LESSON SC 5 – `msg.sender`

University of West Attica
Department of Electrical and Electronics Engineering
Ioannis Christidis
Christoforos Kachris

Support by Ethereum Foundation ESP

What will we accomplish!

In this lesson we will learn about two concepts:

1. `constructor`
2. `msg.sender`

We will continue on our `ICR.sol`.

Owner of Cars = Owner of the Smart Contract

In a previous lesson, we talked about how the owner of the cars is able register their cars inside the SC.

We will build the *ICR SC to only have one owner*. (Later in the course we will expand our logic for multiple owners)

To interact with SC, the owner uses their account that has an `address`.

We could save their `address` in the SC so that we can use it in the ICR functionality.

For example, *the* `registerCar` *function should only be used by the owner*.

For now, let's assume that *the owner is the deployer of the SC*.

So we will create a `public address` *state variable* called `owner`.

```
    address public owner;
```

Access the Deployer the Smart Contract

If you remember from a previous lesson, we mentioned that *when a transaction is made on the blockchain, there is a* `from` *address in the transaction details*.

This `from` address is the address that made the transaction.

Knowing that, *we can access this address inside our SC with the unique global variable,* `msg.sender` that solidity has available.

And since we know that *the* owner *will be the address that deploys the SC*, we can *get their address in the* `constructor`.

The `constructor`, as we saw before, is *a special function that runs only once during the deployment of the SC.* If a SC does not have a `constructor`, the default behavior is equivalent to an empty `constructor`.

So, in the constructor we will make the `owner` address equal to the `msg.sender`.

```
constructor() {
    owner = msg.sender;
}
```

Outro

Nice, we learned about the `msg.sender` and the `constructor`.

Next, we will go over some *access control concepts*.