

『0014』 - Solidity Types - 动态大小字节数组(Dynamically-sized byte array)

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>

一、Dynamically-sized byte array

- `string` 是一个动态尺寸的 UTF-8 编码字符串，它其实是一个特殊的可变字节数组，`string` 是引用类型，而非值类型。
- `bytes` 动态字节数组，引用类型。

根据经验，在我们不确定字节数据大小的情况下，我们可以使用 `string` 或者 `bytes`，而如果我们清楚的知道或者能够将字节书控制在 `bytes1` ~ `bytes32`，那么我们就使用 `bytes1` ~ `bytes32`，这样的话能够降低存储成本。

二、常规字符串 sting 转换为 bytes

`string` 字符串中没有提供 `length` 方法获取字符串长度，也没有提供方法修改某个索引的字节码，不过我们可以将 `string` 转换为 `bytes`，再调用 `length` 方法获取字节长度，当然可以修改某个索引的字节码。

1、源码

```
pragma solidity ^0.4.4;

contract C {

    bytes9 public g = 0x6c697975656368756e;

    string public name = "liyuechun";

    function gByteLength() constant returns (uint) {

        return g.length;
    }
}
```

```

}

function nameBytes() constant returns (bytes) {

    return bytes(name);
}

function nameLength() constant returns (uint) {

    return bytes(name).length;
}

function setNameFirstByteForL(bytes1 z) {

    // 0x4c => "L"
    bytes(name)[0] = z;
}
}

```

2、效果图

The screenshot displays the Remix IDE interface. On the left, the 'browser/ballot.sol' file is open, showing the following Solidity code:

```

1 pragma solidity ^0.4.4;
2
3 contract C {
4     bytes9 public g = 0x6c697975656368756e;
5     string public name = "liyuechun";
6
7     function gByteLength() constant returns (uint) {
8         return g.length;
9     }
10
11     function nameBytes() constant returns (bytes) {
12         return bytes(name);
13     }
14
15     function nameLength() constant returns (uint) {
16         return bytes(name).length;
17     }
18
19     function setNameFirstByteForL(bytes1 z) {
20         // 0x4c => "L"
21         bytes(name)[0] = z;
22     }
23 }

```

On the right side, the 'Run' tab is active, showing the deployment configuration for the contract 'C':

- Environment:** JavaScript VM
- Account:** 0xca3...a733c (100 ether)
- Gas limit:** 3000000
- Gas Price:** 0
- Value:** 0

Below the configuration, there is a section for deployment:

- Contract:** browser/ballot.sol:C
- At Address:** Enter contract's address - i.e. 0x60606...
- Create:** A red button to initiate the deployment.

At the bottom, there are two status boxes:

- 0 pending transactions**
- 0 contract instances**

The bottom status bar shows '[2] only remix transactions, script' and a 'Listen on network' checkbox.

3、说明

```
function nameBytes() constant returns (bytes) {  
  
    return bytes(name);  
}
```

nameBytes 这个函数的功能是将字符串 name 转换为 bytes，并且返回的结果为 0x6c697975656368756e。0x6c697975656368756e 一共为 9 字节，也就是一个英文字母对应一个字节。

```
function nameLength() constant returns (uint) {  
  
    return bytes(name).length;  
}
```

我们之前讲过，string 字符串它并不提供 length 方法帮助我们返回字符串的长度，所以在 nameLength 方法中，我们将 name 转换为 bytes，然后再调用 length 方法来返回字节数，因为一个字节对应一个英文字母，所以返回的字节数量刚好等于字符串的长度。

```
function setNameFirstByteForL(bytes1 z) {  
  
    // 0x4c => "L"  
    bytes(name)[0] = z;  
}
```

如果我们想将 name 字符串中的某个字母进行修改，那么我们直接通过 x[k] = z 的形式进行修改即可。x 是 bytes 类型的字节数组，k 是索引，z 是 byte1 类型的变量值。

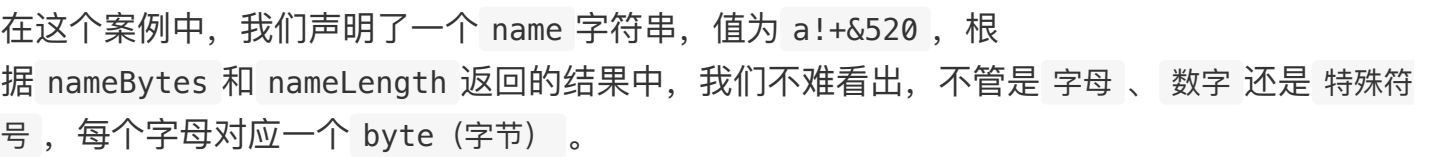
setNameFirstByteForL 方法中，我就将 liyuechun 中的首字母修改成 L，我传入的 z 的值为 0x4c，即大写的 L。

三、汉字字符串或特殊字符的字符串转换为bytes

1、特殊字符

```
pragma solidity ^0.4.4;  
  
contract C {
```

```
function nameBytes() constant returns (bytes) {  
    return bytes(name);  
}  
  
function nameLength() constant returns (uint) {  
    return bytes(name).length;  
}
```



2、中文字符串

```
pragma solidity ^0.4.4;
```

```

contract C {

    string public name = "黎跃春";

    function nameBytes() constant returns (bytes) {

        return bytes(name);
    }

    function nameLength() constant returns (uint) {

        return bytes(name).length;
    }

}

```

The screenshot displays the Remix IDE interface. On the left, the Solidity code for contract C is shown, defining a string variable 'name' with the value '黎跃春' and two functions: 'nameBytes()' which returns the string as bytes, and 'nameLength()' which returns the length of the string in bytes. The main editor area shows the execution results of these functions. The 'nameLength()' function was called, returning a value of 9 (0x00000009). The 'name' variable was also accessed, returning the string '黎跃春'. On the right, the 'Environment' panel shows the current environment is 'JavaScript VM'. Below it, the 'Contract' panel shows the contract 'browser/ballot.sol:C' is selected. The 'At Address' field is empty, and the 'Create' button is visible. The 'Transactions' panel shows '0 pending transactions'. The 'Memory' panel shows the state of the contract's memory at address 0x692...77b3a, with 'nameLength' returning 0: uint256: 9, 'nameBytes' returning 0: bytes: 0xe9bb8ee8b783e698a5, and 'name' returning 0: string: 黎跃春.

在上面的代码中，我们不难看出，黎跃春 转换为 bytes 以后的内容为 0xe9bb8ee8b783e698a5，一共 9 个字节。也就是一个汉字需要通过 3 个字节 来进行存储。那么问题来了，以后我们取字符串时，字符串中最好不要带汉字，否则计算字符串长度时还得特殊处理。

四、创建bytes字节数组

```
pragma solidity ^0.4.4;

contract C {

    bytes public name = new bytes(1);

    function setNameLength(uint length) {

        name.length = length;
    }

    function nameLength() constant returns (uint) {

        return name.length;
    }

}
```

The screenshot displays the Remix IDE interface. On the left, the 'browser' tab is active, showing the Solidity code for 'browser/ballot.sol'. The code defines a contract 'C' with a public byte array 'name' of length 1, a 'setNameLength' function, and a 'nameLength' function. The right-hand panel contains several sections: 'Environment' with settings for JavaScript VM, account (0xca3...a733c), gas limit (3000000), gas price (0), and value (0); a section for 'browser/ballot.sol:C' with an 'At Address' field and a 'Create' button; and two status boxes at the bottom indicating '0 pending transactions' and '0 contract instances'. The bottom of the interface features a console area with a prompt '>' and a 'Listen on network' checkbox.

五、bytes可变数组length和push两个函数的使用案例

```
pragma solidity ^0.4.4;

contract C {

    // 0x6c697975656368756e
    // 初始化一个两个字节空间的字节数组
    bytes public name = new bytes(2);

    // 设置字节数组的长度
    function setNameLength(uint len) {

        name.length = len;
    }

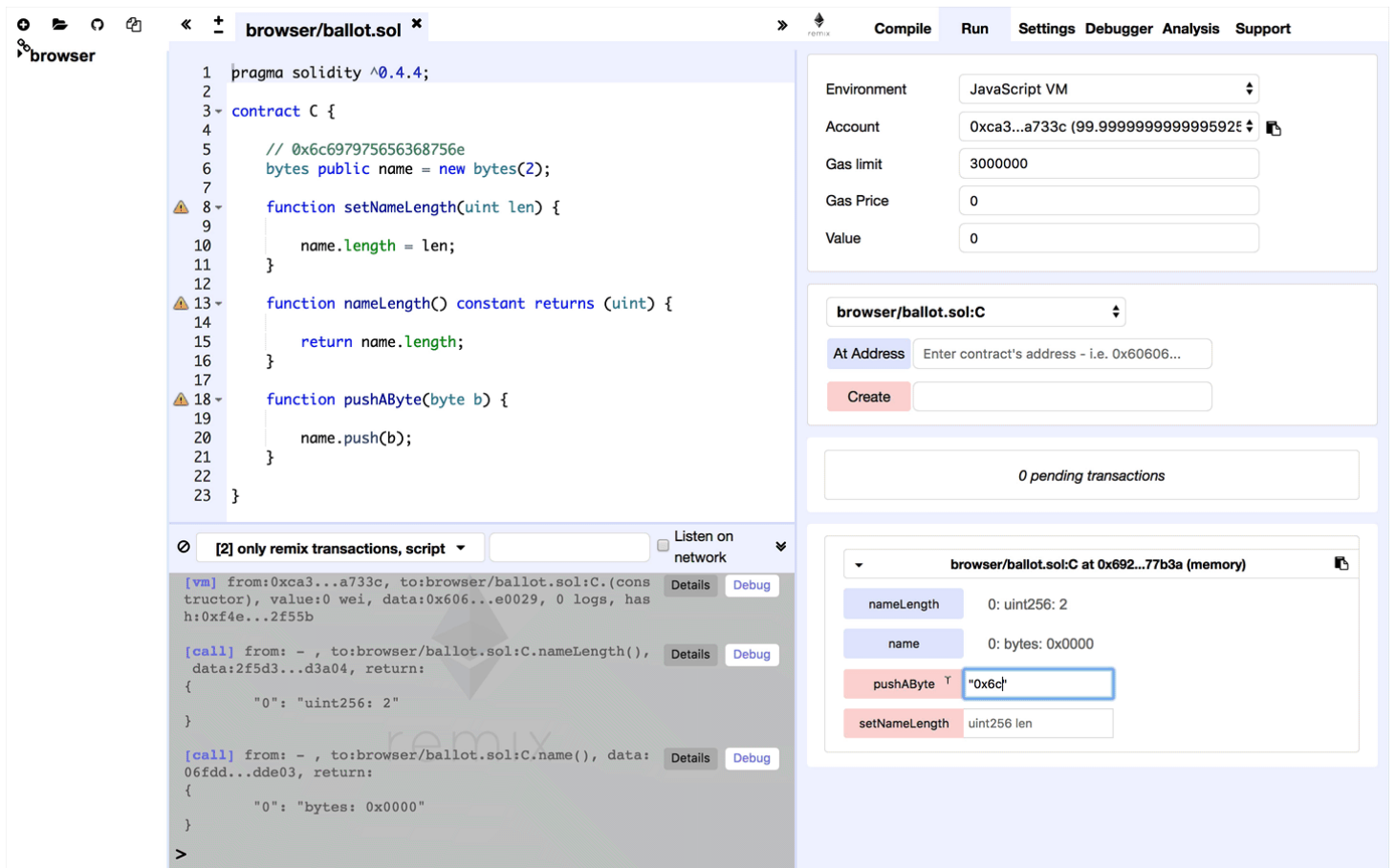
    // 返回字节数组的长度
    function nameLength() constant returns (uint) {

        return name.length;
    }

    // 往字节数组中添加字节
    function pushAByte(byte b) {

        name.push(b);
    }

}
```



说明：当字节数组的长度只有2时，如果你通过push往里面添加了一个字节，那么它的长度将变为3，当字节数组里面有3个字节，但是你通过length方法将其长度修改为2时，字节数组中最后一个字节将被从字节数组中移除。

五、总结

对比分析：

- 不可变字节数组

我们之前的文章中提到过如果我们清楚我们存储的字节大小，那么我们可以通过 `bytes1` , `bytes2` , `bytes3` , `bytes4` ,....., `bytes32` 来声明字节数组变量，不过通过 `bytesI` 来声明的字节数组为不可变字节数组，字节不可修改，字节数组长度不可修改。

- 可变字节数组

我们可以通过 `bytes name = new bytes(length)` - `length` 为字节数组长度，来声明可变大和可修改字节内容的可变字节数组。