# Effect of state variables and location of the variables on gas consumption

Solidity doesn’t allow declaring the location of the variable or objects outside the function and it is defaulted to Storage.

## Gas cost analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable** | **Gas Cost** | **Transaction Cost** | **Exec cost** |
| Deployed | 388579 | 337894 | 337894 |
| Function executed SetBooks | 127417 | 110797 |  |
| getAuthor |  | 24555 |  |
| getBookID |  | 23485 |  |
| getPrice |  | 23507 |  |
| Unit8, uint16 | 410638 | 357076 | 357076 |
| setBooks | 102444 | 89801 |  |
| getAuthor |  | 24555 |  |
| getBookID |  | 23527 |  |
| getPrice |  | 23599 |  |
| getTitle |  | 24599 |  |
| Added one more instance of book as bk1 | 410638 |  |  |
| Added one storage var to function getTitle | 412846 | 358996 |  |
| setBooks | Same as above |  |  |
| Executed getTitle |  | 24618 |  |
| Copied book struct | 412846 | 358996 |  |
|  |  |  |  |
| Added uint128, uint256, uint128 | 388551 | 337870 |  |
| Added uint128, uint128, uint256 | 388579 | 337894 |  |
| Added one string(32) storage var bkTitle | 413177 | 359284 |  |
| Increased string size by 5 chars | 415413 | 361228 |  |
|  |  |  |  |
| **GASCALC01** |  |  |  |
| Deployed - Both functions internal | 102613 | 89228 |  |
|  |  |  |  |
| Reading state vars Once |  |  |  |
| Deployed – Both functions external | 287310 | 249834 |  |
| Deployed – Both functions Public | 287296 | 249822 |  |
| Reading the stat variables twice |  | 25077 |  |
| Reading the stat variables Once |  | 25012 |  |
| GasCalc02 |  |  |  |
| Passing parameter as Memory (pure) | 277944 | 241690 |  |
| As calldata (pure) | 242415 | 210795 |  |
| Covert function from pure to view | 242429 | 210807 |  |
|  |  |  |  |
| Using uint/uint256 | 132137 | 114901 |  |
| Uint8 | 136551 | 118740 |  |
|  |  |  |  |

## Observations

1. Uint8 uses more the gas than uint or uint256
2. Memory variable cost more than calldata
3. Pure cost less than view functions
4. String: More the size more gas. Keep string value to minimum and multiple of 32 bytes
5. Declaring the function as internal costs much less gas than Public so if function is only required to be accessed internally then functions shall be declared internal only.
6. State variable shall be read and written only when required. Store them in memory variables if required multiple times in calculations. Accessing the storage again and again costs more gas.

//SPDX-License-Identifier:MIT

pragma solidity 0.8.6;

contract gascalc01 {

    uint percentage = 30;

    function splitAmountToOwnerAndSeller(uint amount) public view returns (uint amountForSender, uint amountForOwner)

    {

        amountForSender = (amount \* (100 - percentage)) / 100;

        amountForOwner = (amount \* percentage) / 100;

    }

    function splitAmountToOwnerAndSellr1(uint amount) public view returns (uint amountForSender, uint amountForOwner)

    {

        uint ownerPercentage = percentage;

        amountForSender = (amount \* (100 - ownerPercentage)) / 100;

        amountForOwner = (amount \* ownerPercentage) / 100;

    }

}

//SPDX-License-Identifier:MIT

pragma solidity 0.8.6;

contract structure{

    struct Book{

        string title;

        string author;

        uint bookID;

        uint price;

    }

    Book book;

    uint128 price1;

    uint256 price2;

    uint128 price3;

    string bkTitle;

//    Book bk1;

//    string memory btitle;     Gives error, syntax doesn't allow location

//    string bTitle;

function setBkTitle() public {

        bkTitle = "Learning Blockchain is mandatory in 20";   // 32 chars

}

    function setBook() public {

        book= Book("Blokchain for beginners","Ineuron",4,1000);

    }

    function getTitle() public view returns(string memory){

        //  string storage bTitle;

        //  bTitle = book.title;

        return book.title;

    }

    function getAuthor() public view returns(string memory){

        return book.author;

    }

    function getBookId() public view returns(uint){

        return book.bookID;

    }

    function getPrice() public view returns(uint){

        return book.price;

    }

}

//SPDX-License-Identifier:MIT

pragma solidity 0.8.6;

contract structure{

//    string acctName = "iNeuron";

    int mrksPct = 90;

    function dispName() public view returns(int){

        return mrksPct;

    }

}