



Baked buildings

the easy guide to Better Continents 0.10.x

Make the buildings of your Valheim world part of the world itself.

Written by Wubarrk, with Claude (Anthropic) as the drafting agent

For Valheim 1.0 and Better Continents 0.10.x (github.com/RGLabs84/BetterContinents10)

October 2026

Written for players and server owners; no config editing needed beyond one setting.

The pictures are screenshots of the game, taken by an automated test.

Licensed under the GNU General Public License, version 3, WITHOUT ANY WARRANTY. The licence text is the final appendix; the Markdown source, the pictures and the build scripts are attached to this PDF (baking-guide-source.zip). The cover: a baked town block in a test world.

Contents

Start here: what baking does	4
What stays a real piece	4
Where baked pieces come from	5
What every player needs	5
The whole thing in four steps	5
License	6
1 Open the Bake buildings tab	7
Turn the window on	7
Open the tab	7
Who may bake	8
Two rules the tab always follows	8
2 Bake your base	9
Before you start	9
Bake	9
The options	12
What a bake never changes	12
3 Undo and unbake	13
Undo the last change	13
Unbake around you	14
The way out: unbake the whole world	14
When the game crashes in the middle	14
4 Town files	15
A new world with towns	15
A newer town file for a world that exists	15
The town's doors, chests and stations	16
Bushes, mushrooms and flax in a town file	16
Clear ground, the town's ground and paint	16
Export the layer	16
For tool makers	16
5 On a server	17
What the server needs	17

Joining	17
Baking on a server	17
Hosting a game for friends	18
Sharing a world file	18
6 Settings	19
What baked pieces cost your machine	19
Shadows	20
Hide and show	20
7 When it goes wrong	21
The bake refuses	21
Some pieces changed while the bake ran	21
Baked pieces are missing after removing a mod	21
Two pieces in the same place	21
The game crashed in the middle of a bake	22
Removing the layer altogether	22
What to send with a bug report	22
Appendix A The bc_bake console commands	23
Bake	23
Take back	23
Town files	23
Look	24
Where the files are	24
Appendix B placements.bcp for tool makers	25
Appendix C GNU General Public License, version 3	27

Start here: what baking does

Better Continents 0.10.x can **bake** the buildings in a Valheim world: the walls, floors, roofs and beams that players built become part of the world itself, the way its mountains and trees are.

A building you bake looks exactly as it did. You walk on its floors, climb its stairs and hide behind its walls as before. What changes is what the game has to carry around:

- **An ordinary piece is an object.** The game keeps every wall and floor as an object of its own: it saves each one, sends each one to every player who comes near, checks each one for wear and support, and loads each one into the scene. A big base is tens of thousands of objects, and a town is hundreds of thousands.
- **A baked piece is not an object any more.** Better Continents keeps it in the world's *layer*, a compact list saved with the world's settings. Every player's game draws the baked pieces from that list and gives them something solid to stand on, but none of them is an object: nothing to save one by one, nothing to send, nothing that wears or needs support.

The object count is what baking is for. It is what makes a big build heavy, and baking takes the objects away.

A town of 700,000 pieces	As ordinary pieces	Baked
pieces that are objects in the world	700,000	608: its doors, chests and stations
what the world saves and sends, piece by piece	all of them	608
what the layer holds	nothing	700,000 records, about 3 MB

Measured on a real town file of 702,700 records. It also places 12,052 bushes, mushrooms and flowers, which are never baked: they stand as the world's own vegetation does. Your own base is smaller, and the ratio is much the same.

What stays a real piece

Baking only takes pieces that do nothing but stand there. Everything that does something stays a real piece, exactly as it is:

Stays real	Why
doors, gates, hatches	they open and close
chests, item stands, armour stands	they hold things
workbenches, forges, smelters, kilns, fermenters, cooking stations, beehives	they work and hold things
beds, chairs, benches, thrones, tables, banners, rugs	you sleep or sit on them, and they give comfort (with Town, the decor bakes)
fires, braziers, torches, lights	they burn, warm and light (with Town, torches and lanterns become lit copies)
portals, wards, signs, map tables	they hold a link, a name, a text or a map
carts and ships	they move
anything from a mod that Better Continents does not know	it might keep something, so it is never risked

A chair or a bench from a mod that gives no comfort is the one exception: it is baked as a *seat*, and you can still sit on it.

Nothing that gives items is ever baked. Berry bushes, mushrooms, flax, planted crops and items lying on the ground stay what they are: you pick them, they grow back, they rot.

The *Town* option (chapter 2) is for towns, where speed matters more than comfort: it also bakes the decor (tables, banners, carpets, plants, chairs and benches, which you can still sit on) and turns torches, lanterns and candles into lit copies that never need fuel. Doors, chests, stations, beds and the fires you cook on stay real, and become protected parts of the layer that players cannot break or remove.

Where baked pieces come from

There are two ways in, and one world can have both:

1. **You bake in the game.** Stand in your base, open the export window's **Bake buildings** tab, and bake everything within a radius around you. Chapter 2.
2. **A town file.** A tool that builds towns outside the game can write a `placements.bcp` file. Put it in the world's map folder, and Better Continents builds its towns into the world. Chapter 4.

Either way, every bake has a dry run that says what it will do, and an undo.

What every player needs

- **Everyone who joins needs the same Better Continents as the world.** This is already true of every Better Continents world. A player who joins downloads the layer once (a few megabytes for a whole town file) and keeps a copy, so the next join is quick.
- **Without Better Continents, the baked pieces are gone.** The game alone opens the world without its layer, and an older Better Continents (before 0.10.4) cannot read the world's settings at all, so it opens it as a world without Better Continents, with the game's own terrain. Either way baked buildings are neither drawn nor solid there. The settings and the layer are still in the world's files, and come back when this version opens the world again.
- **Before you remove Better Continents from a world, unbake it.** One button makes every baked piece a real piece again. Chapter 3.

Watch out: Valheim's own world rename, move and backup only carry Better Continents' settings file through while Better Continents is installed. A world renamed or restored from a backup by the game alone loses its layer, and with it every baked piece. This was already true of every Better Continents world.

The whole thing in four steps

1. **Turn on the export window** (the `Hud` setting) and press **F7** in your world. Chapter 1.
2. **Open the Bake buildings tab, pick a radius, and press Bake: dry run.** Read what it says. Chapter 2.
3. **Press Confirm.** A few seconds later the pieces are baked. Chapter 2.
4. **Changed your mind? Undo the last change** puts everything back as it was. Chapter 3.

Chapter 5 covers servers and their admins, chapter 6 the settings that change how far baked pieces are drawn, and chapter 7 what to do when something goes wrong. Appendix A lists every `bc_bake` console command.

New in Better Continents 0.10.4 (2026-10).

License

Better Continents: Baked buildings, the easy guide was written by Wubarrk, with Claude (Anthropic's model) as the drafting agent, and is released under the **GNU General Public License, version 3** (GPL-3.0), the licence of the Better Continents Guide it accompanies, so that the two documents can be shipped and changed together under one licence. Copyright 2026 Wubarrk.

It is distributed **WITHOUT ANY WARRANTY**; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. The full text of the licence is the final appendix, [GNU General Public License, version 3](#).

This guide describes Better Continents 0.10.x as its source code is written: the command names, settings, defaults, file names and messages were copied from the code, and the examples of the command output follow what the code writes. The pictures are screenshots taken by an automated test of the game. Where the game shows something different from this guide, the game is right; please report it (see chapter 7).

Better Continents is free software under the GNU Lesser General Public License, version 2.1 (created by billw2012, maintained by Jere Kuusela from 2023 to 2026 and by Wubarrk since September 2026). The Better Continents Guide is GPL-3.0 (written by billw2012). Neither is reproduced here beyond short statements of how the mod behaves. Valheim is the property of Iron Gate Studio; the menu labels quoted here are the game's own English texts, quoted so that you can find the buttons.

Attachment. The file `baking-guide-source.zip` is attached to this PDF; open your PDF viewer's attachments panel to save it. It contains:

- `docs/`: the Markdown source of this guide and its pictures;
- `build_baking_guide.py` and `build_pdf.py`: the build scripts (the second is the Better Continents Guide's renderer, which the first reuses);
- `LICENSE`: the GPL-3.0 text.

1 Open the Bake buildings tab

In short: set `Hud` to `true` in the `[09 BetterContinents.Export]` section of `BetterContinents.cfg`, or tick **Hud** in Configuration Manager. In a world, press **F7** and click **Bake buildings** at the top of the window. Prefer typing? The console's `bc_bake` does the same: appendix A.

Baking lives in the same window as the world export and import: the export window. If you have used the export, you know it already; the third tab is new.

Turn the window on

The window is off when you first install Better Continents. Turn it on once, either way:

- **Configuration Manager** (a separate, free mod that lists every mod's settings in the game): press **F1**, find Better Continents and its section `09 BetterContinents.Export`, tick **Hud**, and press **F1** again.
- **The config file:** open `BepInEx/config/BetterContinents.cfg` in a text editor, find the section `[09 BetterContinents.Export]`, and change `Hud = false` to `Hud = true`. Save the file. If the game is running, the change applies at once.

On a server, the server's own `Hud` setting decides: ask its owner to turn it on in the server's config.

Open the tab

1. **Load your world** and stand in the base you want to bake.
2. **Press F7.**

You see: the export window, and your mouse is free to click.

3. **Click Bake buildings** at the top of the window, next to Export and Import.

You see: the tab, with a line about this world's baked pieces at the top (*no baked pieces yet* in a world that has none), then the sections **Around you**, **Undo and the way out**, **A town file**, **Look**, and at the bottom **What bc_bake said**, where every answer appears.



Screenshot: the Bake buildings tab in a world with no baked pieces yet.

Press **F7** again to close the window. **F9** shows or hides a small status box, which has a **Baking:** line naming the change while one runs.

Who may bake

Where you play	Who can bake and undo
alone (single player)	you
a game you host for friends	you, the host; and players on your admin list
a dedicated server	the server's admins (its <code>adminlist.txt</code>)

Everyone else can still use **Info**, **What is drawn here** and **Hide baked pieces**, which change nothing in the world.

On a server the tab sends your clicks to the server, which checks that you are an admin, does the work, and sends its answer back into **What bc_bake said**. The tab says so at the top: *You are a player on a server: changes go to the server, which makes them for its admins only.*

No cheats, Debug Mode or `devcommands` are needed for any of it.

Two rules the tab always follows

- **Every change starts with a dry run.** The dry run buttons only report what would happen. Nothing in the world changes until you press **Confirm**, and Confirm does exactly what the dry run above it said, at the place where you stood when you ran it. If you change an option after the dry run, Confirm turns grey until you run the dry run again.
- **One change at a time.** While a bake, undo or unbake runs, the tab waits for it to end.

2 Bake your base

In short: stand in the middle of your base, pick a radius in **Around you**, press **Bake: dry run**, read what it says, and press **Confirm**. Every piece within the radius that only stands there becomes part of the world; doors, chests, stations, beds and fires stay real pieces.

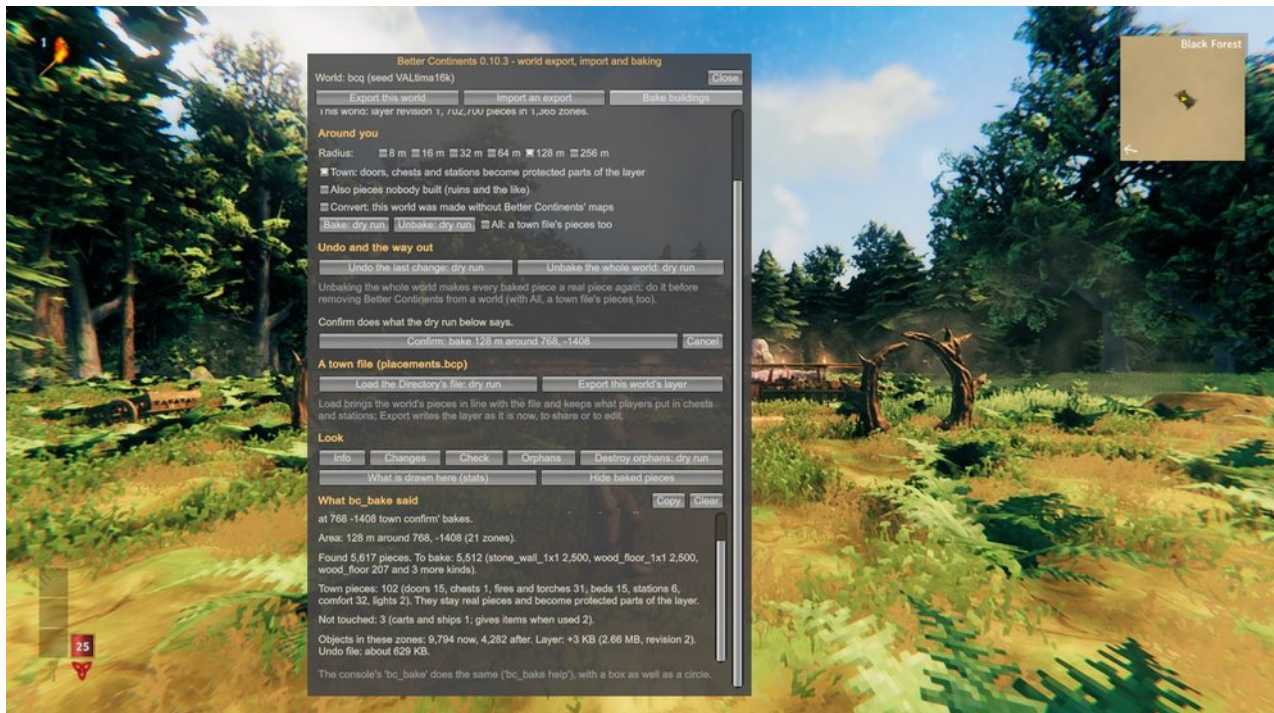
Before you start

- **Stand where the base's middle is.** The bake takes every piece within the radius around the spot where you stand when you press the dry run. Pieces count by their own position, at any height: a tower's top counts when its foot is inside.
- **Pick the radius.** The buttons go from 8 m to 256 m. 32 m covers a house and its yard, 64 m a large base, 128 m and 256 m a village. A bigger radius takes longer, and nothing is lost by baking a base in two or three parts.
- **Wait until you have finished building there.** A baked piece can no longer be repaired, moved or taken down with the hammer. To change a baked building, undo or unbake it first (chapter 3), change it, and bake again.

Bake

1. Press **F7**, click **Bake buildings**, and click a radius under **Around you**.
2. **Leave the options as they are for now:** Town off, Also pieces nobody built off, Convert off. The next section says what each one does.
3. Press **Bake: dry run**.

You see: under **What bc_bake said**, the dry run. Nothing in the world has changed.



Screenshot: the dry run of a bake around a test village, with Town ticked.

A dry run reads like this, line by line:

```
bc_bake area 128 at 768 -1408 town: a dry run, nothing changes. 'bc_bake area 128 at 768 -1408
town confirm' bakes.
Area: 128 m around 768, -1408 (21 zones).
Found 5,617 pieces. To bake: 5,512 (stone_wall_1x1 2,500, wood_floor_1x1 2,500, wood_floor 207
and 3 more kinds).
Town pieces: 102 (doors 15, chests 1, fires and torches 31, beds 15, stations 6, comfort 32,
lights 2). They stay real
pieces and become protected parts of the layer.
Not touched: 3 (carts and ships 1; gives items when used 2).
Objects in these zones: 9,794 now, 4,282 after. Layer: +3 KB (2.66 MB, revision 2). Undo file:
about 629 KB.
```

The dry run of a 128 m bake around a test village with Town ticked, as the game wrote it. Without Town, the third line says Stay pieces, and that they stay as they are.

Line	What it tells you
Area	the circle, and how many of the world's 64 m zones it touches
Found ... To bake	the pieces in the circle, and those that will become part of the world, by kind
Stay pieces (with Town: Town pieces)	pieces that stay real objects: doors, chests, stations and the rest (start here)
Not touched	pieces the bake leaves alone, and why: built by nobody, they move, they give items, or they are baked already
Objects in these zones	how many objects the world holds in those zones now, and after the bake: this is what baking saves
Layer, Undo file	how much the world's layer grows, and the size of the file that lets you undo this bake

4. **Read the dry run.** If a piece you expected to bake is under Stay pieces or Not touched, the line says why. If it is all as you want, go on.
5. **Press Confirm: bake ... m around ...** (the button names the radius and the spot of the dry run).

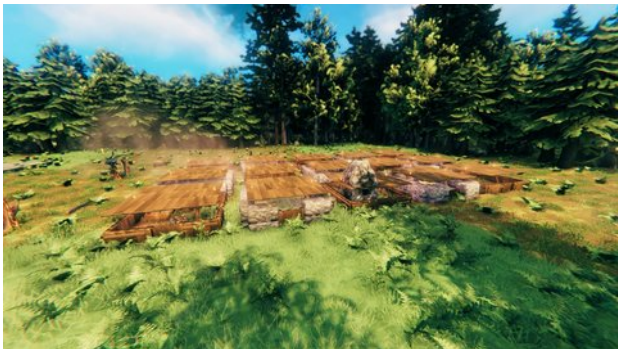
You see: a few lines as the bake runs, then the last one:

```
bc_bake: bake 1 started, 5,512 pieces. Undo file written.
bc_bake: layer revision 2 sent to 0 players; 0 of 0 ready in 0.1 s.
bc_bake: removing 5,512 pieces, 1,000 a frame.
bc_bake: bake 1 done in 0.5 s: 5,512 pieces baked, 102 left as pieces (town pieces, protected
parts of the layer).
Objects here: 9,794 -> 4,282.
'bc_bake undo' puts them back. The world saves as usual ('save' saves now).
```



Screenshot: the tab after Confirm, with the bake's last lines under What bc_bake said.

The base looks exactly as before. The pieces that were baked are no longer objects of the world.



q-town-before.jpg



q-town-after.jpg

Screenshot: the same houses before the bake (left: every piece an object) and after it (right: drawn from the layer).

What the bake of the test village did, counted in the game within 128 m of its middle:

	Before	After
objects (each saved, sent to players and updated by the game)	7,984	2,472
renderers in the scene	15,713	4,689
colliders of real pieces	9,930	4,418
baked pieces drawn by Better Continents	0	5,512, in 12 batches

Every one of the 5,512 baked pieces left the world as an object: none was left behind, and the 102 real pieces resting on baked floors and walls kept their support.

The world saves the bake with its next save, as it saves anything else. Type **save** in the console to save at once.

The options

Town: doors, chests and stations become protected parts of the layer. Use it for a town that other players visit, where every object counts. With it the bake takes much more:

With Town	What happens
tables, banners, carpets, potted plants, statues, garlands	baked like walls. A baked piece gives no comfort, so keep a few real ones where players rest
chairs, benches, stools, thrones	baked as seats: drawn, and you can still sit on them when you are near
wall torches, standing torches, lanterns, candles	baked as lights: the nearest ones (24 by default, chapter 6) burn and never need fuel; the others are drawn unlit. All of them are solid. A baked light gives no warmth, and monsters no longer shy away from it
fire pits, hearths, bonfires, braziers	stay real: cooking stations and cauldrons need a real fire
doors, chests, stations, beds, portals, signs	stay real objects that work as before, and become <i>protected</i> : nobody can damage or remove them, and the weather does not wear them

Without Town, the decor, the lights and the Stay pieces are all left exactly as they are. Undo takes everything back, the protection too. The dry run lists each group (*Comfort: N become decor, Lights: N become lit copies*) before you confirm.

Also pieces nobody built (ruins and the like). Pieces the game placed itself, such as the walls of ruins and villages, have no builder, and a bake leaves them alone. Tick this to bake them too. The dry run warns that a later reset of their zone by a world tool may put the game's own pieces back over the baked ones.

Convert: this world was made without Better Continents' maps. Baking works in any world, including one made without Better Continents' maps. The first bake in such a world makes it a Better Continents world that keeps the game's own ground, biomes, forests and locations; only the layer is added. The dry run says so and asks for Convert. From then on every player needs Better Continents to see the baked pieces (start here).

What a bake never changes

- **The game's rules for your base.** A baked base still counts as a base: beds, workbenches and cooking fires always stay real, so the "safe in home" checks and creature spawns around them work as before. Without Town, comfort and every torch's warmth stay as they were too; with Town, baked decor gives no comfort and baked lights give no warmth.
- **Support.** A baked wall holds what rests on it as firmly as the ground does. Real pieces resting on baked ones never lose support.
- **The look.** Each piece keeps its material variant and its own random shade.
- **Health.** A baked piece is drawn as new. Unbaking it gives back the health it had.

Good to know: a bake waits until every player near the area has the new layer before it takes the pieces out, so nobody falls through a floor. On a busy server that can take a few seconds.

3 Undo and unbake

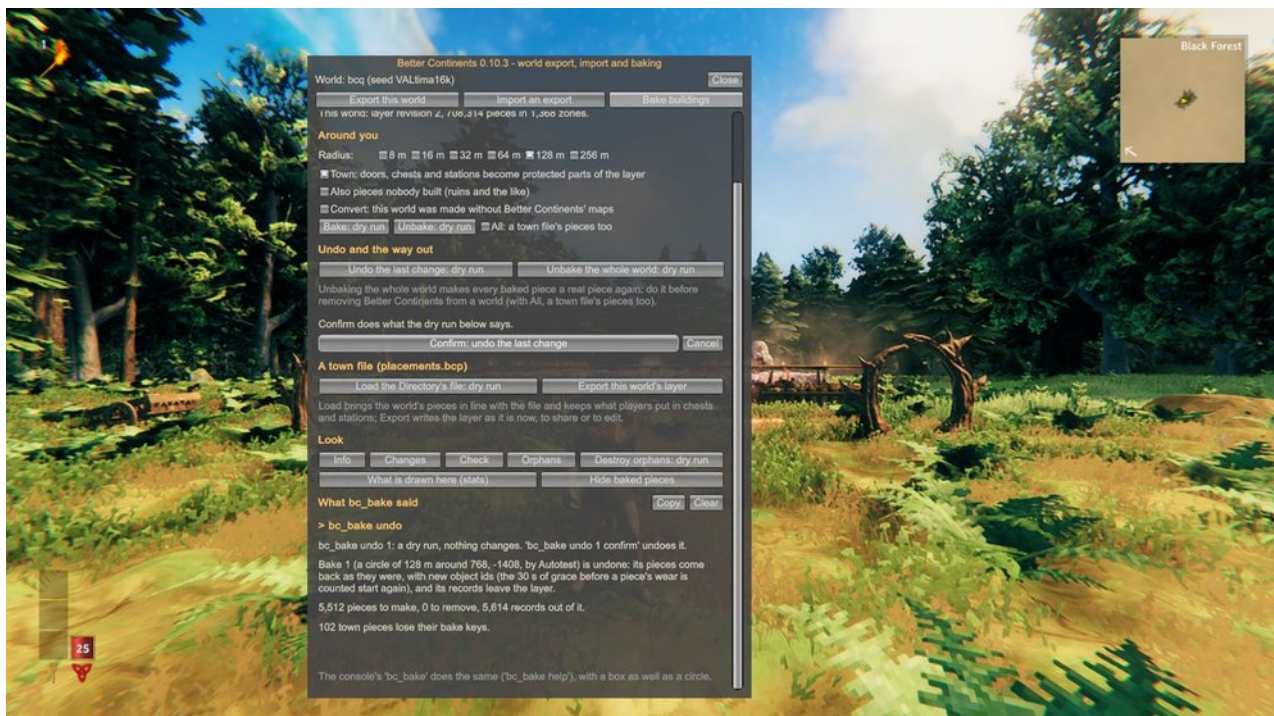
In short: **Undo the last change** puts the world back exactly as it was before the last bake, unbake or drop. **Unbake: dry run** under **Around you** turns baked pieces within the radius back into real pieces. **Unbake the whole world** does it for every baked piece: do that before you remove Better Continents from a world.

Both start with a dry run, like a bake, and both wait for **Confirm**.

Undo the last change

1. Press **Undo the last change: dry run** under **Undo and the way out**.

You see: which change it undoes (its number, what it was, when and who), and how many pieces come back.



Screenshot: the dry run of an undo, right after a bake.

2. Press **Confirm**.

You see: the pieces come back as real objects first, and only then does the layer let go of them, so there is never a moment when a floor is missing.

After an undo, the pieces are exactly the objects they were before the bake: the same kind, place, turn, health, builder and contents. Two small things are new: the game gives them new internal ids (it does so at every load anyway), and their wear clock starts again. Town pieces lose their protection.

Undo also undoes an unbake (the pieces it made go and are baked again) and a drop (chapter 7).

Older changes. The **Changes** button under **Look** lists the world's changes, newest first, with their numbers. The console's `bc_bake undo <number>` undoes an older one, as long as none of its pieces was unbaked since; otherwise it says so, and `unbake` takes the rest back. The newest 20 changes of each world are kept for undo.

Unbake around you

Unbake is undo's free-form sibling: it takes any baked pieces in a circle, from any bake, and makes them real pieces again.

1. **Stand where you want it, and pick a radius** under **Around you**.
2. **Press Unbake: dry run**. It says how many pieces become real, and how many objects the zones will hold after.
3. **Press Confirm**.

An unbaked piece is a real piece again, with the builder, health and look it had when it was baked. You can repair, move or remove it with the hammer as usual.

All: a town file's pieces too. Without it, unbake takes only what was baked in the game. Tick it to unbake a town file's pieces as well (chapter 4); they become real pieces with you as their builder. The next time the world loads a new copy of the town file, those pieces come back as baked ones, so a lasting change belongs in the town file itself.

The way out: unbake the whole world

A baked piece exists only in the world's layer, and only Better Continents reads the layer. If you remove Better Continents from a world, every baked piece is gone from it: not drawn, not solid. An older version is worse: it cannot read the world's settings, so it opens the world with the game's own terrain as well. So, before you do:

1. **Load the world** with this version of Better Continents.
2. **Press Unbake the whole world: dry run**. It says how many real objects this makes.
3. **Press Confirm**, and wait for the last line.
4. **Save** (type **save** in the console), and quit.

Every piece baked in the game is a real piece again. Tick **All** first to unbake a town file's pieces too.

Watch out: unbaking a big world makes a lot of objects at once. Over 50,000 the dry run warns: the world's save, its loading and what each player is sent all grow with them. If the world is to stay light, unbake one area at a time and remove what you no longer need.

Things that give items (bushes, mushrooms, flax) are never baked, so unbake leaves them alone; the dry run counts them as *left alone*.

When the game crashes in the middle

Every bake, undo and unbake writes its undo file before it changes anything, and notes each step it reaches. If the game crashes or the power goes, the next time the world loads Better Continents finishes the change or takes it back whole, and says which in the log. No piece is ever lost, and none is ever both a real piece and a baked one.

The undo files live outside the world folder, in **BetterContinents/bakes/** in Valheim's save folder, one folder per world.

4 Town files

In short: a town file is a `placements.bcp` that a tool wrote outside the game, holding whole towns. Put it in the folder that the `Directory` setting names, next to the world's maps, and make a new world: the towns are part of it from the start. For a world that exists already, **Load the Directory's file: dry run** brings it in.

A town file holds the same kind of records as a bake: pieces with their places, drawn by every player's game. The difference is who made them: a *compiler*, a tool that builds towns from its own designs, wrote the file, and you only hand it to Better Continents.

A new world with towns

1. Put `placements.bcp` in the maps folder that the `Directory` setting in `[00 BetterContinents.Debug]` names (the folder with `heightmap.png` and the other maps).
2. **Make a new world** with Better Continents enabled, as you would with any maps folder.

You see: the world loads with its towns standing.

Better Continents checks the file as it makes the world and keeps it byte for byte in the world's settings file. A damaged file is refused with a line in the log, and the world is made without it.

An export of a world with a layer writes the layer as `placements.bcp` beside its maps, so a world made from the export has the same towns.

A newer town file for a world that exists

When the compiler writes a newer file:

1. Put the new `placements.bcp` in the `Directory`, over the old one.
2. In the world, press **Load the Directory's file: dry run** under **A town file**.

You see: how many of the file's pieces come in, how many of the old ones go, the layer's new size, and what the doors, chests and stations of the towns will do (below).

3. **Press Confirm.**

What a load keeps and replaces:

Part	After a load
the town file's pieces	replaced by the new file's
pieces baked in the game	kept, whatever the file holds
the town's doors, chests and stations (its <i>live</i> pieces)	kept where they did not move; moved, added or removed as the new file says
what players put in the town's chests and stations	kept: a container that holds items is never taken away (orphans, below)

The layer as it was is kept in the world's folder of undo files as `layer-r<revision>.bcp`. To go back, load that file: in the console, `bc_bake load <that file> confirm`.

Tick Convert first to load a town file into a world that was made without Better Continents' maps (chapter 2).

The town's doors, chests and stations

A town file marks some of its pieces *live*: those are made as real objects in the world, so their doors open and their chests hold things. Better Continents makes each one when the game first generates its zone, gives it the town's id, and protects it, so it cannot be broken or removed.

When a newer file, loaded as above, moves or removes such a piece, the world follows at once: the piece is made again at its new place, or taken away. The one exception keeps players' things safe: **a container that holds items stays where it is**, loses its protection, and becomes an *orphan*.

- **Orphans** under **Look** lists them, with where they stand and whether they hold items.
- **Destroy orphans: dry run** shows which would go. Without **items** in the console's **bc_bake orphans destroy items**, the ones that hold items always stay.

Bushes, mushrooms and flax in a town file

A town file may place things that give items: berry bushes, mushrooms, flax, dandelions. These are never baked. Each one is placed **once** as an ordinary object, the first time the game generates its zone, and from then on it is the world's own: players pick it, it grows back, and a reset of its zone puts it back as the game puts back its own vegetation. A newer town file brings its bushes and mushrooms only into zones the game has not generated yet.

A town file can also mark such a kind as **decor** (appendix B): meads on a tavern's shelf, flax in a window box. Then it is drawn like the rest of the town and stands where the file says, but it is not an object: nobody can pick it, and it never grows back or goes away. **bc_bake info** lists those kinds on its last line, the one that starts *Decor:*. A kind the file does not mark stays as above, a real bush or mushroom.

Clear ground, the town's ground and paint

A town file can also say where the game's own trees and rocks must not grow (the streets and squares), lift or lower the ground under a town, and paint it (paths, paving). Better Continents applies all three on every machine, as the world generates. A load that clears more ground removes the game's own trees, rocks and bushes from the newly cleared cells of zones that exist already; nothing a player built is touched, and neither are the town file's own bushes and mushrooms.

Export the layer

Export this world's layer writes the world's layer as it is, town file and bakes together, to **BetterContinents/<world name>/placements.bcp** in Valheim's save folder. A compiler can read it back, or you can keep it as a copy.

For tool makers

Appendix B describes the file in one page and points to the full format.

5 On a server

In short: the server and every player run the same Better Continents. Players download the world's layer once when they join and keep a copy. Admins bake from the Bake buildings tab (or `bc_bake server ...` in the console), and every player gets the change before any piece is taken out.

What the server needs

- **The same Better Continents version on the server and on every player's game.** A player with another version, or without Better Continents, is turned away at the door, as in every Better Continents world.
- **Admins** in the server's `adminlist.txt`, as the game uses it for its own admin commands. Only they can bake, undo, unbake or load.
- `Hud = true` in the server's config, if admins should see the export window. The server's value is the one that counts for everyone on it. Without the window, admins use the console (appendix A).

Nothing else changes in how you run the server.

Joining

When a player joins, the server sends the world's settings as always, then the layer. The layer is compact: a 700,000-piece town file is about 3 MB. The player's game checks it and keeps a copy in `BetterContinents/cache/` in its save folder. The next time, if the layer has not changed, the copy is used and nothing is downloaded. A copy unused for 30 days is deleted.

`Settings Transfer Rate` in the config (the same setting that paces the settings download) paces the layer too.

Baking on a server

An admin bakes as in chapter 2. The tab sends each click to the server, the server checks that the player is an admin, does the work, and its answers appear in **What `bc_bake` said**. In the console, put `server` in front of the command: `bc_bake server area 32 confirm`.

On a dedicated server the radius is around the admin's own character.

What happens when an admin confirms a bake:

1. The server writes the undo file, then adds the pieces to the layer.
2. It sends every player the change. Usually that is a small patch of the zones that changed (tens of kilobytes); a player who missed an earlier change gets the whole layer.
3. It waits until each player says the new pieces are standing on their screen, or has left, or 30 seconds have gone by. The output names any player it stopped waiting for.
4. Only then does it take the real pieces out, 1,000 a frame. Each player sees them go in the same frame they would see any object removed.

So nobody standing in a baked house falls through its floor: the baked floor is under their feet before the real one goes.

Hosting a game for friends

The host is the server. The host bakes at their own console or tab, with no **server** in front. Friends on the host's admin list can bake with **server**, as on a dedicated server.

Sharing a world file

A **.bcworld** share file carries the world's settings but not its layer. A player who joined through a share file downloads the layer once, the first time, as above.

6 Settings

In short: the section `[10 BetterContinents.BakedPlacements]` of `BetterContinents.cfg` changes how far baked pieces are drawn and how they look on **your** screen. Every setting applies at once, and none of them changes the world: each player sets their own.

The defaults suit most machines. Change them in Configuration Manager (F1) or in the config file, while you watch the result.

Setting	Default	Range	What it does
Draw Distance	4	1 to 8 zones	How many zones (64 m each) around you baked pieces are drawn: 4 is a block of 9 x 9 zones, enough for a town's buildings.
Draw Scale	1.5	0.5 to 4	How far each baked piece is drawn, as a multiple of where the game hides the same piece (which follows your level-of-detail setting: about 140 m for a stone wall on High). 1 is the game's own; more shows more of a town from afar.
Detail Scale	1	0.5 to 4	How far baked pieces keep their full detail, as a multiple of where the game switches the same piece to its simpler model (about 30 m for a stone wall on High).
Shadows	All	All, Near, Off	Which baked pieces cast shadows. All: every one, as the game's own pieces do. Near: only those in full detail. Off: none. Every baked piece receives shadows.
Light Distance	90	0 to 200 m	How far from you a baked light can burn. Beyond it, baked torches and lanterns are drawn unlit. 0: none burns.
Light Count	24	0 to 128	How many baked lights burn at once: the nearest ones within Light Distance. The others are drawn unlit. Each burning light costs frame time, so a town full of torches stays fast.
Light Shadows	4	0 to 32	How many of the burning baked lights cast shadows: the nearest ones. A light's shadows cost memory and frame time, and some graphics settings allow every light to cast them. 0: none does.
Seat Distance	12	0 to 40 m	How far from you a baked world's chairs, benches and thrones can be sat on: within it each gets its seat, farther out it is only drawn. 0 turns the seats off.
Tints	(empty)	name=r,g,b; ...	Changes the colours a town file tints its pieces with, by the tint's name in the file, for example <code>plaster=1.65,1.58,1.48;</code> <code>brick=1.25,0.72,0.56</code> . Empty: the file's own colours.
Diagnostics	false		Writes the materials of each kind of baked piece to the log once. For bug reports.

Light Distance, Light Count, Light Shadows and Seat Distance apply to the lights and seats of a bake with Town (chapter 2) and of a town file (chapter 4). In a town with many torches, Light Count and Light Shadows are what keep the frame rate up: measured in a blueprint town with 199 baked lights, lighting all of them near you cost a third of the frame rate.

What baked pieces cost your machine

Baked pieces are drawn the fast way: every piece of a kind in view goes to the graphics card in one batch, instead of one object at a time. So what a town costs grows with the number of pieces in view, not with the number of objects in the world, and there are no objects for the game to update.

Measured in a world with a 702,700-piece town file, standing in its largest town: within two zones of the player the file holds 38,969 pieces, and 36 of them are real objects (the town's doors, chests and stations). Better Continents drew 27,990 pieces in 120 batches. As ordinary pieces, the other 38,933 would each be an object for the game to load, save and send, with about 11 renderers each to keep in the scene.

Shadows

Most of what a big town costs your graphics card is its shadows, not its pieces: the game draws every shadow-casting piece again for each band of its sun shadows. So Better Continents draws the shadows of baked pieces near you from merged shadow-only shapes, a few per zone, which it builds in the background on your other processor cores as you walk; the pieces you see are drawn as before. The shadows look the same. A few kinds of piece cast their own shadows as before (thatch roofs, beds, rugs and banners among them). In the busiest view of a test town of 33 blueprint buildings, the frame rate went from 38 to 63 fps, with the town's shadows using about 350 MB of graphics memory (measured on one machine). `bc_bake stats` shows them on its *Shadow proxies* lines.

If your frame rate drops in a big town, lower **Light Count** and **Light Shadows** first, then **Draw Scale** (to 1), then **Shadows** (to Near), then **Draw Distance**.

Hide and show

Hide baked pieces under **Look** stops drawing them on your screen, to compare. They stay solid: you still stand on their floors. **Show baked pieces** brings them back. **What is drawn here (stats)** lists, for what is in view now, how many pieces of each kind are drawn, in how many batches, and how long it took.

7 When it goes wrong

In short: every answer of the tab says what it did or why it did not. The log (`BepInEx/LogOutput.log`) has every line too. A crash in the middle of a change is finished or taken back at the next load, and no piece is ever lost.

The bake refuses

It says	What it means, and what to do
<i>This world is not a Better Continents world. Baking makes it one ... Add 'convert' to go ahead.</i>	The world was made without Better Continents' maps. Read start-here's "What every player needs", then tick Convert and run the dry run again.
<i>(names) would not be able to join the converted world</i>	Every player connected must run this Better Continents version before a world is converted. Ask them to update, or wait until they leave.
<i>this world has a Better Continents settings file that cannot be read</i>	The world's Better Continents file was written by a newer version, or is damaged. Better Continents never writes over it. Open the world with the version that made it, or restore the file from a backup.
<i>the pieces in this area have N different sets of values ... Bake a smaller area.</i>	The area holds more kinds of piece state than one bake can keep. Bake it in two or three parts.
<i>the layer cannot hold this bake</i>	The layer is at one of its limits (the reason follows). Bake a smaller area.
<i>Nothing to bake</i>	Every piece in the circle stays real or is left alone; the dry run's lists say why.
the Confirm button is grey	An option changed after the dry run, or another change is still running. Run the dry run again.

Some pieces changed while the bake ran

N pieces changed while the bake ran: their records stay. Someone moved or removed a piece between the dry run and the end of the bake. Those pieces were not taken out, and their baked copy is drawn too. **Undo the last change**, then bake again.

Baked pieces are missing after removing a mod

A baked piece from a mod is drawn only while that mod is installed. Without it, its pieces are not drawn and not solid, but they stay in the layer: put the mod back, and they come back. **Info** under **Look** lists the kinds this game does not have.

Two pieces in the same place

Check under **Look** reads the whole layer: damaged zones, and pieces of a town file and of a bake that stand in the same place. A bake inside a town file's town never takes the town's own pieces (they are not objects), but a newer town file can put a piece where a player's baked piece stands. Unbake the one you do not want, or drop the bake (appendix A).

The game crashed in the middle of a bake

Load the world again. Before anything else, Better Continents reads the undo file of the change that was running and either finishes it or takes it back whole. The log says which (look for `bc_bake`). Every piece is either a real piece or a baked one, never both and never neither, whatever the moment of the crash, and whether or not the game saved in between.

Changes under **Look** shows the change's state afterwards: *Settled* (finished), *Abandoned* (taken back) or *Undone*.

Removing the layer altogether

The console's `bc_bake drop <world name>`, typed in the main menu, takes the layer out of a world's files, and keeps the old file as `BetterContinents/pre-drop/<world name>/BetterContinents.pre-drop` in Valheim's save folder. It refuses a world with pieces baked in the game unless you add `confirm`, because those pieces are then gone for good: unbake the world first (chapter 3). A world that was converted (chapter 2) becomes the game's own world again.

What to send with a bug report

- The log, `BepInEx/LogOutput.log`, from the session where it happened.
- The output of **Info** and **Changes** (the **Copy** button copies the whole answer pane).
- With **Diagnostics** on (chapter 6), the log also lists the materials of each baked kind, which helps when a piece looks wrong.

Appendix A

The bc_bake console commands

Open the console with **F5**. **bc_bake** needs no cheats and no Debug Mode. Every command that changes the world is a dry run unless the line ends with **confirm**. Words like **town**, **any**, **convert**, **all** and **confirm** can stand anywhere after the command. **bc_bake help** prints this list in the game.

On a server, an admin puts **server** after **bc_bake** (for example **bc_bake server area 32 confirm**); the answer comes back to the admin's console. A player who is not an admin can use the read-only commands only.

Bake

Command	Does
bc_bake area <radius> [at <x> <z>]	a bake of the pieces within the radius (4 to 256 m) around you, or around the point x z. Add town , any , convert , confirm as in chapter 2
bc_bake box <x1> <z1> <x2> <z2>	the same for a box, up to 512 x 512 m

Take back

Command	Does
bc_bake undo [<n>]	undoes the newest change, or change n, from its undo file. Add confirm
bc_bake unbake <radius> [at <x> <z>]	turns baked pieces in the circle back into real pieces. all takes a town file's too. Add confirm
bc_bake unbake box <x1> <z1> <x2> <z2>	the same for a box
bc_bake unbake world [all]	every baked piece of the world: the way out before removing Better Continents. Add confirm
bc_bake drop compiler	takes the town file's pieces out of the layer without making real pieces. Add confirm ; undo brings them back
bc_bake drop bake <n>	the same for the pieces of bake n
bc_bake drop <world name>	in the main menu: removes a world's layer from its files (chapter 7). Add confirm when it holds pieces baked in the game

Town files

Command	Does
bc_bake load [<file>]	loads a town file into the running world: the Directory's placements.bcp , or the file named. convert as for a bake. Add confirm
bc_bake export [<file>]	writes the world's layer as it is, by default to BetterContinents/<world name>/placements.bcp in Valheim's save folder
bc_bake orphans	lists the town's live pieces whose record is gone and that were left standing (chapter 4)
bc_bake orphans destroy [items]	destroys them; the ones that hold items only with items . Add confirm

Look

Command	Does
<code>bc_bake info</code>	what this world's layer holds: its revision, where its pieces come from, how many of each role, the town's live pieces standing, its size, the kinds this game lacks, and the things that give items
<code>bc_bake list</code>	the changes made to this world: number, kind, time, who, area, pieces, state, and whether it can still be undone
<code>bc_bake check [all <zx> <zz>]</code>	reads the layer for damage and for pieces of a town file and a bake in the same place
<code>bc_bake stats</code>	on a player's game: what is drawn in view now, by kind, and what it costs
<code>bc_bake hide, bc_bake show</code>	on a player's game: stops and starts drawing baked pieces (they stay solid)
<code>bc_bake help</code>	the list of commands

Where the files are

File	Where
the layer	inside the world's <code>BetterContinents</code> settings file, in the world's folder
undo files	<code>BetterContinents/bakes/<world name>-<world id>/</code> in Valheim's save folder, on the machine that runs the world
a layer kept before a load	the same folder, <code>layer-r<revision>.bcp</code>
a player's copy of the layer	<code>BetterContinents/cache/</code> in Valheim's save folder, on each player's machine

Appendix B

placements.bcp for tool makers

This page is for people who write tools that make town files. Players can skip it.

`placements.bcp` is *format 1* of a baked layer: one file for the whole world, little-endian, with a CRC-32 at the end. The full layout, every field and every check a reader makes, is in `BakedFormat.cs` in Better Continents' source, whose comments are the reference. In outline:

Part	Holds
header (32 bytes)	BCPL, format 1, header flags (ground, paint, clear masks, registry), revision, counts
producer	a short text naming the tool that wrote the file
palette	one entry per kind, world-wide: up to 8 candidate prefab names, each with its anchor; a role; a collision kind; a Unity layer; flags; a tint; tags
zone index	one 28-byte row per 64 m zone, sorted by z then x: where its block is, how many records, its Live count, its flags, its height range
zone blocks	one raw Deflate stream per zone: the records column by column, then the zone's clear mask, ground and paint
trailer	CRC-32 (zlib's) of everything before it

Records. Each record is a palette index, x and z inside the zone (1/1024 m), y in millimetres, and a yaw in 65,536 steps; optional runs add a full rotation, a scale and an id. Positions are the piece's *anchor* point; the palette's anchor offsets turn them into the prefab's pivot.

Roles.

Role	Better Continents
0 Static	draws it and gives it a collider; it is never an object
1 Live	makes it a real, protected object when its zone first generates, with the record's id and the entry's tags
2 Copy	draws it unlit and gives a collider; within Light Distance, the Light Count nearest become lit copies of the piece
3 Seat	draws it and gives a collider; within Seat Distance, a chair to sit on

Palette flags (one byte per entry): 1 UniformScale, 2 NoShadows, 4 Protect, 8 LOD0Only, 16 NoCopyLight, 32 Decor. A file with a flag Better Continents does not know is refused.

Pieces that give items (any prefab with a Pickable, PickableItem, ItemDrop or Plant) are placed once as ordinary objects when their zone generates, whatever their role. Write them as Static records. To make such a kind scenery instead, set flag 32 (Decor) on its palette entry: Better Continents then draws its records like any other Static or Copy record and places no object for them (nothing to pick, no regrowth). Only a Static or Copy entry may carry it: a Live or Seat entry with flag 32 makes the whole file refuse to load (*it is marked decor (flag 32), which only a Static or Copy entry can be*). Give decor kinds collision 0 (None) or Boxes: a pickable's own colliders (collision 3) are often triggers. A world that already placed a kind as real objects keeps them when a newer file marks the kind decor, until their zone is reset or a player picks them.

Zone sections. A clear mask (16 x 16 cells of 4 m) keeps the game's vegetation out; zone flag 16 keeps it out of the whole zone. Ground (65 x 65 vertices, 1 m apart, with a weight each) replaces the world's height; paint (64 x 64 cells) lays dirt, paving or cultivated soil. Extras are kept as they are, for the tool's own use.

What Better Continents adds when it changes a layer. Header flag 8 and a registry of in-game bakes; record flags 8 (the bake a record belongs to, and its saved values) and 16 (a random-material seed). A tool's own file never sets these. A file with a header flag Better Continents does not know is refused with a

line naming it, so a newer tool's file never loads half-read.

Checks. Better Continents reads every byte before it uses a file: the CRC, the header, every palette entry, the index's order, each block's counts and runs, the Live ids (unique, and present on every Live record), and the totals. The first problem stops the read, and the log names the zone and record.

Testing your file. Put it in a maps folder and make a new world (chapter 4), or load it into a running world with `bc_bake load <file>`: the dry run reports its counts without changing anything. `bc_bake check all` then reads every zone, and `bc_bake export` writes back what Better Continents holds, for a byte-for-byte comparison.

Appendix C

GNU General Public License, version 3

The GNU General Public License, version 3, reproduced verbatim from LICENSE.

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<https://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require,

such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to

"keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or

requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this license. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this license (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically

receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered

work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>
```

```
This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.
```

```
This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
GNU General Public License for more details.
```

```
You should have received a copy of the GNU General Public License
along with this program. If not, see <https://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program> Copyright (C) <year> <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <https://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <https://www.gnu.org/licenses/why-not-lgpl.html>.