What's open source?

Justin
Mar 31, 2020

The TL;DR

Open source *technically* means that an app's code is publicly available, but more *generally* refers to the movement of **communities of developers** building **transparent, open codebases** free for everyone to use.

- If an application is **open source**, its code is freely readable and usable
- Many of the **everyday basics** that our applications use are open source, like Linux, PostgreSQL, Nginx, Node, and React
- Open source codebases are supported by **communities of developers** who improve and **contribute** to them for free
- Open source faces a significant **"tragedy of the commons" problem**: people use it without giving back to the community
- New business models **built on open source** are becoming a popular way of selling software as a service

Open source might be going through its midlife crisis right now, but it's powerful and it's here to stay. There's a lot to cover, so sit tight and scroll!

What's open source?

All of the applications that we use (and pay handsomely for!) are just a bunch of code. To understand what open source means, let's start with explaining what the dominant software model has been for the past 20 years: **closed source**.

→ *What does closed source mean?*

In most cases, if you have access to an application's code, you have the application: that's why application code is usually *private* and even classified as intellectual property. The best example of this is Microsoft Windows; it's *by far* the most popular desktop operating system, and it's **closed source**. That means that Microsoft keeps the Windows code private, and *licenses* it to businesses,

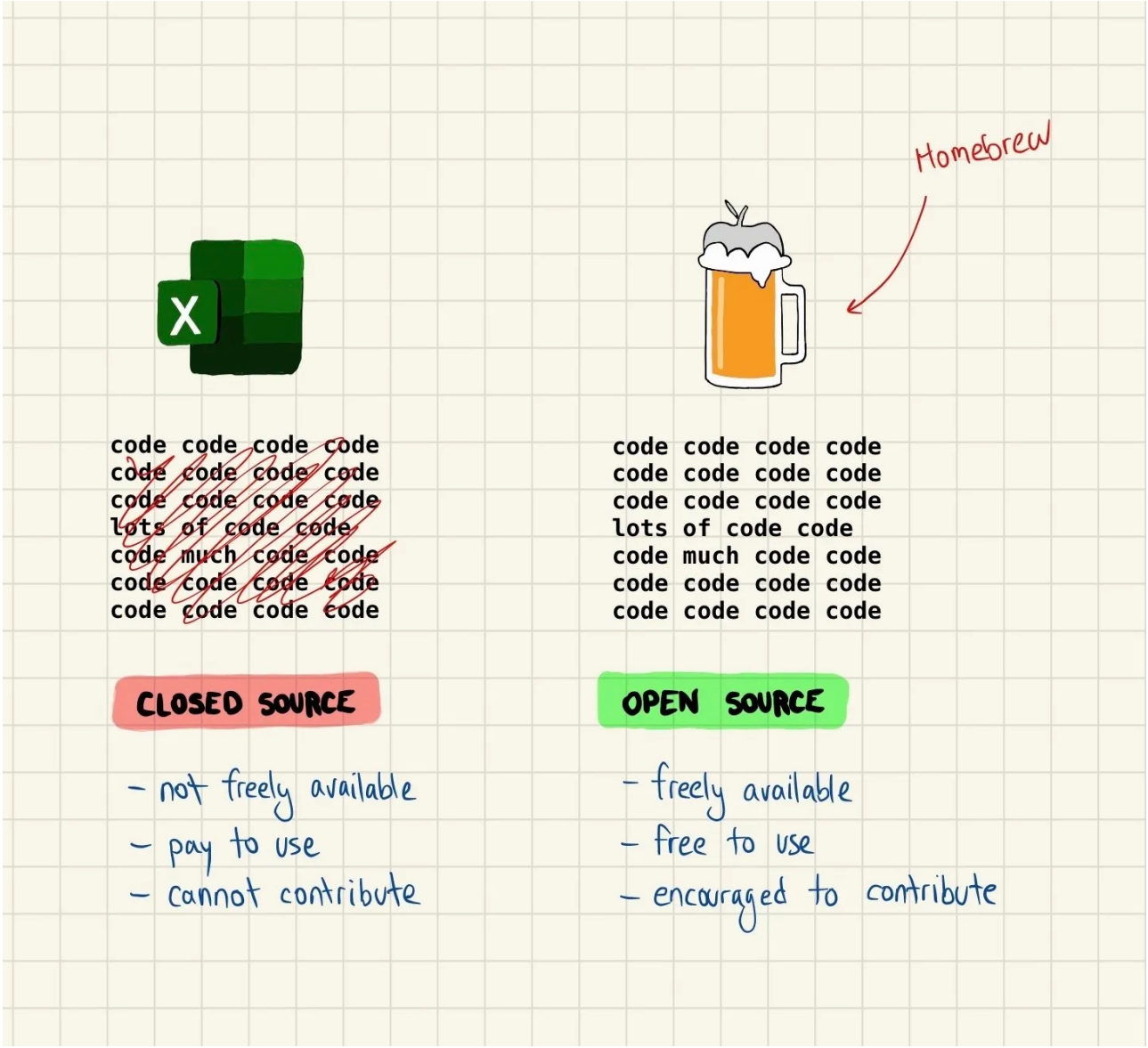
who pay for it. If you somehow got access to the Microsoft Windows codebase and used it without licensing it, that would be **pirating, and it's illegal**. Bill Gates is angry at you now.

Most applications that you use today are closed source: Twitter, Gmail, Google Docs, Excel, Tik Tok, whatever. Engineers at those companies work very hard on writing the code that powers those apps, and it's understandably not publicly available.

→ *What does open source mean?*

Surprise surprise, open source means that an application's code is *not* private: anyone can read and use it. It's harder to point to examples of **user facing** applications that are open source, because very few apps like Excel are; the model is much more popular for **developer tooling**. A lot of the tools that our apps are built on, like Linux (operating systems) and Homebrew (package managers), are all open source.

To make this a little more concrete, let's dive deeper into **Homebrew**. It's a package manager for Mac – a tool that developers use to install things and manage versions – and it's completely open source. Homebrew's code is freely available on Github here, and anyone can make suggestions or write code to improve it and add new features.



> **😰 Don't sweat the details 😰**
>
> Because most popular open source projects are developer tools, it's sometimes hard for non-developers to understand what the deal is. You don't need to understand what Homebrew does to understand open source: just think of it as an application like Gmail.
>
> **😰 Don't sweat the details 😰**

Open source is all about community

So far, I've given you the *technical* definition of open source: publicly available, free code. But that's only the tip of the iceberg. What open source *really* means today is an **active, engaged community of supporters**. Once an open source

application gets big enough, there are usually 3 groups of these kind of supporters helping it out:

1. **Core committers** – people spending significant time on building the codebase and managing what needs to get done
2. **Contributors** – people who occasionally help out by completing tasks that core committers outline, or making their own suggestions
3. **Sponsors** – companies and individuals that *financially* support the open source application

Let's walk through another good example – **Apache Airflow** – and identify who all of these people are. Airflow is a developer tool for running certain programs at certain times (I used it a lot at my last job). There's a group of 20 on the Project Management Committee: they're the core committers. If you head over to Airflow's Github though, you'll see that there are over 1,000 (!) committers: most of these are contributors. Finally, Apache (the parent organization) has big name sponsors like Google, Facebook, and Microsoft.

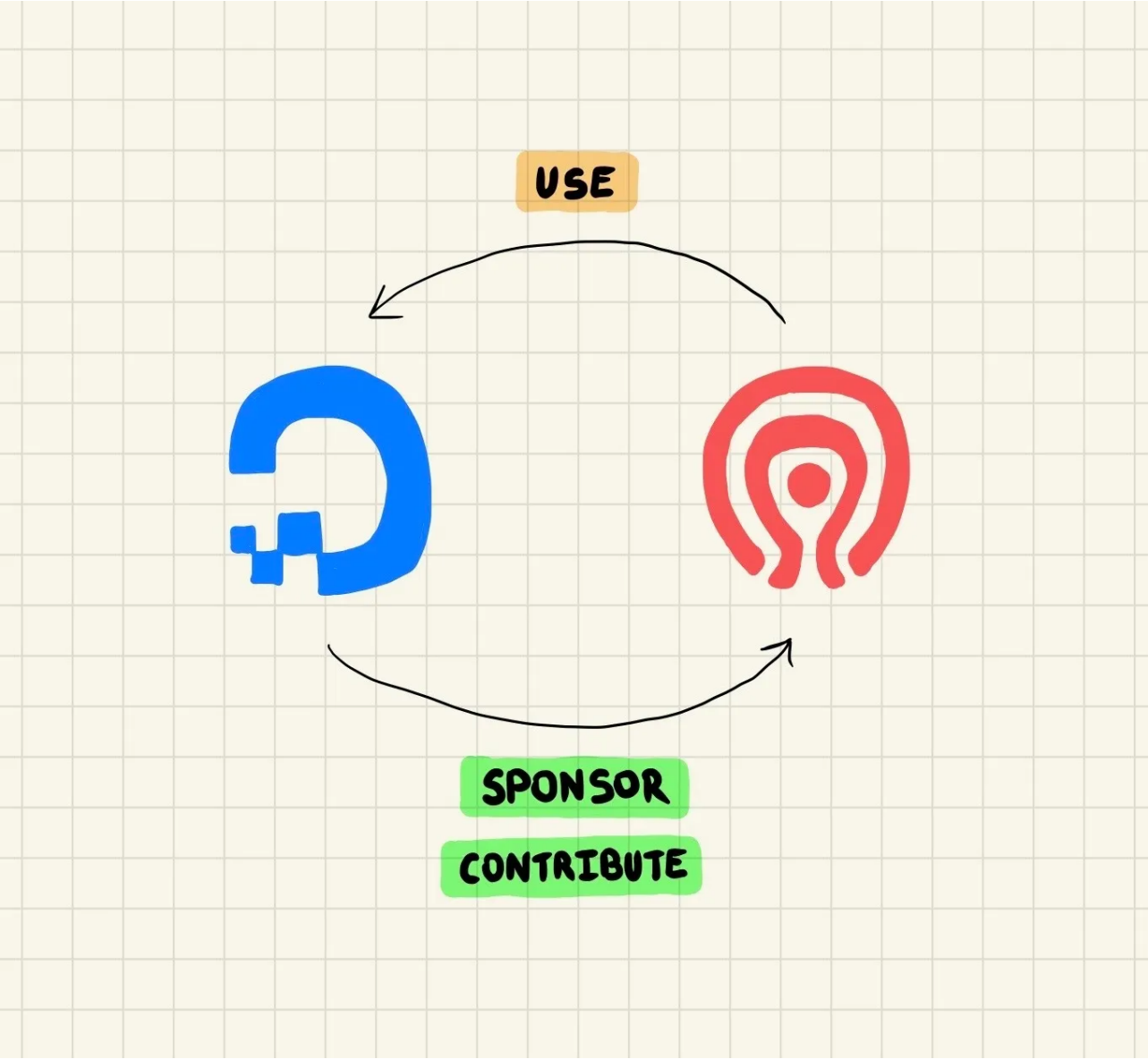
Most of the time, contributors are developers working on the application's core codebase. But sometimes they're less technical folks; you can also contribute to an application's **documentation, or even marketing site**. My former employer DigitalOcean runs a great yearly program called Hacktoberfest that encourages people from all professional backgrounds to contribute to open source in whatever ways they can.

The ugly parts of open source

Not everything is rosy in open source land. Because open source libraries and apps are free to use, they often face a "tragedy of the commons" problem: most users never contribute anything back to the communities that they take from. This is a well documented problem, and not a new one; but it continues to place a strain on core committers who are expected to produce a high quality, functioning application with **almost nothing in return**.

One thing that compounds this problem is **corporate use** of open source. Companies build their applications on hundreds of open source packages, but often give nothing back to those packages because they're free. Forward

thinking software organizations make sure to set aside budget for sponsoring the software that they use, and let their developers take work time to contribute code to these open source repositories. DigitalOcean uses an open source technology called Ceph to build one of their storage products, and gives back by sponsoring the Ceph Foundation.



The problem isn't one sided: contributing to open source can often be a complex, frustrating, and even hostile process. Jumping into someone else's codebase is a daunting task, and it's difficult for developers to *really* add value to an open source repository without spending significant time gaining context and getting comfortable. And open source communities can, at times, be toxic towards new members. Most developers are familiar with Linus Torvalds, the creator of Linux and Git – he's famous for being a colossal asshole. This is rare – most open source communities are welcoming and kind – but it happens.

Open source as a business

All of the open source projects that we've covered so far are **not for profit** – they're often managed by foundations, and nobody is getting rich off of them. But that's not the only model in open source: some companies keep their applications open source, but offer **value-add services and support** on top of that software for a hefty fee. MongoDB, Elastic, and Confluent are examples of billion dollar companies built on this business model.

To get a better sense of what this means, let's walk through Elastic's core product. It's called Elasticsearch, and it's a developer tool for storing and searching unstructured data. Elasticsearch is open source and free on Github, and you can run it on servers yourself. But that's difficult and takes a bunch of developer time. Instead, you can pay Elastic to **deploy Elasticsearch for you**; they'll take care of the tedious work and include value-add services like plugins and support.

Not all open source business models are created equal, and some companies have struggled to make this arrangement work. MongoDB famously changed their license and stopped developing their open source offering. Sentry had to do the same. The best example of this is probably Docker; 30% of developers use it, but the parent company was never able to figure out a business model that made money. The pattern seems to be that paid open source works well when the software is difficult to deploy infrastructure-wise, but the book is still open.

Terms and concepts covered

```
Closed source
```

```
Open source
```

```
Committers
```

Contributors

Sponsors

Further reading

- It's not realistic to support every piece of open source software you use, but contributing is a lot easier than you might think
- Open source as a business model is still in its relatively early stages, and companies are still figuring things out
- A first person account of the struggles of being an open source maintainer

Comments

Write a comment...