ULB UNIVERSITÉ LIBRE DE BRUXELLES

CoDe

INFO-H-301
Programmation orientée objet

TP6
Threads

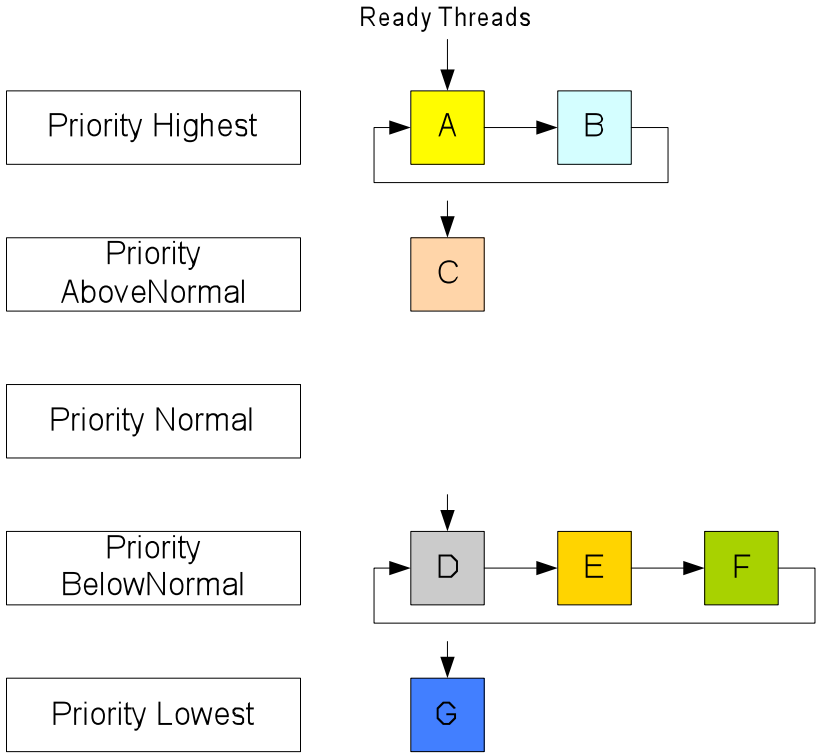
F. Servais & B. Verhaegen

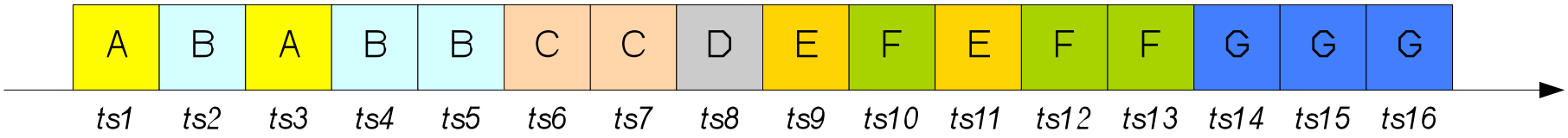
Thread

- Thread ~ Processus
- But : exécuter des opérations en parallèle.
- Exemple : Un logiciel de traitement de texte vérifie l'orthographe pendant que l'utilisateur tape du texte.
  - Un thread qui gère le clavier
  - Un thread qui gère la correction orthographique
- Pour donner une illusion de parallélisme sur un processeur, l'OS exécute les processus et threads de façon **concurrente** et **synchronisée**.

Ordonnancement

- Un des rôles de l'OS est d'orchestrer correctement les processus et threads.
- Pour cela, il se base sur les priorités





Thread en Java

- Deux manières de faire :
  - hériter de la classe `Thread`
  - implémenter l'interface `Runnable` (préféré)

- Une classe « *thread* » doit contenir une méthode `void run()` qui défini le comportement et le cycle de vie de ce *thread*.
- Une fois la méthode `run()` finie, le *thread* s'arrête et est détruit.
- Pour démarrer un *thread*, il faut appeler sa méthode `start();`

Exemple complet

```
public class MyTimer implements Runnable {

    String string;
    int waitTime;
    Thread thread;

    public MyTimer(String string, int waitTime){
        this.string = string;
        this.waitTime = waitTime;
        this.thread = new Thread(this);
        thread.start();
    }

    public void run() {
        // TODO Auto-generated method stub
        try{
            while(true){
                System.out.println(string);
                Thread.sleep(waitTime);
            }
        } catch (Exception e) {};
    }

    public static void main(String[] args) {
        MyTimer first = new MyTimer("first",2000);
        MyTimer second = new MyTimer("second",3000);
    }
}
```

Un objet Thread

Creation de l'objet Thread en lui donnant en paramètre un objet implémentant Runnable

Comportement du Thread

Qu'affiche à l'écran ce programme ?

Remarque : deadlocks

- « Étreinte mortelle »
- Situation d'interblocage

- Exemple :
  - soit deux threads `t1` et `t2` et deux ressources `r1` et `r2`
  - `t1` a la main et bloque `r1`
  - `t2` prend la main et bloque `r2`
  - `t1` prend la main et veut bloquer `r2`. Il doit attendre que `r2` soit libre et donc que `t2` la libère.
  - `t2` prend la main et veut bloquer `r1`. Il doit attendre que `r1` soit libre et donc que `t1` la libère.
  - Le système est bloqué