

Working from older sources unfortunately means that your changes may sometimes be too obsolete or too divergent for easy re-integration into FreeBSD. Chances of this can be minimized somewhat by subscribing to the FreeBSD announcements mailing list and the FreeBSD-CURRENT mailing list lists, where discussions on the current state of the system take place.

Assuming that you can manage to secure fairly up-to-date sources to base your changes on, the next step is to produce a set of diffs to send to the FreeBSD maintainers. This is done with the diff(1) command.

The preferred diff(1) format for submitting patches is the unified output format generated by `diff -u`.

```
% diff -u oldfile newfile
```

or

```
% diff -u -r -N olddir newdir
```

would generate a set of unified diffs for the given source file or directory hierarchy.

See diff(1) for more information.

Once you have a set of diffs (which you may test with the patch(1) command), you may submit them for inclusion with FreeBSD as a Bugzilla problem report. *Do not* just send the diffs to the FreeBSD technical discussions mailing list or they will get lost! We greatly appreciate your submission (this is a volunteer project!); because we are busy, we may not be able to address it immediately, but it will remain in the PR database until we do. Indicate your submission by including `[PATCH]` in the synopsis of the report.

If you feel it appropriate (for example you have added, deleted, or renamed files), bundle your changes into a `tar` file.

If your change is of a potentially sensitive nature, such as if you are unsure of copyright issues governing its further distribution then you should send it to core@FreeBSD.org directly rather than submitting as a bug report. The core@FreeBSD.org reaches a much smaller group of people who do much of the day-to-day work on FreeBSD. Note that this group is also *very busy* and so you should only send mail to them where it is truly necessary.

Please refer to intro(9) and style(9) for some information on coding style. We would appreciate it if you were at least aware of this information before submitting code.

2.4. New Code or Major Value-Added Packages

In the case of a significant contribution of a large body work, or the addition of an important new feature to FreeBSD, it becomes almost always necessary to either send changes as tar files or upload them to a web or FTP site for other people to access. If you do not have access to a web or FTP site, ask on an appropriate FreeBSD mailing list for someone to host the changes for you.

When working with large amounts of code, the touchy subject of copyrights also invariably comes up. FreeBSD prefers free software licenses such as BSD or ISC. Copyleft licenses such as GPLv2 are sometimes permitted. The complete listing can be found on the core team licensing policy page.

2.5. Money or Hardware

We are always very happy to accept donations to further the cause of the FreeBSD Project and, in a volunteer effort like ours, a little can go a long way! Donations of hardware are also very important to expanding our list of supported peripherals since we generally lack the funds to buy such items ourselves.

2.5.1. Donating Funds

The FreeBSD Foundation is a non-profit, tax-exempt foundation established to further the goals of the FreeBSD Project. As a 501(c)3 entity, the Foundation is generally exempt from US federal income tax as well as Colorado State income tax. Donations to a tax-exempt entity are often deductible from taxable federal income.

Donations may be sent in check form to:

> The FreeBSD Foundation
> 3980 Broadway Street
> STE #103-107
> Boulder CO 80304
> USA

The FreeBSD Foundation is also able to accept online donations through various payment options.

More information about the FreeBSD Foundation can be found in The FreeBSD Foundation — an Introduction. To contact the Foundation by email, write to info@FreeBSDFoundation.org.

2.5.2. Donating Hardware

The FreeBSD Project happily accepts donations of hardware that it can find good use for. If you are interested in donating hardware, please contact the Donations Liaison Office.

3. Contributing to ports

3.1. Adopting an unmaintained port

3.1.1. Choosing an unmaintained port

Taking over maintainership of ports that are unmaintained is a great way to get involved. Unmaintained ports are only updated and fixed when somebody volunteers to work on them. There are a large number of unmaintained ports. It is a good idea to start with adopting a port that you use regularly.

Unmaintained ports have their `MAINTAINER` set to `ports@FreeBSD.org`. Many unmaintained ports can have pending updates, this can be seen at the FreeBSD Ports distfile scanner.

On PortsFallout can be seen a list of unmaintained ports with errors.

Some ports affect a large number of others due to dependencies and secondary port relationships. Generally, we want people to have some experience before they maintain such ports.

You can find out whether or not a port has dependencies or secondary ports by looking at a primary index of ports called INDEX. (The name of the file varies by release of FreeBSD; for instance, INDEX-13.) Some ports have conditional dependencies that are not included in a default

INDEX build. We expect you to be able to recognize such ports by looking through other ports' Makefile's.

3.1.2. How to adopt the port

First make sure you understand your The challenge for port maintainers. Also read the Porter's Handbook. *Please do not commit yourself to more than you feel you can comfortably handle.*

You may request maintainership of any unmaintained port as soon as you wish. Simply set `MAINTAINER` to your own email address and send a Problem Report with the change. If the port has build errors or needs updating, you may wish to include any other changes in the same PR. This will help because many committers are less willing to assign maintainership to someone who does not have a known track record with FreeBSD. Submitting PRs that fix build errors or update ports are the best ways to establish one.

File your PR with Product `Ports & Packages`. A committer will examine your PR, commit the changes, and finally close the PR. Sometimes this process can take a little while (committers are volunteers, too :).

3.2. The challenge for port maintainers

This section will give you an idea of why ports need to be maintained and outline the responsibilities of a port maintainer.

3.2.1. Why ports require maintenance

Creating a port is a once-off task. Ensuring that a port is up to date and continues to build and run requires an ongoing maintenance effort. Maintainers are the people who dedicate some of their time to meeting these goals.

The foremost reason ports need maintenance is to bring the latest and greatest in third party software to the FreeBSD community. An additional challenge is to keep individual ports working within the Ports Collection framework as it evolves.

As a maintainer, you will need to manage the following challenges:

- **New software versions and updates.** New versions and updates of existing ported software become available all the time, and these need to be incorporated into the Ports Collection to provide up-to-date software.
- **Changes to dependencies.** If significant changes are made to the dependencies of your port, it may need to be updated so that it will continue to work correctly.
- **Changes affecting dependent ports.** If other ports depend on a port that you maintain, changes to your port may require coordination with other maintainers.
- **Interaction with other users, maintainers and developers.** Part of being a maintainer is taking on a support role. You are not expected to provide general support (but we welcome it if you choose to do so). What you should provide is a point of coordination for FreeBSD-specific issues regarding your ports.

- **Bug hunting.** A port may be affected by bugs which are specific to FreeBSD. You will need to investigate, find, and fix these bugs when they are reported. Thoroughly testing a port to identify problems before they make their way into the Ports Collection is even better.
- **Changes to ports infrastructure and policy.** Occasionally the systems that are used to build ports and packages are updated or a new recommendation affecting the infrastructure is made. You should be aware of these changes in case your ports are affected and require updating.
- **Changes to the base system.** FreeBSD is under constant development. Changes to software, libraries, the kernel or even policy changes can cause flow-on change requirements to ports.

3.2.2. Maintainer responsibilities

3.2.2.1. Keep your ports up to date

This section outlines the process to follow to keep your ports up to date.

This is an overview. More information about upgrading a port is available in the Porter's Handbook.

1. Watch for updates

   Monitor the upstream vendor for new versions, updates and security fixes for the software. Announcement mailing lists or news web pages are useful for doing this. Sometimes users will contact you and ask when your port will be updated. If you are busy with other things or for any reason just cannot update it at the moment, ask if they will help you by submitting an update.

   You may also receive automated email from the `FreeBSD Ports Version Check` informing you that a newer version of your port's distfile is available. More information about that system (including how to stop future emails) will be provided in the message.

2. Incorporate changes

   When they become available, incorporate the changes into the port. You need to be able to generate a patch between the original port and your updated port.

3. Review and test

   Thoroughly review and test your changes:

   - Build, install and test your port on as many platforms and architectures as you can. It is common for a port to work on one branch or platform and fail on another.
   - Make sure your port's dependencies are complete. The recommended way of doing this is by installing your own ports tinderbox. See Resources for ports maintainers and contributors for more information.
   - Check that the packing list is up to date. This involves adding in any new files and directories and removing unused entries.
   - Verify your port using portlint(1) as a guide. See Resources for ports maintainers and

contributors for important information about using portlint.

- Consider whether changes to your port might cause any other ports to break. If this is the case, coordinate the changes with the maintainers of those ports. This is especially important if your update changes the shared library version; in this case, at the very least, the dependent ports will need to get a `PORTREVISION` bump so that they will automatically be upgraded by automated tools such as ports-mgmt/poudriere.

4. Submit changes

   Send your update by submitting a Problem Report with an explanation of the changes and a patch containing the differences between the original port and the updated one. Please refer to Writing FreeBSD Problem Reports for information on how to write a really good PR.

   Please do not submit a shar(1) archive of the entire port; instead, use git-format-patch(1) or diff(1) `-ruN`. In this way, committers can much more easily see exactly what changes are being made. The Porter's Handbook section on Upgrading has more information.

5. Wait

   At some stage a committer will deal with your PR. It may take minutes, or it may take one or two weeks - so please be patient. If it takes any longer, please seek for help on mailing lists (FreeBSD ports mailing list), IRC: #bsdports on EFNet or #freebsd-ports on Libera for example.

6. Give feedback

   If a committer finds a problem with your changes, they will most likely refer it back to you. A prompt response will help get your PR committed faster, and is better for maintaining a thread of conversation when trying to resolve any problems.

7. And Finally

   Your changes will be committed and your port will have been updated. The PR will then be closed by the committer. That is it!

3.2.2.2. Ensure your ports continue to build correctly

This section is about discovering and fixing problems that stop your ports from building correctly.

FreeBSD only guarantees that the Ports Collection works on the `-STABLE` branches. In theory, you should be able to get by with running the latest release of each stable branch (since the ABIs are not supposed to change) but if you can run the branch, that is even better.

Since the majority of FreeBSD installations run on PC-compatible machines (what is termed the `i386` architecture), we expect you to keep the port working on that architecture. We prefer that ports also work on the `amd64` architecture running native. It is completely fair to ask for help if you do not have one of these machines.

The usual failure modes for non-x86 machines are that the original programmers assumed that, for instance, pointers are int-s, or that a relatively lax older gcc compiler was being used. More and more, application authors are reworking their code to remove these assumptions - but if the author is not actively maintaining their code, you may need to do this yourself.

These are the tasks you need to perform to ensure your port is able to be built:

1. Watch for build failures

   Check your mail for mail from pkg-fallout@FreeBSD.org and the distfiles scanner to see if any of the port which are failing to build are out of date.

2. Collect information

   Once you are aware of a problem, collect information to help you fix it. Build errors reported by pkg-fallout are accompanied by logs which will show you where the build failed. If the failure was reported to you by a user, ask them to send you information which may help in diagnosing the problem, such as:

   - Build logs
   - The commands and options used to build the port (including options set in /etc/make.conf)
   - A list of packages installed on their system as shown by pkg-info(8)
   - The version of FreeBSD they are running as shown by uname(1) -a
   - When their ports collection was last updated
   - When their ports tree and INDEX was last updated

3. Investigate and find a solution

   Unfortunately there is no straightforward process to follow to do this. Remember, though: if you are stuck, ask for help! The FreeBSD ports mailing list is a good place to start, and the upstream developers are often very helpful.

4. Submit changes

   Just as with updating a port, you should now incorporate changes, review and test, submit your changes in a PR, and provide feedback if required.

5. Send patches to upstream authors

   In some cases, you will have to make patches to the port to make it run on FreeBSD. Some (but not all) upstream authors will accept such patches back into their code for the next release. If so, this may even help their users on other BSD-based systems as well and perhaps save duplicated effort. Please consider sending any applicable patches to the authors as a courtesy.

3.2.2.3. Investigate bug reports and Bugzilla Problem Reports related to your port

This section is about discovering and fixing bugs.

FreeBSD-specific bugs are generally caused by assumptions about the build and runtime environments that do not apply to FreeBSD. You are less likely to encounter a problem of this type, but it can be more subtle and difficult to diagnose.

These are the tasks you need to perform to ensure your port continues to work as intended:

1. Respond to bug reports

   Bugs may be reported to you through email via the Bugzilla Problem Report database. Bugs may also be reported directly to you by users.

   You should respond to PRs and other reports within 14 days, but please try not to take that long. Try to respond as soon as possible, even if it is just to say you need some more time before you can work on the PR.

   If you have not responded after 14 days, any committer may commit from a PR that you have not responded to via a `maintainer-timeout`.

2. Collect information

   If the person reporting the bug has not also provided a fix, you need to collect the information that will allow you to generate one.

   If the bug is reproducible, you can collect most of the required information yourself. If not, ask the person who reported the bug to collect the information for you, such as:

   - A detailed description of their actions, expected program behavior and actual behavior
   - Copies of input data used to trigger the bug
   - Information about their build and execution environment - for example, a list of installed packages and the output of env(1)
   - Core dumps
   - Stack traces

3. Eliminate incorrect reports

   Some bug reports may be incorrect. For example, the user may have simply misused the program; or their installed packages may be out of date and require updating. Sometimes a reported bug is not specific to FreeBSD. In this case report the bug to the upstream developers. If the bug is within your capabilities to fix, you can also patch the port so that the fix is applied before the next upstream release.

4. Find a solution

   As with build errors, you will need to sort out a fix to the problem. Again, remember to ask if you are stuck!

5. Submit or approve changes

   Just as with updating a port, you should now incorporate changes, review and test, and submit your changes in a Bugzilla PR (or send a follow-up if a PR already exists for the problem). If another user has submitted changes in the PR, you can also send a follow-up saying whether or not you approve the changes.

3.2.2.4. Providing support

Part of being a maintainer is providing support - not for the software in general - but for the port and any FreeBSD-specific quirks and problems. Users may contact you with questions, suggestions, problems and patches. Most of the time their correspondence will be specific to FreeBSD.

Occasionally you may have to invoke your skills in diplomacy, and kindly point users seeking general support to the appropriate resources. Less frequently you will encounter a person asking why the `RPMS` are not up to date or how can they get the software to run under Foo Linux. Take the opportunity to tell them that your port is up to date (if it is, of course!), and suggest that they try FreeBSD.

Sometimes users and developers will decide that you are a busy person whose time is valuable and do some of the work for you. For example, they might:

- submit a Bugzilla PR or send you patches to update your port,
- investigate and perhaps provide a fix to a PR, or
- otherwise submit changes to your port.

In these cases your main obligation is to respond in a timely manner. Again, the timeout for non-responsive maintainers is 14 days. After this period changes may be committed unapproved. They have taken the trouble to do this for you; so please try to at least respond promptly. Then review, approve, modify or discuss their changes with them as soon as possible.

If you can make them feel that their contribution is appreciated (and it should be) you will have a better chance persuading them to do more things for you in the future :-).

3.3. Finding and fixing a broken port

There are some really good places to find a port that needs some attention.

You can use the web interface to the Problem Report database to search through and view unresolved Bugzilla PRs. The majority of ports PRs are updates, but with a little searching and skimming over synopses you should be able to find something interesting to work on.

PortsFallout shows port issues gathered from the FreeBSD package building.

It is OK to send changes for a maintained port as well, but remember to ask the maintainer in case they are already working on the problem.

Once you have found a bug or problem, collect information, investigate and fix! If there is an

existing PR, follow up to that. Otherwise create a new PR. Your changes will be reviewed and, if everything checks out, committed.

3.4. When to call it quits

As your interests and commitments change, you may find that you no longer have time to continue some (or all) of your ports contributions. That is fine! Please let us know if you are no longer using a port or have otherwise lost time or interest in being a maintainer. In this way we can go ahead and allow other people to try to work on existing problems with the port without waiting for your response. Remember, FreeBSD is a volunteer project, so if maintaining a port is no fun any more, it is probably time to let someone else do it!

In any case, the Ports Management Team (`portmgr`) reserves the right to reset your maintainership if you have not actively maintained your port in some time. (Currently, this is set to 3 months.) By this, we mean that there are unresolved problems or pending updates that have not been worked on during that time.

3.5. Resources for ports maintainers and contributors

The Porter's Handbook is your hitchhiker's guide to the ports system. Keep it handy!

Writing FreeBSD Problem Reports describes how to best formulate and submit a Problem Report. Following this article will greatly assist us in reducing the time needed to handle your PRs.

The Bugzilla Problem Report database.

The FreeBSD Ports distfile scanner (portscout) can show you ports for which the distfiles are not fetchable. You can check on your own ports or use it to find ports that need their `MASTER_SITES` updated.

ports-mgmt/poudriere is the most thorough way to test a port through the entire cycle of installation, packaging, and deinstallation. Documentation is located at the poudriere GitHub repository

portlint(1) is an application which can be used to verify that your port conforms to many important stylistic and functional guidelines. portlint is a simple heuristic application, so you should use it *only as a guide*. If portlint suggests changes which seem unreasonable, consult the Porter's Handbook or ask for advice.

The FreeBSD ports mailing list is for general ports-related discussion. It is a good place to ask for help. You can subscribe, or read and search the list archives. Reading the archives of the FreeBSD ports bugs mailing list and the SVN commit messages for the ports tree for head/ may also be of interest.

PortsFallout is a place to help in searching for the FreeBSD package-fallout archive.

4. Getting Started in Other Areas

Looking for something interesting to get started that is not mentioned elsewhere in this article? The FreeBSD Project has several Wiki pages containing areas within which new contributors can get ideas on how to get started.

The Junior Jobs page has a list of projects that might be of interest to people just getting started in FreeBSD, and want to work on interesting things to get their feet wet.

The Ideas Page contains various "nice to have" or "interesting" things to work on in the Project.