Payable

Functions and addresses declared `payable` can receive `ether` into the contract.

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.3;

contract Payable {
    // Payable address can receive Ether
    address payable public owner;

    // Payable constructor can receive Ether
    constructor() payable {
        owner = payable(msg.sender);
    }

    // Function to deposit Ether into this contract.
    // Call this function along with some Ether.
    // The balance of this contract will be automatically updated.
    function deposit() public payable {}

    // Call this function along with some Ether.
    // The function will throw an error since this function is not payable.
    function notPayable() public {}

    // Function to withdraw all Ether from this contract.
    function withdraw() public {
        // get the amount of Ether stored in this contract
        uint amount = address(this).balance;

        // send all Ether to owner
        // Owner can receive Ether since the address of owner is payable
        (bool success, ) = owner.call{value: amount}("");
        require(success, "Failed to send Ether");
    }

    // Function to transfer Ether from this contract to address from input
    function transfer(address payable _to, uint _amount) public {
        // Note that "to" is declared as payable
        (bool success, ) = _to.call{value: _amount}("");
        require(success, "Failed to send Ether");
    }
}
```

Try on Remix