

Para saber mais: As interfaces do Pool de threads

Nesse capítulo usamos o pool de threads no lado do servidor. Vimos que existem várias implementações, por exemplo

`CachedThreadPool` :

```
ExecutorService pool = Executors.newCachedThreadPool();
```

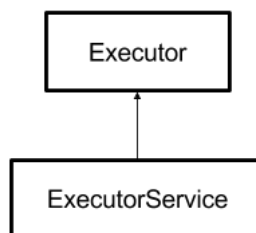
ou o `SingleThreadExecutor` :

```
ExecutorService pool = Executors.newSingleThreadExecutor();
```

Todas as implementações tem em comum que eles implementam a mesma interface: [link do JavaDoc](https://docs.oracle.com/javase/10/docs/api/java/util/concurrent/ExecutorService.html) (<https://docs.oracle.com/javase/10/docs/api/java/util/concurrent/ExecutorService.html>)

No entanto, olhando quais são os métodos que a interface define não encontramos o método `execute` que já usamos durante a aula. Isto é porque o `ExecutorService` estende uma outra interface `Executor` que possui apenas um método, justamente aquele `execute` .

java.util.concurrent.*



```
package java.util.concurrent;

public interface Executor {
    void execute(Runnable command);
}

public interface ExecutorService extends Executor {

    //outras definições de métodos
}
```

`ExecutorService` é um `Executor` ! Faz sentido? Ou seja, poderíamos ter escrito no nosso projeto:

```
Executor pool = Executors.newCachedThreadPool();
```

Por enquanto seria suficiente, mas vamos utilizar ainda métodos específicos da interface `ExecutorService` como o `submit` e `shutdown` . Aguarde!

