Wafer Stage Simulator Documentation

## Overview

The wafer stage simulator emulates the movement of a wafer stage in semiconductor

manufacturing, supporting uniform motion.

## Key Features

* Simulates constant-speed moves.
* Reads input from a text file named “*moves.txt*” which contains a series of Moves. They are characterized by a Target Position and a Speed.
* Provides logging functionality for detailed movement tracking.
* Customizable FPS for smooth motion visualization.

## Class Documentation

* **WaferStage** - This is a class that is used to wrap the logic of a wafer stage.

It has two methods, inherited from the Moveable and Printable abstract classes. They run simultaneously, as one of them wraps the logic of updating the current position of the wafer stage with intent of reaching a target position with a specific speed, while the other one is responsible for visualizing the wafer stage on the console. Logging of the position update is supported in the Move method as well as logging of the expected and the actual time that the movement took.

* **FileManager -** This is a class responsible for handling the file reading and writing by following the RAII principles. It has two methods - ReadLine() and WriteLine() which are used for reading a row from a file and writing a row in a file.
* **MovesParser -** This is a class that parses the lines from the input file (“*moves.txt*”) and adds Move objects to a vector. This logic is located in the ParseMoves() method.
* **Logger -** responsible for managing logging operations, specifically for recording events related to a Wafer Stage's movement. It interacts with the FileManager to handle file I/O for logging data.
* **Moveable -** The Moveable class serves as an abstract base class (interface) that defines the contract for objects capable of executing movement. It enforces any derived class to implement the Move function, which outlines how the movement should be executed for that specific type.
* **Printable -** The Printable class defines an interface for objects that can be printed. It mandates that any derived class must implement the Print function, which handles the logic for displaying or outputting the object's state or information.