Fall 2024

SEC-205: Distributed Ledger and Blockchain

Lab/Practical Session 1: Introduction to Solidity

Charnon Pattiyanon, Ph.D.

charnon@cmkl.ac.th

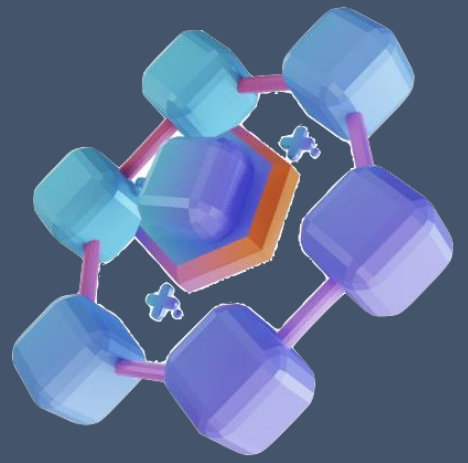
Assistant Director of IT and Instructor

Department of Artificial Intelligence and Computer Engineering (AiCE)

CMKL University

What is Solidity?

- **Programming Language** that creates smart contracts and instructs them how to behave.



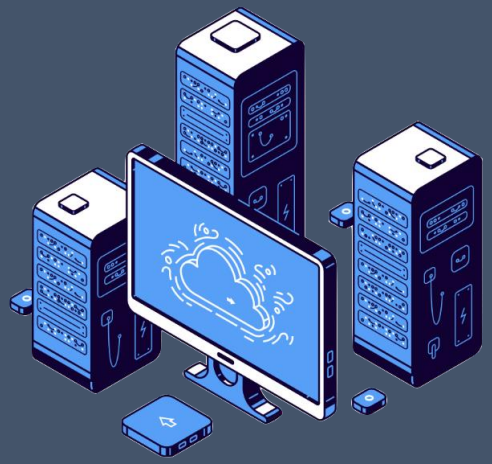
Code **Smart Contract** **Blockchain**

Web 2.0 Applications vs. Web 3.0 Applications

Web 2.0 Applications

User

Applications



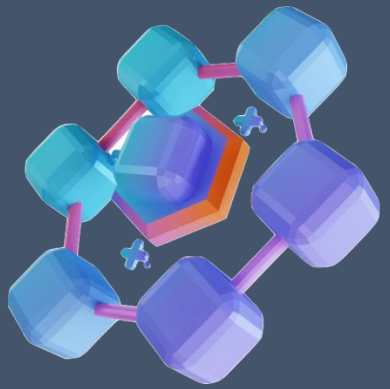
Server

Web 3.0 Applications

User

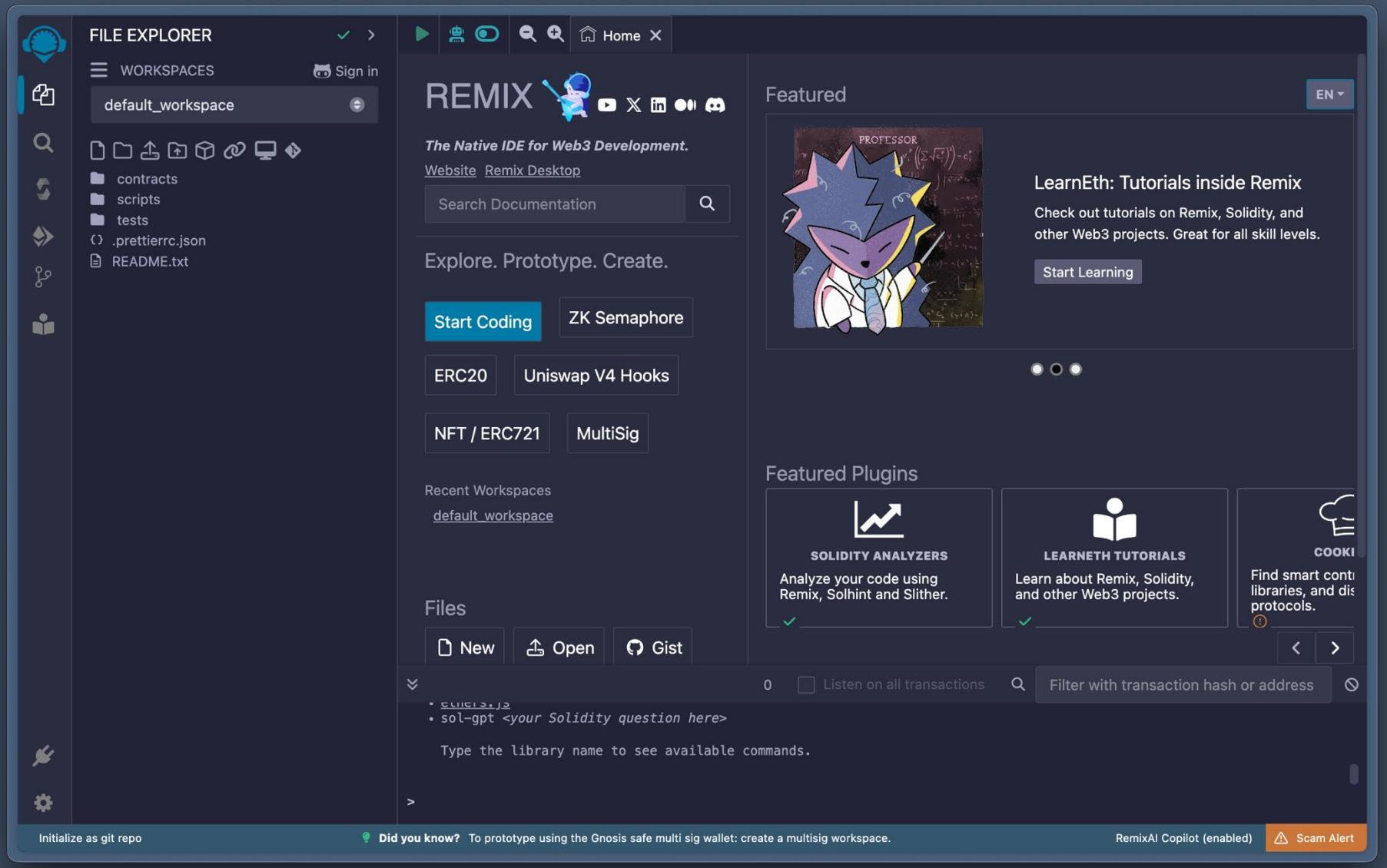
Crypto Wallet

DApp



Blockchain

Solidity Integrated Development Environment (IDE)



Your First Smart Contract

Hello World Smart Contract!

Variables

- **Address**

```
pragma solidity ^0.8.26

contract Fundamentals {

  // This is just a string
  address userAddress = 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4;

}
```

- **String**

```
pragma solidity ^0.8.26

contract Fundamentals {

  // This is just a string
  string message = "Hello Solidity";

}
```

Bool

```
pragma solidity ^0.8.26

contract Fundamentals {

  // Basically a speification of true or false
  bool isReady = true;

}
```

Variables

- **UINT (Unsigned Integer)**

```solidity
pragma solidity ^0.8.26

contract Fundamentals {

  // Small number for the number of days
  // in a week
  uint8 WEEK = 7;


  // A number represents 1 days?
  uint256 MAX_WAIT_TIME = 1 days
}
```

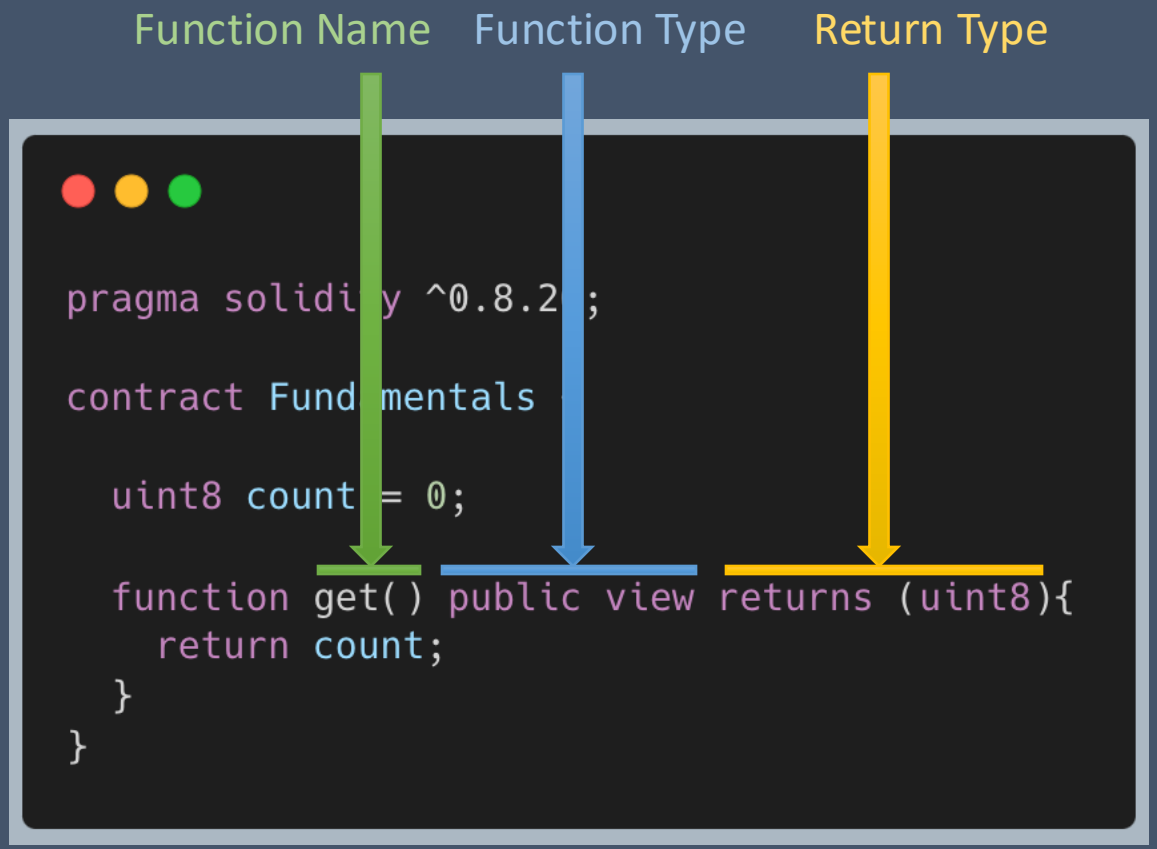
Uint8 (8 bits)	0 to 255
Uint16 (16 bits)	0 to 65,536
Uint128 (128 bits)	0 to 340,282,366,920,938,463,463,374,607,431,768,211,456
Uint256 (256 bits)	0 to 57,896,044,618,658,097,711,785,492,504,343,953,926,634,992,332,820,282,019,728,792,003,956,564,819,967

Simple Calculator Smart Contract

Variables and Functions

Functions

- It is just simply a repeatable task that you don’t want to rewrite it many times.



Function Visibility

- **Public** function can be used internally and externally.

```
pragma solidity ^0.8.26;

contract Fundamentals {

  uint8 count = 0;

  function get() public view returns (uint8){
    return count;
  }
}
```

- **Private** function can be used within the contract only.

```
pragma solidity ^0.8.26;

contract Fundamentals {

  uint256 private myPrivateVar;

  function myPrivateFunction() private {
    // Function Implementation
  }
}
```

Function Visibility

- **Internal** function can be used within the contract and other inheriting contracts.

```solidity
pragma solidity ^0.8.26;

contract BasicCalculator {

  uint256 public result;

  function add (uint256 a, uint256 b) internal {
    result = a + b;
  }

  function subtract (uint256 a, uint256 b) internal {
    result = a - b;
  }
}

contract AdvancedCalculator is BasicCalculator {
  function multiply (uint256 a, uint256 b) internal {
    result = a * b;
  }

  function divide (uint256 a, uint256 b) internal {
    result = a / b;
  }

  function performOperation(uint256 a, uint256 b, uint8 operation) public {
    if (operation == 0) add(a,b);
    else if (operation == 1) subtract(a,b);
    else if (operation == 2) multiply(a,b);
    else if (operation == 3) divide(a,b);
    else revert("Invalid Operation");
  }
}
```

Function Visibility

- **External** function can be accessed only from external contracts or accounts.

```
pragma solidity ^0.8.26;

contract Calculator {

  uint256 public result;

  function add (uint256 num) external {
    result += num;
  }

  function subtract (uint256 num) external {
    result -= num;
  }

  function multiply (uint256 num) external {
    result *= num;
  }

  function get() external view returns (uint256){
    return result;
  }
}
}
```

End of the Lab! 

Please feel free to ask any questions.

If you need further discussion, please contact me:

- Email me at charnon@cmkl.ac.th
- Appoint me for 1-on-1 discussion during the office hours.