

『0008』 - Solidity中public、internal、private在状态变量和函数中的使用以及Solidity智能合约继承、重写

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>

在上一小节中我们在函数参数中使用 `storage` 这个关键字时，当前的函数必须是 `internal` 或者 `private` 类型。

在本小节中，我（微信：liyc1215）将重点为大家介绍属性和函数的使用权限。

状态变量、函数的权限

一、public

备注：为了演示方便，我直接通过 <https://remix.ethereum.org/> 来进行演示。

`public` 类型的状态变量和函数的权限最大，可供外部、子合约、合约内部访问。

```
pragma solidity ^0.4.4;

contract Animal {

    string _birthDay; // 生日
    int public _age; // 年龄
    int internal _weight; // 身高
    string private _name; // 姓名

    function Animal() {
        _age = 29;
        _weight = 170;
        _name = "Lucky dog";
        _birthDay = "2011-01-01";
    }

    function birthDay() constant returns (string) {
```

```

    return _birthDay;
}

function age() constant public returns (int) {
    return _age;
}

function height() constant internal returns (int) {
    return _weight;
}

function name() constant private returns (string) {
    return _name;
}

}

```

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'browser/ballot.sol' file is open, displaying Solidity code for a contract named 'Animal'. The code includes state variables for birth date, age, weight, and name, and functions for setting and retrieving these values. On the right, the 'Run' tab is active, showing the execution environment (JavaScript VM) and the contract's state. The 'At Address' field is empty, and the 'Create' button is visible. Below the 'Create' button, the execution results are shown, indicating that the contract is running at address 0x692...77b3a. The results table shows the following values:

Variable	Value
birthDay	string: 2011-01-01
age	int256: 29
_age	int256: 29

在这个合约中，我们通过运行结果不难看出，可供外部调用的一个有三个函数，分别为 `birthDay`，`_age`，`age`，也许有人会问，为什么外部可以调用 `_age` 函数呢，为什么外部可以调用 `_age` 函数呢，为什么外部可以调用 `_age` 函数呢，原因是因为我们的状态变量 `_age` 的权限是 `public`，当一个状态变量的权限为 `public` 类型时，它就会自动生成一个可供外部调用的 `get` 函数。在我们这个合约中，因为 `_age` 是 `public` 类型，所以在合约中其实会有一个默认的和状态变量同名的 `get` 函数，如下所示：

```

function _age() constant public returns (int) {
    return _age;
}

```

在我们显示声明的四个函数中：

```
function birthDay() constant returns (string) {  
    return _birthDay;  
}  
  
function age() constant public returns (int) {  
    return _age;  
}  
  
function height() constant internal returns (int) {  
    return _weight;  
}  
  
function name() constant private returns (string) {  
    return _name;  
}
```

由上面的运行结果，我们知道，这四个函数中，只有 `birthDay`、`age` 函数可供外部访问，**【PS：age 函数是我显示声明的，_age 函数是因为状态变量 `_age` 为 `public` 自动生成的，因为状态变量默认为 `internal` 类型，所以不会自动生成可供外部访问的和状态变量同名的函数】**，换句话说，只有 `public` 类型的函数才可以供外部访问，由此可知，函数声明时，它默认为 `public` 类型，而状态变量声明时，默认为 `internal` 类型。

小结：

- 状态变量声明时，默认为 `internal` 类型，只有显示声明为 `public` 类型的状态变量才会自动生成一个和状态变量同名的 `get` 函数以供外部获取当前状态变量的值。
- 函数声明时默认为 `public` 类型，和显示声明为 `public` 类型的函数一样，都可供外部访问。

二、internal

- `internal` 类型的状态变量可供外部和子合约调用。
- `internal` 类型的函数和 `private` 类型的函数一样，智能合约自己内部调用，它和其他语言中的 `protected` 不完全一样。

```
pragma solidity ^0.4.4;  
  
contract Animal {  
  
    string _birthDay; // 生日  
    int public _age; // 年龄
```

```
int internal _weight; // 身高
string private _name; // 姓名

function Animal() {
    _age = 29;
    _weight = 170;
    _name = "Lucky dog";
    _birthDay = "2011-01-01";
}

function birthDay() constant returns (string) {
    return _birthDay;
}

function age() constant public returns (int) {
    return _age;
}

function height() constant internal returns (int) {
    return _weight;
}

function name() constant private returns (string) {
    return _name;
}

}

contract Person is Animal {

    function Person() {

        _age = 50;
        _weight = 270;
        _birthDay = "2017-01-01";

    }

}
```

The screenshot displays the Remix IDE interface. On the left, the Solidity code editor shows a contract named `Animal` with state variables `_birthDay`, `_age`, `_weight`, and `_name`. It includes functions `Animal()`, `birthDay()`, `age()`, `height()`, and `name()`. A second contract, `Person`, is defined as inheriting from `Animal` and implementing the `Person()` function. The `Person` contract's code is highlighted with a red box. On the right, the 'Run' tab is active, showing the 'Environment' section with 'JavaScript VM' selected. Below this, the 'At Address' field is set to 'browser/ballot.sol:Person', and the 'Create' button is visible. A red arrow points to the 'Create' button. At the bottom of the 'Run' tab, a transaction list shows a transaction for 'browser/ballot.sol:Person at 0x692...77b3a' with a status of '(memory)'. The transaction details show the state variables: `birthDay` (string: 2017-01-01), `age` (int256: 50), and `_age` (int256: 50).

在这个案例中，`contract Person is Animal`，`Person` 合约继承了 `Animal` 合约的 `public/internal` 的所有状态变量，但是只能继承父合约中的所有的 `public` 类型的函数，不能继承 `internal/private` 的函数，不能继承 `internal/private` 的函数，不能继承 `internal/private` 的函数。

三、private

我们在 `person` 合约中尝试调用 `_name` 状态变量，你会发现，编译没法通过。

« **browser/ballot.sol** ✕

```
3 contract Animal {
4
5     string _birthDay; // 生日
6     int public _age; // 年龄
7     int internal _weight; // 身高
8     string private _name; // 姓名
9
10    function Animal() {
11        _age = 29;
12        _weight = 170;
13        _name = "Lucky dog";
14        _birthDay = "2011-01-01";
15    }
16
17    function birthDay() constant returns (string) {
18        return _birthDay;
19    }
20
21    function age() constant public returns (int) {
22        return _age;
23    }
24
25    function height() constant internal returns (int) {
26        return _weight;
27    }
28
29    function name() constant private returns (string) {
30        return _name;
31    }
32
33 }
34
35
36 contract Person is Animal {
37
38    function Person() {
39
40        _age = 50;
41        _weight = 270;
42        _birthDay = "2017-01-01";
43
44        _name = "liyuechun";|
45    }
```



✕ browser/ballot.sol:44:9: DeclarationError: Undeclared identifier.
4 _name = "liyuechun";
4 ^----^

⊗ [2] only remix transactions, script ▼

☐ Listen on network

因为 `_name` 状态变量在 `Animal` 合约中属于 `private` 私有类型，只能在 `Animal` 内部使用，所以到我们在子合约 `Person` 中尝试使用时，就会报错。

四、重写

子合约可以将父合约的 `public` 类型的函数，只能继承`public`类型的函数，只能继承`public`类型的函数，只能继承`public`类型的函数，我们可以直接调用继承过来的函数，当然，我们还可以对继承过来的函数进行重写。

- 重写前

```
pragma solidity ^0.4.4;

contract Animal {

    string _birthDay; // 生日
    int public _age; // 年龄
    int internal _weight; // 身高
    string private _name; // 姓名

    function Animal() {
        _age = 29;
        _weight = 170;
        _name = "Lucky dog";
        _birthDay = "2011-01-01";
    }

    function birthDay() constant returns (string) {
        return _birthDay;
    }

    function age() constant public returns (int) {
        return _age;
    }

    function height() constant internal returns (int) {
        return _weight;
    }

    function name() constant private returns (string) {
        return _name;
    }

}

contract Person is Animal {
```

« + browser/ballot.sol ✕

1 pragma solidity ^0.4.4;

2

3 contract Animal {

4

5 string _birthDay; // 生日

6 int public _age; // 年龄

7 int internal _weight; // 身高

8 string private _name; // 姓名

9

10 function Animal() {

11 _age = 29;

12 _weight = 170;

13 _name = "Lucky dog";

14 _birthDay = "2011-01-01";

15 }

16

17 function birthDay() constant returns (string) {

18 return _birthDay;

19 }

20

21 function age() constant public returns (int) {

22 return _age;

23 }

24

25 function height() constant internal returns (int) {

26 return _weight;

27 }

28

29 function name() constant private returns (string) {

30 return _name;

31 }

32 }

33 }

34

35

36 contract Person is Animal {

37

38

39 }

40

» remix Compile Run Settings Debugger Analysis Support

Environment JavaScript VM

Account 0xca3...a733c (99.999999999999698t

Gas limit 3000000

Value 0

browser/ballot.sol:Person

At Address Enter contract's address - i.e. 0x60606..

Create

0 pending transactions

browser/ballot.sol:Person at 0x692...77b3a (memory)

birthDay string: 2011-01-01

age int256: 29

_age int256: 29

- 重写后

```
pragma solidity ^0.4.4;
```

```
contract Animal {
```

```
    string _birthDay; // 生日
    int public _age; // 年龄
    int internal _weight; // 身高
    string private _name; // 姓名
```

```
    function Animal() {
        _age = 29;
        _weight = 170;
        _name = "Lucky dog";
        _birthDay = "2011-01-01";
    }
```

```
    function birthDay() constant returns (string) {
        return _birthDay;
    }
```

```
    function age() constant public returns (int) {
        return _age;
    }
```

```
    function height() constant internal returns (int) {
```



```

        return _weight;
    }

    function name() constant private returns (string) {
        return _name;
    }
}

contract Person is Animal {

    function birthDay() constant returns (string) {

        return "2020-12-15";
    }

}

```

The screenshot shows the Solidity compiler interface. On the left, the source code for `browser/ballot.sol` is displayed. On the right, the execution environment is shown with various settings and a list of contracts.

Source Code (Left):

```

1 pragma solidity ^0.4.4;
2
3 contract Animal {
4
5     string _birthDay; // 生日
6     int public _age; // 年龄
7     int internal _weight; // 身高
8     string private _name; // 姓名
9
10    function Animal() {
11        _age = 29;
12        _weight = 170;
13        _name = "Lucky dog";
14        _birthDay = "2011-01-01";
15    }
16
17    function birthDay() constant returns (string) {
18        return _birthDay;
19    }
20
21    function age() constant public returns (int) {
22        return _age;
23    }
24
25    function height() constant internal returns (int) {
26        return _weight;
27    }
28
29    function name() constant private returns (string) {
30        return _name;
31    }
32 }
33
34
35
36 contract Person is Animal {
37
38    function birthDay() constant returns (string) {
39
40        return "2020-12-15";
41    }
42 }
43
44

```

Execution Environment (Right):

- Environment: JavaScript VM
- Account: 0xca3...a733c (99.9999999999997267)
- Gas limit: 3000000
- Value: 0

The **Contracts** tab is selected, showing a list of contracts. The contract `browser/ballot.sol:Person` is selected, and its details are shown below:

- At Address: Enter contract's address - i.e. 0x60606..
- Create: [button]
- 0 pending transactions
- Contract: `browser/ballot.sol:Person at 0x692...77b3a (memory)`
- Variables:
 - `birthDay`: string: 2020-12-15
 - `age`: int256: 29
 - `_age`: int256: 29

Red boxes and arrows highlight the following:

- The `birthDay()` function in the `Person` contract, which overrides the `birthDay()` function in the `Animal` contract.
- The `birthDay` variable in the `Person` contract, which is a constant string.
- The `age` variable in the `Person` contract, which is a constant integer.

小结

本篇文章主要全面介绍了合约中状态变量和函数中 `public`、`internal`、`private` 三种权限在合约内部、外部以及子合约中的应用。通过本篇教程的学习，我相信你一定会进一步了解状态变量的继承以及函数继承和重写。接下来的系列文章中，我们将进一步讲解Solidity中相关的语法以及

开发中的注意事项。

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