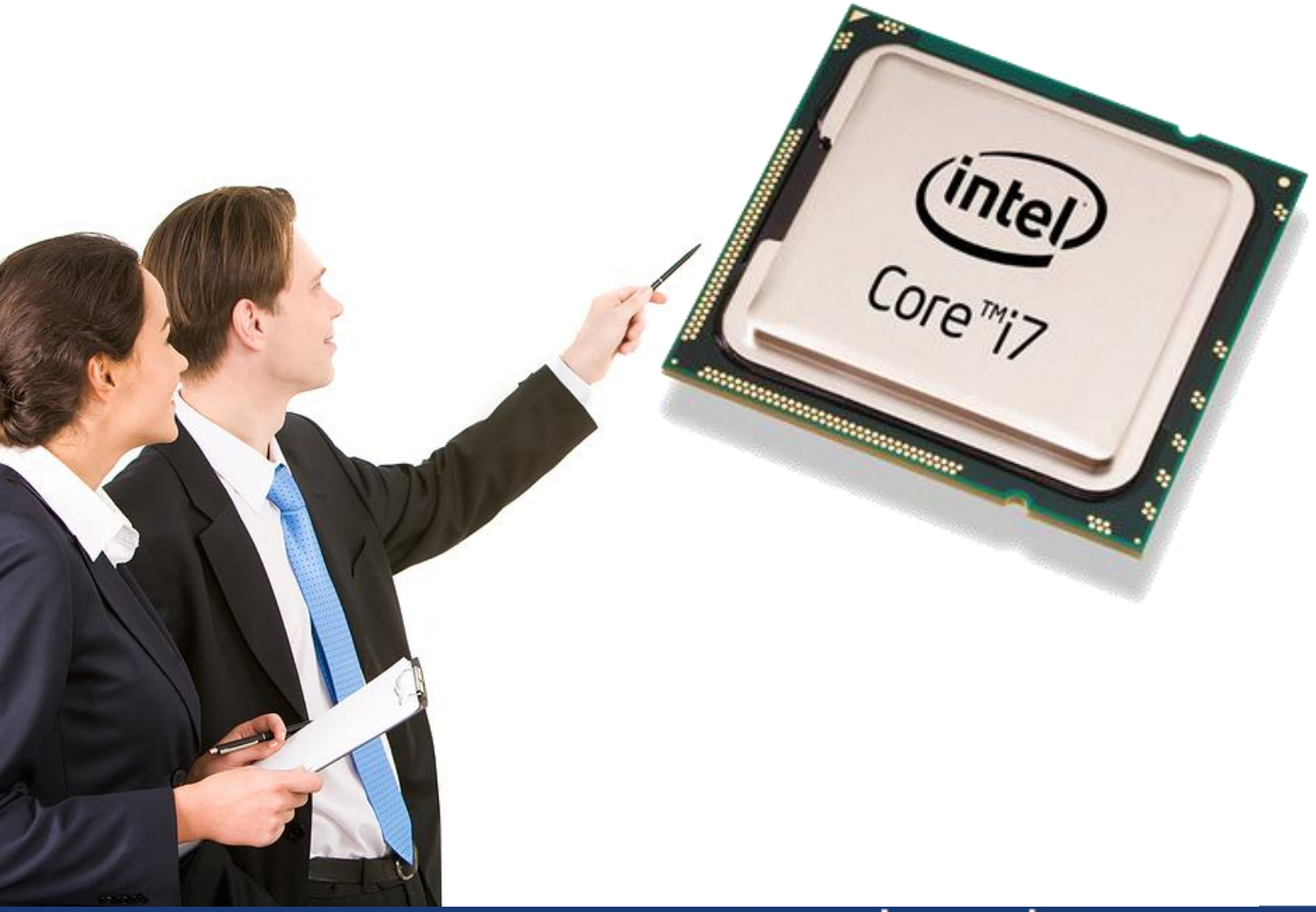
Java Orientado a Objetos
Threads



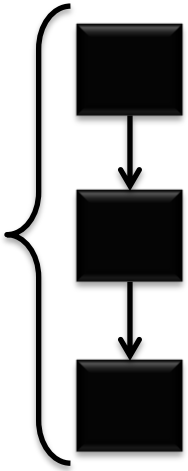
3WAY NETWORKS

Threads

Pense em **threads** como processos do sistema operacional

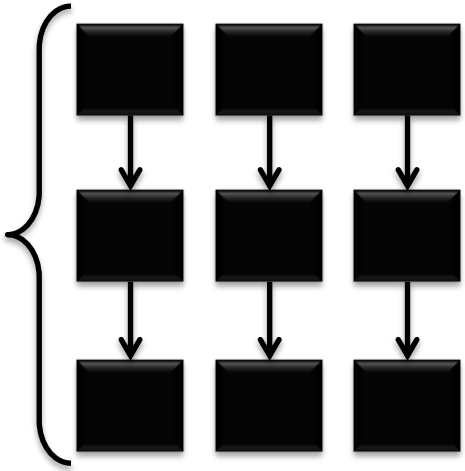
Sequencial

Programa

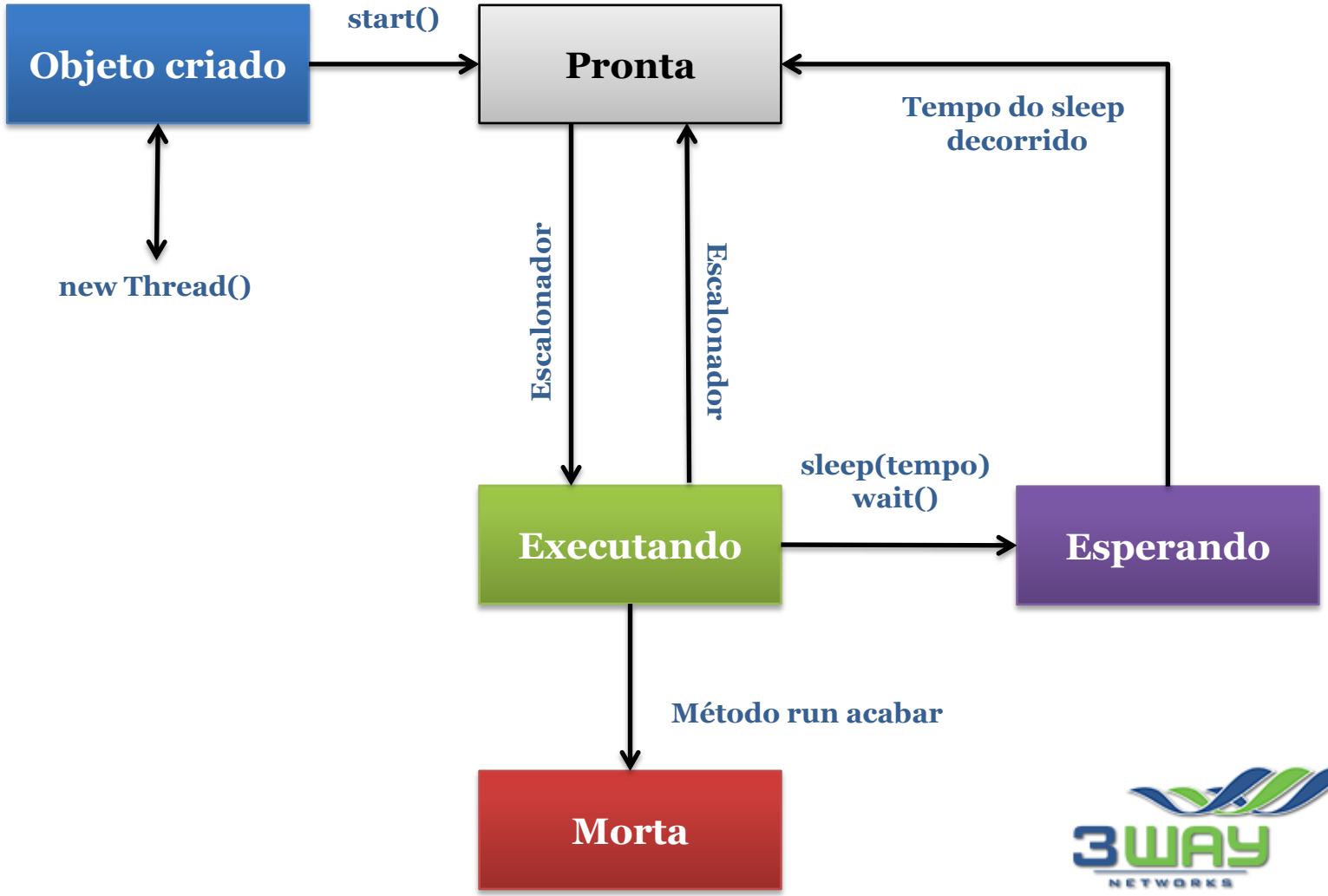


MultiThread

Programa



Ciclo de vida de uma Thread



Criando Threads

```java
public class HerancaThread extends Thread {

    @Override
    public void run() {
        // conteudo da thread
        System.out.println("extends Thread");
        for (int i = 0; i < 10; i++) {
            System.out.println(i);
        }
    }
}
```

Estender a classe **Thread**

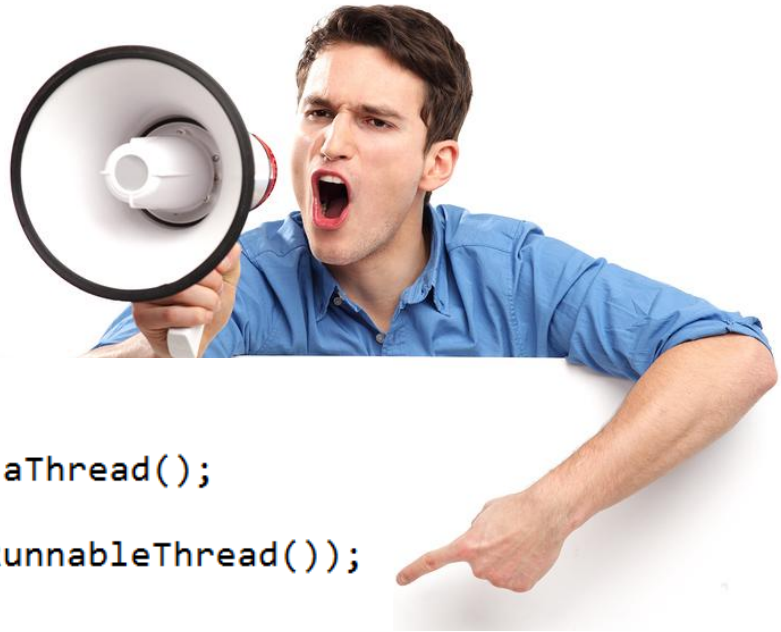
A classe **É UMA** hread

```java
public class RunnableThread implements Runnable {

    @Override
    public void run() {
        // conteudo da thread
        System.out.println("implements Runnable");
        for (int i = 0; i < 10; i++) {
            System.out.println(i);
        }
    }

}
```

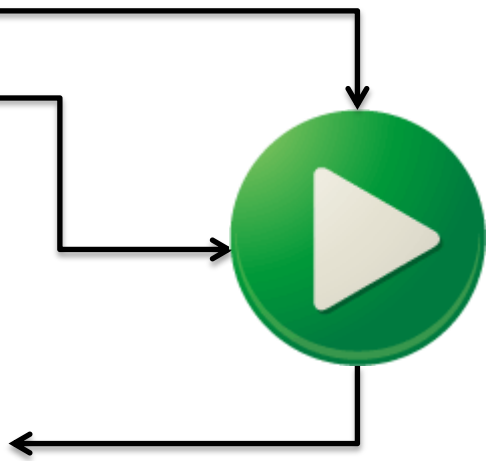
Implementar a interface **Runnable**

A classe **TEM UMA** thread

Iniciando Threads



```
public static void main(String[] args) {

    HerancaThread threadSimples = new HerancaThread();

    Thread threadRunnable = new Thread(new RunnableThread());

    threadSimples.start();

    threadRunnable.start();
}
```

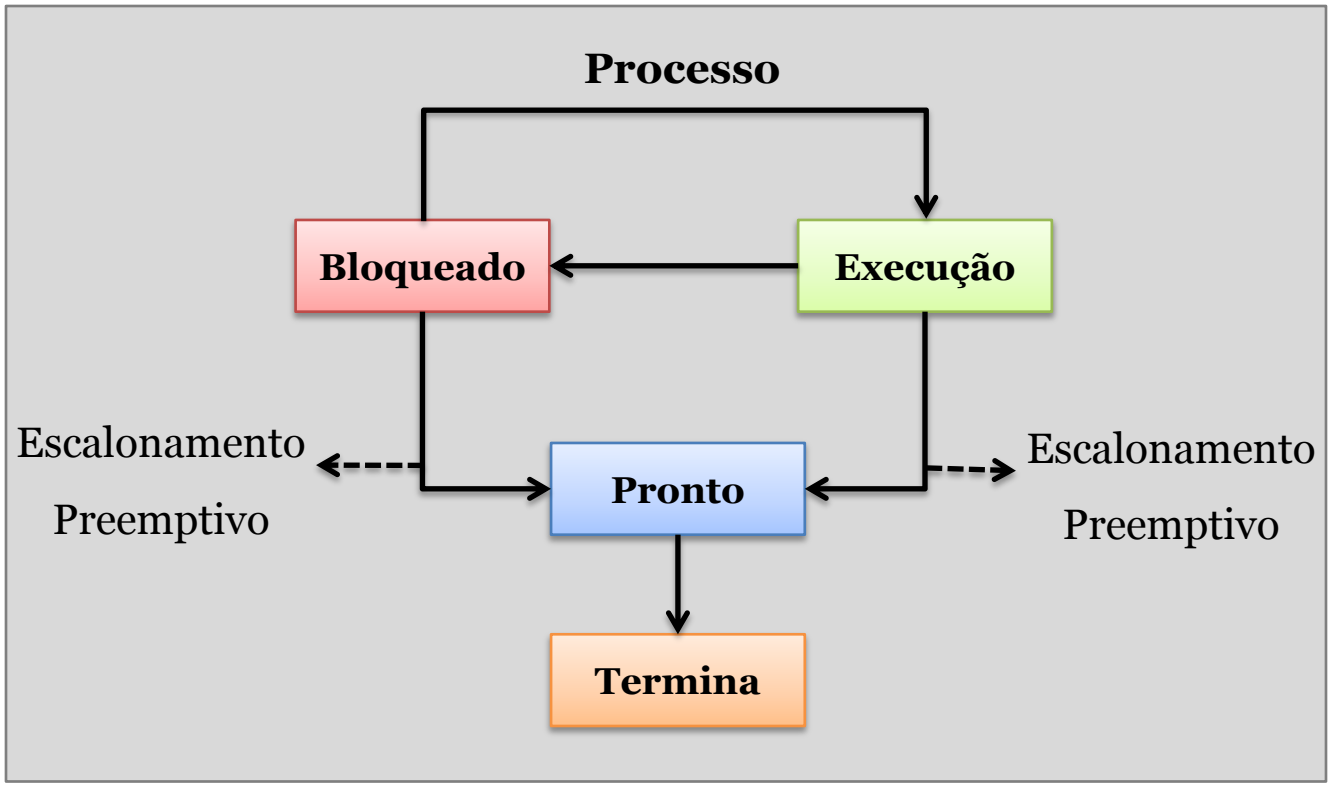


Inicia execução do método run()

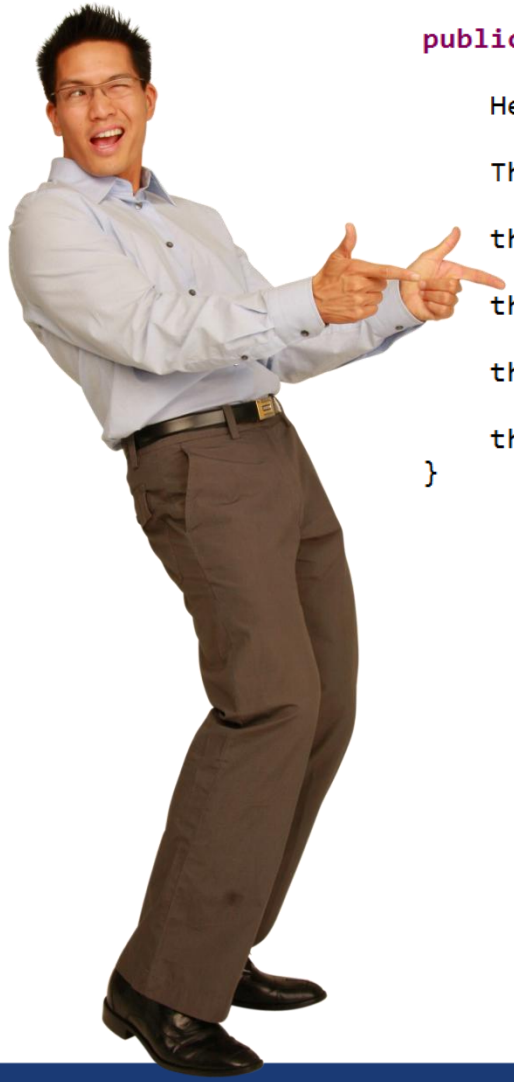
presente nas Threads

Escalonamento da Thread

O Escalonador seleciona um entre os processos em memória prontos para executar e aloca a CPU para ele



Prioridades de uma Thread



```
public static void main(String[] args) {
    HerancaThread threadSimples = new HerancaThread();
    Thread threadRunnable = new Thread(new RunnableThread());
    threadSimples.setPriority(Thread.MAX_PRIORITY);
    threadSimples.start();
    threadRunnable.setPriority(Thread.MIN_PRIORITY);
    threadRunnable.start();
}
```

Maior a chance de ser executado antes

Sincronização

Métodos **synchronized** têm um lock para acesso ao objeto.

métodos sem o modificador **synchronized** não exigem o lock do objeto para acesso.

Objeto

Cada objeto tem um lock de acesso.

Bloqueando acesso Concorrente

A coordenação é feita com métodos da classe Object: