**개발 function 정리**

|  |  |  |
| --- | --- | --- |
| **Function** | **Interface** | **Description** |
| createPair(address tokenA, address tokenB) external returns (address pair) | IDelioswapFactory | create the token pair |
| getPair(address tokenA, address tokenB) external view returns (address pair) | IDelioswapFactory | Returns the address of the pair for tokenA and tokenB |
| getReserves() external view returns (uint112 reserve0, uint112 reserve1, uint32 blockTimestampLast); | IDelioswapPair | Returns the reserves of token0 and token1 pairs created |
| swapExactETHForTokens(uint amountOutMin, address[] calldata path, address to, uint deadline) external payable returns (uint[] memory amounts) | IDelioswapRouter01 | Swaps an exact amount of tokens for as much ETH as possible |
| swapExactTokensForETH(uint amountIn, uint amountOutMin, address[] calldata path, address to, uint deadline) external returns (uint[] memory amounts) | IDelioswapRouter01 | Swaps an exact amount of ETH for as many output tokens as possible |
| approve(address spender, uint value) external returns (bool) | IDelioswapPair | approving tokens to be used by spender |
| allowance(address owner, address spender) external view returns (uint) | IDelioswapPair | Allow an address to give an allowance to another address to be able to retrieve tokens from it |
| totalSupply() external view returns (uint) | IDelioswapPair | Returns the amount of LP tokens in existence |
| addLiquidityETH(address token, uint amountTokenDesired, uint amountTokenMin, uint amountETHMin, address to, uint deadline) external payable returns (uint amountToken, uint amountETH, uint liquidity) | IDelioswapRouter01 | Adds liquidity to an ERC-20⇄WETH pool with ETH |
| removeLiquidityETH(address token, uint liquidity, uint amountTokenMin, uint amountETHMin, address to, uint deadline) external returns (uint amountToken, uint amountETH) | IDelioswapRouter01 | Removes liquidity from an ERC-20⇄WETH pool and receive ETH. |
| buyTokens(address \_investor) public payable | Sale | Provides Service to end user, they can purchase Security token from there metamask wallet account |

1. **CreatePair-** If you add liquidity, you always need to match it with two pairs. So, for example you need to add your Token A on the one side, and Ether on the other side of the equation. If the pair is not in the system yet, then it will be deployed. Deployment works, using the address of the factory, the address of Token A and the address of Token B, to deploy the pair.
   1. FactoryContract-it implements the IDelioswapFactory interface and define createPair() function
   2. function createPair(address tokenA, address tokenB) external returns (address pair)로 사용- create the token pair that is sold and bought for tokenA and tokenB if one doesn’t exist already.
   3. token과 token의 pair를 만들어줌 - Create pair and emits event PairCreated(). Emitted each time a pair is created via [createPair](https://uniswap.org/docs/v2/smart-contracts/factory/#createpair).
2. **getPair-** Returns the address of the pair for tokenA and tokenB, if it has been created, else address (0)
   1. FactoryContract**-** it implements the IDelioswapFactory interface and getPair() function
   2. getPair(tokenAaddress, tokenBaddres)**-** Returns the address of the pair for tokenA and tokenB, if it has been created, else address(0)
   3. token과 token이 pair**-** unique address of the pair for tokenA and tokenB
3. **getReserves-** Returns the reserves of token0 and token1 pairs created
   1. PairContract에 있음- it implements the IDelioswapPair interface and getReserves() function
   2. getReserves()- Used to get the reserves of token0 and token1 for which pair created
   3. Pair에 대한 Reserves를 리턴함 ()- Returns the reserves of token0 and token1 for which pair created
4. **swapExactToeknsForETH-** Swaps an exact amount of tokens for as much ETH as possible
   1. Router2Contract에 있음it implements router interface and define swapExactToeknsForETH() function
   2. swapExactTokensForETH(amount, min, path, account, deadline)- Swaps an exact amount of tokens for as much ETH as possible, along the route determined by the path, deadline refers to Unix timestamp after which the transaction will revert.
   3. Token을 주고 ETH를 받는 The input token amount and all subsequent output token amounts.
5. **swapExactETHForTokens**- Swaps an exact amount of ETH for as many output tokens as possible, along the route determined by the path.
   1. Router2Contract에 있음it implements router interface and define swapExactETHForTokens() function
   2. swapExactETHForTokens(min, path, account, deadline) Swaps an exact amount of ETH for as many output tokens as possible, along the route determined by the path, deadline refers to Unix timestamp after which the transaction will revert.
   3. ETH를 주고 token을 받는 swap- The input token amount and all subsequent output token amounts
6. **Approve** -It is used for approving tokens to be used by spender. It allows spender to either transfer the tokens or burn those tokens from Approver.
   1. PairContract에 있음it implements the IDelioswapPair interface and define Approve () function
   2. approve(address, amount)-approving tokens to be used by spender.
   3. msg.sender가 address에게 amount를 허락해줌 -address refers to the address of token and amount refers to number of tokens
7. **allowance**- Allow an address to give an allowance to another address to be able to retrieve tokens from it.
   1. PairContract에 있음- it implements the IDelioswapPair interface and define Approve () function
   2. Allowance(addressA, addressB) The ERC-20 standard allow an address to give an allowance to another address to be able to retrieve tokens from it.
   3. addressA가 addressB에게 얼마나 approve한지 값을 리턴 Allow an addressA to give an allowance to another address and returns the remaining number of tokens that the spender will be allowed to spend on behalf of owner.
8. **totalSupply-** Returns the amount of LP tokens in existence for specific pair in the pool
   1. PairContract에 있음- it implements the IDelioswapPair interface and define Approve () function
   2. totalSupply()- Provide amount of LP tokens in existence for particular pair in the pool
   3. pair의 LP토큰 총 발행량을 리턴- LP tokens for particular pair
9. **addLiquidityETH** - Adds liquidity to an ERC-20⇄WETH pool with ETH.
   1. Router2Contract에 있음 it implements the IDelioswapRouter01 interface and define addLiquidityETH () function
   2. addLiquidityETH(tokenAddresss, tokenAmount, tokenMin, ETHMin, account, deadline)- Adds liquidity to an ERC-20⇄WETH pool with ETH.
   3. ETH와 tokenAmount를 저장함In return get The amount of token sent to the pool, The amount of ETH sent to the pool and The amount of liquidity tokens minted.
10. **removeLiquidityETH-** Removes liquidity from an ERC-20⇄WETH pool and receive ETH.
    1. Router2Contract에 있음 it implements the IDelioswapRouter01 interface and define removeLiquidityETH () function
    2. removeLiquidityETH(tokenAddress, LPTokenAmount, outputTokenMin, outputETHMin, account, deadline)- Removes liquidity from an ERC-20⇄WETH pool and receive ETH and token
    3. LPToken을 주고 ETH와 Token을 받음 LPToken - The amount of liquidity tokens to remove and get back equilant amount of ETH and desired Token
11. **buyTokens**
    1. SaleContract에 있음- Provides service to admin where they can sell Security tokens to the market
    2. buyToken(account)- Provides Service to end user, they can purchase Security token from there metamask wallet account
    3. ETH를 주면 발행된 token을 지급함- Currently supports ETH to purchase STO Token