Contributing to the FreeBSD Ports Collection

BY MATEUSZ PIOTROWSKI

Why would you use ports?

Many FreeBSD releases ago, the FreeBSD Ports Collection was the primary way to install third-party software. Users would build software they needed from ports. In theory, there were some binary packages available, but the overall support for them wasn't that great. Package management tools were cumbersome. Package repositories contained outdated packages. Building software from ports was a necessity.

The situation began to change around the time when the new pkg(8) package management tool emerged. Nowadays, FreeBSD packages repositories are one of the largest and most up-to-date in the open-source world (see the graphs on repology.org). Most FreeBSD users use binary packages instead of compiling ports themselves and it has been like that since I started using using FreeBSD, which was around version 10.3.

Although binary packages are great, people use FreeBSD Ports Collection directly all the time. Both FreeBSD maintainers and FreeBSD users use it all the time. Why would FreeBSD users use it? Because ports make it really easy to tailor binary packages to very specific needs. Would you like to rebuild the Nginx package with a custom patch? No problem. Would you like to add an unusual backend to your collectd daemon? Easy. Would you like to get a debug build of Python? No big deal.

In this article I would like to give you some insights about contributing to the FreeBSD Ports Collection. How to get started? Why to get started? What does it take to submit a patch? Keep reading if you want to know the answers to those questions.

Hello, my name is Mateusz, I would like to contribute to FreeBSD

This is a message I see all the time, whether on the mailing lists, Discord or IRC channels.

Typically, it gets answered with a bunch of links to bug trackers and wiki pages with project ideas. You would expect that long-time contributors love to introduce newcomers to their projects. This is true (and that is a sign of a healthy open-source community). What is also true is that long-time contributors are reluctant to reply to such messages. Why is that so? Well, they know that there is a high chance that they won't hear back from that newcomer ever again. Yes, you heard me right. Basically, this is not how you get started contributing to open source. Don't get me wrong. I used to send similar messages all the time in the past myself. Why? Because it felt like the right way to start! I had the time and motivation, I just needed the community to give me an interesting project to focus on. Isn't that simple?

It turns out that it works slightly differently. Becoming absorbed in a project listed on some project ideas page is borderline impossible. Project ideas land on wiki pages because no one had the motivation to spend the the necessary amount of time to do the work. How could it possibly be picked up by a newcomer if it doesn't spark joy even among seasoned contributors? I am not sure. Some project ideas just have to wait for their champion.

How to get started then? The truth is that you need to find the area you are passionate about on your own. Here is what you can do. Start using FreeBSD regularly. Explore the system and pay attention to what annoys you. Ask yourself such questions as:

1. How do I get my laptop to suspend automatically if the battery is running low?
2. Could the GPU setup documentation be more straightforward?
3. How cool would be to have a proper icon for Xpdf in application launchers and menus?

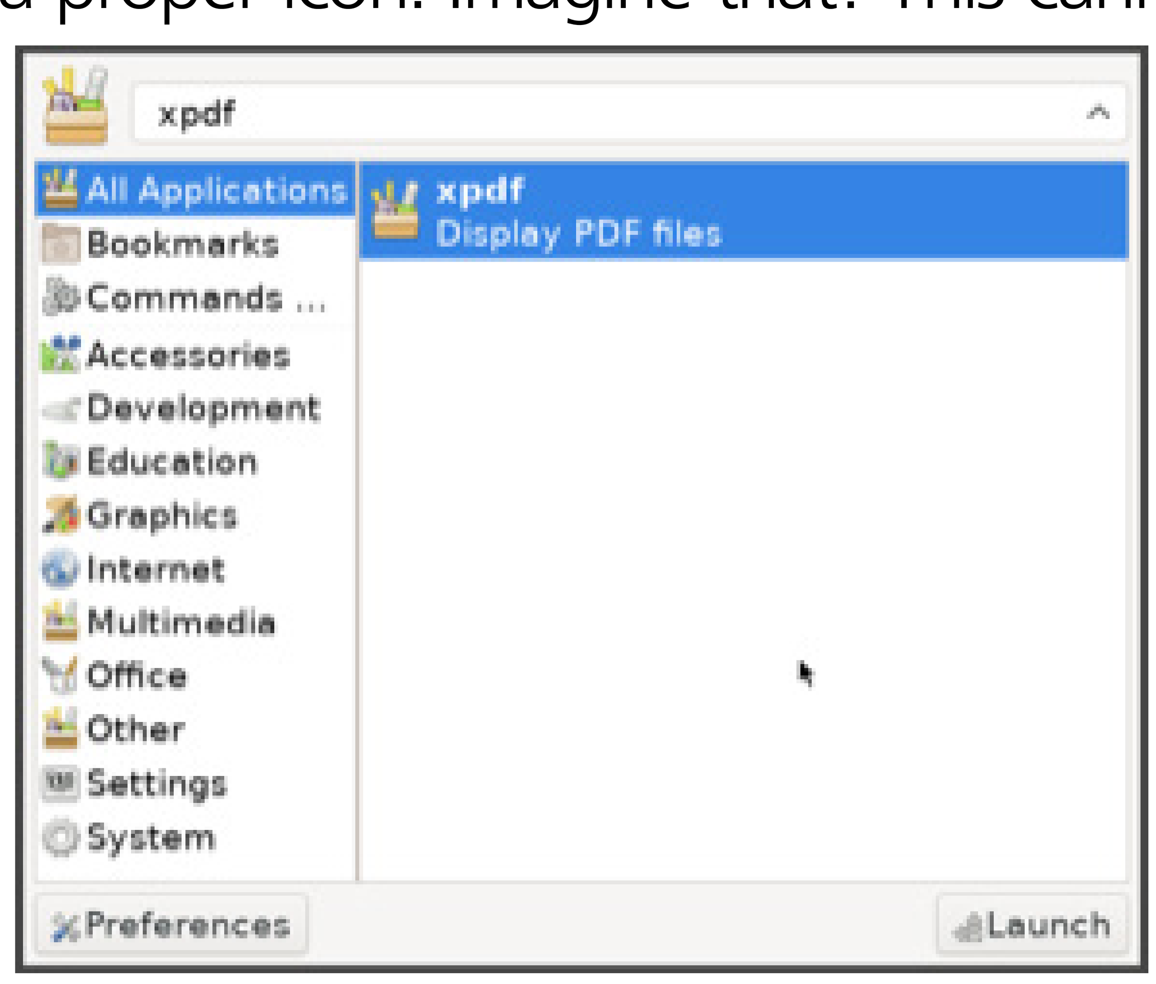
The greatest motivation to contribute comes from an itch to scratch. A problem so annoying that you decide to fix it on your own. A problem so interesting that resisting the urge to to fix it straightaway is futile. A problem so common that solving it surely will give you all the street cred at the next conference. In other words, the easiest way to start contributing is to work on something you need. Of course, you are going to get stuck at some point. What is different this time is that you have a specific problem at hand. These problems attract a good deal of attention in the community. Remember those "unreachable" seasoned contributors? Trust me, they are going to answer your questions much more often now as you seem to be quite motivated to solve some interesting problems there. Because it's much more fun helping somebody out with a problem rather than helping somebody out with finding a problem.

Scratching an Itch (Missing Xpdf Icons Edition)

This part of the article describes a workflow of developing a patch for the FreeBSD ports. When I was starting hacking on FreeBSD ports, I often wondered how others develop their patches. (Now that I think about, I am still fascinated by the efficiency of certain ports committers. Perhaps more than ever.) For some reason exact development workflows are infrequently described in the official FreeBSD documentation. Without further ado, let's see what it takes to cook up a ports patch.

First Encounter

It is a sunny Saturady evening, the snow is melting. You are computing important things on your FreeBSD desktop. You are going through your ever-growing todo list. One item at a time. The next task requires a PDF reader. No problem. Let's use our venerable Xpdf for that. So you fire up Xfce Application Finder and search for Xpdf. Everything is going smoothly. And then you see it. The Xpdf entry does not have a proper icon. Imagine that! This cannot be. We need to fix this.



Before we start hacking on the ports tree, let's understand why the Xpdf icon is missing. Xfce Application Finder generates the list of entries based on the desktop files located in `/usr/local/share/applications.`

The one called `/usr/local/share/applications/xpdf.desktop` describes the Xpdf entry. Let's see if there is something related to icons in that file.

```
$ grep -i icon /usr/local/share/applications/xpdf.desktop
Icon=xpdf
```

The icon's name is xpdf. Let's see if we can find such an icon in `/usr/local/share/icons`.

```
$ find /usr/local/share/icons -name '*xpdf*'
```

The output of our find(1) one-liner is empty. The Xpdf icon is not installed. At this point, we probably need to take a look at the ports tree.

Developing a Patch

The first thing we need is a copy of the FreeBSD ports tree. You can read up on the details in the FreeBSD Handbook (https://docs.freebsd.org/en/books/handbook/ports/#ports-using). Ultimately, the following command is all we need:

```
$ git clone https://git.FreeBSD.org/ports.git ~/ports
```

Now let's examine the Xpdf port. How do we find it among all the ports? There are a couple of different ways to do it.

The easiest way is to ask pkg(8) about the origin of the package.

```
$ pkg search -o xpdf
japanese/xpdf                   Japanese font support for xpdf
graphics/xpdf                   Display PDF files and convert them to other formats
graphics/xpdf3                  Display PDF files and convert them to other formats
graphics/xpdf4                  Display PDF files and convert them to other formats
print/xpdfopen                  Command line utility for PDF viewers
```

pkg-search(8) searches the package repository catalogue looking for package names matching "xpdf". The `-o` option tells pkg-search(8) to show the origin of the package in the output. The origin is the official term for the directory name of a port within the ports tree. This is exactly what we are looking for.

Tip: Sometimes I don't know the name of the package that installed a file I'd like to fix. In those cases I use pkg-which(8):

```
$ pkg which /usr/local/share/applications/xpdf.desktop
/usr/local/share/applications/xpdf.desktop was installed by package xpdf-4.03,1
```

OK, so we learnt that the origin of the Xpdf package is `graphics/xpdf`. Let's see what is inside the port's directory:

```
$ cd ~/ports/graphics/xpdf
$ ls
Makefile
```

Aha! For those of you, who are new to ports: what we see here does not look like a typical port. Usually, you expect to find other files like `distinfo` containing checksums of source code archives, `pkg-descr` containing a longer description of the port, and `pkg-plist` listing all the files this port installs. Let's see what's inside the `Makefile`:

```
$ cat -n Makefile
     1  VERSIONS=                3 4
     2  XPDF_VERSION?=           4
     3
     4  MASTERDIR=               ${.CURDIR}/../xpdf${XPDF_VERSION}
     5
     6  .include "${MASTERDIR}/Makefile"
```

See the `MASTERDIR` variable on line 4? It means that `graphics/xpdf` is a master port. When this port is build, it's actually `graphics/xpdf4` driving the whole process. (BTW, Xpdf version 4 is apparently the default, judging by line 2). Down the rabbit hole!

```
$ cd ~/ports/graphics/xpdf4
$ ls
distinfo    files       Makefile    pkg-descr   pkg-message pkg-plist
```

Just as expected. High time we grabbed a copy of the source code. We do it with the `extract` target defined by the FreeBSD Ports framework.

```
make extract
```

All the source code and build artifacts live in directory `./work` in the port's directory. E.g., the source code of Xpdf gets extracted into `work/xpdf-4.03`.

Tip: It is a good idea to run the `patch` target as well. The reason is that we want all the local FreeBSD ports patches applied to the freshly extracted, unmodified source code.

```
make patch
```

Alright, we've got all the basics covered. Let's start code spelunking. Are there any icons in the Xpdf sources?

```
$ find work/ -name *icon*
work/xpdf-4.03/xpdf-qt/indicator-icon-err5.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err2.svg
work/xpdf-4.03/xpdf-qt/indicator-icon1.svg
work/xpdf-4.03/xpdf-qt/indicator-icon6.svg
work/xpdf-4.03/xpdf-qt/icons.qrc
work/xpdf-4.03/xpdf-qt/xpdf-icon.svg
work/xpdf-4.03/xpdf-qt/indicator-icon7.svg
work/xpdf-4.03/xpdf-qt/indicator-icon0.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err3.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err4.svg
work/xpdf-4.03/xpdf-qt/indicator-icon3.svg
```

```
work/xpdf-4.03/xpdf-qt/indicator-icon4.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err7.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err0.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err1.svg
work/xpdf-4.03/xpdf-qt/indicator-icon-err6.svg
work/xpdf-4.03/xpdf-qt/xpdf-icon.ico
work/xpdf-4.03/xpdf-qt/indicator-icon5.svg
work/xpdf-4.03/xpdf-qt/indicator-icon2.svg
```

Excellent! `xpdf-icon.ico` and `xpdf-icon.svg` look promising. These are the files we need to get installed into `/usr/local/share/icons`. In order to do that we need edit the port's `Makefile` and expand the install targets. This port already has a `post-install` target so let's add four more lines to it. We will use `${INSTALL_DATA}` to install icon files, and `${MKDIR}` to create directories. These variables are examples of numerous wrapper variables defined in the FreeBSD Ports framework. If you'd like to know more about those variable take a look at the output of, e.g., `make -V INSTALL_DATA`. The patch should look like this so far:

```
diff --git a/graphics/xpdf4/Makefile b/graphics/xpdf4/Makefile
index bd81dd1a16be..36bd84d97e7e 100644
--- a/graphics/xpdf4/Makefile
+++ b/graphics/xpdf4/Makefile
@@ -70,5 +71,9 @@ post-install:
                ${INSTALL_DATA} ${WRKDIR}/xpdf-man.conf \
                        ${STAGEDIR}${PREFIX}/etc/man.d/xpdf.conf
                ${INSTALL_DATA} ${FILESDIR}/xpdf.desktop ${STAGEDIR}${DESKTOPDIR}
+               ${MKDIR} ${STAGEDIR}${PREFIX}/share/icons/hicolor/256x256
+               ${INSTALL_DATA} ${WRKSRC}/xpdf-qt/xpdf-icon.ico
${STAGEDIR}${PREFIX}/share/icons/hicolor/256x256/xpdf.png
+               ${MKDIR} ${STAGEDIR}${PREFIX}/share/icons/hicolor/scalable
+               ${INSTALL_DATA} ${WRKSRC}/xpdf-qt/xpdf-icon.svg
${STAGEDIR}${PREFIX}/share/icons/hicolor/scalable/xpdf.svg

 .include <bsd.port.mk>
```

This is nice. Since we are installing two new files now, we need to add them to the packing list (`pkg-plist`). The list can be regenerated with `make makeplist`, but we'll do it by hand this time. Here's the patch:

```
diff --git a/graphics/xpdf4/pkg-plist b/graphics/xpdf4/pkg-plist
index e6cd3e15dd75..7eee2ae85bc6 100644
--- a/graphics/xpdf4/pkg-plist
+++ b/graphics/xpdf4/pkg-plist
@@ -10,6 +10,8 @@ libexec/xpdf/pdftotext
 %%GUI%%libexec/xpdf/xpdf
 %%GUI%%bin/xpdf
 %%GUI%%%%DESKTOPDIR%%/xpdf.desktop
+%%GUI%%share/icons/hicolor/256x256/xpdf.png
+%%GUI%%share/icons/hicolor/scalable/xpdf.svg
```

```
etc/man.d/xpdf.conf
%%DATADIR%%/man/man1/pdfdetach.1.gz
%%DATADIR%%/man/man1/pdffonts.1.gz
```

Paths in that list are relative to `${PREFIX}` (`/usr/local` by default). `%%GUI%%` at the beginning of the line means that those files are only going to be installed if this port is built with the `GUI` option enabled (apparently, some people like to have their Xpdf software headless).

The last bit we need to take care of is to bump the port's revision number. Once the change lands in the ports tree, port builders must know to rebuilt the package with our modifications. The easiest way to bump the revision is to use `portedit` (it's a part of the `portfmt` package):

```
$ portedit bump-revision -i Makefile
```

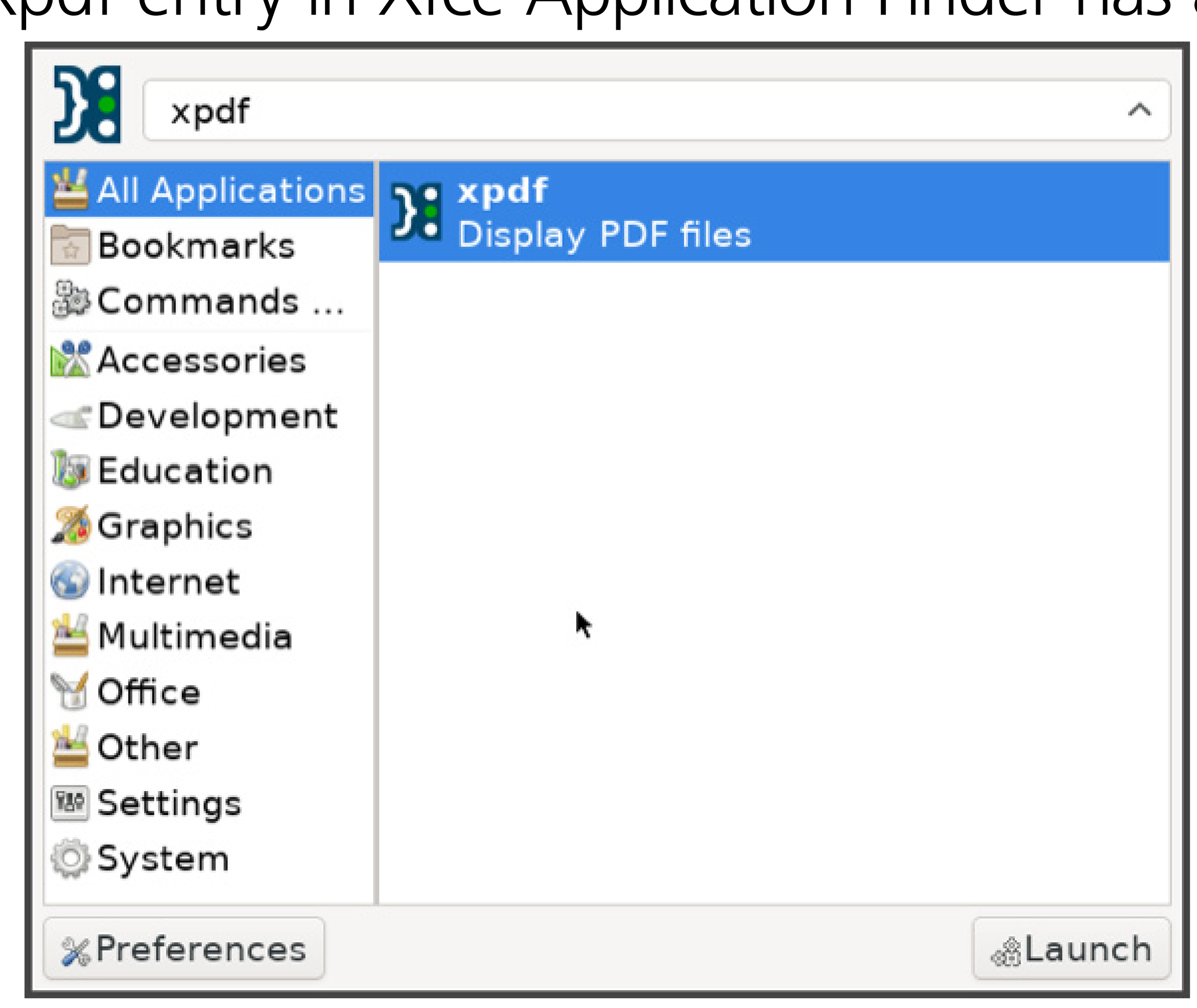
As a result we should see the following in the diff:

```
diff --git a/graphics/xpdf4/Makefile b/graphics/xpdf4/Makefile
index bd81dd1a16be..36bd84d97e7e 100644
--- a/graphics/xpdf4/Makefile
+++ b/graphics/xpdf4/Makefile
@@ -1,5 +1,6 @@
 PORTNAME=      xpdf
 PORTVERSION=   4.03
+PORTREVISION=  1
 PORTEPOCH=     1
 CATEGORIES=    graphics print
 MASTER_SITES=  https://dl.xpdfreader.com/
```

Great! Now let's test our changes. For that we need to build and reinstall Xpdf. The following command is enough. You may want to run `make missing` first and install the dependencies with pkg(8) to save time.

```
make reinstall
```

Time to check if the Xpdf entry in Xfce Application Finder has an icon now.



Success!

We need to test the patch a bit more before we submit it for review.

1. Does Xpdf still work? (Launch the newly reinstalled Xpdf and see if everything™ is in order.)
2. Does our patch work as exptected? (We've seen the icon in Xfce Application Finder so the answer is yes.)
3. Can you build Xpdf in Poudriere? (Hmm?)

Poudriere setup is well explained in the "Testing the Port" chapter of the FreeBSD Porter's Handbook: https://docs.freebsd.org/en/books/porters-handbook/testing/.

Submitting the Patch

FreeBSD Bugzilla is the service where contributors upload patches with suggested changes: https://bugs.freebsd.org. The whole process is fairly straightforward. First, you create an account and log in. Then you open a new problem report (PR) by clicking "New" in the navigation bar at the top. Remember to prefix the summary with "`graphics/xpdf4`". This way the port's maintainer will get notified about the PR (some other tips on how to write a good PR are written down here: https://wiki.freebsd.org/Bugzilla/DosAndDonts). Sometimes, in addition to opening a PR on Bugzilla, people submit their patches to Phabricator. This other service has a nicer interface for code reviews.

Oh, BTW, the problem of missing Xpdf icons is based on a true story. I've reported the issue with Xpdf on Bugzilla (https://bugs.freebsd.org/bugzilla/show_bug.cgi?id=261376) and posted my patch to Phabricator (https://reviews.freebsd.org/D33984). The Xpdf maintainer reviewed my patch and gave the green light to commit the change. Since I am a ports committer, I committed the change myself.

Getting help

One day you will embark on your own journey of contributing to the FreeBSD ports tree. Many feel overwhelmed and terrified when facing the challenge. Fear not! The FreeBSD community is always ready to help you out. IRC channels, mailing lists, forums, and most recently Discord are all great venues to talk to other FreeBSD folks and ask questions.

Most importantly, have fun hacking on FreeBSD and enjoy your time among the FreeBSD folks. See you at the zoo!

MATEUSZ PIOTROWSKI is a FreeBSD ports and documentation committer based in Berlin. He enjoys troubleshooting bugs, scripting automation, and designing robust software systems (always thoroughly documenting everything along the way). Recently, his interests have drifted toward tracing and performance engineering. When he is not hacking on the supposedly deterministic circuitry of modern software, he is exploring the ever-changing dynamics within society and culture.