**Solidity** 存储位置

**1** 、**Solidity** 有五个存储位置：

**全局storage**: 合约中的全部状态变量；

**局部storage：**在函数中声明的存储位置为storage的变量；

**内存(memory)：**在函数中声明的存储位置为memory的变量,；

**stack:** 函数中声明的值类型的变量；

**calldata:** 外部函数 (external) 参数的存储方式强制是 calldata，并且只读;

引用类型：

**1、数组(T[]或者T[N])、**

**2、结构体(struct)、**

**3、字符串(string)、**

**4、动态字节数组(bytes)**

**作为局部变量时，上述4项才能声明成storage/memory 。**

**映射虽然是引用类型，但只能在存放storage中**

2、函数参数类型及其对函数的影响：

外部函数的参数存储在 calldata，可以传递数组(array)，不能传结构体(struct)和映射(mapping)，都是传值，且calldata只读；**外部函数现在可以返回变长数组。**

public/internal/private 函数(参数：Memory 数组)：

1、调用时可传入 Storage 数组， 传值效果，创建 临时拷贝；

2、传入Memory 数组，外部调用时为传值；内部调用时为**传引用**。

Memory 参数可读写， 赋值给局部 memory 变量是引用关系，赋值给 storage 状态 变量是拷贝 关系，不能赋值给 storage 局部变量(因指针无法指向) .

internal/private 函数(**Storage 数组参数**)，调用函数时必需传入**Storage 数组变量**， 为传**引用**。

**3、存储位置对【变量能够执行操作】的影响**

1、Storage 状态变量数组可使用数组全部既定操作，可以 new、可以字面量数组赋值，变长数组

可以修改 length、可以 push。

2、Storage 局部变量（数组/结构）只能是引用指针，指向 Storage 存储的数组/结构，不能 new、

不能字面量数组赋值。（所以 Storage 局部变量要明确指向状态变量/Storage 参数，否则指针

可能出现意外）。

3、Memory 数组采用 new 定长后（不能用字面量数组赋值定长），不能改 length，不能 push。

Memory 数组元素不能存映射。

4、外部函数 Memory 数组参数接收到的是 ABI 编码数组。

多维数组：申明 uint[][5]，读取元素 uint[2][1] 注意一维二维下标位置是反的。

数组 push 操作是主要消耗 gas，多维数组操作特别耗 gas，无法支持大型多维数组。

bytes 和 string 类型的变量是特殊的数组，外部函数 calldata 参数是 abi 紧打包。

string 是 utf-8 字符，不支持 length 和下标访问，而 bytes 的 length 和下标访问都支持

可使用 bytes(s).length / bytes(s)[7]='x'; 使用 bytes()转换后，只是得到一个引用。