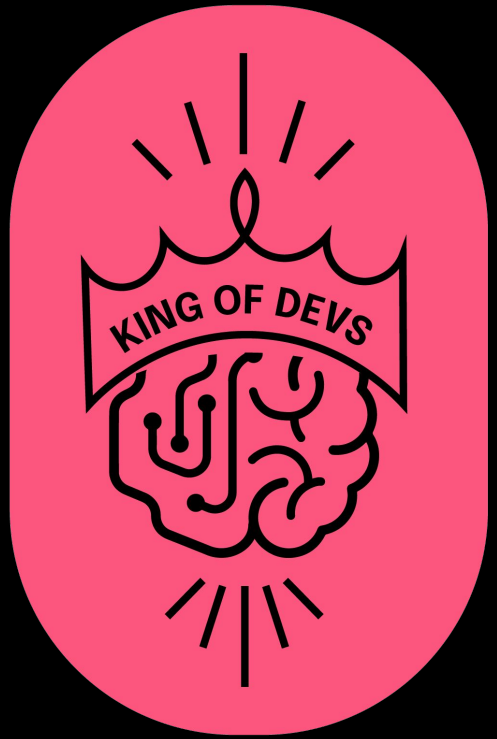
KING OF DEVS

KING OF



WORKSHOP DAY 1

Useful Open Zeppelin contracts

Tokens

- ERC20
- ERC721
- ERC1155

Access

- Ownable

Security

- Pausable

Utils

- Enumerable Set

Access

- Ownable

```
abstract contract Ownable is Context {
    address private _owner;
```

```
modifier onlyOwner() {
    _checkOwner();
    _;
}
```

Access

- Ownable

```
function _checkOwner() internal view virtual {
    require(owner() == _msgSender(), "Ownable: caller is not the owner");
}
```

Security

- Pausable

```
modifier whenNotPaused() {
    _requireNotPaused();
    _;
}
```

```
function _requireNotPaused() internal view virtual {
    require(!paused(), "Pausable: paused");
}
```

Security

- Pausable

```
modifier whenPaused() {
    _requirePaused();
    _;
}
```

```
function _requirePaused() internal view virtual {
    require(paused(), "Pausable: not paused");
}
```

Tokens - ERC20

```
import "@openzeppelin/contracts/token/ERC20/ERC20.sol";

contract GLDToken is ERC20 {
    constructor(uint256 initialSupply) ERC20("Gold", "GLD") {
        _mint(msg.sender, initialSupply);
    }
}
```

- https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/ERC20.sol

ERC20 - Extensions

- ERC20Burnable

```
function burn(uint256 amount) public virtual {
    _burn(_msgSender(), amount);
}
```

```
function burnFrom(address account, uint256 amount) public virtual {
    _spendAllowance(account, _msgSender(), amount);
    _burn(account, amount);
}
```

ERC20 - Extensions

- ERC20Pausable

```
function _beforeTokenTransfer(
    address from,
    address to,
    uint256 amount
) internal virtual override {
    super._beforeTokenTransfer(from, to, amount);

    require(!paused(), "ERC20Pausable: token transfer while paused");
}
```

ERC20 - Presets

- ERC20PresetFixedSupply

```
constructor(
    string memory name,
    string memory symbol,
    uint256 initialSupply,
    address owner
) ERC20(name, symbol) {
    _mint(owner, initialSupply);
}
```

Nota: Es burnable además

ERC20 - Presets

- ERC20PresetMinterPauser

```
function mint(address to, uint256 amount) public virtual {
    require(hasRole(MINTER_ROLE, _msgSender()), "ERC20PresetMinterPauser: must have minter role to mint");
    _mint(to, amount);
}
```

```
function pause() public virtual {
    require(hasRole(PAUSER_ROLE, _msgSender()), "ERC20PresetMinterPauser: must have pauser role to pause");
    _pause();
}
```

Nota: Es burnable además

Utils

- T= AddressSet (t= *address*)
- T= Bytes32Set (t= *bytes32*)
- T= UintSet (t= *uint*)

Library Enumerableset

```
function add(T,t) bool
function remove(T,t) bool
function contains(T,t) bool
function length(T) uint256
function at(T,uint256) t
```

GRACIAS