An Easier Way to Deploy Your Smart Contracts

TL;DR - Deploying your smart contracts shouldn't be difficult, and the process should be flexible. I give a sneak peek at the next feature we're building: aneasy to use interface to deploy your Truffle projects withTruffle Teams ! It's going to be awesome, andyou can be the first to use it ! I'll be giving ahands-on workshop from 9AM-12PM on Friday, August 2nd, atTruffleCon 2019 that will walk you through through the entire Truffle Teams lifecycle, including thenever-before-used Deployments interface ! See you there!

So! You've finished implementing your dApp, you've tested it onGanache , and you're ready to push it to a testnet or Mainnet! Congratula—oh wait.

It's not that easy.

There are a handful of solutions, and none havereally made it painlessly easy. Even our owntruffle migrate (akatruffle deploy) has left much to be desired.

Well it's about to get a whole lot easier with Truffle TeamsSkip to the good stuff.

The Not-So-Easy Ways to Deploy Your Smart Contracts ¶

So what's wrong with what's already made? Why makeyet another tool? I hear you. Let's take a look at whatis available, and if you're still meh, I'd love to talk to you atTruffleCon !

Remix

OhRemix , we all have fond memories of you. And many people still use Remix for all sorts of use cases! Remix gives you an in-browser Ethereum IDE, letting you develop, compile, test, debug, and deploy smart contracts for Ethereum. And it's probably the least amount of work needed to throw a single smart contract onto a testnet or Mainnet (i.e. when deploying a single.sol file).

Remix, as of version0.7.7 , gives you a simpleDeploy button. This will send a transaction to theweb3 provider of your choice (many useMetaMask), and you're off to the races!

Great, right?

Well, Remix gets harder to use as you add more contracts, import 3rd party contracts, work as a whole team, etc. It's definitely flexible and able to handle these use cases (though not without legwork and configuration), but it's not my personal preference for developing large, complex dApps.

Truffle CLI

EnterTruffle .

It's a fantastic framework (not to toot our own horns) that lets you develop your dApp in a flexible and extensible way. It provides structure to keep you sane in this chaotic world of Web 3.0.

It even gives you a mechanism to define what deploying (aka: "migrating") means for your application. You can specify what contracts get deployed in which order with whatever arguments you require. For instance, your smart contracts may depend on the deployment of a past contract, which you can easily code in migration scripts.

And it's as easy as runningtruffle migrate !

...and if you were deploying directly toGanache , that'd be entirely true.

But that's not the case when we start talking about testnets or Mainnet.

Theeasiest way to deploy to an external network is by usingTruffle's HD Wallet Provider . However, this requires you to somehow get your mnemonic or private key into thetruffle-config.js config file,without committing it to your provider source control provider (PLEASE don't do this - you'll get your Ether stolen). To use an account you created with MetaMask, you'd need to export your private key from MetaMask (or your mnemonic, if you're feeling lucky 😱) and create the boilerplate code to read in the key from an environment variable (to prevent committing it to your repository and sharing it publicly).

In all truthfulness, this isn't too bad, but it's also a bit too much config for me. I'd much rather use MetaMask directly to authenticate transactions during deployments.

Custom Management Web Interface¶

Some devs will create a small front-end web interface which usesweb3.js to deploy their smart contracts. I have done this personally in myGame of Thrones Death Pool project, and I chalked it up to "well I can put this on a website and others can deploy their own versions."

Sometimes this is necessary, but it's more work if it's not absolutely needed for your application.

My Version of an Ideal Deployment Interface¶

In an ideal world, I'd like these features in a deployment interface:

- Lets me use MetaMask directly for signing transactions* Remix ✔ | Truffle CLI ✕ | Custom Webapp ✔
-
    - A bonus win here is support for Ledger and Trezor wallets!
- I can use a framework/toolset/etc. that lets me build complex apps and supports complex deployment processes* Remix ✕ | Truffle CLI ✔ | Custom Webapp ✔
- I don't need to do a lot of work to go from finishing development to deploying* Remix ✔ | Truffle CLI ✔ | Custom Webapp ✕

Introducing: Truffle Teams Deployments

If you haven't heard about our latest tool,Truffle Teams , you really should check it out. Truffle Teams gives you zero-config Continuous Integration (CI) for your Truffle projects, and it also lets you monitor what's happening with your deployed contracts. Truffle Teams meshes the full Truffle Suite into a cohesive application that enables better collaboration within development teams (even teams of 1!).

The next big feature we're working on is Deployments. We already compile your contracts from your Truffle project for you when you make a commit to GitHub; why can't we migrate/deploy them as well? Further, Truffle Teams is a web application, allowing for easy integration withMetaMask .

We're looking at this from aDevOps perspective, and we're trying to create an extensible system that will support projects and teams ofany size .

Connect Your GitHub Repository to Truffle Teams¶

To get started with Truffle Teams, you must first connect your GitHub repository. This is a simple step, and we have instructions on how to do thishere .

Setup Your Migrations Scripts¶

If you haven't built a Truffle project, you should really give it a try; thePet Shop Tutorial is a great place to start as it gives a good survey of the different facets of Truffle, including themigration scripts. These scripts let you define the behavior of deploying your smart contracts using a language you're already familiar with. In the tutorial, you deploy a single contract, but it has the flexibility to do so much more (i.e. you can deploy one contract, get its address, and use that in the constructor when deploying another contract).

Setting up these scripts is part of the development lifecycle of creating a Truffle project, so I won't cover that here. But that's great! You should already be done with this step!!

Commit to GitHub¶

Welp. That was easy. 😄

Monitor the Status of Your Build¶

Head over to https://my.truffleteams.com and click on theBUILDS tab if you're not there already. You can see a build has been queued or has started to process. You can click on the repository name in the card to see more details. You can also get to this page by clicking on the yellow dot in GitHub next to your commit and pressingDetails .

All Your Tests Pass, Let's Deploy! (but only to a testnet just to be sure)¶

NOTE: This is just a sneak peek of features to come! You won't be able to follow along (yet, butstay tuned !). Great! Your builds are passing, and you're ready to deploy to a testnet (i.e. Ropsten) to see if everything works.

Now head over to theDEPLOYMENTS tab and select your repository.

You'll now see a screen that shows a list of builds for your commits on the left.

Deploying a build is as simple as hovering over the commit, pressing a button, and selecting which network you'd like to deploy to.

Truffle Teams will then start the migration/deployment process. Each transaction is sent to the web application and

MetaMask will prompt you to confirm transactions. By leveraging MetaMask, we can give youLedger andTrezor hardware wallet support too!

Happy With a Released Version? Promote it to Production (aka deploy to Mainnet)¶

But we're not done yet! You've finished your testing on Ropsten, and you're now ready to deploy to Mainnet. With the upcoming version of Teams, you can select that deployment and promote it to production (aka Mainnet) with just a couple of clicks.

And that's it! Pretty easy if you ask me.¶

Don't Believe Me? Be The First to Try it at My TruffleCon 2019 Workshop! ¶

I really hope you're going to TruffleCon; it's going to be awesome. Even more so because you canbe the first to try Truffle Teams Deployments in my workshop! In myPush It! Push It Real Good: Truffle Teams Intro & Deployments (9AM-12PM on Friday, Aug, 2nd) workshop, we'll cover this entire lifecycle: hooking up your repository, getting your first build to pass, deploying to Ropsten, and finally monitoring transactions made against your contracts (we didn't talk about this here; check out thedocs for more details on contract monitoring).

Hopefully there will be time for me to answer any questions you have, and maybe even help get your own Truffle project hooked up with Truffle Teams! If not, I will be available during the office hours at 5PM on Friday! I'm also always happy to chat anytime throughout the conference!