hardhat开发编译测试部署智能合约实操

* 项目搭建

首先，新建一个目录名为hardhat-tutorial，进入该目录，初始化一个node项目，执行npm init -y

文本

描述已自动生成

然后安装hardhat作为开发时依赖，执行

npm install -D hardhat

文本

描述已自动生成

接下来创建hardhat项目，执行

npx hardhat

会有两种选择：

1. 创建一个sample项目
2. 仅仅创建一个空的hardhat.config.js文件

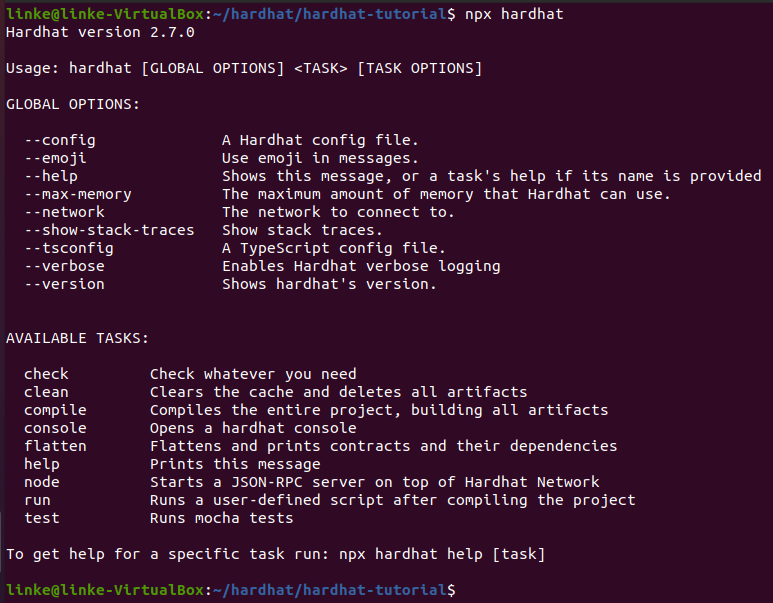
如果选择第二种，一个空的hardhat.config.js文件会被创建，然后我们可以根据自己的需求配置相关的依赖。

文本

描述已自动生成

这里我选择第二种，创建一个空的hardhat.config.js文件。创建hardhat项目成功之后会自动创建hardhat.config.js文件。

接下来如果想看hardhat能够为我们提供什么功能，有些什么命令，可以再次执行命令npx hardhat



在开发智能合约之前，还需要安装好以下几个依赖：waffle, ethers, chai，这些package用于连接以太坊节点以及智能合约的测试，执行以下命令

npm install -D @nomiclabs/hardhat-ethers ethers @nomiclabs/hardhat-waffle ethereum-waffle chai

文本

描述已自动生成

接下来需要在hardhat.config.js配置文件中引入waffle package

图形用户界面, 文本

描述已自动生成

只需要引入waffle，不需要引入ethers，因为waffle依赖于ethers，waffle就会引入ethers

同时，把solidity版本改为0.8.0

* 创建智能合约

当前的目录结构如下

文本

描述已自动生成

创建一个contracts目录用于存放智能合约代码，执行

mkdir contracts

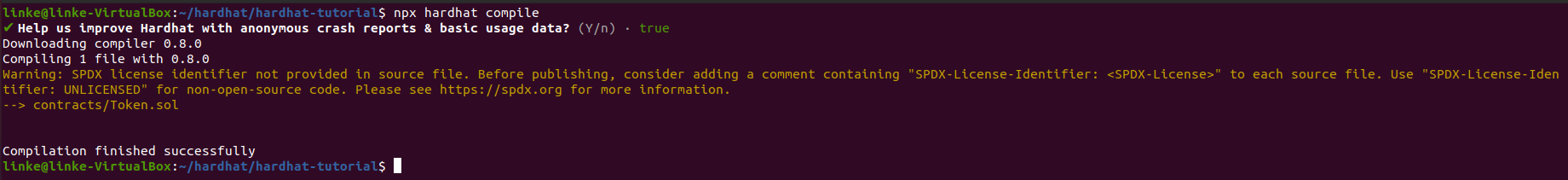
然后再在contracts下创建一个Token.sol智能合约，这是一个非常简单的token合约，内容为

文本

描述已自动生成

合约已经创建好了，首先编译一下，执行

npx hardhat compile



编译成功。

会看到多了两个目录

文本

描述已自动生成

文本

描述已自动生成

artifacts/contracts/Token.sol/Token.json文件包含合约编译后产生的abi和字节码。

接下来创建一个test目录，用于存放测试用例

mkdir test

再创建一个test/Token.js文件，用于存放Token.sol的测试用例，内容如下



beforeEach是指在执行每个测试用例之前要执行的操作，describe是一个测试套件（test suite），it是一个测试用例。可以看到，一共有2个测试套件，共4个测试用例。