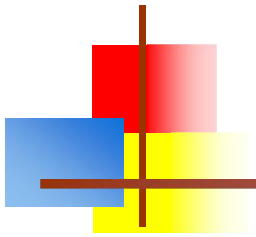
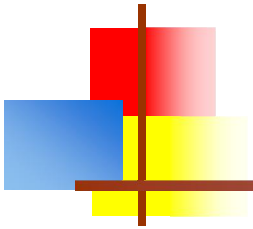


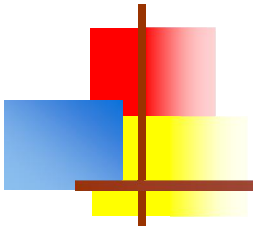
Concept de Thread



Utilisation d'un Thread

- Un **Thread** est un «**processus**» :
  - Léger : pas de réservation de ressources système (fichiers, canaux, signaux),
  - Simple : 1 registre Compteur de Programme et une pile.

- Sur un ordinateur, à un instant donné,
  - il n'y a qu'UN SEUL THREAD en cours d'exécution, et éventuellement d'autres threads en attente d'exécution

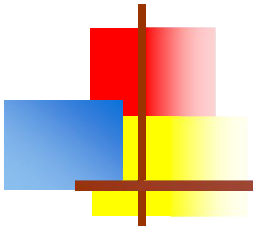


Utilisation d'un Thread

Deux méthodes, un point commun :

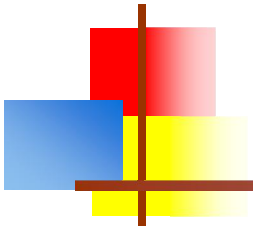
« écrire une méthode **run()** qui fait
ce que doit faire le thread. »

- Méthode 1 : **Dériver** une classe de **Thread**.
- Méthode 2 : **implémenter l'interface Runnable**.



Utilisation d'un Thread

- Par implémentation de l'interface
  - Usage
    - `public void MaClasse implements Runnable`
  - Avantages et inconvénients
    - ☺ Meilleur sur le plan orienté objet
    - ☺ La classe peut hériter d'une autre classe
    - ☺ Consistance



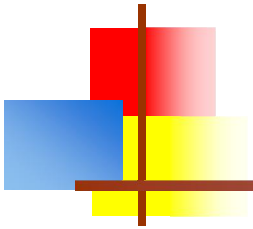
Utilisation d'un Thread

```
public class animeframe extends JFrame implements
Runnable {

   private Thread anim;


   …
   public animeframe () {
   anim = new Thread(this);
   anim.start();  }

public void paint (Graphics g) {
  g. draw…
 }
```



Utilisation d'un Thread

```
public void run () { // corps du thread d'animation
while (true) {
…
try { Thread.sleep (dt); }
catch (InterruptedException e) {}
   repaint () ; }

                                    }          // fin de run

 public static void main (String args[]) {
    new animframe().setVisible(true) ; }
}
```