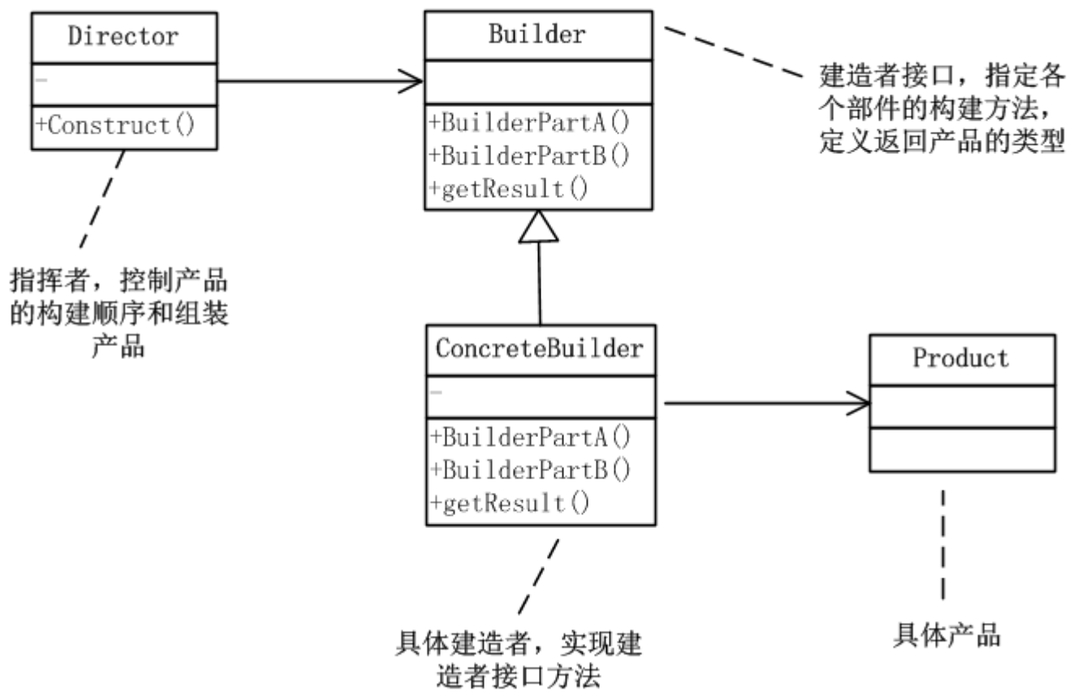
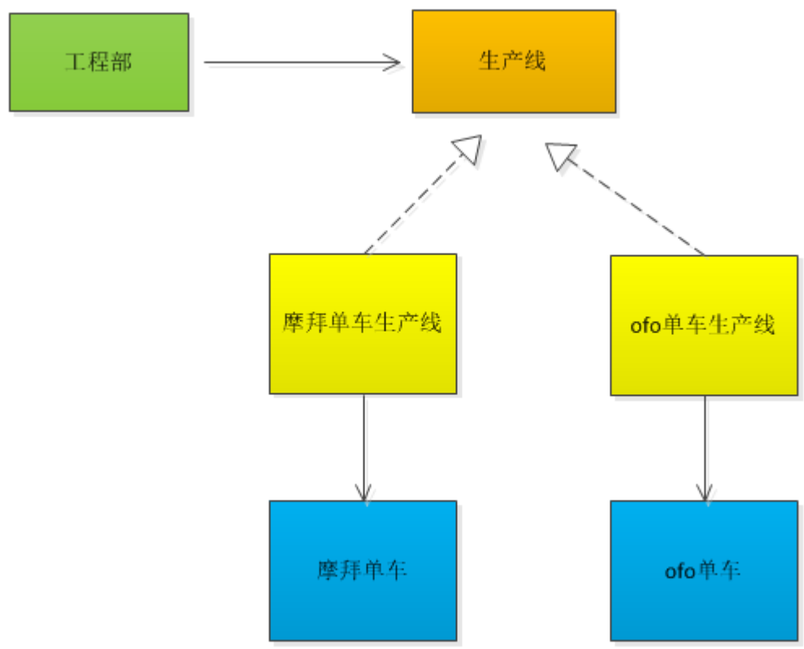
3. 创建型模式-建造者模式

概念：
建造者模式的定义如下：将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示．建造者模式的作用，就是将"构建"和"表示"分离，以达到解耦的作用．

结构图:



我们用制造自行车为例子讲解建造者模式，自行车由车架 轮胎 GPS等部件组成，自行车制造公司的工程部门相当于指挥者，生产部门相当于建造者，当今共享单车做的比较大的摩拜和ofo相当于客户，单车就是产品了．结构图如下所示：



优点：
1)产品的建造和表示分离，实现了解耦．

2)隐藏了产品的建造细节，用户只需关心产品的表示，而不需要了解是如何创建产品的．

3)体现了开闭原则，如上代码所示，如果需要再生产其他共享单车，只需要再开一条生产线即可，不影响其他生产线的作业．

In [10]:
```
#coding:utf8

from abc import ABCMeta, abstractmethod

class Bike(object):

    @property
    def tires(self):
        return self.__tires

    @tires.setter
    def tires(self, value):
        self.__tires = value

    @property
    def frame(self):
        return self.__frame

    @frame.setter
    def frame(self, value):
        self.__frame = value

    @property
    def gps(self):
        return self.__gps

    @gps.setter
    def gps(self, value):
        self.__gps = value


class bikeBuilder:
    __metaclass__ = ABCMeta
    @abstractmethod
    def buildTires(self):
        pass
    @abstractmethod
    def buildFrame(self):
        pass
    @abstractmethod
    def buildGps(self):
        pass
    @abstractmethod
    def getBike(self):
        pass

class ofoBikeBulder(bikeBuilder):
    def __init__(self):
        self.bike = Bike()
    def buildTires(self):
        self.bike.tires = "black tires"
    def buildFrame(self):
        self.bike.frame = "yellow frame"
    def buildGps(self):
        self.bike.gps = "american gps"
    def getBike(self):
        print("frame: %s tires: %s gps:%s" %(self.bike.frame, self.bike.tires, self.bike.gps))

class moBikeBulder(bikeBuilder):
    def __init__(self):
        self.bike = Bike()
    def buildTires(self):
        self.bike.tires = "orange tires"
    def buildFrame(self):
        self.bike.frame = "gray frame"
    def buildGps(self):
        self.bike.gps = "beidou gps"
    def getBike(self):
        print("frame: %s tires: %s gps:%s" %(self.bike.frame, self.bike.tires, self.bike.gps))


class Director:
    def __init__(self, bikeBuilder):
        self.bikeBuilder = bikeBuilder
    def construct(self):
        self.bikeBuilder.buildTires()
        self.bikeBuilder.buildFrame()
        self.bikeBuilder.buildGps()

if __name__ == '__main__':
    ofo = ofoBikeBulder()
    ofo_bike = Director(ofo)
    ofo_bike.construct()
    ofo.getBike()
    print('#' * 20)
    mobike = moBikeBulder()
    mo_bike = Director(mobike)
    mo_bike.construct()
    mobike.getBike()
```

```
frame: yellow frame tires: black tires gps:american gps
####################
frame: gray frame tires: orange tires gps:beidou gps
```