

## Tumbler Lock

N NsN

[VIEW IN BROWSER](#)

updated 10. 10. 2022 | published 10. 10. 2022

### Summary

I tried to make a lock to demonstrate how locks and lockpicking works, and accidentally made a really hard to pick lock.

[Learning](#) > [Engineering](#)

Tags: [lock](#) [lockpicking](#) [openscad](#) [parametric](#) [tumblerlock](#)

This is a practical example on how simple tumbler locks work.

### Components

There are 3 different variations of the inner plug.

- plugA: This plug features a wide window to make the pins easy to see, however because the pins would fall into that window, the lock can only turn counter clockwise.
- plugB: This plug has viewing slots cut in, that should be just small enough that the pins can't fall into them. Because of this the plug can turn clockwise.
- plugC: This plug has little guards under the pins preventing them from falling out. It can be turned in any direction, but is harder to print, the quality of the print might be worse.

Unfortunately some adjustments to make it easily printable also make it quite difficult to pick the lock with normal diamond, rake or snake picks, so a special lockpick is also included.

### **Non printed parts**

You will need one spring of 4mm diameter and approximately 15mm length for each pin. Springs from a ball point pen should work.

I've experimented with printed springs in a flexure design, but all of them made the final design unreasonably large.

### **Printing**

All components should be printed in the orientation they are in the file.

Most things should be fine with 2 perimeters and 10% infill, PLA works well and adaptive layer height can be used to increase detail and reducing print time. For the outer cylinder it can be useful to increase the perimeter count to 3, to make the whole hull solid.

Only the key and the pick need supports on build plate enabled. If you have good bed adhesion, you can also flip the key upside down, and it should print without supports.

### **Assembly**

Add the key pins in the right order to each channel. Add a driver pin on top of each, the rounded part of both the key and the driver pins should point towards the key.

Add a spring to each channel and slide the cylinder lock in place. This works best without a key in the lock.

### **Picking the lock**

In trying to make the lock easier to print and improving the tolerances, it looks like I made the lock a bit hard to pick.

The keyway is surprisingly tight, since the shortest keypins are quite small. Both driver and keypins are also rounded / chamfered at the ends, to make the movement smoother, but this seems to have made picking them harder.

### **Generating your own keys**

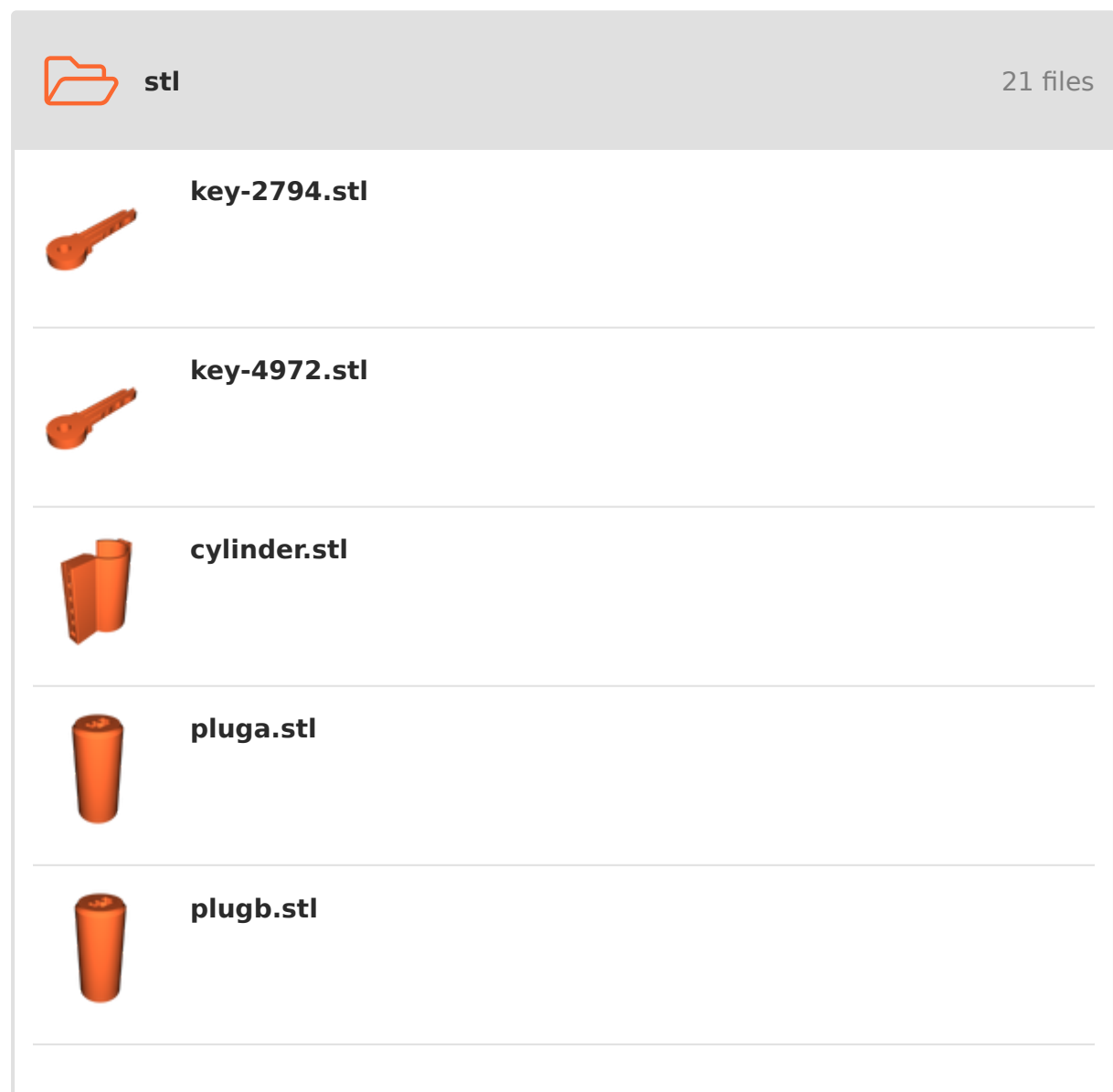
You can use the provided python script to create your own keys and locks. You will need an installation of `openscad`.

This script has only been tested on ubuntu and openscad version 22.09.23 so far. The full commandline would look like this:

```
python3 generateLock.py "4,9,7,2" --quality 8 --out stl --exec openscad-nightly
```

- With `4,9,7,2` being the positions to cut. You can use any number of integers between 1 and 9.
- Quality dictates how long the rendering will take, 1 being fastest and 10 being slowest.
- `stl` is the folder in which the results should be placed.
- And `--exec openscad-nightly` telling the script which openscad binary to use.

## Model files





**plugc.stl**

---



**pick.stl**

---



**wrench.stl**

---



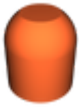
**cylinderlock.stl**

---



**keypin1.stl**

---



**keypin0.stl**

---



**keypin2.stl**

---



**keypin3.stl**

---



**keypin4.stl**

---



**keypin5.stl**

---



**keypin6.stl**



**keypin7.stl**



**keypin8.stl**



**keypin9.stl**



**driverpin.stl**



**keypins.stl**



**src**

8 files



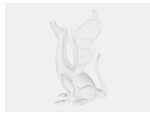
**automation.scad**



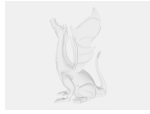
**key.scad**



**plug.scad**



**cylinder.scad**



**settings.scad**



**combined.scad**



**tools.scad**



**pins.scad**



**generatelock.py**

## Other files



**readme.txt**

## License

This work is licensed under a  
[Creative Commons \(4.0 International License\)](#)



**Attribution-ShareAlike**

✖ | Sharing without ATTRIBUTION

- ✓ | Remix Culture allowed
- ✓ | Commercial Use
- ✓ | Free Cultural Works
- ✓ | Meets Open Definition