**Zksync Local Node**

**Introduction**

👀❗**IMPORTANT**  
This lesson is optional. If you encounter difficulties installing or understanding the required tools, feel free to proceed to the next section and continue using Anvil to test your smart contract locally.

In the previous lessons, we learned about deploying smart contracts with the forge create and forge script commands on our **local Anvil chain**. In this lesson, we will set up and run a **ZKsync local environment**.

**Local Node Setup**

To deploy locally on a ZKsync local chain, you'll need additional tools: Docker, Node.js, and zksync-cli.

1. **Docker**: Start the [Docker](https://www.docker.com/) daemon. On Mac OS, you can start it using the Docker application interface. On Linux, use commands like sudo systemctl start docker and sudo systemctl stop docker will manage Docker lifecycles. Verify the installation with docker --version and docker ps commands.
2. **Node.js and npm**: Install [Node.js](https://nodejs.org/en) and [npm](https://www.npmjs.com/). Follow the Node.js documentation to install the right version for your operating system. Verify the installations with npm --version and node --version commands.
3. **zksync-cli**: Once Docker and Node.js are installed, you can install the [zksync-cli](https://www.npmjs.com/package/zksync-cli) to manage your local ZKsync development environment. Run npx zksync-cli dev config to set up your configuration. Choose the in-memory node option for a quick startup without persistent state and avoid additional options like a portal or block explorer unless you want to explore them independently.

To start your local ZKsync node, run npx zksync-cli dev start. This command spins up a ZKsync node in Docker and runs it in the background. Verify the process is running with docker ps.

🗒️ **NOTE**  
If Docker isn’t running, the npx zksync-cli dev start command will fail. Ensure Docker is running before attempting to start the ZKsync node again.

**Deployment**

The ZKsync deployment process is similar to previous deployments. We will use the same commands, but this time, we will append the --zksync and --legacy flags. Note that the forge script command is not well supported in ZKsync, so we will use forge create instead.

**Conclusion**

Setting up a local ZKsync node involves a few additional tools, including Docker, Node.js, npm, and zksync-cli: they will help creating a robust ZKsync development environment and allowing test and deployment of smart contracts on a ZKsync local chain.