**Multi Threading vs async and await Difference:**

**(https://www.baeldung.com/cs/async-vs-multi-threading)**

**Multithreading programming is all about concurrent execution of different functions. Async programming is about non-blocking execution between functions**, and we can apply async with single-threaded or multithreaded programming.

So, multithreading is one form of asynchronous programming.

Let’s take a simple analogy; you have a friend, and you decided to make dinner together.

Async is when you say to your friend, “You go to the store and buy pasta. **Let me know when you get back**, to make dinner together. **Meanwhile, I’ll prepare sauce and drinks**.”

Threading is, “You boil the water. I’ll heat the tomato sauce. While the water is boiling, ask me and I’ll put the pasta in. When the sauce is hot, you can add cheese. When both are done, I’ll sit down and you serve dinner. Then we eat.”. **In the threading analogy, we can see the sequence of “When, Do” events, which represent the sequential set of instructions per each person (thread).**

From that analogy, we can conclude that **Multithreading is about workers, Asynchronous is about tasks.**

**Which One To Use?**

**Choosing between the two programming models depends mainly on *performance*.**

Given all possible combinations between sync/async and single/multi-threading, which model should perform better?

**In a nutshell, for large scale applications with a lot of I/O operations and different computations, using asynchronous multithreading programming flow, will utilize the computation resources, and take care of non-blocking functions.** This is the programming model of any OS!

With more power, comes more responsibility! So if we decided to implement this model, we have to take care of different issues like race condition, deadlocks, shared resources, and callbacks events.