Java's Threads

Threads are the fundamental building blocks, to support concurrency, in a Java application.

They're essential, because they allow us to perform multiple tasks simultaneously, within a single process.

The java.util.Thread Class

You can see that this class implements the Runnable interface, which has a single abstract method, the run method.

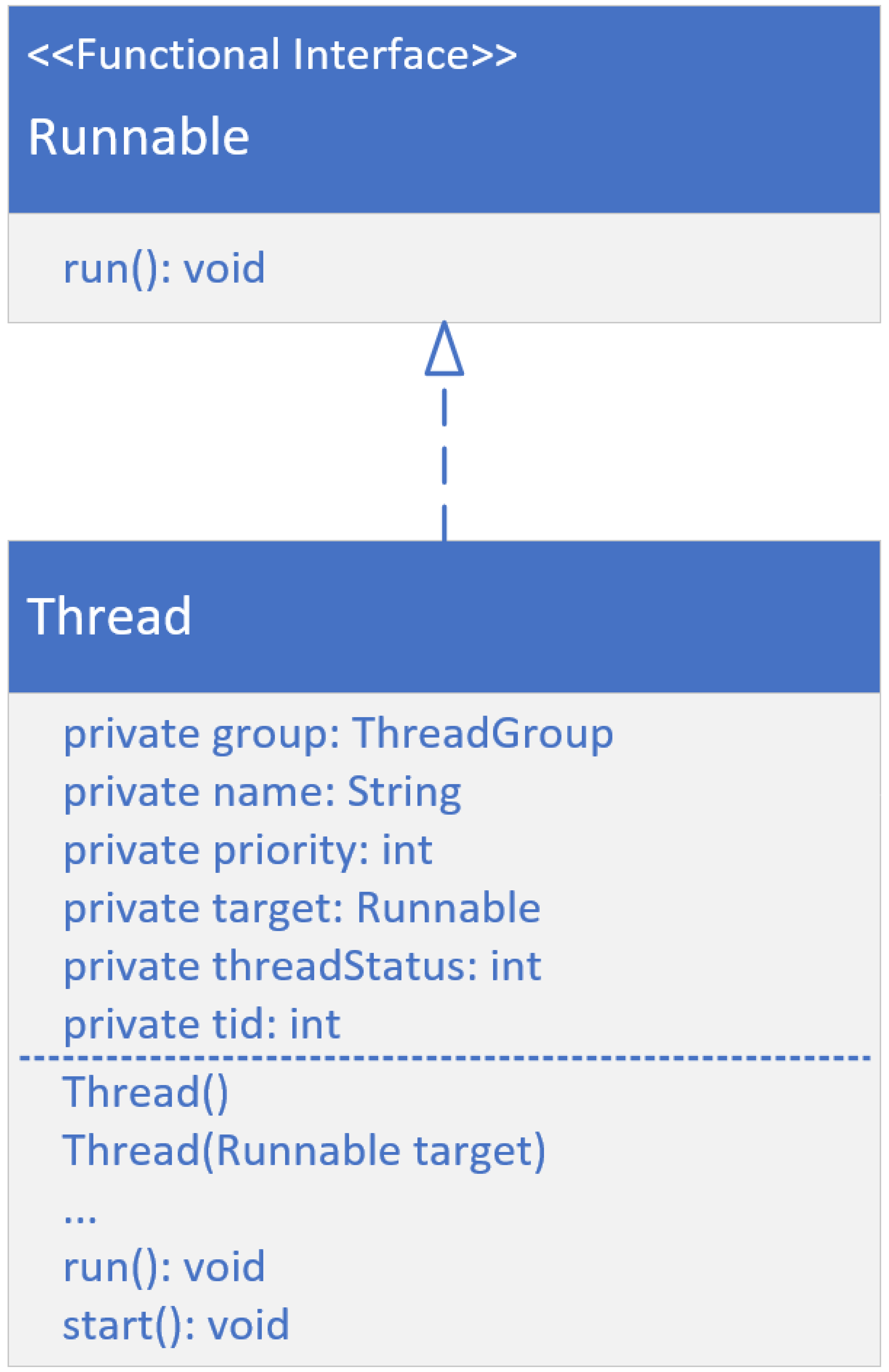
Each instance of a thread has some state.

The fields displayed here are all encapsulated and this includes thread group, a name, a priority, a status, and a thread id.

A thread can be constructed with no arguments.

It can be constructed by passing a Runnable instance to it.

The first method is the run method, which has to be overridden, since it's declared abstract on the Runnable interface.



Thread Priority

Thread priority is a value from 1 to 10.

The Thread class has three pre-defined priorities, included as constants.

```
Thread.MIN_PRIORITY = 1 (low)

Thread.NORM_PRIORITY  = 5 (default)

Thread.MAX_PRIORITY = 10 (high)
```

Higher-priority threads have a better chance of being scheduled, by a thread scheduler, over the lower-priority threads.

However, priority behavior can vary across different operating systems and JVM implementations.

You can think of this priority as more of a suggestion, to the thread management process.

Creating a Thread Instance

- Extend the Thread class, and create an instance of this new subclass.
- Create a new instance of Thread, and pass it any instance that implements the Runnable interface. This includes passing a lambda expression.
- Use an Executor, to create one or more threads for you.