

『0013』 - Solidity Types - 固定大小字节数组(Fixed-size byte arrays)

孔壹学院：国内区块链职业教育领先品牌

作者：黎跃春，区块链、高可用架构工程师

微信：liyc1215 QQ群：348924182 博客：<http://liyuechun.org>

固定大小字节数组(Fixed-size byte arrays)

固定大小字节数组可以通过 `bytes1` , `bytes2` , `bytes3` , ..., `bytes32` 来进行声明。

PS: `byte` 的别名就是 `byte1` 。

- `bytes1` 只能存储 一个 字节，也就是二进制 8位 的内容。
- `bytes2` 只能存储 两个 字节，也就是二进制 16位 的内容。
- `bytes3` 只能存储 三个 字节，也就是二进制 24位 的内容。
-
- `bytes32` 能存储 三十二个 字节，也就是二进制 $32 * 8 = 256$ 位的内容。

```
pragma solidity ^0.4.4;

contract C {

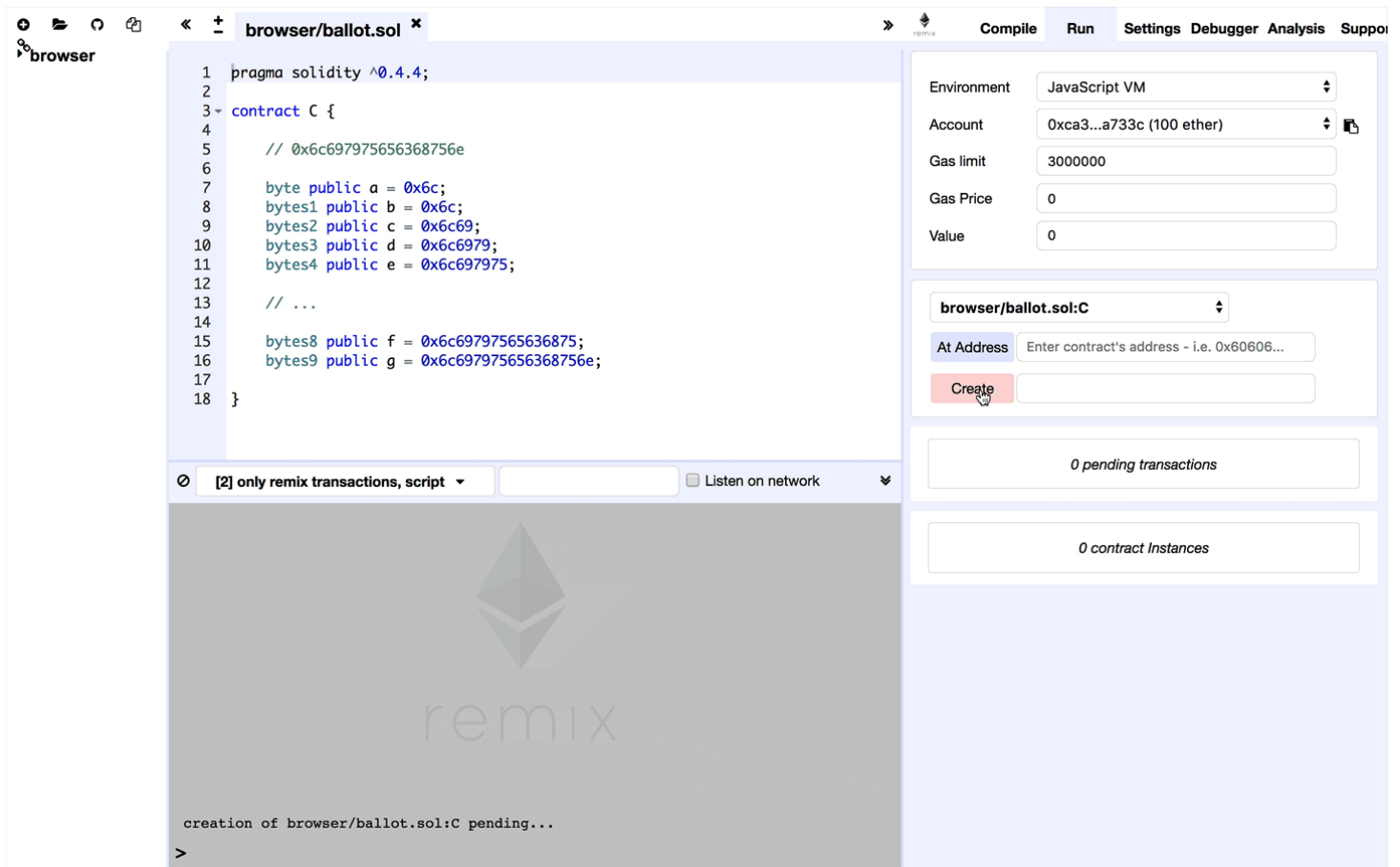
    // 0x6c697975656368756e

    byte public a = 0x6c; // 0110 1100
    bytes1 public b = 0x6c; // 0110 1100
    bytes2 public c = 0x6c69; // 0110 1100 0110 1001
    bytes3 public d = 0x6c6979; // 0110 1100 0110 1001 0111 1001
    bytes4 public e = 0x6c697975; // 0110 1100 0110 1001 0111 1001 0111 0101

    // ...

    bytes8 public f = 0x6c69797565636875; // 0110 1100 0110 1001 0111 1001 0
    111 0101 0110 0101 0110 0011 0110 1000 0111 0101
    bytes9 public g = 0x6c697975656368756e; // // 0110 1100 0110 1001 0111 1
    001 0111 0101 0110 0101 0110 0011 0110 1000 0111 0101 0110 1110

}
```



说明

0x 6c 69 79 75 65 63 68 75 6e 是一个十六进制的整数，它的二进制码是 0b 0110 1100 0110 1001 0111 1001 0111 0101 0110 0101 0110 0011 0110 1000 0111 0101 0110 1110，在计算机中 0b 0110 1100 0110 1001 0111 1001 0111 0101 0110 0101 0110 0011 0110 1000 0111 0101 0110 1110 二进制码存储的内容其实就是 liyuechun 我名字的全拼。我们都知道，在计算机中，所以的内容，不管是图片、文字、视频，任何资料我们都可以转换成 二进制 码在计算机中进行存储。

在计算机中，一个字母 或者 一个数字 的存储空间为 一个字节，也就是 8位 二进制位。一个汉字 占 两个字节，也就是 16位。

0x6c697975656368756e 中，0x6c 是一个字节，因为16进制中，一个数字等价于二进制中的4位，两个数字等价于8位，刚好一个字节，0x6c 用二进制来表示是 0b 0110 1100，0x6c 对应的内容为 l,而 0x6c69 对应的内容为 li,以此内推 0x6c697975656368756e 对应的内容为 liyuechun。

PS:

byte 和 bytes1 等价，只能存储一个字节，当超过它的存储范围时就会报错，如下图所示：

操作运算符

- 比较运算符: `<=`, `<`, `==`, `!=`, `>=`, `>`

```
pragma solidity ^0.4.4;
```

```
contract C{
```

```
    // 0x 6c 69 79 75 65 63 68 75 6e -> liyuechun
```

```
    // 1 2 3 4 5 6 7 8 9 A B C D E F
```

```
    bytes1 b10 = 0x6c; // l -> 0110 1100 -> 12 * 1 + 6 * 16 = 108
```

```
    bytes1 b11 = 0x69; // i -> 0110 1001 -> 9 * 1 + 6 * 16 = 105
```

```
    // <=, <, ==, !=, >=, >
```

```
    function test1() constant returns (bool) {
```

```
        return b10 <= b11; // false
```

```
    }
```

```

function test2() constant returns (bool) {

    return b10 < b11; // false
}

function test3() constant returns (bool) {

    return b10 == b11; // false
}

function test4() constant returns (bool) {

    return b10 >= b11; // true
}

function test5() constant returns (bool) {

    return b10 > b11; // true
}
}

```

- 位操作符: `&`, `|`, `^(异或)`, `~ (取反)`, `<< (左移)`, `>> (右移)`

```

pragma solidity ^0.4.4;

contract C{

    // 0x 6c 69 79 75 65 63 68 75 6e  -> liyuechun

    // 1 2 3 4 5 6 7 8 9 A B C D E F

    bytes1 b10 = 0x6c; // l -> 0110 1100  -> 12 * 1 + 6 * 16 = 108
    bytes1 b11 = 0x69; // i -> 0110 1001  -> 9 * 1 + 6 * 16 = 105

    //  &, |, ^(异或), ~ (取反), << (左移), >> (右移)

    function test1() constant returns (bytes1) {

        return b10 & b11;
        // 0110 1100 -> 0x6c
        // 0110 1001 -> 0x69
        // 0110 1000 -> 0x68
    }
}

```

```

function test2() constant returns (bytes1) {

    return b10 | b11;
    // 0110 1100 -> 0x6c
    // 0110 1001 -> 0x69
    // 0111 1101 -> 0x6d
}

function test3() constant returns (bytes1) {

    return ~b10;
    // 0110 1100 -> 0x6c
    // 1001 0011 -> 0x93
}

function test4() constant returns (bytes1) {

    return b10 << 1;
    // 0110 1100 -> 0x6c
    // 1101 1000 -> 0xd8
}

function test5() constant returns (bytes1) {

    return b10 >> 1;
    // 0110 1100 -> 0x6c
    // 0011 0110 -> 0x36
}
}

```

- 索引访问：如果 `x` 是一个 `bytesI` ,那么可以通过 `x[k]` ($0 < k < I$) 获取对应索引的字节，
PS: `x[k]`是只读，不可写。

成员函数

- `.length` 返回字节的个数。（只读）

```

pragma solidity ^0.4.4;

contract C {

    bytes9 public g = 0x6c697975656368756e;

    function gByteLength() constant returns (uint) {

```

```
return g.length;
```

```
}
```

```
}
```

The screenshot shows the Remix IDE interface. On the left, the 'ContractDefinition C' tab displays the following Solidity code:

```
1 pragma solidity ^0.4.4;
2
3 contract C {
4
5     bytes9 public g = 0x6c697975656368756e;
6
7     function gByteLength() constant returns (uint) {
8         return g.length;
9     }
10 }
11
12 }
```

On the right, the 'Run' tab is active, showing the 'Environment' section with the following settings:

- Environment: JavaScript VM
- Account: 0xca3...a733c (99.999999999999840€)
- Gas limit: 3000000
- Gas Price: 0
- Value: 0

Below the environment settings, the 'browser/ballot.sol:C' section shows the 'At Address' field and a 'Create' button. The '0 pending transactions' section is also visible.

At the bottom, the 'Logs' section shows the execution results:

```
creation of browser/ballot.sol:C pending...
[vm] from:0xca3...a733c, to:browser/ballot.sol:C.(constructor), value:0 wei, data:0x606...30029, 0 logs, hash:0x358...bdd4c
[call] from: - , to:browser/ballot.sol:C.gByteLength(), data:a01a2...a2aaf, return:
{
  "0": "uint256: 9"
}
[call] from: - , to:browser/ballot.sol:C.g(), data:e2179...79b8e, return:
{
  "0": "bytes9: 0x6c697975656368756e"
}
```

不可变深度解析

长度不可变

```
pragma solidity ^0.4.4;
```

```
contract C {
```

```
    bytes9 name = 0x6c697975656368756e;
```

```
    function setNameLength(uint length) {
```

```
        // 报错
```

```
        name.length = length;
```

```
    }
```

}

```
1 pragma solidity ^0.4.4;
2
3 contract C {
4
5
6     bytes9 name = 0x6c697975656368756e;
7
8     function setNameLength(uint length) {
9
10         name.length = length;
11     }
12
13
14
15
16 }
```

browser/ballot.sol:10:9: TypeError: Expression has to be an lvalue
name.length = length;
^-----^

browser/ballot.sol:10:23: TypeError: Type uint256 is not implicitly convertible to expected type bytes9
name.length = length;
^-----^

内部字节不可修改

```
pragma solidity ^0.4.4;

contract C {

    bytes9 name = 0x6c697975656368756e;

    function setNameFirstByte(byte b) {

        name[0] = b;
    }

}
```

```
1 pragma solidity ^0.4.4;
2
3 contract C {
4
5
6     bytes9 name = 0x6c697975656368756e;
7
8     function setNameFirstByte(byte b) {
9
10         name[0] = b;
11     }
12
13 }
```

browser/ballot.sol:10:9: TypeError: Expression has to be an lvalue
name[0] = b;
^-----^

总结

`bytesI(1 <= I <= 32)` 可以声明固定字节大小的字节数组变量，一旦声明，内部的字节和字节数组长度不可修改，当然可以通过索引读取(只读)对应索引的字节，或者通过 `length` 读取字节数组的字节数。

技术交流

- 区块链技术交流QQ群：348924182
- 「区块链部落」官方公众号



长按，识别二维码，加关注