# Contributing to void-packages

void-packages is the backbone of the Void Linux distribution. It contains all the definitions to build packages from source.

This document describes how you, as a contributor, can help with adding packages, correcting bugs and adding features to void-packages.

## Getting your packages into Void by yourself

If you really want to get a package into Void Linux, we recommend you package it yourself.

We provide a [comprehensive Manual](./Manual.md) on how to create new packages.

There's also a [manual for xbps-src](./README.md), which is used

to build package files from templates.

For this guide, we assume you have basic knowledge about [git](http://git-scm.org), as well as a [GitHub Account](http://github.com).

Please note that we do not accept any packages containing non-release versions, such as specific git- or svn-revisions anymore.

### Creating a new template

You can use the helper tool `xnew`, from the [xtools](https://github.com/chneukirchen/xtools) package, to create new templates:

$ xnew pkgname subpkg1 subpkg2 ...

Templates must have the name `void-packages/srcpkgs/<pkgname>/template`, where `pkgname` is the same as the `pkgname` variable in the template.

For deeper insights on the contents of template files, please read the [manual](./Manual.md), and be sure to browse the existing template files in the `srcpkgs` directory of this repository for concrete examples.

When you've finished working on the template file, please check it with `xlint` helper from the [xtools](https://github.com/chneukirchen/xtools) package:

$ xlint template

If `xlint` reports any issues, resolve them before committing.

### Committing your changes

Once you have built your template file or files, the commit message should have one of the following forms:

\* for new packages, use ```New package: <pkgname>-<version>``` ([example](https://github.com/void-linux/void-packages/commit/176d9655429188aac10cd229827f99b72982ab10)).

\* for package updates, use ```<pkgname>: update to <version>.``` ([example](https://github.com/void-linux/void-packages/commit/b6b82dcbd4aeea5fc37a32e4b6a8dd8bd980d5a3)).

\* for template modifications without a version change, use ```<pkgname>: <reason>``` ([example](https://github.com/void-linux/void-packages/commit/8b68d6bf1eb997cd5e7c095acd040e2c5379c91d)).

\* for package removals, use ```<pkgname>: remove package``` ([example](https://github.com/void-linux/void-packages/commit/83784632d94deee5d038c8e1c4c1dffa922fca21)).

\* for `common/shlibs` modifications, use `common/shlibs: <pkgname>` ([example](https://github.com/void-linux/void-packages/commit/613651c91811cb4fd2e1a6be701c87072d759a9f)).

If you want to describe your changes in more detail, add an empty line followed by those details ([example](https://github.com/void-linux/void-packages/commit/f1c45a502086ba1952f23ace9084a870ce437bc6)).

Pull requests are automatically submitted for Continuous Integration (CI) testing to ensure packages build on various combinations of libc and architecture.

Packages that take longer than 50 minutes to build (for example, Firefox or the Linux kernel) will fail CI and should include `[ci skip]` in the last line of the last commit message in the PR to avoid wasting CI builder time.

Use your best judgment on build times based on your local building experience. If you skip CI when submitting a PR, please build and cross-build for a variety of architectures locally, with both glibc and musl, and note your local results in PR comments.

Make sure to cover 64-bit and 32-bit architectures.

Runtime testing of packages and building with the `-Q` flag or with `XBPS\_CHECK\_PKGS=yes` set in the environment or `etc/conf` are strongly encouraged.

New packages will not be accepted unless they have been runtime tested.

For further information, please consult the [manual](https://github.com/void-linux/void-packages/blob/master/Manual.md).

`xbump`, available in the [xtools](https://github.com/chneukirchen/xtools) package, can be used to commit a new or updated package:

$ xbump <pkgname> <git commit options>

`xbump` will use `git commit` to commit the changes with the appropriate commit message. For more fine-grained control over the commit, specific options can be passed to `git commit` by adding them after the package name.

After committing your changes, please check that the package builds successfully. From the top level directory of your local copy of the `void-packages` repository, run:

$ ./xbps-src pkg <pkgname>

Your package must build successfully for at least x86, but we recommend trying to build for armv\* as well, e.g.:

$ ./xbps-src -a armv7l pkg <pkgname>

For further details, see the output of `./xbps-src -h`.

### Starting a pull request

Once you have successfully built the package, you can start a pull request.

Most pull requests should only contain a single package and dependencies which are not part of void-packages yet.

If you make updates to packages containing a soname bump, you also need to update `common/shlibs` and revbump all packages that are dependant.

There should be a commit for each package revbump, and those commits should be part of the same pull request.

When you make changes to your pull request, please \*do not close and reopen your pull request\*. Instead, just forcibly git push, overwriting any old commits. Closing and opening your pull requests repeatedly spams the Void maintainers.

#### Travis

Once you have started a pull request, you will get instant feedback from Travis. It will check if the templates you have changed

comply with the our guidelines. At the moment not all packages comply with the rules, so if you update a package, it may happen that Travis reports errors about places you haven't touched. Please feel free to fix those errors too.

#### Review

Most of the time your pull request will contain mistakes. It's nothing bad, it just happens.

Reviewers will comment on your pull request and point out which changes are needed before the template can be included.

We recommend having only a single commit for pull request, so if you need to make changes in commits but already have a pull request, use the following commands:

$ git add <file>

$ git commit --amend

$ git push -f

#### Closing the pull request

Once you have applied all requested changes, the reviewers will merge your request.

If the pull request becomes inactive for some days, the reviewers may or may not warn you when they are about to close it.

If it stays inactive further, it will be closed.

Please abstain from temporarily closing a pull request while revising the templates. Instead, leave a comment on the PR describing what still needs work, or add "[WIP]" to the PR title. Only close your pull request if you're sure you don't want your changes to be included.

#### Publishing the package

Once the reviewers have merged the pull request, our [build server](http://build.voidlinux.org) is automatically triggered and builds

all packages in the pull request for all supported platforms. Upon completion, the packages are available to all Void Linux users.