Work Stealing Thread Pool

The work stealing thread pool is used for parallelism, and concurrent execution of tasks.

Each worker thread has its own task queue.

When a worker thread finishes its own tasks, and its queue is empty, it can "steal" tasks from the back of other worker threads' queues.

This helps to balance the workload among threads, reduces idle time, and optimizes resource usage.

The ForkJoinPool

The ForkJoinPool class is Java's implementation of the Work Stealing Pool.

It's based on the fork-join, or divide and conquer algorithm of computing.

This algorithm

- breaks down a complex task into smaller subtasks,
- processes them independently and in parallel,
- and then combines the results to solve the original problem.

Types of Parallelism

There are two main types of parallelism:

- **Task parallelism** divides a program into smaller tasks that get executed concurrently. Each task can run on a separate thread or processor core.
- **Data parallelism** processes different parts of the same data concurrently.