Django signals are executed synchronously by default. This means that when a signal is sent, the receiver functions are executed immediately in the same thread and process that sent the signal.

For example, I'll create a signal that prints a message whenever a new instance of a model is saved.

```
# models.py
from django.db import models
from django.db.models.signals import post_save
from django.dispatch import receiver

class MyModel(models.Model):
    name = models.CharField(max_length=100)

@receiver(post_save, sender=MyModel)
def my_model_saved(sender, instance, created, **kwargs):
    print(f'MyModel instance saved: {instance.name}, created: {created}')
```

In this code, we have a model called **MyModel** and a signal receiver function **my_model_saved** that listens for the **post_save** signal. This function will print a message whenever a **MyModel** instance is saved.