Chapter 6

Building Your First Ethereum App

Episode 6.01

Preparing Your First Truffle Project

Building Your First App

- Start by doing!
- Limited coding knowledge is ok
- Start from the beginning

Creating a Truffle Project

- Identify project to Truffle
- Organizes activities into projects
- Project artifacts stored in directories
- To create a new project
  - Create a new directory
  - Ask Truffle to initialize it
- After creating, edit Truffle config file to configure your project

Episode 6.02

Writing a Simple Smart Contract

Simple Smart Contract Design

- A few things to consider
  - Pay a fee to access blockchain data
  - Can't change blockchain data
  - Data stored on the blockchain is available to everyone
- Spending time designing your app saves time later

Hello World Code

```
pragma solidity ^0.5.0;

contract HelloWorld {
    string private helloMessage = "Hello world";

    function getHelloMessage() public view returns
(string memory) {
        return helloMessage;
    }
}
```

Episode 6.03

Compiling Your Simple Smart Contract

Running Your Simple Smart Contract

- Write source code
- Compile
  - Translates into bytecode that the EVM can execute
- Deploy to Ethereum blockchain
- Call (invoke) a function

To download different versions of the Solidity compiler, go to:
https://github.com/ethereum/solc-bin/tree/gh-pages/bin

Episode 6.04

Deploying Code and Invoking Functions

- Deploying Code Commands

Command	truffle deploy --reset
Result	Deploys smart contract to your blockchain. In this case, since we've pointed Truffle to use the test blockchain, Ganache, it will deploy to the testing environment

Command	truffle console
Result	Launches Truffle console and the development environment

Command	HelloWorld.deployed().then(function(instance) {return instance });
Result	Points to the smart contract code

Command	HelloWorld.deployed().then(function(instance) {return instance.getHelloMessage() })
Result	Invokes the function to return “Hello world” message