Topic: Django Signals

==========

Question 1: By default are django signals executed synchronously or asynchronously? Please support your answer with a code snippet that conclusively proves your stance. The code does not need to be elegant and production ready, we just need to understand your logic.

Ans:-By default, Django signals are executed synchronously. This means that the signal handlers are called immediately when the signal is sent, and the sender waits for all handlers to finish before continuing.

CODE:-

```
import time
from django.core.signals import request_finished
from django.dispatch import receiver
@receiver(request_finished)
def my_signal_handler(sender, **kwargs):
    print("Signal handler started")
    time.sleep(5)
    print("Signal handler finished")
print("Sending signal")
request_finished.send(sender=None)
print("Signal sent")
```

Question 2: Do django signals run in the same thread as the caller? Please support your answer with a code snippet that conclusively proves your stance. The code does not need to be elegant and production ready, we just need to understand your logic.

Ans:- Yes, Django signals run in the same thread as the caller by default.

CODE:-

```
import threading
```

```
from django.core.signals import request_finished

from django.dispatch import receiver

@receiver(request_finished)

def my_signal_handler(sender, **kwargs):

    print(f"Signal handler thread: {threading.current_thread().name}")

print(f"Main thread: {threading.current_thread().name}")

request_finished.send(sender=None)
```

Question 3: **By default do django signals run in the same database transaction as the caller?**

Please support your answer with a code snippet that conclusively proves your stance. The code does not need to be elegant and production ready, we just need to understand your logic.

Ans:- By default, Django signals do not run in the same database transaction as the caller. However, you can use the transaction.on_commit hook to ensure that a signal handler runs only after the transaction is successfully committed.

CODE:-

```
from django.db import transaction

from django.db.models.signals import post_save

from django.dispatch import receiver

from myapp.models import MyModel

@receiver(post_save, sender=MyModel)

def my_signal_handler(sender, instance, **kwargs):

    def on_commit_handler():

        print("Signal handler executed after transaction commit")

    transaction.on_commit(on_commit_handler)

instance = MyModel.objects.create(field='value')
```