

python之父类的init方法

写代码的时候忽然要实现继承父类所有的方法并且修改父类的__init__参数，这样就叫方法重载。

方法重载(overloading method): 子类修改父类的一些参数就叫重载。
方法重写(overriding method): 子类不想原封不动的继承父类的方法，需要进行覆盖。# 修改父类的方法
#重写父类的方法的目的是为了给他扩展功能，父类的方法已经不能满足需求

1. 子类继承父类时，如果 `init` 方法被重写，可以使用super函数调用父类的 `init` 方法，否则被改写的 `__init__` 方法会覆盖原有的自动继承的 `__init__` 方法。
2. 类的继承，就是子类继承除了构造函数之外的所有的东西， `init` 方法不是构造方法（函数）
3. 子类继承父类时， `init` 方法被自动继承，并在创建实例时，自动调用。

核心思想就一句话，先调用一下你要重写的父类方法

方法重载

```
1
2 class A():
3     def __init__(self,name=None,models=None):
4
5         self.name=name
6         self.models=models
7
8     def function(self):
9
10        print(self.name)
11        print(self.models)
12
13 class B(A):
14     def __init__(self):
15
16        A.__init__(self,name=None,models=None)
17
18        self.models='modeify'
19
20 #主函数，代码执行的开始
21 if __name__ == '__main__':
22
23     a=A()
24     a.function()
25
26
27     b=B()
28     b.function()
```

执行结果：

```
None
None
None
modeify
```

方法重写

```
1 class A():
2     def __init__(self,name=None,models=None):
3         self.name=name
4         self.models=models
5
6     def function(self):
7         print(self.name)
8         print(self.models)
9
10 class B(A):
11     def __init__(self):
12
13         A.__init__(self,name=None,models=None)
14         self.models='modeify'
15     def function(self):
16
17         print(1)
18         #主函数，代码执行的开始
19
20 if __name__ == '__main__':
21     a=A()
22     a.function()
23
24     b=B()
25     b.function()
```

执行结果

```
None
None
1
```