한 번에 끝내는
블록체인 개발 A to Z

Chapter 3

Defi 기초 컨셉 구현

Chapter 3

Defi 기초 컨셉 구현

유니스왑v1 CPMM 구현(2)

Goal

CPMM 기반의 ETH<->ERC20 스왑 기능을 구현한다.

Swap

- 지난 클립에서 작성한 getOutputAmount를 활용하면 스왑 기능을 쉽게 구현 할 수 있다.
- Swap은 내가 입력한 InputAmount를 Exchange로 보내고 이에 상응하는 OutputAmount를 계산해서 나에게 보내주는 것이다.

```
function getOutputAmount(uint256 inputAmount, uint256 inputReserve, uint256 outputReserve)
    uint256 numerator = (inputAmount * outputReserve);
    uint256 denominator = (inputReserve + inputAmount);
    return numerator / denominator;
}
```

구현 및 테스트(CPMM)

---

git clone https://github.com/GrayWorld-io/lec_fc_defi

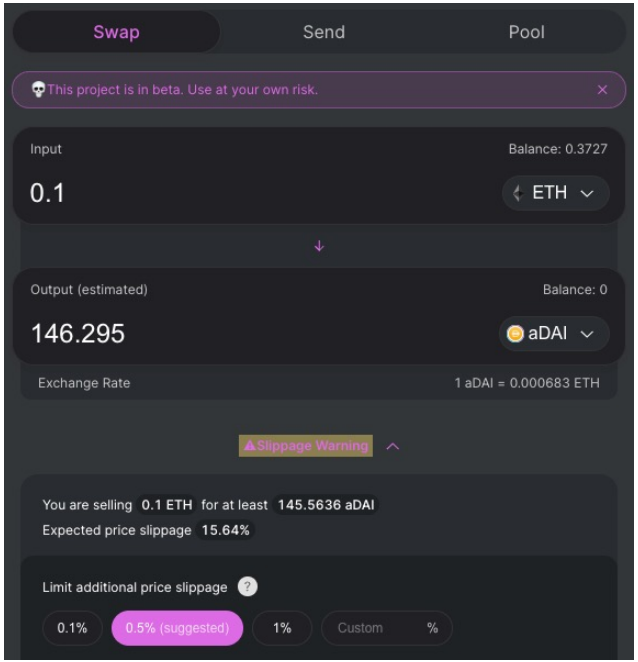
Swap

이 값은 프론트앤드에서 입력한 값이다.

```
// ETH -> ERC20
function ethToTokenSwap(uint256 _minTokens) public payable {

    uint256 tokenReserve = token.balanceOf(address(this));
    // calculate amount out
    uint256 outputAmount = getOutputAmount(msg.value, address(this).balance - msg.value, tokenReserve);

    require(outputAmount >= _minTokens, "Insufficient output Amount");

    //transfer token out
    require(token.transfer(msg.sender, outputAmount));
}
```

- getOutputAmount의 파라미터로 msg.value를 빼주는 이유는 ethToTokenSwap 함수가 실행 될 때 이미 ETH가 Exchange Contract로 넘어왔기 때문에 넘어오기 전에 풀에 있던 ETH 개수를 가져오기 위함이다.
- require 조건문은 minTokens에서 이미 슬리피지가 계산되어 입력되기 때문에 OutputAmount는 minToken보다 커야 한다.

Swap


Swap
Send
Pool
This project is in beta. Use at your own risk.
Input
Balance: 0.3727
0.1
ETH
Output (estimated)
Balance: 0
146.295
aDAI
Exchange Rate
1 aDAI = 0.000683 ETH
Slippage Warning
You are selling 0.1 ETH for at least 145.5636 aDAI
Expected price slippage 15.64%
Limit additional price slippage
0.1%
0.5% (suggested)
1%
Custom
%

Swap(Frontend)

```
if (swapType === ETH_TO_TOKEN) {
  const reserveETH = outputReserveETH
  const reserveToken = outputReserveToken

  if (amount && reserveETH && reserveToken) {
    try {
      const calculatedDependentValue =
        independentField === INPUT
          ? calculateEtherTokenOutputFromInput(amount, reserveETH, reserveToken)
          : calculateEtherTokenInputFromOutput(amount, reserveETH, reserveToken)
```

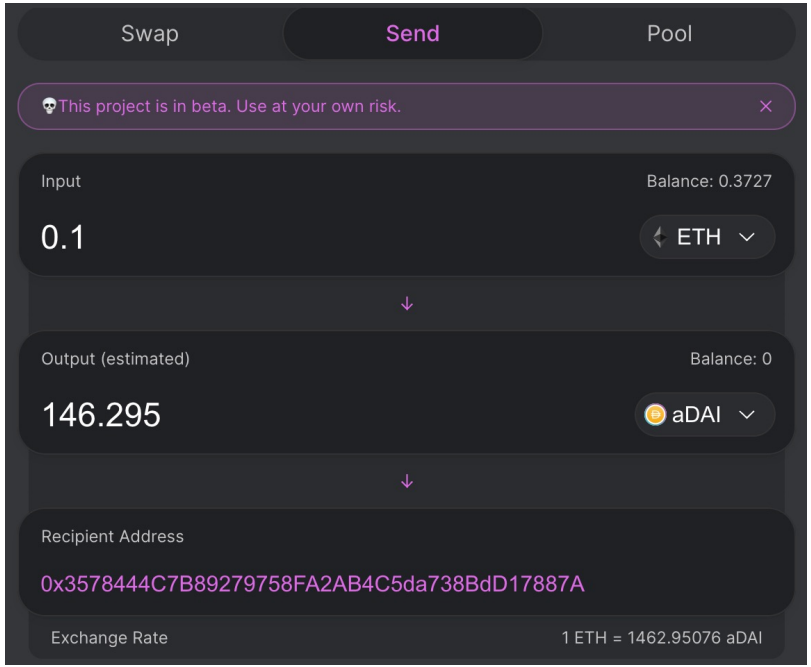
InputAmount를 통해 OutputAmount를 가져오기 때문에 independentField는 INPUT이다.

```
function calculateEtherTokenOutputFromInput(inputAmount, inputReserve, outputReserve) {
  const inputAmountWithFee = inputAmount.mul(ethers.utils.bigNumberify(997))
  const numerator = inputAmountWithFee.mul(outputReserve)
  const denominator = inputReserve.mul(ethers.utils.bigNumberify(1000)).add(inputAmountWithFee)
  return numerator.div(denominator)
}
```

Swap

```
function ethToTokenSwapInput(uint256 min_tokens, uint256 deadline) public payable returns (uint256) {
  return ethToTokenInput(msg.value, min_tokens, deadline, msg.sender, msg.sender);
}

function ethToTokenTransferInput(uint256 min_tokens, uint256 deadline, address recipient) public payable
  require(recipient != address(this) && recipient != address(0));
  return ethToTokenInput(msg.value, min_tokens, deadline, msg.sender, recipient);
}

function ethToTokenSwapOutput(uint256 tokens_bought, uint256 deadline) public payable returns(uint256) {
  return ethToTokenOutput(tokens_bought, msg.value, deadline, msg.sender, msg.sender);
}

function ethToTokenTransferOutput(uint256 tokens_bought, uint256 deadline, address recipient) public payable
  require(recipient != address(this) && recipient != address(0));
  return ethToTokenOutput(tokens_bought, msg.value, deadline, msg.sender, recipient);
}
```

Swap(Send)


Swap
Send
Pool
This project is in beta. Use at your own risk.
Input
Balance: 0.3727
0.1
ETH
Output (estimated)
Balance: 0
146.295
aDAI
Recipient Address
0x3578444C7B89279758FA2AB4C5da738BdD17887A
Exchange Rate
1 ETH = 1462.95076 aDAI

다음 강의

처음에 구현한 유동성 공급 함수를 개선한다.