Cryptozombies

Chapter 3: Advanced Solidity Concepts

Contents

Immutability of Contracts

Contracts can't be changed once deployed, so be careful what you deploy since if you need to patch-fix something, you need to deploy another contract and inform the users of your old contract of the address of the new contract.

Ownable

The Ownable contract allows us to add 'guards' to our methods and check if the caller is actually the owner.

Function Modifiers

A modifier is a piece of code that can be executed before executing a method. You simply add the modifier to the method signature and the piece of code inside the modifier is executed, useful for 'guards'.

Struct packing

Packing smaller size variables together inside structs allow us to save gas in the long run by making the variables take up less storage space. This doesn't work outside of structs

Time in Solidity

'now' returns the current UNIX timestamp since Jan. 1st 1970. ("year 2038" problem also explained). Now we must make a design decision, do we either spend more gas and make the contract 'future-proof' or stick to the 'traditional' way of storing time, uint32.

Zombies Cooldowns

We added cooldowns for our zombies, so they can't feed every time they want.

Abuse of public functions

We need to take time to study our external and public functions to see if we can find a way to 'abuse' them. We found an exploit so we make it internal.

_;

Weirdest syntax ever, but whatever. Used in modifiers. This returns the control to the method in which the modifier was used in.

Zombie modifiers

We created our own guards (aka modifiers, I still prefer to call them guards), so we can check the zombie's level for some special power-ups.

View functions

They don't cost gas, becayse they don't actually change anything on the blockchain. Pretty useful.

Storage

Well, it's pretty expensive, so instead of doing memory-wise expensive writes, let's do more expensive reads. This will save us money in the long run.

For Loops

Basically, JS for loops