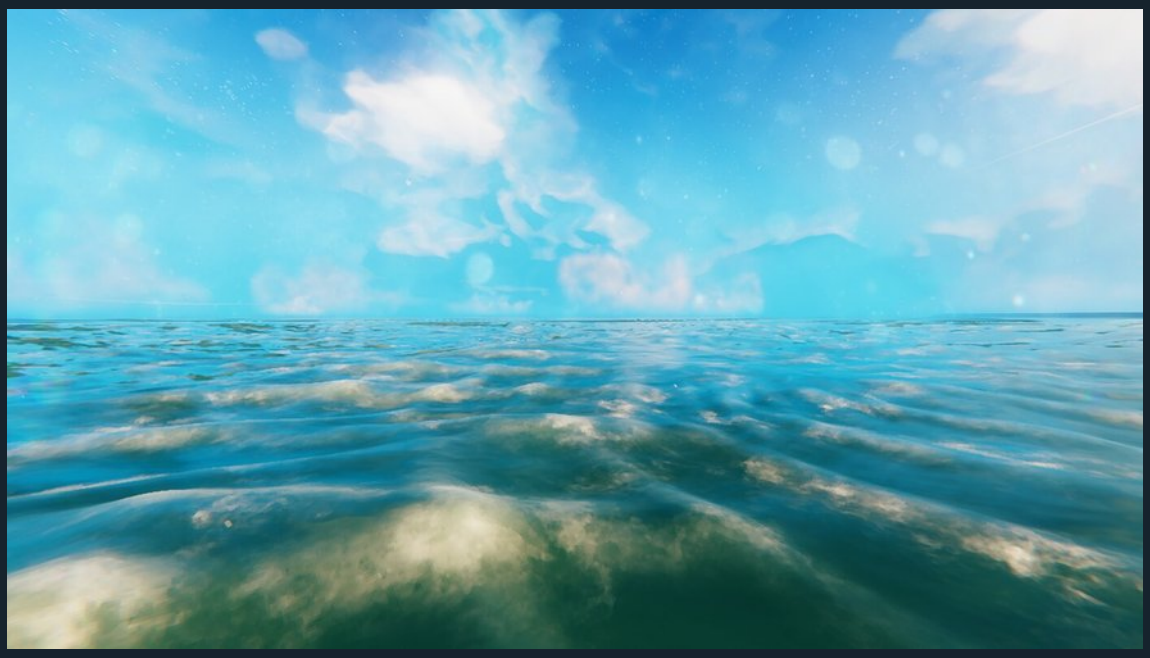


Water and lakes

the easy guide to Better Continents 0.10.x

Waves to the edge of a big world, and lakes above the sea.

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, server owners and map makers. Lakes and clearing are experimental.

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 (water-guide-source.zip). The cover: the open sea 35.75 km from the middle of a test world.

Contents

Start here: what changes in the water . . . 4
Who gets it, and who does not . . . 4
What it costs . . . 5
Lakes and clearing (experimental) . . . 5
Where to go next . . . 5

License . . . 6

1 The waves and the storm . . . 7
Near water and far water . . . 7
The waves . . . 7
It looks like the game's water . . . 7
The storm . . . 8
What you see is what you feel . . . 9
The world's edge . . . 9

2 Settings . . . 10
Own Water . . . 10
Soft Far Sea . . . 10
What these settings never do . . . 11
The game's own graphics settings . . . 11
The lake settings . . . 11

3 The console: bc_water and bc genloc . . . 12
bc_water: what draws the sea, and why . . . 12
Two words for testing (Debug Mode) . . . 13
bc genloc: make the world's locations again . . . 13

4 When it goes wrong . . . 15
The game's water is drawn, and you expected Better Continents' . . . 15
Another mod changes the waves . . . 15
Graphics Overdrive and other graphics mods . . . 16
Expand World Size and Expand World Data . . . 16
Dedicated servers . . . 16
Other versions of Better Continents . . . 16
What to send with a bug report . . . 16

5 Lakes (experimental) . . . 17
What a lake is . . . 17
Make one in the game: bc_lake here . . . 18
Look after your lakes . . . 18
How deep a lake looks: Lake Depth Scale . . . 19
What a lake is to the water rules . . . 19
In multiplayer . . . 19
Settings: [12 BetterContinents.Lakes] . . . 20
What lakes do not do (yet) . . . 20

6 Lake files, and worlds from Tarn (experimental) . . . 21
Where it goes, and when it is read . . . 21
The format, line by line . . . 21
bc_lake load: a file into a running world . . . 22
bc_lake export: the world's lakes to a file . . . 22
A world from Tarn . . . 22
The Tarn mod installed beside Better Continents . . . 23

7 Clearing (experimental) . . . 24
Where it goes . . . 24
The file . . . 24
Make one with Python . . . 25
The mask in a running world . . . 25
The other way: a white vegetation map . . . 26

Appendix A Commands and settings . . . 27
bc_water (any player) . . . 27
bc genloc (the machine that runs the world, cheats on) . . . 27
bc_lake (experimental) . . . 27
Settings . . . 28
Files . . . 29

Appendix B For map makers . . . 30
The sea follows your maps . . . 30
Lakes (experimental) . . . 30
Clearing (experimental) . . . 30
What ships with the world . . . 31

Appendix C GNU General Public License, version 3 . . . 32

Start here: what changes in the water

Valheim's water shader was written for a world 10 km across. Two circles of that world are written into its code: past about 12 km from a point 4 km south of the middle the sea is drawn **calm** (the game's Deep North sea), and past about 12 km from a point 4 km north of the middle it is drawn **red** (the game's Ashlands sea). No setting moves them. On a bigger world, almost all of the sea lies past those circles, so it was drawn calm, and the far sea looked flat however hard the wind blew.

Better Continents 0.10.x draws that sea with **its own water**: the game's look, but with the world's own rules.

- **Waves to the edge.** Full waves and white crests out to the edge of the world, however big it is.
- **Calm only in the world's own Deep North.** Where your world's Deep North is (its biome map, or the game's own ring scaled to the world's size), the sea is calm, and the waves die down over its first 200 m, as the game's do.
- **Red only where the world's hot sea is.** The Ashlands' red is drawn where your world's hot sea is (its heat map, or the ring scaled to its size), with the game's soft, foamy 25 m edge.
- **What you see is what you feel.** Boats and swimmers ride the waves that are drawn, on every machine, a dedicated server included. In the tests the drawn and the felt sea agreed to within 4 mm.

It works **in place**: no new world data, nothing to make again. An existing world gets it the moment it loads.



Screenshot: the open sea 19.5 km from the middle of a world that reaches 65 km from its middle, seen from 2 m above the water at the same moment. Left: Better Continents' own water, with waves and crests. Right: the game's own water shader at the same spot, which is what earlier versions of Better Continents showed there: a calm sea.

Who gets it, and who does not

Your world	Its sea
A Better Continents world with a biome map or a heat map, or made since 0.10 at a size other than the game's (or sized by Expand World Size)	Better Continents' own water, whenever its sea differs from the game's
A Better Continents world without maps, at the game's own size (World Size 10,000, Edge Size 500)	the game's water, exactly
A world made without Better Continents (vanilla), or a world that keeps the game's own terrain	the game's water, exactly
A world with Better Continents switched off (the Disabled preset)	the game's water, exactly

Where nothing in the world's rules differs from the game's, nothing changes: Better Continents compares its map of the sea with the game's two circles, and when they agree it leaves the game's water alone.

What it costs

Nothing is sent when a player joins. Each machine works out the world's sea from the world's settings, on its other processor cores, while the world loads: under a second for a world 130 km across, measured on one machine with many cores (fewer cores take longer). In the tests the frame time was the same with Better Continents' water and with the game's (on one machine, whose screen held both at 90 frames a second).

Lakes and clearing (experimental)

> **Watch out: Lakes and clearing are experimental for now.** They come from **Tarn**, the mod by Raven Iron (Wubarrk and NomadicWar). They work in the tests and on test worlds, but details may still change. **Try them on a copy of a world first.**

Valheim has one water level, so every lake in the hills is a dry hollow. A world can now carry a list of **lakes**, each at its own height, filled with the game's own water: you swim in them, sail on them, and see them on the map. A **clearing mask** keeps trees, rocks and bushes off ground a map maker marked, such as roads and lake beds. Chapters 5 to 7.

Where to go next

1. **What the waves do,** in calm weather and in a storm: chapter 1.
2. **The two settings** (Own Water and Soft Far Sea): chapter 2.
3. **The console:** `bc_water` says what is drawn and why; `bc genloc` makes a world's locations again and reports what changed: chapter 3.
4. **Something looks wrong:** chapter 4.
5. **Lakes** (chapters 5 and 6) and **clearing** (chapter 7), both experimental.

Appendix A lists every command and setting; appendix B is one page for map makers.

New in Better Continents 0.10.5 (2026-10).

License

Better Continents: Water and lakes, 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 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 formats and messages were copied from the code, and the examples of command output are lines the game wrote in automated tests. 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 4).

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). Lakes and clearing come from **Tarn**, by **Raven Iron** (Wubarrk and NomadicWar), contributed to Better Continents under the LGPL-2.1; their file formats are Tarn's. 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 setting names quoted here are the game's own English texts, quoted so that you can find them.

Attachment. The file `water-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_water_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 The waves and the storm

In short: in a world that Better Continents shapes, the sea has the game's waves everywhere, calm only in the world's own Deep North and red only where its own hot sea is. It looks like the game's water, and boats and swimmers feel exactly the waves that are drawn.

Near water and far water

The game draws its sea in two parts, and so does Better Continents:

	Near you	Farther out
What it is	the water of each 64 m zone loaded around you	one flat plane that follows you to the horizon
How far	to about 130 m at the game's usual settings	from there to the horizon
Waves	real waves: the surface moves up and down, and the game's Tessellation setting adds detail	flat, with the game's moving ripple texture, as in the game
Depth colour	from the depth of the sea under each zone	deep-sea colour, or the depth under it with Soft Far Sea (chapter 2)

So a storm's swell is a real swell near you, and the sea far away is the game's flat plane with its ripples, as it has always been. What changed is that the near water now has its waves wherever the world says it should.

The waves

In open sea, the waves are the game's own waves: the same ten waves, the same heights and speeds, growing and falling with the wind. Better Continents changes only **where** the sea is calm and where it is red. Two things decide that, both from the world's own rules, the ones the weather, a swimmer's heat and a ship's burn already follow:

- **The world's Deep North** (the biome map's Deep North, or on a world without one, the game's ring scaled to the world's size). There the sea is calm. The waves die down over the first 200 m into it, as the game's do, and a small ripple stays.
- **The world's hot sea** (the heat map, or on a world without one, the game's Ashlands ring scaled to the world's size). There the water is red.

Everywhere else the sea has full waves, out to the world's edge.

> **Good to know:** a world without a Deep North has no calm sea at all. For the 130 km test world of these pictures, `bc_water` reported *0.0 % of the world is Deep North (calm), 0.0 % is hot sea (red)*: waves everywhere. A world of World Size 24,500 made without maps reported 19 % calm and 19 % hot: the game's two rings, scaled to that world.

It looks like the game's water

The colours, the glare, the foam, the refraction, the fog and the depth colours are the game's. The test below put Better Continents' water on a world where it would not normally be used, a world of the game's own size (with `bc water force on`, chapter 3), so that it could be compared with the game's own water at the same places and the same moment.



Screenshot: the edge of the red sea, from 60 m up, in a world of the game's own size. Left: Better Continents' water; right: the game's, at the same place and the same moment of the water's clock. Both draw the red with the same broken edge.

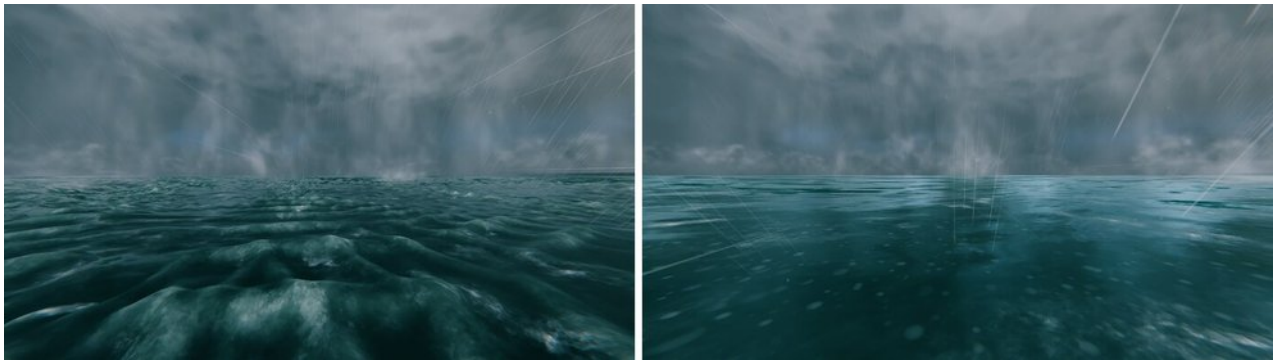


Screenshot: the calm sea 337 m inside the game's Deep North ring of the same world, from 30 m up. Left: Better Continents' water; right: the game's.

One thing is left out on purpose: the game draws large pale **bubble patches** on what it takes for the Ashlands sea, and on a big world that was most of the sea. Better Continents' water draws none; its hot sea is red, with the same foam and waves as any sea.

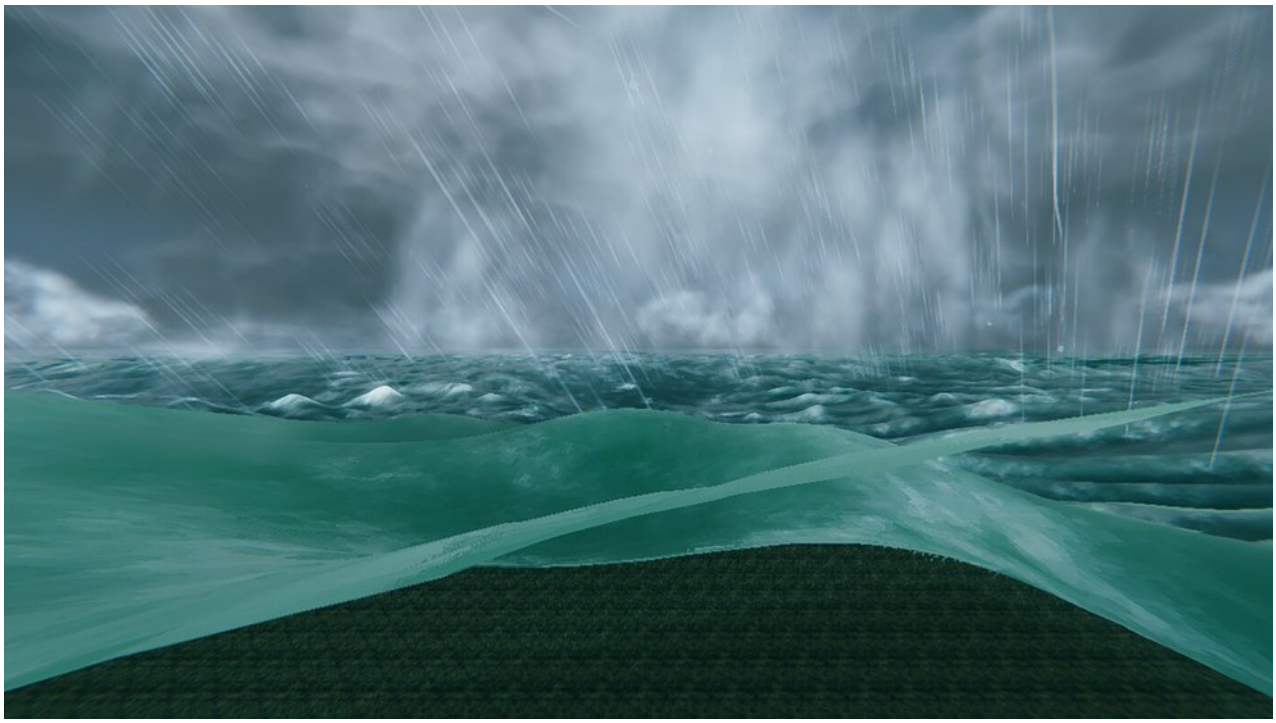
The storm

In a storm the wind is at its strongest, and so are the waves. Past the game's circles, the game drew a calm sea even then.



Screenshot: a thunderstorm at full wind, 35.75 km from the middle of the test world, 2 m above the water. Left: Better Continents' water, a real swell. Right: the game's water shader at the same place: calm.

At that spot the surface under the camera rose and fell 1.6 m in four seconds of the storm. Near the world's edge it was more:



Screenshot: the same storm 61.75 km from the middle, near the edge of the test world. The camera stands 2 m above the level of still water, and the wave in front of it rises higher. The surface under the camera moved 3.3 m in four seconds.

What you see is what you feel

Boats, swimmers and anything that floats ask the game where the water's surface is. Better Continents answers with the same waves its water draws: the same wave sum, the same calm and the same clock. So a boat rides the swell you see, and sits still on a calm sea you see as calm.

- **On every machine.** The answer comes from the world's settings, which every player has, and never from a player's graphics. A dedicated server answers the same way, without drawing anything.
- **Measured.** A probe in the game (`bc water probe`, chapter 3) compares the drawn surface with the felt one at 81 places in front of the camera. In the test world, at four distances out to 61.75 km and in storms, the largest difference was 4 mm.
- **With Own Water off** (chapter 2) a player sees the game's water, but boats still ride the world's waves. That is the one case where drawn and felt differ, and it is that player's choice.

> **Good to know:** in the game itself, boats on the calm sea feel its small ripple about ten times as strongly as it is drawn. With Better Continents' water the calm sea feels as calm as it looks.

The world's edge

At the edge of the world the water drops away as before, and the last 200 m of sea before it carry a band of foam. Lakes near the edge (chapter 5) do the same.

2 Settings

> **In short:** two settings in `[11 BetterContinents.Water]` of `BetterContinents.cfg`, **Own Water** and **Soft Far Sea**. Both are on by default, both are each player's own, and both apply at once. Neither changes the world, and neither changes the waves that boats feel.

Change them in Configuration Manager (F1) or in the config file, while you watch the sea.

Setting	Default	What it does
Own Water	true	Draws the sea with Better Continents' own water in worlds whose sea is not the game's: waves out to the world's edge, calm only in its own Deep North, and the Ashlands' red where its hot sea is. Off: the game's water.
Soft Far Sea	true	With Own Water: the far sea past the loaded zones takes the depth of the sea under it, so that it looks like the near water where they meet. Off: the far sea is drawn as deep sea everywhere, as the game draws it.

Own Water

On (the default), your game draws Better Continents' water wherever chapter 1 says it should. In a world whose sea is the game's (a vanilla world, or a Better Continents world without maps at the game's size), it draws nothing of its own either way.

Off, your game draws the game's own water shader everywhere, and Better Continents goes back to what versions 0.10.3 and 0.10.4 did with it:

- the game's shader would paint red all the sea past its own ring; Better Continents sets the colours of each 64 m zone of water to the world's own hot sea instead, so that red shows only where the world is hot (in steps of 64 m, and the far sea in one colour);
- on a world that Better Continents shapes, the game's pale bubble patches are left out too;
- the sea past the game's calm circle is drawn calm, as the game draws it.

Boats and swimmers **still ride the world's waves** with Own Water off: what floats follows the world, never one player's choice of how to draw it. So with Own Water off, a boat may rock on water your screen shows as calm.

Turn it off to compare, or if your graphics device has trouble with it. The other players are not affected.

> **Good to know:** your game also draws the game's water by itself, whatever this setting says, when it cannot draw Better Continents' water: chapter 4 lists when, and the log says why.

Soft Far Sea

Seen from high up, the sea is two things (chapter 1): the near water of the loaded zones, coloured by the depth of the sea under each one, and the far plane beyond, which the game always colours as deep sea. Over a shallow coast the two meet in straight light and dark strips, the edges of the loaded zones. The game's own worlds show the same strips on any coast seen from high up; big worlds only make long flights more common.

With **Soft Far Sea** on, the far plane takes the depth of the sea under it from a grid of points 64 m apart, by the same rule the near water uses, so where the two meet they have the same colour. It only applies while Own Water draws the sea. Off, the far plane is deep sea everywhere, as in the game.

What these settings never do

- **They never change the world.** They are not saved in it, not exported, not imported, and not sent to other players.
- **They never change the waves that are felt.** Every machine works out the felt waves from the world's settings alone.
- **They never need a restart** or a new world: the sea changes the moment you change them.

The game's own graphics settings

- **Tessellation** (the game's graphics setting) chooses between a finer and a simpler drawing of the near water, for Better Continents' water exactly as for the game's.
- Graphics mods that tune the game's water material keep working: chapter 4.

The lake settings

Lakes have their own section, `[12 BetterContinents.Lakes]`, described in chapter 5: **Lake Depth Scale** (the one world setting), **Lake Build Range**, **Lake Build Budget**, **Lake Sound** and **Lakes On Map**.

3 The console: bc_water and bc genloc

> **In short:** `bc_water` tells you whether Better Continents draws the sea in this world, and why. Any player can type it, with no cheats. Two more words, for testing, need Debug Mode. `bc genloc` makes a world's locations again, as the game's own `genloc` does, and tells you what it found and what changed.

Open the console with **F5**.

bc_water: what draws the sea, and why

`bc_water` only reads: it changes nothing, so every player may use it on their own game, a player connected to a server included. It answers with a few lines that start with `Water:`. In the 130 km test world of chapter 1 it said:

```
Water: BC's sea is a candidate for this world (the world has a biome map, a heat map,
  a world laid out to its own size).
Water: the sea field: on.
Water: 4096 x 4096 texels of 31.738 m over +-65000 m, built in 760 ms on 31 thread(s).
Water: 0.0 % of the world is Deep North (calm), 0.0 % is hot sea (red), its Deep North
  is not the game's circle, its hot sea is not the game's ring.
Water: boats, swimmers and fish feel the world's waves (the field's W, the drawn clock).
Water: the far sea's depth grid: 2034 x 2034 texels of 64 m.
Water: drawn with BetterContinents/Water: 0 zone water surface(s) and 26 far sea
  (LOD 250). Soft Far Sea is on.
Water: 0 material value(s) did not survive a shader change and were put back (none:
  Unity keeps them by name); the zone depth arrays, which no shader declares, were put
  back 1 time(s).
```

Each line is one line on the console; the long ones are broken here to fit the page.

Line	What it tells you
a candidate … (why)	whether this world can have Better Continents' sea, and what makes it one: a biome map, a heat map, a size of its own, or Expand World Size
the sea field	the map of the world's sea that every machine works out at load: whether it is ready, its size, and how long it took
% Deep North, % hot sea	how much of the world is calm and how much is red, and whether either is the game's own circle
boats, swimmers … feel	whose waves are felt: the world's, or the game's own where the world's calm sea is the game's
drawn with …	what draws the sea on this machine now; when it is the game's water, the line says why instead
another mod … patches …	appears only when another mod changes the waves; see chapter 4

In a world whose sea is the game's, the answer is short. In a world of the game's size made without maps:

```
Water: BC's sea is not a candidate for this world (a vanilla-size world without maps:
  the game's own sea).
Water: the sea field: off: a vanilla-size world without maps: the game's own sea.
Water: the game's own water is drawn: this world's sea is the game's.
```

With Own Water off, the last line says *the game's own water is drawn: Own Water is off.*

Two words for testing (Debug Mode)

These two need **Debug Mode** on the machine that types them (`Debug Mode = true` in `[00 BetterContinents.Debug]` of `BetterContinents.cfg`). On the machine that runs the world they are also in the cheat tree, as `bc water force on` and `bc water probe`, with cheats on.

`bc_water force on` draws Better Continents' water in a world where it would not (a world of the game's own size without maps, or one whose sea is the game's), so that you can compare it with the game's own water at the same places: chapter 1's red-edge and calm-sea pictures were made that way. It changes only the drawing on your own game; the waves that are felt stay as they are. It is not saved: `bc_water force off`, or leaving the world, gives the rules back.

```
Water: force is ON: BC's sea is a candidate here whatever the world's rules say, its
  field is built from the world's rules and BC's shader is drawn even where the sea is
  the game's own; the felt waves stay as they are. ...
```

`bc_water probe` measures, at 81 places of the water in front of the camera, how far the drawn surface is from the surface that boats feel, and writes one line to the console and the log:

```
Water probe: n=81 median=0.000 p99=0.002 max=0.003 (drawn minus felt, metres);
  camera 30960.4 -17875.0
```

`n` is how many of the places were used, then the middle, the 99th-percentile and the largest difference, in metres. A few millimetres is what the tests measured. If it cannot measure, it says why: `Water probe: cannot: ...` (for example where Better Continents' water is not drawn: `bc_water force on` first).

bc genloc: make the world's locations again

The game's own `genloc` command makes a world's locations again. `bc genloc` runs the same generation and adds:

- **a report:** what it found when it starts, and what changed when it ends;
- **the new location icons**, sent to every player in the world when it ends (the game sends none after a `genloc`);
- **a stuck run freed:** when the game's last run threw an error midway, it leaves its slot taken, and the game's `genloc` then does nothing at all and says nothing. `bc genloc` sees that nothing is running there, frees the slot, and starts a new run;
- `bc genloc alt`, the game's own alt-biome branch;
- `bc genloc pins`, which puts only the location map's pins and the start position back into the game's location list (what Upgrade World's `locations_restore` can leave out), and makes no locations.

```
bc genloc         makes the locations again; says what it found, and what changed
bc genloc alt     the game's own alt-biome branch (the same as 'genloc alt')
bc genloc pins    only the location map's pins and the start position
bc genloc help    a short help
```

It is part of the `bc` cheat tree: it runs on the machine that runs the world (single player, or the host's game), with cheats on (`devcommands`; Debug Mode turns them on when you spawn). On a dedicated server it has to be typed into the server's own console. A player connected to someone else's world is told to run it there.

What the generation does. Every location that is not placed yet is cleared and rolled again, in the zones that are not generated yet. Locations already placed stay, and so does everything in the zones already generated. The new ones appear as players reach their zones. It starts with a line like this one:

```
Genloc: starting. 33 location(s) now: 33 placed (kept, with every zone that is
  generated), 0 not placed (cleared, then rolled again in the zones that are not
  generated yet). The new ones are placed as players reach their zones. Stay in the
  world until it says it is done: a save in the middle keeps only the placed ones.
```

and ends, seconds or minutes later, with one line that counts what changed:

```
Genloc: done in 23.5 s: 33 -> 12315 location(s), 33 placed ones kept, 12282 to be
  placed as players reach them; types that changed: AbandonedLogCabin02 0->33,
  AbandonedLogCabin03 0->33, ... ; the location icons were sent to the clients.
```

It also lists the locations that are still short of their number (*short of their quantity: SwampRuin2 15/30, …*), the same ones the game warns about, and writes a table by type to the log. The game stays playable while it runs.

> **Watch out: stay in the world until it says it is done.** A save in the middle of the run keeps only the placed locations: in a test, a world that was left in the middle came back with 33 of its 12,308 locations, and `bc genloc` then made the rest again. Type `bc genloc` a second time while it runs and it only says how far it is.

4 When it goes wrong

> **In short:** when your game cannot draw Better Continents' water, it draws the game's water instead, and the world is unchanged for everyone else. `bc_water` says which water is drawn and why; the log (`BepInEx/LogOutput.log`) has a line that starts with `Water:` for every reason that is a failure or a choice.

The game's water is drawn, and you expected Better Continents'

Type `bc_water` (chapter 3). Its last line says *the game's own water is drawn:* and the reason:

It says	What it means, and what to do
this world's sea is the game's	The world is a vanilla world, a Better Continents world at the game's own size without maps, a world that keeps the game's own terrain, or one whose calm and red seas come out exactly as the game's circles. Nothing to do: the game's water is right there. The first line of `bc_water` says which.
the sea field is not ready yet	The world has just loaded, and the map of its sea is still being worked out on your other cores (under a second on a fast machine, longer on a slow one). Wait a moment.
Own Water is off	Your own choice (chapter 2).
another mod has replaced the game's water shader (...)	A graphics or water mod draws the sea with its own shader. Better Continents steps aside on your game and leaves that mod's water alone.
the water shader is not supported on this graphics device	Your graphics device cannot run the water. A Mac is one: the water is built for Direct3D 11, Vulkan and OpenGL Core, not for Metal.
this graphics device has no two-channel 8 bit texture format (RG16)	Your graphics device lacks a texture format the water needs.
the water bundle could not be loaded (...), this build of Better Continents carries no water shader	The water's shader file inside `BetterContinents.dll` did not load. Reinstall Better Continents from its package.

In every one of these cases the waves that are **felt** stay the world's: your boat still rides the waves the other players see. Only your screen shows the game's water.

> **Good to know:** the tests ran on Linux, with Vulkan. The Direct3D 11 programs that Windows uses are in the same shader file, built and checked with the others. If your Windows game draws the game's water, `bc_water` and the log say why; please report it.

Another mod changes the waves

A mod that changes how the game works out its waves can make what is drawn and what is felt differ. Better Continents does not fight it. It says so in the log once, and in `bc_water`:

```
Water: another mod, <its name>, patches <the game's method>: the drawn waves may not
  match what boats feel.
```

If boats then rock on water that looks still, or sit still on drawn waves, that mod is the place to look.

Graphics Overdrive and other graphics mods

Better Continents' water keeps every setting of the game's water material under its own name, so mods that tune those settings keep working. Wubarrk's **Graphics Overdrive** was installed for the tests: its water dials (tessellation, normals, refraction) and its fog apply to Better Continents' water as to the game's, and its ray-traced water follows the world's calm sea. A mod that replaces the water shader altogether is the one case where Better Continents steps aside (above).

Expand World Size and Expand World Data

Better Continents' water asks the world where its Deep North and its hot sea are through the game's own functions, which those mods change too. So the calm and red seas are drawn where they put them, and a world sized by Expand World Size always gets Better Continents' sea (when its sea differs from the game's).

Dedicated servers

A dedicated server draws nothing, so it loads no shader and makes no textures. It still works out the calm part of the world's sea, once the first zone water starts on it, and answers every question about the waves exactly as the players' games do. A boat simulated by the server rides the same waves as a boat simulated by a player.

Nothing about the water is sent to players: each machine works it out from the world's settings, which every player already gets when joining.

Other versions of Better Continents

- The water adds **no data to the world**. A world without lakes or a clearing mask (chapters 5 to 7) still opens in Better Continents 0.10.4, with that version's water.
- Every player in a world needs the same Better Continents version as the machine that runs it, as always: a join with another version is refused.
- A world **with** lakes or a clearing mask needs 0.10.5 or later (chapter 5).

What to send with a bug report

- The log, `BepInEx/LogOutput.log`, from the session where it happened (its `Water:` lines say what the water did).
- What `bc_water` says, in the same session.
- Where you were (the coordinates, or a screenshot with the map open), and the weather.

5 Lakes (experimental)

> **Watch out: lakes are experimental for now.** They work in the tests and on test worlds, but details may still change, and a world that carries lakes can only be opened by Better Continents 0.10.5 or later. **Try them on a copy of a world first:** copy the world's folder (`worlds_local/<world name>/` in Valheim's save folder) somewhere safe before you add lakes, and keep the copy until you are happy.

Valheim has one water level, the sea's, at a height of 30 m. A hollow in the hills stays dry however deep it is. A world can now carry a **list of lakes**, each with its own water level, and Better Continents fills each one with the game's own water.

Lakes come from **Tarn**, the mod by Raven Iron (Wubarrk and NomadicWar); its lake files work unchanged (chapter 6).

What a lake is

A lake is one line of the world's lake list: a **name**, a **point** in the water, the **level** of its surface, and a **box** it may fill. Better Continents floods the ground from the point: every place lower than the level that joins it, inside the box (and 12 m around it), becomes water. The water is the game's own:

- you **swim** in it and come out **Wet**, a boat **floats** and sails on it, and the view under its surface is the game's underwater view;
- it is drawn on the **minimap and the large map** as water (a lake of only a pixel or two of the map may be left off);
- a quiet **water sound** plays near it, rising with the wind.



Screenshot: a lake made by `bc_lake here`'s spill search in a hollow about 50 m above the sea (its level 80.41 m, 16.9 m deep at the deepest), in a test world made without maps. The trees in the water stand there because nothing cleared the lake's bed: a lake does not clear it by itself (chapter 7).

Lakes are built as you come near and taken down behind you, a few milliseconds of each frame, so a world can hold many of them (up to 5,000). A dedicated server builds none: it keeps the list and sends it to the players.

Make one in the game: bc_lake here

Stand in the low part of a dry hollow, open the console (**F5**) and type:

```
bc_lake here
```

Better Continents looks at the ground within 120 m of you, finds the height at which water would spill out of the hollow, and fills it to 0.3 m under that. It names the lake (`lake1`, `lake2` and so on), adds it to the world's list, and says what it made: its level, where it would spill, how deep it is and how much water it holds. The water appears within moments.

```
lake1: level 80.41 m (spills at 80.71), deepest 16.9 m, 30992 m2 wet. Saved with the world at
  its next save; sent to 0 players.
```

Written from the code, with the numbers of a lake that the same spill search made in a game test.

Words	What they change
`bc_lake here 200`	looks 200 m around you instead of 120 m (8 to 1,000 m), for a bigger hollow
`bc_lake here 120 1`	fills to 1 m under the spill point instead of 0.3 m (0 to 20 m): a lower lake with more shore
`bc_lake at <x> <z> [radius] [margin]`	the same at a place you name, without standing there

It refuses, and says why, when there is no lake to make:

It says	What to do
this hollow drains to the sea: it spills at … m, the sea is 30.0 m	The hollow is open to the sea, or its rim is within 0.2 m of it: no lake can stand there.
you are not in a hollow: it would spill at … m and the ground here is higher. Stand in the low part.	Move to the lowest part of the hollow and try again.
the water still reaches the edge of the hollow after the level was lowered … times by 0.2 m …	The rim has a gap. Try a smaller margin, a bigger radius, or another place.

> **Good to know:** lakes are built on a grid of 4 m. When a thin gap in the rim lets that grid's water out, `here` lowers the level 0.2 m at a time (ten times at most) until the lake holds, and says so.

A lake is saved with the world at its next save (`save` in the console saves at once) and sent to every player in the world at once.

Look after your lakes

Command	Does
`bc_lake list [n]`	the n nearest lakes (10 by default): name, point, level, distance, and whether it is built
`bc_lake info`	how many lakes the world has, its depth scale, the nearest three, whether any lie in the Deep North or the hot sea, and the clearing mask
`bc_lake remove <name>`	takes that lake out (every lake of that name)
`bc_lake remove all`	a dry run that says how many would go; add `confirm` to remove them all
`bc_lake depth [scale]`	shows, or sets, how deep lakes look (below)

Before `remove` changes anything, the old list is written to `lakes-before-<time>.txt` in `BetterContinents/<world name>/` in Valheim's save folder, so nothing is lost: to bring it back, copy it as `lakes.txt` into a folder of its own and load it (`bc_lake load <that folder> confirm`, chapter 6). When the lakes change, the map is drawn again, which may pause the game for a moment.

How deep a lake looks: Lake Depth Scale

The game's water takes its colour, its foam and the size of its waves from how deep it is: full colour and full waves at 10 m. Lakes would all look like open sea by that rule, so their depth is scaled down: **Lake Depth Scale** is the share of a lake's real depth that the water takes it to be. At the default 0.5, a lake looks and moves like sea only where it is 20 m deep, and is calmer and greener where it is shallower.

```
bc_lake depth          shows the world's scale
bc_lake depth 0.3      calmer and greener (0 to 3; try 0.2 to 1)
```

It is **one value for the whole world**, kept with its lakes, because it also sets the waves that boats and swimmers feel in a lake: every player must agree on it. `bc_lake depth` changes it at once for everybody. The setting of the same name in the config only gives a **new** world its first value.

What a lake is to the water rules

A lake follows the same rules as the sea (chapter 1):

- **In the world's Deep North it is calm** (that is how the code works; no game test has seen a lake there yet). Elsewhere it has the sea's waves, scaled by its depth (above).
- **In the world's hot sea it is red, and it burns** (how the code works; not yet seen in a game test): swimmers and boats in it take heat damage, as at sea. `bc_lake here`, `load` and `info` say so when a lake lies there.
- **With Own Water on,** Better Continents' water draws the lake; with it off, the game's does. Either way boats feel the lake's waves as drawn by Better Continents' rules. In the tests the drawn and the felt surface of a lake agreed to within a few centimetres in a strong wind.
- Lakes never change the sea: the sea's calm and red parts come from the world's rules alone.

In multiplayer

- The lake list is part of the world's settings: every player gets it when joining (and keeps a copy for the next join), and every change is sent to the players in the world at once.
- **Every player needs Better Continents 0.10.5 or later**, the same version as the machine that runs the world.
- Only the machine that runs the world changes lakes: the player in single player, the host in a hosted game. On a dedicated server an admin (on the server's `adminlist.txt`) puts `server` first: `bc_lake server here` makes a lake where that admin stands. Any player may use `list` and `info`, and the export words where the server allows exports.
- A player who also has the **Tarn** mod installed sees dry hollows where the others see lakes: chapter 6.

Settings: [12 BetterContinents.Lakes]

Setting	Default	Range	What it does
Lake Depth Scale	0.5	0 to 3	How deep the water takes a lake to be, as a share of its real depth, for the lakes of a **new** world. An existing world keeps its own (`bc_lake depth`).
Lake Build Range	400	150 to 2000 m	Lakes within this many metres of you are built (their water and the places you can swim in it), and taken down again 150 m further out.
Lake Build Budget	2	0.5 to 20 ms	Milliseconds of each frame your game spends making the water of lakes.
Lake Sound	0.35	0 to 1	Volume of the water sound near a lake (the game's ocean loop, quieter). 0 = off.
Lakes On Map	true		Draws the world's lakes on the large map and the minimap.

All but the first are each player's own and apply at once.

What lakes do not do (yet)

- **A lake does not clear its own bed.** Trees, rocks and bushes that stood in the hollow stand in the water. A clearing mask over the lake beds keeps them out of zones generated after it (chapter 7).
- **Grass in a lake bed** follows the ground's paint, not the water: on a map, paint the beds as dirt, or grass shows through the water.
- **A lake follows the world's generated ground,** not ground that players dug, raised or levelled.
- **An older Better Continents** cannot read a world with lakes: it opens it as a world without Better Continents, with the game's own terrain. Nothing is lost: the lakes stay in the world's files and come back with 0.10.5 or later. Do not let an older version save the world, and keep your copy.

6 Lake files, and worlds from Tarn (experimental)

> **Watch out: lakes are experimental for now:** details may still change. **Try a lake file on a copy of a world first**, and keep the copy until you are happy.

A map maker gives a world many lakes at once with a **lake file**, `lakes.txt`. It is Tarn's own format, so a file made for Tarn works as it is.

Where it goes, and when it is read

Put `lakes.txt` in your **maps folder**, the folder the `Directory` setting names (beside `heightmap.png` and the other maps). The name `tarn-lakes.txt` is taken too, when there is no `lakes.txt`; if both are there, `lakes.txt` wins and the log says so.

- **A new world** made **From Config** (or with `bc_import`, or from a world export's folder) reads it when it is made, and keeps the lakes in its own settings from then on. The file is not read again.
- **A world that already exists** takes the file with `bc_lake load` (below).

The log says what was read: `Lakes: <path>: 172 lakes`, and a warning for each line it could not use.

The format, line by line

One lake per line, nine fields separated by semicolons:

```
# Tarn lakes: name;x;z;level;minX;minZ;maxX;maxZ;depth
Vikedal1;-4210.5;3190.25;92;-4420;3050;-4010;3330;11.4
```

A lake file of one lake. The first line is a comment; a file Better Continents writes starts with it, followed by "(world metres, absolute heights; x, z = a point in the water)".

#	Field	Example	What it is
1	name	`Vikedal1`	the lake's name; no semicolon, at most 64 characters (a longer one is cut)
2	x	`-4210.5`	a point in the water, east-west, in world metres: where the flood starts (best the deepest point)
3	z	`3190.25`	the same point, north-south
4	level	`92`	the height of the water's surface, as the game counts heights: the sea is at 30, so `92` is 62 m above the sea
5	minX	`-4420`	the box the lake may fill: its west edge
6	minZ	`3050`	its south edge
7	maxX	`-4010`	its east edge
8	maxZ	`3330`	its north edge
9	depth	`11.4`	how deep the lake is at its deepest, in metres; kept and written back, while the water itself takes its depth from the ground

The rules the reader follows:

- Numbers use a **dot** for decimals and no thousands separator, whatever your computer's language.
- A line that starts with `#` is a comment, and a blank line is skipped. Fields after the ninth are ignored.
- A line with fewer than nine fields, or a field that is not a number, is **left out**, with a warning that names its line (`lakes.txt: line 6: ...`).

- A lake whose box is empty, whose point lies outside its box, or whose level is under 30.5 m is **kept, with a warning**: you may know better, but check it. A level under 30.5 m sits in the sea's own waves.
- A lake without a name is called `lake1`, `lake2` and so on. A world holds at most 5,000 lakes.

How the water fills the box. From the point, the ground is looked at on a 4 m grid over the box and 12 m around it. Every place lower than the level that joins the point becomes water, and the shore one step beyond. If the point itself is dry, the lowest ground within 12 m of it is taken; if that is dry too, the lake is not built and the log says *its point (x, z) is dry*. If the water reaches the edge of the grid, the shore is cut there and the log warns that *the basin is not closed there*: make the box bigger, or the level lower.

bc_lake load: a file into a running world

```
bc_lake load                 a dry run: what the maps folder's lakes.txt and
                             clearmask.bin hold, against what the world has
bc_lake load confirm         loads them: the world's lakes (and mask) are replaced
bc_lake load <folder> ...    the same from another folder
```

The dry run lists the file's warnings (the first ten; the log has all of them) and says what the world's depth scale will be: the world keeps its own if it has lakes already. With `confirm`, the lakes the world had are first written to `lakes-before-<time>.txt` (and the mask to `clearmask-before-<time>.bin`) in `BetterContinents/<world name>/` in Valheim's save folder. The new lakes are sent to the players in the world at once and saved with the world at its next save.

A world made without Better Continents (vanilla) can take lakes too: add `convert` (`bc_lake load <folder> convert confirm`). The world then becomes a Better Continents world that keeps the game's own ground, biomes, vegetation and locations, and carries the lakes. From then on every player needs this Better Continents version to join it (the load is refused, and names them, while a player in the world has another); the game alone, or an older Better Continents, opens it without the lakes.

bc_lake export: the world's lakes to a file

```
bc_lake export               writes lakes.txt and clearmask.bin to a new folder,
                             lakes-<time>, in BetterContinents/<world name>/
bc_lake export <folder>      to that folder
```

It never writes over a file that is already there unless you add `confirm`, and never inside the game's world saves. The numbers are written exactly (the world's lakes come back to the bit), in a form Tarn reads too. A **world export** (F7, or `bc_export`) writes `lakes.txt` and `clearmask.bin` into its folder as well, so a world made from that folder has the same lakes and mask.

A world from Tarn

Tarn kept each world's files in `BepInEx/config/Tarn/`: `<world name>.txt` (the lakes) and `<world name>.clear` (the clearing mask). Better Continents reads them **once, by itself**, when the world loads on the machine that runs it, if:

- the world is a Better Continents world,
- it has no lakes (for the `.txt`) or no mask (for the `.clear`) of its own yet, and
- the Tarn mod is not installed.

The log says `Lakes: imported 172 lakes from <path>; they are saved with the world at the next save`. From then on the world carries them itself and the files are not looked at again. **Tarn's folder is never written to.** An imported world keeps the depth look Tarn gave it (Tarn's own `DepthScale`, when you had set one).

By hand: `bc_lake import` (a dry run) and `bc_lake import confirm`, or `bc_lake import <folder> confirm` for another folder. A world made without Better Continents needs `convert` as well, as for `load`.

The Tarn mod installed beside Better Continents

If the **Tarn** mod is installed too, Better Continents **steps aside**: its own lakes and clearing stay off, so that the two never both build water. Nothing is lost: the world's lakes and mask are kept and saved as before, and come back the moment Tarn is removed. Better Continents says so in the log, in `bc_lake info`, and once on the screen when a world with lakes or a clearing mask loads:

```
Better Continents' lakes are off while the Tarn mod is installed. Remove Tarn to use
  Better Continents' own.
```

The words of `bc_lake` that would change the lakes answer with the same sentence and do nothing.

> **Watch out:** in multiplayer, a player who has Tarn installed while the world's lakes are Better Continents' sees dry hollows where the others see lakes, and their boats and swimmers fall through. When you move a world to Better Continents' lakes, **remove Tarn from every machine**, the server's included. The files keep working with either.

7 Clearing (experimental)

> **Watch out: clearing is experimental for now:** details may still change, and a world that carries a clearing mask can only be opened by Better Continents 0.10.5 or later. **Try it on a copy of a world first.**

A **clearing mask** is a black-and-white picture of the world's ground, one bit a pixel. Where a pixel is marked, the game places **no trees, rocks, bushes or other vegetation** (the branches and stones you pick up included). Map makers use it for roads and lake beds. It is Tarn's `.clear` file, unchanged.

- **Only what would have stood on a marked spot is left out.** Almost everything else is placed exactly as without the mask.
- **It acts when a zone is generated**, the first time anyone comes near. Zones generated before the mask was in the world keep their plants: make a **new world**, or regenerate those zones (below).
- **Grass is not vegetation:** it follows the ground's paint. Paint roads and lake beds as dirt or paved ground on the paint map, or grass grows through.
- **Locations are not affected:** a village or a dungeon entrance on a marked spot is placed as usual.

Where it goes

Put `clearmask.bin` in your **maps folder** (the `Directory` setting), beside the other maps; `tarn-clear.bin` is taken when there is no `clearmask.bin`. A new world made From Config (or with `bc_import`, or from a world export's folder) reads it when it is made and keeps it. An existing world takes one with `bc_lake load` (chapter 6) or `bc_lake mask load <file>`.

When a world with a mask loads, the log says what it holds, and the first time a spot is kept clear it says where:

```
Clearing: mask 4096 x 4096 over +-10500 m, <K> km2 kept free of vegetation
Clearing: the first spot kept clear of vegetation is at (<x>, <z>), in the zone
  (<zx>, <zz>)
```

The second line comes once, the first time the mask keeps a spot clear: it proves that the mask is at work.

The file

Bytes	What
0 to 7	the eight letters `TARNCLR1`
8 to 11	the size: how many pixels across (and down), a 32-bit whole number, least significant byte first (4096 for the Ravenrest map)
12 to 15	the half-width the mask covers, in metres, a 32-bit floating-point number, least significant byte first (10500 for the Ravenrest map)
16 to the end	size x size bits, one per pixel, eight to a byte, **the first pixel in the highest bit**; the rows from **north to south**, each row from west to east

The mask is square and centred on the world's middle: it covers from minus to plus the half-width on both axes. Pixel (row r, column c) is centred at x = (c + 0.5) x p - half-width and z = half-width - (r + 0.5) x p, where p = 2 x half-width / size is the width of a pixel. A marked pixel keeps clear its square and 0.25 m more on each side, so that neighbouring pixels leave no seam. A mask may be up to 16,384 pixels across, and its pixels at least 1 cm wide.

Line it up with your maps. Better Continents' maps cover World Size + Edge Size each way from the middle (10,500 m at the game's own size), with north at the top of the picture. A mask of the same size,

drawn the same way up, with that half-width, lies over them to within half a pixel.

Make one with Python

A road picture painted white where the ground is to be kept clear, the same size as your other maps, becomes a mask with a few lines of Python (with numpy and Pillow):

```
import numpy as np
from PIL import Image

HALF_WIDTH = 10500.0   # World Size + Edge Size of your world (10,500 m at the game's)
# white = keep clear; north at the top, as your other maps
picture = Image.open("roads.png").convert("L")
size = picture.size[0]
assert picture.size == (size, size), "the mask must be square"
marked = np.asarray(picture) > 127       # rows north to south, each west to east
bits = np.packbits(marked.reshape(-1))   # 8 pixels a byte, the first in the highest bit
with open("clearmask.bin", "wb") as f:
    f.write(b"TARNCLR1")
    f.write(np.array([size], dtype="<i4").tobytes())
    f.write(np.array([HALF_WIDTH], dtype="<f4").tobytes())
    f.write(bits.tobytes())
print(size, "x", size, ":", int(marked.sum()), "pixels marked")
```

Put the file in your maps folder and make a new world. In the game, `bc_lake mask at <x> <z>` tells whether a point is cleared, and `bc_lake info` shows the mask's size and how many square kilometres it keeps clear.

The mask in a running world

Command	Does
`bc_lake mask at <x> <z>`	whether that point is kept clear, and which pixel of the mask it is in
`bc_lake mask load <file>`	a dry run of putting that mask in the world; add `confirm` to do it (the old mask is first written to `clearmask-before-<time>.bin` in `BetterContinents/<world name>/`)
`bc_lake mask export <file>`	writes the world's mask to that file, in Tarn's format (`confirm` to write over a file)
`bc_lake regen`	says how many generated zones the lakes and the mask touch; with `confirm` it regenerates the world's zones

`bc_lake regen confirm` **regenerates every generated zone that is not protected,** as `bc regen` does: Better Continents has no regeneration by region. Zones within one zone of anything a player built or worked on, tombstones, the zones of players on other machines, and you inside a dungeon are kept as they are; players, what they built, tombstones and tamed animals are never removed. Read the dry run before you confirm, and keep your copy of the world.

> **Good to know:** only the machine that runs the world places vegetation, so only it uses the mask. Players get a copy of it with the world's settings (an export from a player's game then has it too), but a mask loaded during play is not sent to them: they do not need it.

The other way: a white vegetation map

A vegetation map (`vegetationmap.png`) can keep ground clear too: its pure white pixels allow no vegetation at all. That is the way to clear ground when your maps already have a vegetation map. The mask's strengths are Tarn's files, working as they are, and one bit a pixel: a 16,384-pixel mask is 32 MB on disk, and a fraction of that once the world packs it.

Appendix A

Commands and settings

Open the console with **F5**. Words in `< >` are yours to fill in; words in `[ ]` may be left out.

bc_water (any player)

Command	Needs	Does
`bc_water`	nothing	whether Better Continents' sea is on in this world and why, the map of the sea, and what draws the water on your game (chapter 3)
`bc_water force on`	Debug Mode	draws Better Continents' water where it would not be drawn, to compare it with the game's; your game only, not saved
`bc_water force off`	Debug Mode	the world's own rules again (leaving the world does the same)
`bc_water probe`	Debug Mode	the drawn surface against the felt one at 81 places in front of the camera: one `Water probe:` line

On the machine that runs the world the same words are in the cheat tree as `bc water ...`, with cheats on.

bc genloc (the machine that runs the world, cheats on)

Command	Does
`bc genloc`	makes the world's locations again, as the game's `genloc` does; says what it found, and what changed when it ends; sends the new location icons to the players; frees the game's slot when a run that threw left it taken. Stay in the world until it says it is done
`bc genloc alt`	the game's own alt-biome branch (`genloc alt`)
`bc genloc pins`	puts only the location map's pins and the start position into the game's location list, and makes no locations; says so when the world has neither
`bc genloc help`	a short help

bc_lake (experimental)

> **Watch out:** lakes and clearing are **experimental** for now. Try them on a copy of a world first.

Every word that replaces or removes in bulk is a **dry run** unless the line has `confirm`; a single `here`, `at` or `depth` is made at once. `confirm` and `convert` may stand anywhere after the word. `bc_lake help` prints the list in the game. No cheats and no Debug Mode are needed.

Command	Who	Does
`bc_lake help`	anyone	the words, and what each does
`bc_lake info`	anyone	the world's lakes and mask in words
`bc_lake list [n]`	anyone	the n nearest lakes (10 by default; up to 1,000)
`bc_lake mask at <x> <z>`	anyone	whether that point is kept clear
`bc_lake here [radius] [margin]`	the world's machine	fills the hollow you stand in to `margin` under its spill point (radius 8 to 1,000 m, 120 by default; margin 0 to 20 m, 0.3 by default)
`bc_lake at <x> <z> [radius] [margin]`	the world's machine	the same at a place
`bc_lake remove <name>`	the world's machine	removes the lakes of that name

Command	Who	Does
`bc_lake remove all`	the world's machine	removes every lake; `confirm`
`bc_lake depth [scale]`	the world's machine	shows or sets the world's Lake Depth Scale (0 to 3)
`bc_lake load [folder]`	the world's machine	the maps folder's (or the folder's) `lakes.txt` and `clearmask.bin` into the world; `confirm`; `convert` for a world made without Better Continents
`bc_lake import [path]`	the world's machine	Tarn's `<world name>.txt` and `.clear` from `BepInEx/config/Tarn` (or the folder); `confirm`; `convert` as for `load`
`bc_lake mask load <file>`	the world's machine	the mask alone; `confirm`; `convert` as for `load`
`bc_lake export [folder]`	where exports are allowed	writes `lakes.txt` and `clearmask.bin`; `confirm` to write over files already there
`bc_lake mask export <file>`	where exports are allowed	writes the mask; `confirm` to write over a file
`bc_lake regen`	the world's machine	how many generated zones the lakes and the mask touch; with `confirm`, regenerates every unprotected generated zone
`bc_lake server <words>`	an admin	runs the words on the dedicated server; its answer comes back to your console

"The world's machine" is the player in single player and the host in a hosted game. A player connected to a server who types a changing word is told to ask the server: `bc_lake server <the same words>`, which the server takes from admins only (its `adminlist.txt`). The host's cheat tree has the same words as `bc lake ...`.

While the **Tarn** mod is installed, the words that change lakes or the mask refuse and say why (chapter 6).

Settings

`BetterContinents.cfg`, or Configuration Manager (F1).

Section	Setting	Default	Whose	What it does
`[11 BetterContinents.Water]`	**Own Water**	true	each player's, at once	draws Better Continents' water where the world's sea is not the game's
	Soft Far Sea	true	each player's, at once	with Own Water, the far sea takes the depth of the sea under it
`[12 BetterContinents.Lakes]`	**Lake Depth Scale**	0.5 (0 to 3)	the world's	how deep the water takes lakes to be, for a new world's lakes; `bc_lake depth` in a running world
	Lake Build Range	400 (150 to 2000 m)	each player's, at once	lakes within this distance are built, and taken down 150 m further out
	Lake Build Budget	2 (0.5 to 20 ms)	each player's, at once	time of each frame spent making lake water
	Lake Sound	0.35 (0 to 1)	each player's, at once	the water sound near lakes; 0 = off
	Lakes On Map	true	each player's, at once	lakes drawn on the map and the minimap
`[00 BetterContinents.Debug]`	**Debug Mode**	false	each machine's	needed by `bc_water force` and `bc_water probe`

Files

File	Where	What
`lakes.txt` (or `tarn-lakes.txt`)	the maps folder (`Directory`)	the lake list a new world reads (chapter 6)
`clearmask.bin` (or `tarn-clear.bin`)	the maps folder	the clearing mask a new world reads (chapter 7)
`<world name>.txt`, `<world name>.clear`	`BepInEx/config/Tarn/`	Tarn's own files: imported once, never written to
`lakes-before-<time>.txt`, `clearmask-before-<time>.bin`	`BetterContinents/<world name>/` in Valheim's save folder	what a `remove`, `load`, `import` or `mask load` replaced
`lakes-<time>/`	the same folder	what `bc_lake export` writes when you name no folder
the world's lakes and mask	inside the world's `BetterContinents` settings file, in the world's folder	saved with the world at each save

Appendix B

For map makers

One page of what this guide means for the maps you draw. The AI skill document in the same package (`BetterContinents-AI-Skill.pdf`) has the same in full, with code.

The sea follows your maps

- **Your biome map's Deep North is the calm sea.** The waves die down over its first 200 m. Paint Deep North over the sea only where you want it calm; with no Deep North on the map, the world has no calm sea at all.
- **Your heat map's hot sea is the red sea**, wherever the heat over the sea is above black, with a 25 m soft edge and no more fading than your heat map draws. Fade the heat out over the sea where the red should end, and keep it black everywhere else.
- **Without a biome map or a heat map,** a world made at another size than the game's has the game's two rings, scaled to its size; at the game's own size it has the game's water.
- **Check it** in the world: `bc_water` says how much of the world is calm and how much is hot, and whether either is the game's own circle (chapter 3).

Lakes (experimental)

> **Watch out:** lakes and clearing are **experimental** for now. Try them on a copy of a world first.

- **A lake is a line**, `name;x;z;level;minX;minZ;maxX;maxZ;depth` (chapter 6), in `lakes.txt` in your maps folder.
- **The level is the game's height**: the sea is at 30, so a lake whose water stands 62 m above the sea has level 92. Work it out from the heights your heightmap gives, not from the picture's pixel values.
- **Find the level the way the ground allows it.** Start at the basin's deepest point, raise the water until it would spill over the lowest point of the rim, and set the level a little under that (Better Continents' own `here` takes 0.3 m). Put the point at the deepest place, and the box around everything under the level that joins it, with some room.
- **Make the basin closed.** The water is laid on a 4 m grid of the generated ground; a gap in the rim narrower than that can still let it out, and the log then warns that *the basin is not closed*. Dig basins a little wider than you need.
- **Keep levels above 30.5 m.** Lower than that, a lake sits in the sea's waves.
- **Try one first in the game:** `bc_lake here` in your basin makes the lake Better Continents would make, and `bc_lake export` writes it out in the file's format, to compare with yours.

Clearing (experimental)

- **A mask is one bit a pixel** (chapter 7), `clearmask.bin` in your maps folder, north at the top, the same half-width as your maps (World Size + Edge Size).
- **Mark roads one pixel wider and lake beds a few pixels wider** than they are drawn, so that nothing leans in from the edge (the Ravenrest map marks roads + 1 pixel and lake beds + 3 pixels at 5.1 m a pixel).

- **Paint the same ground as dirt or paved ground** in the paint map: the mask keeps trees and rocks off, but grass follows the paint.
- **Test it on a new world.** A mask acts only on zones generated after it is in the world.

What ships with the world

- `lakes.txt` and `clearmask.bin` are read when a world is made From Config, and kept in the world's own settings: the world needs no file afterwards.
- A world export writes them back out, and a `.bcworld` share file carries them.
- Every player of a world with lakes or a mask needs Better Continents 0.10.5 or later. Tell your players.

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>.
```