// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

// Parent Contract

contract ParentContract {

address public owner;

// Constructor

constructor() {

owner = msg.sender; // Set contract creator as owner

}

// Public function

function getOwner() public view returns (address) {

return owner;

}

// Visibility: Internal function

function \_getBalance() internal view returns (uint) {

return owner.balance; // Return owner's balance

}

// Error handling with require

function transferOwnership(address newOwner) public {

require(newOwner != address(0), "New owner address invalid");

owner = newOwner;

}

}

// Child Contract inheriting ParentContract

contract ChildContract is ParentContract {

// Payable function

function deposit() public payable {

// Payable function to receive Ether

}

// Public function using internal function

function showOwnerBalance() public view returns (uint) {

return \_getBalance(); // Access internal function from parent contract

}

// Struct definition

struct Person {

string name;

uint age;

}

// Mapping

mapping(address => Person) public people;

// Function to add people to mapping

function addPerson(string memory \_name, uint \_age) public {

people[msg.sender] = Person(\_name, \_age);

}

// Enum definition

enum Status { Active, Inactive }

// Fixed array

uint[5] public fixedArray = [1, 2, 3, 4, 5];

// Dynamic array

uint[] public dynamicArray;

// Special arrays (bytes and string)

bytes public byteArray;

string public textString = "Hello, Solidity";

// Function to add to dynamic array

function addToArray(uint \_value) public {

dynamicArray.push(\_value);

}

}

A screenshot of a computer

Description automatically generated