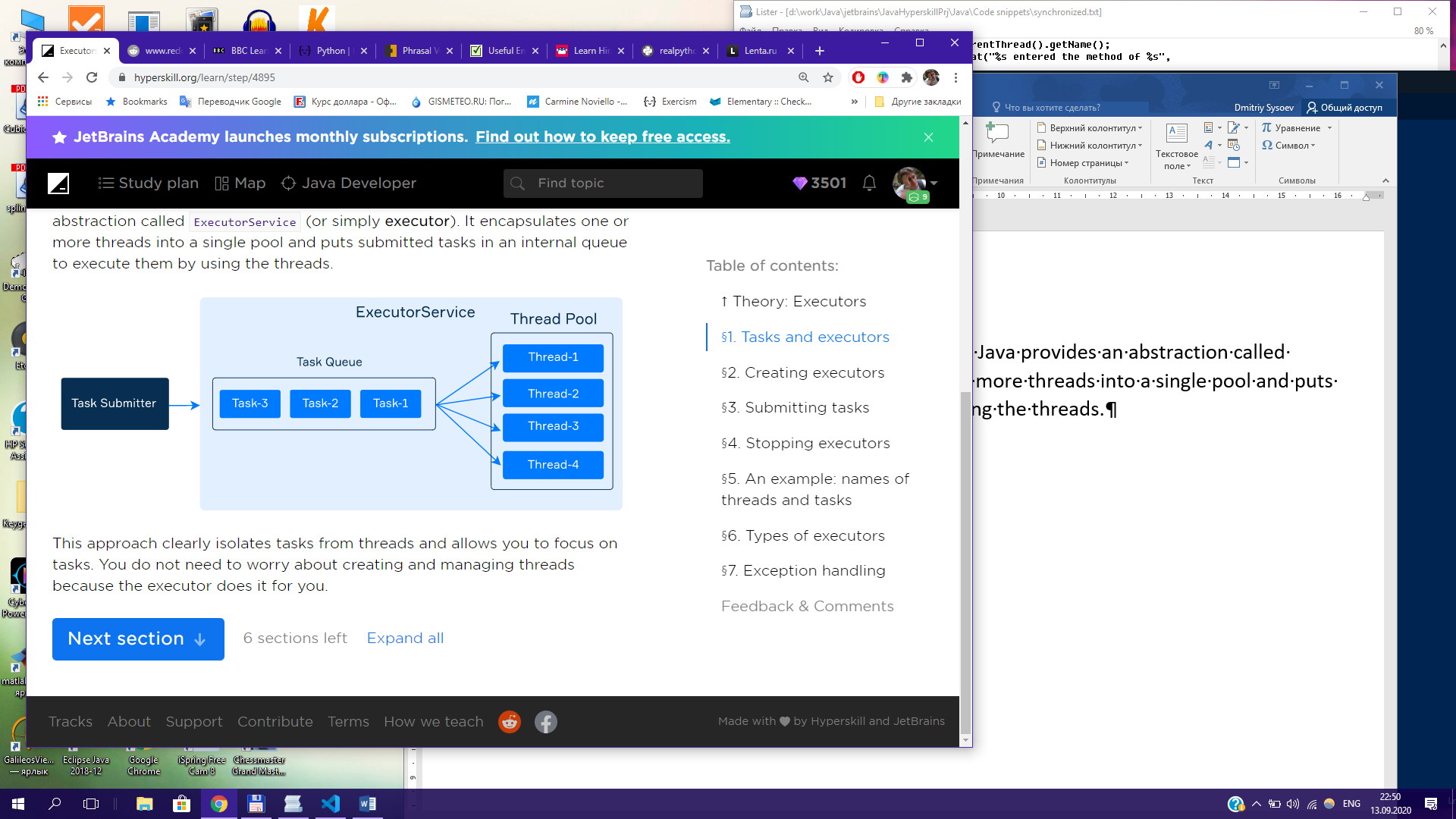
# Task and executors

To simplify the development of multi-threaded applications, Java provides an abstraction called ExecutorService (or simply executor). It encapsulates one or more threads into a single pool and puts submitted tasks in an internal queue to execute them by using the threads.



This approach clearly isolates tasks from threads and allows you to focus on tasks. You do not need to worry about creating and managing threads because the executor does it for you.

All types of executors are located in the **java.util.concurrent** package. This package also contains a convenient utility class Executors for creating different types of ExecutorServices’s.

ExecutorService executor = Executors.newFixedThreadPool(4);

An executor has the submit method that accepts a Runnable task to be executed. Since Runnable is a functional interface, it is possible to use a lambda expression as a task.

As an example, here we submit a task that prints “Hello!” to the standard output.

Executor.submit( () -> System.out.println(“Hello!”));

After invoking submit, the current thread does not wait for the task to complete. It just adds the task to the executor’s internal queue to be executed asynchronously by one of the threads.