What does UIPath do?

Creating robots that will destroy us

Justin
Apr 26, 2021

12 1

The TL;DR

UIPath helps people **automate rote, repetitive manual tasks** like updating spreadsheets, creating documents, and sending emails.

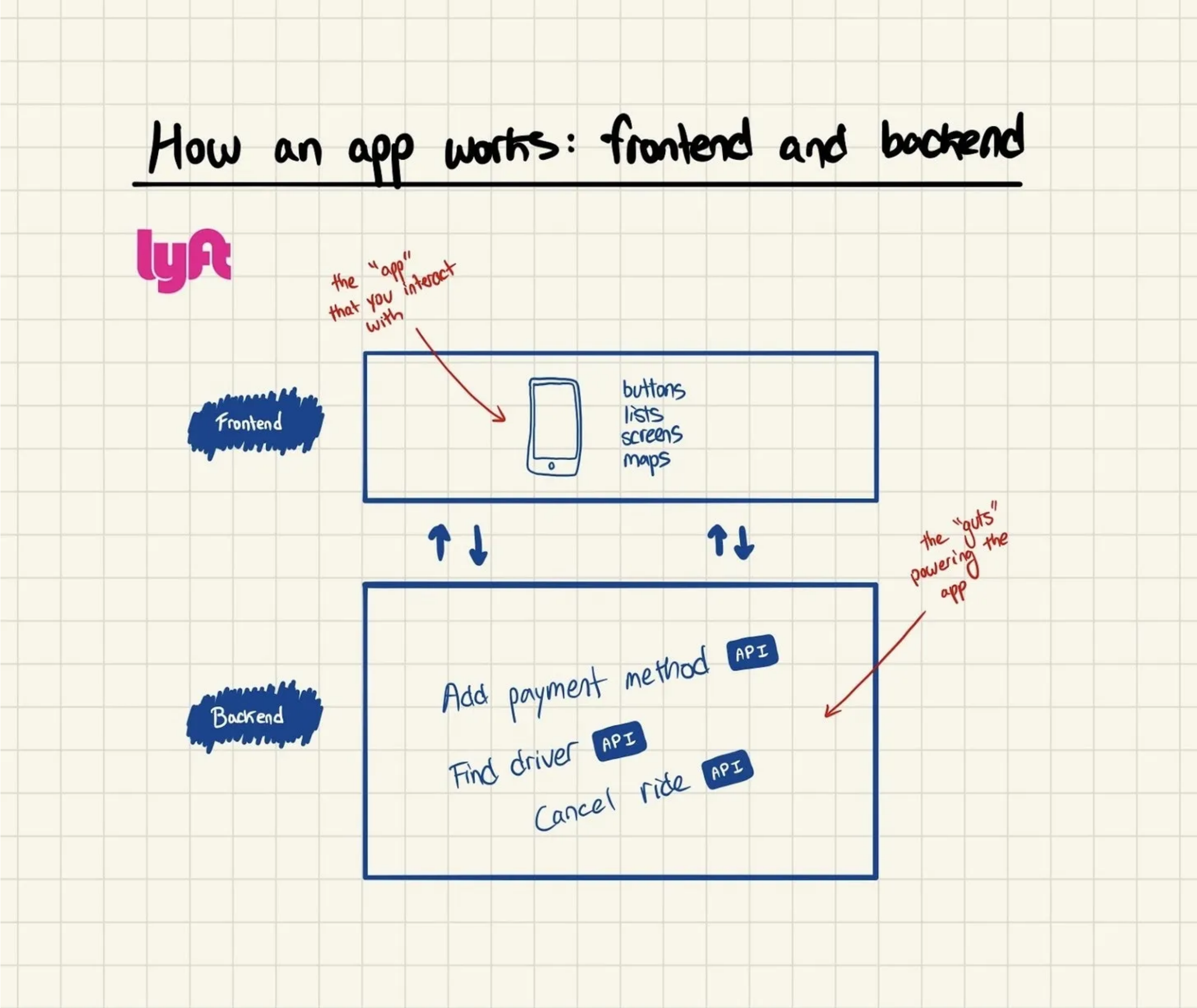
- Work that we do tends to be **repetitive and manual** – pretty much anything with a *process* like forms, compliance, etc.
- Traditional approaches to automation use **exposed APIs from software** to hack together scripts – but many apps don't have those APIs available
- UIPath literally **records the work you do** (click here, scroll there) and then lets you replay it, automatically, whenever you want
- You can also build basic **forms**, **workflows** based on conditions, make **HTTP requests** – pretty much anything

UIPath is *definitely* enterprise-first (just check out their website), and it's working – they went public at a $35B valuation in April 2021.

How automation generally works

Automation is the buzziest of buzzwords, so it's worth getting into what it actually means, and more importantly, how people (developers?) tend to make it happen.

Software (well, most software) is built on top of APIs. There's a nice user interface with buttons and fields and text, but every time you actually do something (load a page, submit a form, move things around), there's probably a server involved, and your app is interacting with it via APIs.


How an app works: frontend and backend
lyft
the "app" that you interact with
Frontend
buttons
lists
screens
maps
the "guts" powering the app
Backend
Add payment method
API
Find driver
API
Cancel ride
API

(Stolen from the original [What's an API post](https://technically.substack.com))

Companies design their user interfaces around what problems they think they're solving, and what they think their users want to do. When Google designed Gmail, they figured that people would usually want to send emails, adjust their settings, and search their inbox – and so that's what the Gmail app lets you do.

But the APIs that *power* the Gmail app are much more robust than that – developers tend to design them very granularly, and they can be more powerful than what the Gmail UI lets you do with them. The Gmail UI, for example, doesn't really let you **mail merge** (send the same email to multiple people and templatize content) – but the APIs that power Gmail can do it very easily. The problem is just that you're limited to *interacting with those APIs* via the Gmail user interface.

With that distinction in mind, automation is usually just **interacting with APIs directly**. If there's a Gmail API that sends an email to an intended recipient, you could just write a simple Python script that hits that API in a loop until all email addresses in a list have an email sent to them. The key is just being able to write code (lol), since APIs are programmatic interfaces, and as such, they require programming.

The problem with API-first automation is that not every app gives you access to the APIs that you need! That can manifest in two different ways:

1. **The app doesn't give you access to its APIs** – this is more common than not, and happens for many legitimate reasons (e.g. they're not secure enough).
2. **The app doesn't really work via APIs** – apps like Excel build most of the application logic into the frontend itself, so there are few or no APIs so speak of.

So if you're working in Excel, moving files around on your Mac, or filling out a document, how do you automate that?

> 🔍 **Deeper Look** 🔍
>
> While companies may not *officially* expose their APIs for general use, you can often **reverse engineer them** and work with them anyway. You won't know much about how they'll behave and won't have documentation to read, but it's worth noting that some developers do this anyway.
>
> 🔍 **Deeper Look** 🔍

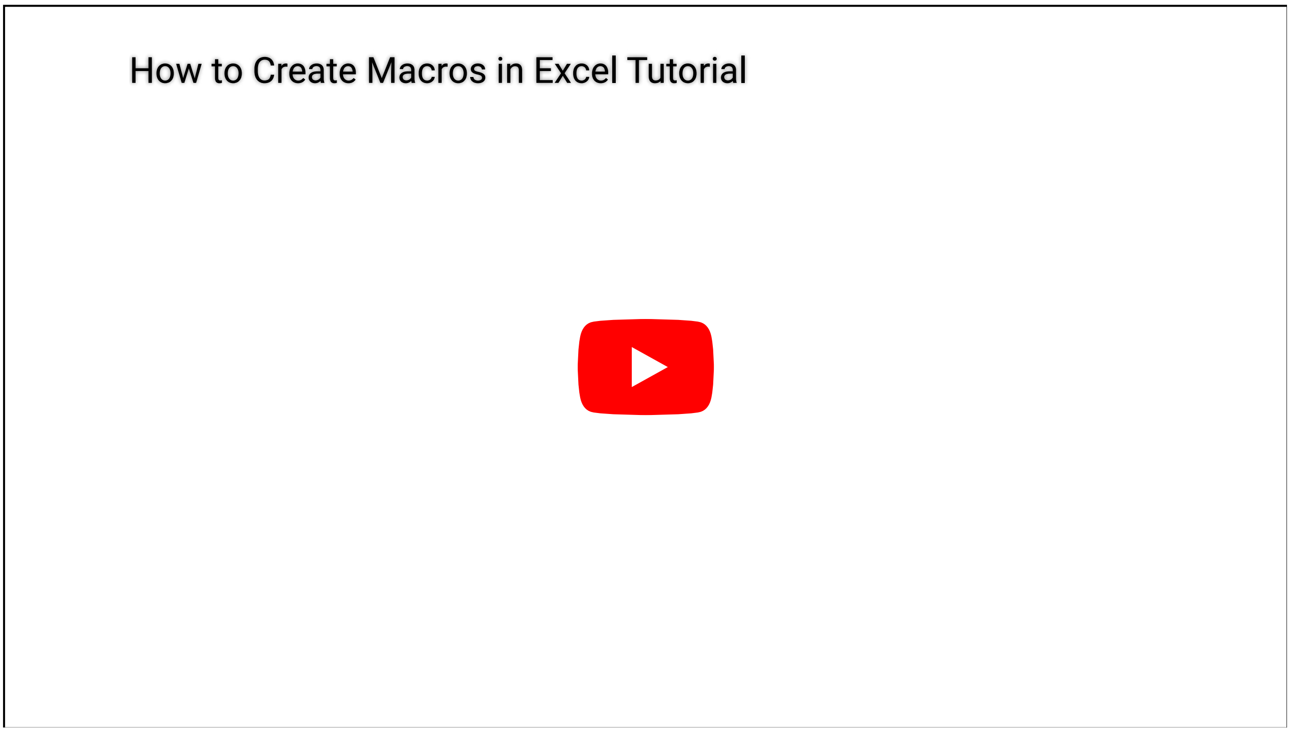
RPA and the "robot" concept

The other side of automation doesn't use APIs at all – instead, it basically **mimics exactly whatyou would be doing** inside of an app or operating system. If you've ever recorded an Excel macro, this will sound familiar; there are basically a couple of steps:

1. Open up your recording app (like UIPath)
2. Click "record"

3. Do the manual task
4. Stop recording
5. “Play” the task whenever you want

It’s sort of like recording a “how to” video, but for your computer – and it’s much different than using APIs, because *anyone can do this*. The hot acronym for this these days is **Robotic Process Automation**, or **RPA**.



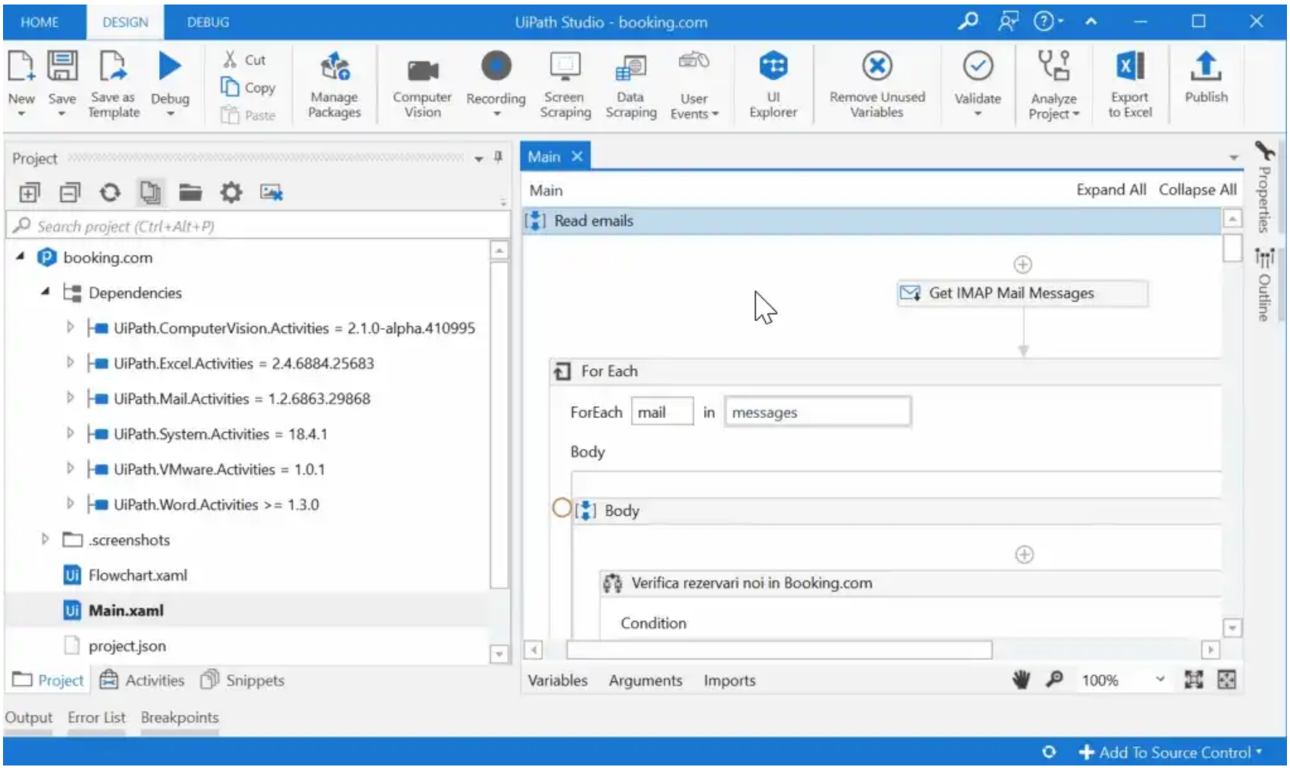
(Recording an Excel macro)

If you’ve had a job before (sorry), you can probably think of one or two things that you’ve that really, *probably*, should be automated by now – and how you’d be able to do it with robots. At my first startup job, I was in charge of our **monthly metrics report** – I needed to run a basic SQL query, copy the results, paste them into an Excel spreadsheet, update a bunch of column names, and email the sheet to a few people on the leadership team. Would have been a great candidate for RPA. But alas, I was young and dumb – and now I’m just dumb.

This, in a nutshell, is most of what UIPath does – it’s a product (sorry, a *platform*) that helps teams robotically automate their tasks.

The UIPath product

The *main* value proposition of UIPath – and their core product – is the ability to record your manual tasks, save them as a **process**, and replay them whenever you want, instantly. Here's their app for recording tasks (UIPath Studio):



You can read a basic tutorial on how to do recordings here. And beyond recording, you can also build workflows visually by choosing available apps and options and *chaining them together* on certain conditions.

But where UIPath gets really interesting is the ability to **add other shit** (this is the technical term) to automations. A couple of big ones:

1. **Forms**

You can create basic web forms in UIPath and then work with that data in a process. There's a good example on their site about how banks use it to automate KYC (know your customer) processes – they'll make customers fill out a form, then automatically create templated documents for them to fill out and sign.

2. **Business logic**

UIPath lets you loop through lists of things, set conditions based on the current date, and set processes to run on a schedule. So once you've got the basics of your automation done, you can fine tune the specifics easily.

3. **Working with external data**

You can read and write data from external sources in UIPath: you can get emails from an inbox, read data from a database, and even make an HTTP request. And when you combine this with something like a form, that gives you the ability to create (admittedly rudimentary) UIs on top of data – which makes UIPath start to look more and more like an app builder.

> 🚨 **Confusion Alert** 🚨
>
> UIPath lets you build workflows in two ways – via recording your actions, *and* visually through their UI workflow builder. For more custom work like logic, loops, and working with data, recording doesn't quite cut it.
>
> 🚨 **Confusion Alert** 🚨

These days, I think it's fair to say that UIPath ends up combining **both types of automation** – working with data and APIs, as well as RPA and good ol' recording what you're doing. And that's their marketing pitch, pretty much – automate everything, no matter how or why. And like with every generic "builder" they face the core problem of people like me showing up to the website and wondering "what would I actually *do* with this" – which is why they put so much time into diving into use cases.

For more depth on UIPath financials and comps, check out John's newsletter.

1 Comment

Write a comment...

Ali Rohde Writes Ali Rohde Jobs Apr 26, 2021

This, in a nutshell, is most of what UIPath does – it's a product (sorry, a _platform_) that helps teams robotically automate their tasks.

^ Why I read Technically

♡ 2 Reply Collapse •••