7月18日

①

Solidity 存储位置

1、Solidity 有四个存储位置：

storage: 存储

memory: 内存

stack：堆栈（包含在内存中）

calldata: 调用数据（包含在内存中）

2、Solidity 变量分为：

局部变量：

全局变量：

3、Solidity 变量类型分为：

值类型

类型关键字	类型	说明
bool	布尔	
int/ uint	整数	从 uint8 到 uint256， 从 int8 到 int256
fixed/ ufixed	定点数	fixed/ ufixed
address： address payable：	地址	address：持有 20 个字节的值（以太坊地址的大小）。 address payable：与相同 address，但具有额外的成员 transfer 和 send。
	合约	
bytes1 ， bytes2 ， bytes3，...，bytes32	固定大小的字节数组	
	地址字面量	

	字符串字面量	
	十六进制字面量	
	Unicode 字面量	

引用类型：

Type[]	动态数组
Bytes	动态大小的字节数组
String	动态大小的 UTF-8 编码的字符串
Struct	结构体
Mapping	映射

数据存储方式细节总结：

变量作用范围	变量类型	存放位置	
状态变量	值类型	Storage（存储）	----------
	引用类型		
局部变量	值类型	Stack（堆栈）	
	引用类型	Storage（存储）	

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

内部函数（internal）参数（包括返回的参数）的存储方式默认是 memory；

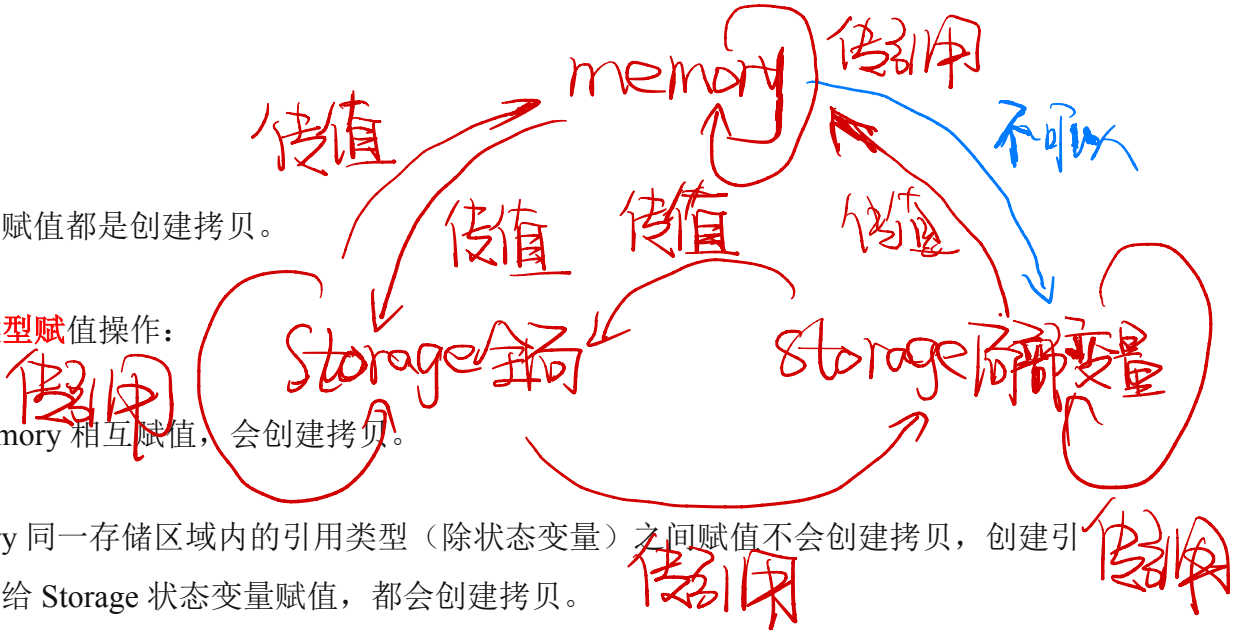
值类型声明时，不允许指定 storage/memory；

映射只能存储在 storage 中；

数组/结构体作为局部变量，才能声明成 storage/memory 。

2、存储位置对赋值的影响：

，值类型之间赋值都是创建拷贝。



下面指**引用类型赋**值操作：

Storage 与 memory 相互赋值，会创建拷贝。

storage/memory 同一存储区域内的引用类型（除状态变量）之间赋值不会创建拷贝，创建引用；所有变量给 Storage 状态变量赋值，都会创建拷贝。

Storage 状态赋值给 Storage 局部变量：**不**创建拷贝，仅创建一个**引用。**

Memory 引用类型赋值给 Memory 引用类型：**不**创建拷贝，仅创建一个**引用。**

memory 变量不能给 storage 局部变量赋值，但可以给 storage 状态变量赋值。

3、对函数参数及返回值的影响：

外部函数的参数存储在 calldata，可以传递数组，不能是结构和映射，都是传值，都是**只读**。

外部函数现在版本可以返回变长数组了（现在很多网上资料说不能返回变长数组，是错误的）。

public/internal/private 函数（Memory 数组参数），调用时可填 Storage 数组，**传值**效果，创建临时拷贝；填 Memory 数组，外部调用**传值**效果（拷贝），内部调用**传址**效果（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），调用函数时必需填 Storage 数组变量，**传址**效果（引用）。

4、数组

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。

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）。

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。

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。