**Understanding Shared Pages, Reentrant Code, and Throttling**

**Shared Pages**

* **Definition:** Memory regions that can be accessed by multiple processes or threads simultaneously.
* **Benefits:**
  + Efficient resource utilization.
  + Reduced memory footprint.
  + Improved performance in certain scenarios.
* **Challenges:**
  + Synchronization issues to prevent race conditions.
  + Potential for data corruption.
* **Common Use Cases:**
  + Shared libraries
  + Data structures accessed by multiple threads
  + Memory-mapped files

**Reentrant Code**

* **Definition:** Code that can be safely executed by multiple concurrent threads or processes without causing interference.
* **Characteristics:**
  + Doesn't modify global variables or static data.
  + Uses thread-local storage for thread-specific data.
  + Avoids non-reentrant system calls.
* **Importance:**
  + Essential for multi-threaded and multi-process applications.
  + Helps prevent race conditions and data corruption.
* **Examples:**
  + Pure functions
  + Thread-safe libraries

**Throttling**

* **Definition:** A technique used to control the rate at which a process or system performs operations.
* **Purpose:**
  + Prevent overloading of resources.
  + Ensure fair resource allocation.
  + Improve system stability.
* **Methods:**
  + Token bucket algorithm
  + Leaky bucket algorithm
  + Rate limiting
* **Applications:**
  + Network traffic management
  + API rate limiting
  + Database query optimization
  + *Modern Operating Systems* by Andrew Tanenbaum and Herbert Bos
  + **Concurrency and Parallelism:**
  + *The Art of Concurrency* by Doug Lea
  + *Concurrent Programming on Windows* by Joe Duffy