Django Signal

```
from django.db import models
from django.db.models import signals


def pre_save_example(sender, instance, **kwargs):
    print("sender1 : ", sender)
    print("instance1 : ", instance.id)  #None
    print("Save is called 1")


def post_save_example(sender, instance, **kwargs):
    print("sender2 : ", sender)
    print("instance2 : ", instance.id)  #obj id
    print("Save is called 2")


class Customer(models.Model):
    name = models.CharField(max_length=16)
    description = models.CharField(max_length=32)


signals.pre_save.connect(receiver=pre_save_example, sender=Customer)
signals.post_save.connect(receiver=post_save_example, sender=Customer)
```

```
from django.db import models
from django.db.models import signals

def create_customer(sender, instance, created, **kwargs):
      print "Save is called"

class Customer(models.Model):
      name = models.CharField(max_length=16)
      description = models.CharField(max_length=32)

signals.post_save.connect(receiver=create_customer, sender=Customer)
```

```
In [1]: obj = Customer(name='foo', description='foo in detail')
In [2]: obj.save()
Save is called
```

```
from django.db import models
from django.db.models import signals
from django.dispatch import receiver

class Customer(models.Model):
      name = models.CharField(max_length=16)
      description = models.CharField(max_length=32)

@receiver(signals.pre_save, sender=Customer)
def create_customer(sender, instance, created, **kwargs):
      print "customer created"
```

```
from django.db.models.signals import post_save, post_delete, pre_save

class TodoList(models.Model):
      @staticmethod def pre_save(sender, instance, **kwargs):
      #do anything you want
pre_save.connect(TodoList.pre_save,
TodoList,dispatch_uid="sightera.yourpackage.models.TodoList")
from django.db.models.signals import post_save
from django.dispatch import receiver

class TransactionDetail(models.Model):
      product = models.ForeignKey(Product)

# method for updating
@receiver(post_save, sender=TransactionDetail,
dispatch_uid="update_stock_count")
def update_stock(sender, instance, **kwargs):
      instance.product.stock -= instance.amount
      instance.product.save()
```