Threads

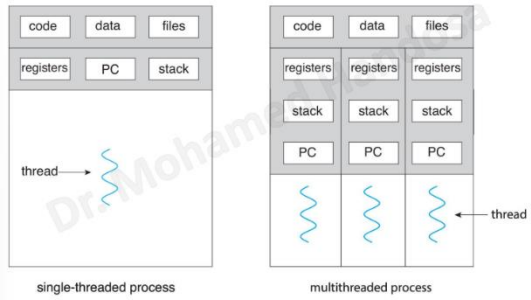
اللَّهُمَّ صَلِّ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّدٍ، كَمَا صَلَّيْتَ عَلَى إِبْرَاهِيمَ، وَبَارِكْ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّد،
كَمَا بَارَكْتَ عَلَى آلِ إِبْرَاهِيمَ، فِي الْعَالَمِينَ، إِنَّكَ حَمِيدٌ مَجِيدٌ.

By: Mohamed Gamal Maklad

❖ **Threads**:

- thread is a **single sequence of execution**. مجموعه من الأوامر بتتنفذ بالترتيب
- thread **is the basic unit of CPU utilization**. هو الوحده البنائيه الاساسيه للسي بي يو
- **traditional process has a single thread of control**.
- ***single-threaded process*** can perform **one task at a time**.
- ***A multi-threaded process*** can perform **multiple tasks at a time**.
- thread comprises(مكوناته) a **thread ID, a PC, a register set, and a stack**.
- **Process threads share code section, data section, and resources**.

ال process اللي عندها signal thread متقدرش تعمل غير تاسك واحده لوحده اما بقي اللي عندها multiple thread تقدر تعمل أكتر من تاسك في نفس الوقت مع بعض علي التوازي



- شايف هنا عندي multiple thread ف ال process بتعمل شير لل data , aode , file وده طبعا هيوفر عليا وهيكون اوفر في الوقت و الفلوس كمان

❖ **Benefits of Multithreaded Programs**:

- ***Responsiveness.*** A **blocked thread doesn't block the process.** لو ثريد قفلت ف العمليه بتفضل شغاله عادي
- ***Resource sharing.*** **Threads share the resources of the process**.
- ***Economy***. Thread **creation has less overhead than process creation**. بيتشاركوا ف الحاجه زي ما قولنا ف كده اوفر ليا
- ***Scalability***. **Threads can run in parallel** on a **multiprocessor machine** زيادة الإنتاج بسبب ان هما شغالين توازي زي ما قولنا

❖ **Multi-core Programming Challenges**: تعالي بقي نشوف التحديات اللي بتقابله او الصعوبات ايه

- ***Identifying tasks***. **divide program into tasks** that **can run in parallel**. يقسم البرنامج لتاسكات تشتغل مع بعض علي التوازي
- ***Balance***. **Ensure that these tasks perform equal work** of **equal value**. يتأكد انه مقسمهم قد بعض
- ***Data splitting***. **Divide data between tasks that run on separate cores**. يقسم بقي الداتا علي التاسكات دي
- ***Data dependency***. **Examine data to find dependencies between tasks**. ممكن ثريد تحتاج حاجه من ثريد تانيه علشان تكمل
- ***Testing and debugging***. Testing a **program with many execution paths**. لو في خطأ هيكون صعب انك تكتشف هو فين بالظبط

❖ **Types of Parallelism**:

- ***Data Parallelism***: **Distribution of data across multiple cores**

في النوع ده احنا بنقسم الداتا اللي عندنا بمعني لو عندنا ثريد مفروض يصحح ورق الامتحان مثلا وفي 100 سؤال فبدل ماكان ثريد يعلم سؤال ب سؤال لا هو بيروح يعمل مثلا 4 ثريد كل واحد بيعلم 25 سؤال و كلهم بيشتغلوا توالي ف كده هيقلل الوقت للربع و لو بصينا هنلاقي كلهم بيعملوا نفس الشغلانه كلهم بيعلموا 25 سؤال

- ***Task Parallelism***: **Distribution of tasks across multiple cores**

هنا بقي بيقم التاسكات علي الثريد بحيث كل واحد بيعمل تاسك معين يعني مثلا انت h تطلب منك ف البيت تروح تجيب خضار و تروح تجيب حاجات من البقاله فقولت انا مالي هو كل حاجه عليا لا انا هاخد اخويا الصغير هو يعمل حاجه وانا اعمل حاجه فتاخد اخوك الصغير و هو يروح البقاله وانت تروح تجيب خضار كده انت قسمت المهام بحيث كل واحد بيعمل حاجه مختلفه بس في نفس الوقت ف كده وفرت وقت برضه

❖ **User Threads versus Kernel Threads**:

- **Support for threads may be provided either**
  - At the **user level, for user threads**. **ممكن الثريد يتعمل عن طريق اليوزر ب انه يستخدم مكاتب جاهزه**
  - By the **operating system, for kernel threads**. **او نظام التشغيل اللي يعمل الثريد هو**
- ***User threads*** **are managed without OS support**. **هنا في اليوزر ثريد نظام التشغيل ملوش دعوه بيها**
- ***Kernel threads*** are supported and managed directly by the OS.
- Most modern OSs **support kernel threads** (e.g., Windows, Linux).

❖ **Multithreading Models**: تعالي نشوف الأنواع

❖ **Many-to-One Model**:

- **Only one user thread can access the kernel at a time**.
- **Threads cannot run in parallel on multi-core systems**.
- **entire process will block if a thread makes a blocking system call**.
- Thread management is done in user space **by a thread library not OS.**

هنا بيكون في اكتر من user thread و واحده بس kernel thread كلهم بيحاولوا يعملوا access عليها اول ما واحد ينجح يقوم عامل lock ولما يخلص ممكن ثريد تاني يعمل access بحيث واحد بس اللي يقدر يعمل اكسيس

❖ **One-to-One Model**:

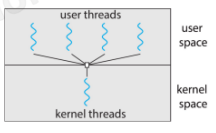
- **Multiple threads** can **access the kernel at a given time**. **اكتر من ثريد بتشتغل مع الكيرنل عادي**
- **Multiple threads can run in parallel on multi-core systems**. **الثريد بتشتغل علي التوازي**
- When a thread makes a blocking system call, **another thread can run**.**لو الثريد قفلت ممكن واحده تانيه تشتغل عادي**
- **OS creates a kernel thread per user thread (can burden performance)**

كل ما يتم انشاء user thread تقوم OS عامله kernel thread هتقولي طب ما ده حلو هقولك لا ده لو زاد اوي هيأثر علي الجهاز

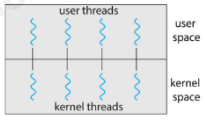
❖ **Many-to-Many Model**:

- **Multiple threads can access the kernel at a given time.**
- **Multiple threads can run in parallel on multi-core systems.**
- **When a thread makes a blocking system call, another thread can run.**
- **OS creates a set of kernel threads (less than or equal to user threads)**
- **Advantages of Many-to-Many Model**:
  - Does not suffer from the **drawback of the many-to-one model.** **Multiple user threads can run in parallel on a multicore system**.
  - Does not suffer from the **drawback of the one-to-one model**. **Increasing the number of user threads does not affect system performance**

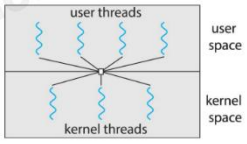
جه في بالك سؤال مهو كده زيه زي one to one هقولك لأ هو one to one كان كل user thread له kernel thread هنا لأ OS بتقوله انا اه هعملك لكل user thread واحده kernel thread بس اخري عدد معين لو كترت عنه تستني بقي kernel thread تفضي انا مش هعمل اكتر من كده علشان احافظ علي الجهاز هتقولي يعني OS قال كده هقولك اه و شوح ب ايده كمان 😊



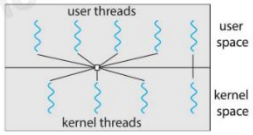
Many to one



One to One



Many to Many



Two-Level

بيقولك ان معظم OS بتستخدم one to one لانهم شايفين ان فكرة انك تعمل kernel thread لكل user thread عادي لان البروسيسور بقت اقوي بس المكتبات الحديثه بتستخدم many to many ف هنا بقي في TWO level بيعمل ميكس بين الاتنين

❖ **Two-Level Model**:

- **Most operating systems use the one-to-one model**.
  - ***In practice***, it is difficult to implement the many-to-many model.
  - **No need to limit the number of kernel threads on modern multicore systems**.
- **Modern concurrency libraries use the many-to-many model.**
  - **Developers identify tasks and the library maps them to kernel threads**.

❖ **Explicit Threading**:

- The **server creates a thread for each request.** معني كده انه ون - ون
- **thread is discarded once it has completed its work.** اول ما الثريد يخلص شغله بقفله
  - **Each new request requires creating a new thread** (i.e., **overhead**).
- The number of threads that can be created has **no upper bound.** كده هيعملي مشكلة ان أداء النظام بيقل بسبب كترة الثريد
  - Too many threads that work **concurrently can exhaust system resources**

❖ **Implicit Threading**:

هنا بقي بدل ما كان المبرمج بيكتب thread لا هو هنا بيعمل function وفي مكتبه بقي موجوده مسئوله عن عمل ال thread

- **Helps developing modern applications with hundreds of threads.**
- **Programmers identify tasks (not threads) that can run in parallel**.
  - The programmer **writes a task as a function**.
  - ***Run-time libraries*** **(not programmers) create and manage threads**.
- Each task is mapped to a separate thread using the ***many-to-many model***.
- **Two implicit threading approaches**:
  - ***Thread pool***. works well for **asynchronous** (**independent**) tasks.
  - ***Fork-join***. works well for tasks that are ***synchronous*** (**cooperating)**

❖ **Thread Pool**:

- At startup, **the server creates a set of threads in a pool.**
- When the server receives a request, it submits it to the pool.
  - If the **pool has a thread**, the **request is served immediately**.
  - If the **pool is empty**, the **task is queued until a thread becomes free**.
- Once **a thread completes serving the request, it returns to the pool**.
- **Thread pools offer two benefits**:
  - **Serving a request with an existing thread is faster than creating a thread**.
  - **Limiting the number of threads** helps **avoid exhausting system resources**.
- ***A dynamic thread pool*** can **adjust the number of threads in the pool**

هنا بيكون عندنا pool خلينا نشبه ب مخزن يجي ال server اول حاجه يعملها انه بينشئ thread يملي بيها المخزن ده قبل محد أصلا يطلب طب جاله request يقوم مديله thread افرض بقي اللي عنده خلص يقوم أي حد يطلب بعد كده يقف يستني لحد ما thread تاني يخلص و بعد كده ياخد thread لما يفضي

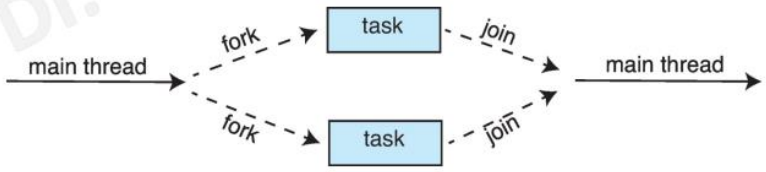
في هنا ميزه ان بدل ما ال thread كانت بتتدمر بعد ما تخلص شغلها لا هي بقي بترجع pool و يعاد استخدامها تاني

Dynamic thread دي نقصد بيها انه بينشئ thread في الأول ع قد ما الجهاز بتاعك يستحمل بحيث ما يأثرش ع الأداء

❖ **Fork-Join**:

هنا بقي ي هندسه اول ما المهم تيج يلل thread يقوم عامل children و مقسم عليهم المهام ويخليهم يشتغلوا بالتوازي مع بعض اول ما يخلوا يجمع النتائج منهم

- In traditional fork-join, the main parent thread does the following:
  1. **Create (i.e., forks) one or more child threads**.
  2. **Wait for the created threads (children) to terminate**.
  3. **Join the threads, at which point it can retrieve and combine their results**.
- ***Implicit fork-join* is a synchronous version of the thread pool**.
  - **Threads are not constructed but parallel tasks are designated**.
  - **A library manages the creation of threads and assigns tasks to those threads**.



❖ **Thread Cancellation**:

- **Refers to terminating a thread before it has completed**. **ده يعني بيقفل الثريد بعد ما تخلص شغلها**
- Cancellation of a ***target thread*** can be ***asynchronous*** ***or*** ***deferred***.**الثريد اللي هتتقفل اسمه الثريد الهدف و بتتقفل بطريقتين**
- **Asynchronous cancellation:**
  - **One thread immediately terminates the target thread**. **ده بتقفل الثريد فجأه**
- **Deferred cancellation**:
  - The target thread periodically **checks whether it should terminate.** **هنا بقي بيشوف هل الثريد ينفع يتقفل ولا لسه**
  - **This allows the target thread to terminate itself in an orderly fashion**

اللَّهُمَّ صَلِّ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّدٍ، كَمَا صَلَّيْتَ عَلَى إِبْرَاهِيمَ، وَبَارِكْ عَلَى مُحَمَّدٍ وَعَلَى آلِ مُحَمَّد،

كَمَا بَارَكْتَ عَلَى آلِ إِبْرَاهِيمَ، فِي الْعَالَمِينَ، إِنَّكَ حَمِيدٌ مَجِيدٌ.

By: Mohamed Gamal Maklad