Concurrency and Multithreading Lab

Introduction to Concurrency and Multithreading

Concurrency refers to the ability of a system to handle multiple tasks simultaneously. In a concurrent system, multiple tasks can make progress without waiting for others to complete, leading to more efficient resource utilization.

Multithreading, on the other hand, involves the execution of multiple threads within a single process. Each thread can perform a different task simultaneously, allowing for parallelism in execution.

Java Concurrency Utilities

Java provides several concurrency utilities that facilitate the development of thread-safe applications. These utilities include concurrent collections such as ConcurrentHashMap and CopyOnWriteArrayList, which are designed to handle concurrent access from multiple threads.

ConcurrentHashMap is a thread-safe implementation of a hash map, allowing for safe concurrent reads and writes. CopyOnWriteArrayList is a thread-safe implementation of a list that creates a copy of the array with each write operation, ensuring thread safety without using locks.

Lab Implementation

The lab is implemented using the MVC architecture, with the controller handling HTTP requests and the service layer managing the business logic. The TaskService class is responsible for managing tasks using concurrent collections.

Console outputs have been added to demonstrate how concurrency is handled within the application. Each operation, such as adding, retrieving, updating, or deleting tasks, is logged with the thread name to show how different threads manage these operations concurrently.

Performance Comparison

A performance comparison was conducted to measure the efficiency of concurrent collections versus non-concurrent collections. The comparison involved inserting a large number of entries into a ConcurrentHashMap and a non-concurrent map, then measuring the time taken for each operation.

The results indicate that concurrent collections provide better performance in multi-threaded environments, where multiple threads perform read/write operations simultaneously. This efficiency comes from the thread-safe mechanisms built into concurrent collections.

Conclusion

This lab has demonstrated the concepts of concurrency and multithreading using Java's concurrency utilities. The use of concurrent collections like ConcurrentHashMap and CopyOnWriteArrayList has been shown to improve the efficiency and safety of operations in a multi-threaded environment.

Understanding these concepts is crucial for developing robust applications that can handle the demands of modern, concurrent systems.