**Castelazo - THREAD\_EXECUTORS\_RUNNABLES**

**Threads**

Threading is a facility to allow multiple tasks to run concurrently within a single process. Threads are independent, concurrent execution through a program, and each thread has its own stack. Threads are implemented by “Extending Thread Class”. Extending Thread Class is required to override ***run()*** method. The actual logic to be executed by thread is located inside the ***run()*** method. You need to call the start() method to start executing the thread object.

Disadvantages of using this method are as follows:

* Creating a new thread causes some performance overhead.
* Too many threads can lead to reduced performance, as the CPU needs to switch between these threads.
* You cannot easily control the number of threads, therefore you may run into out of memory errors due to too many threads.

An example of using the extending thread code is found here <https://github.com/castelazo-giovanni/CIT360-WINTER2019/blob/master/JAVA%20CODE%20EXAMPLES/codeExamples/src/Thread_Runnable/creatingThread.java>

**Runnables**

A Thread can be created by extending ***Thread*** class also, but Java only allows one class to extend. It will not allow multiple inheritance. This is why it’s always better to create a thread by implementing a ***Runnable*** interface. This is how Java allows you to implement multiple interfaces at a time. By implementing ***Runnable*** interface, you need to provide an implementation for the ***run()*** method.

To run this implementation class, first create a ***Thread*** object, then pass ***Runnable*** implementation class object to its constructor, finally call the **start()** method on thread class in order to start executing the ***run()*** method.

Implementing the **Runnable** interface does not create a ***Thread*** object, it only defines an entry point for threads in your object. It allows you to pass the object to the Thread(Runnable implementation) constructor.

An example of using the runnable code is found here <https://github.com/castelazo-giovanni/CIT360-WINTER2019/blob/master/JAVA%20CODE%20EXAMPLES/codeExamples/src/Thread_Runnable/creatingRunnable.java>

**Executors**

In order to use executors, you must first import the correct libraries. The ***java.util.concurrent*** package defines three executor interfaces.

***Executor*** is a simple interface that supports the launching of new tasks. It provides a single method, execute, designed to be a drop-in replacement for a common thread-creation idiom.

***ExecutorService*** is a subinterface of ***Executor*** which adds features that help manage the lifecycle of both of the individual tasks and of the executor itself. When using the ***ExecutorService***, you must remember to shut it down when you are done. This is done with the shutdown() method.

***ScheduledExecutorService*** is a subinterface of ***ExecutorService*** that supports future and/or periodic execution of tasks. This allows you to run task after a specified delay. It also lets you run tasks at different intervals.

An example of using the executor code is found here <https://github.com/castelazo-giovanni/CIT360-WINTER2019/blob/master/JAVA%20CODE%20EXAMPLES/codeExamples/src/Thread_Runnable/creatingExecutor.java>