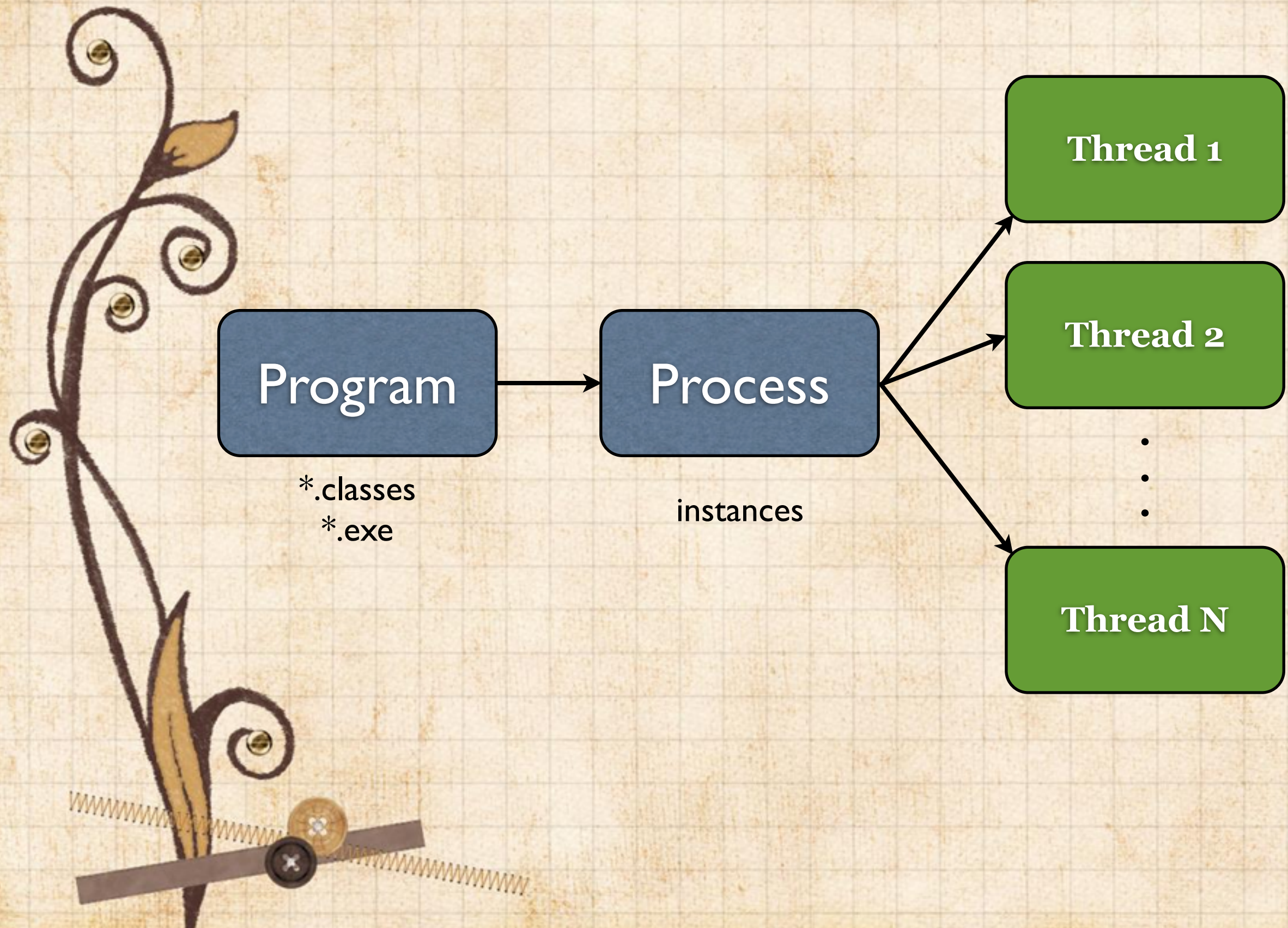
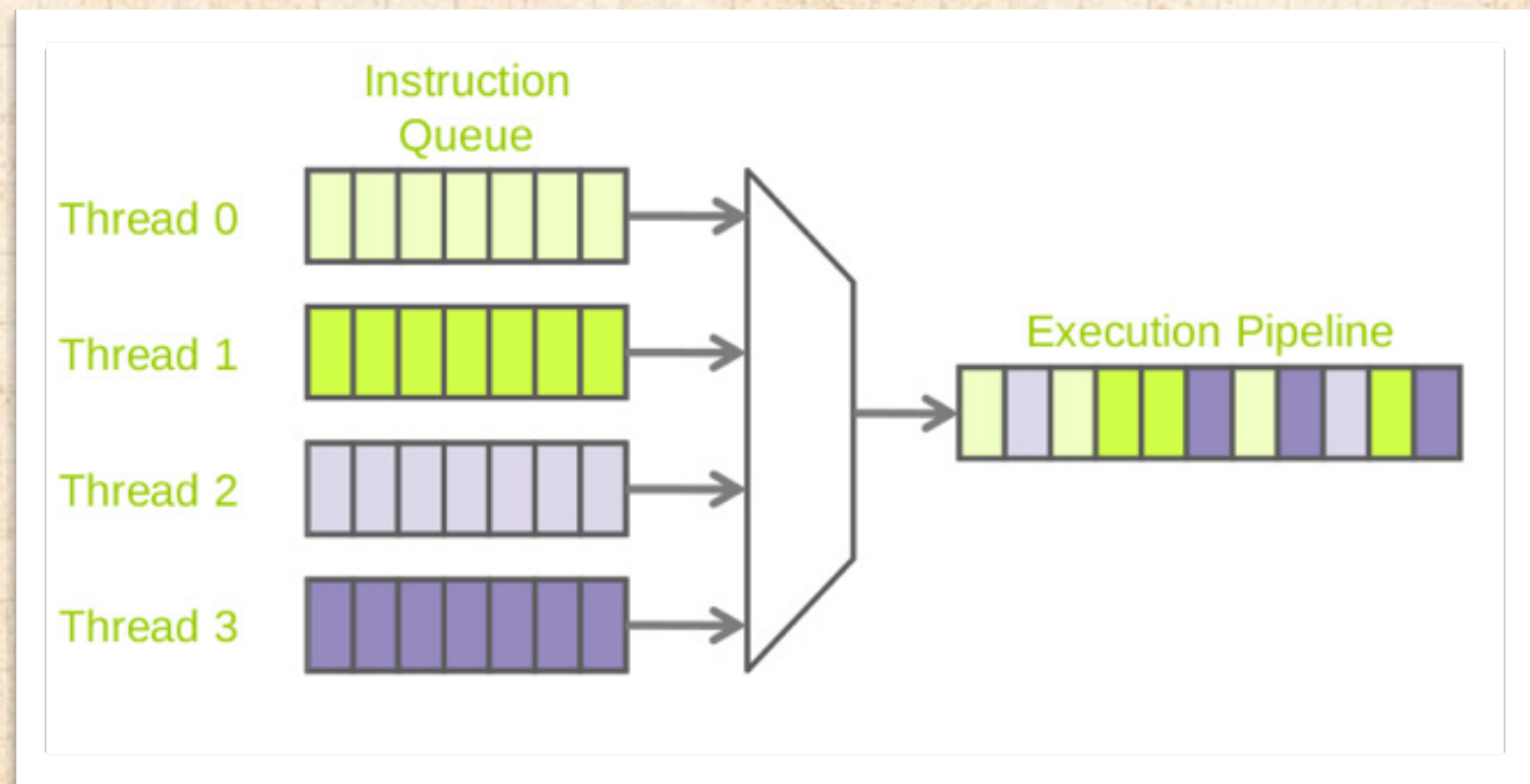
A decorative illustration on the left side of the slide. It features a dark brown, swirling vine with several golden-yellow leaves. At the bottom of the vine, there is a dark brown horizontal bar with two circular buttons (one gold, one dark brown) and a coiled spring mechanism.

Java 物件導向程式語言

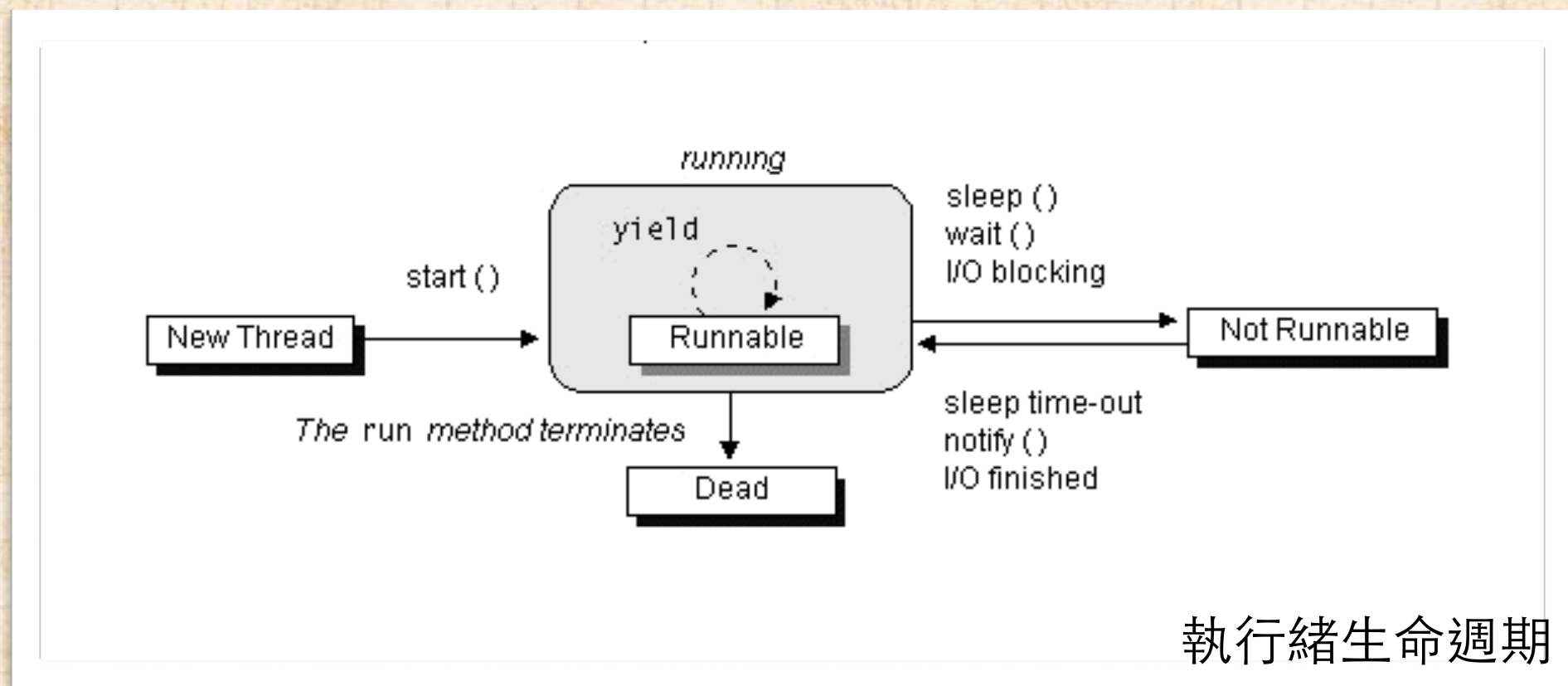
多執行緒程式設計



多執行緒處理佇列



Thread Life Cycle



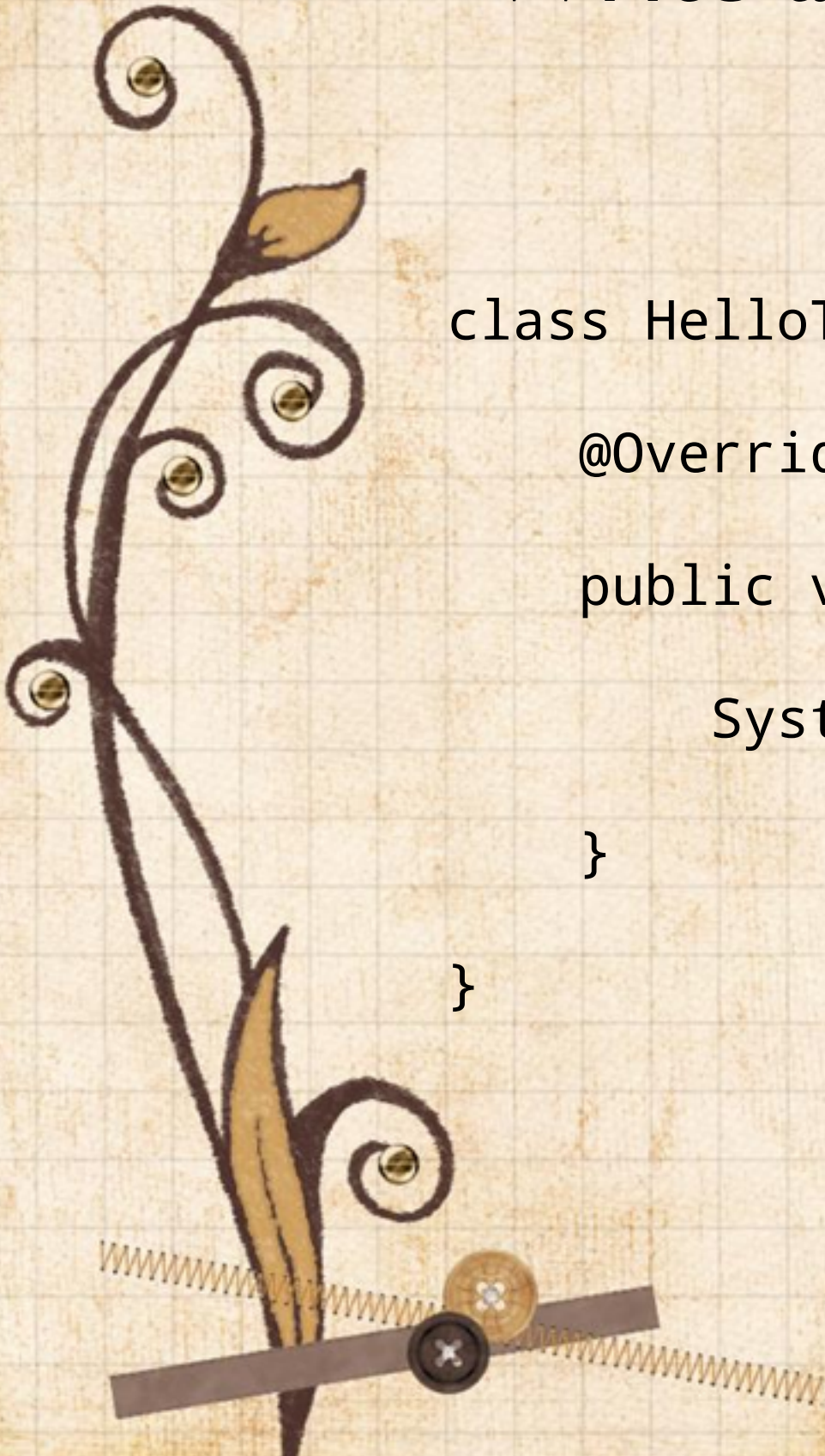
Create a Thread

```
Thread thread = new Thread();
```



Write a HelloThread class

```
class HelloThread extends Thread {  
    @Override  
    public void run() {  
        System.out.println("Hello");  
    }  
}
```




Thread .start()

```
Thread thread1 = new HelloThread();  
thread1.start();
```



HelloThread with constructor



```
class HelloThread extends Thread {  
    private final String who;  
    public HelloThread(String name) {  
        who = name;  
    }  
    @Override  
    public void run() {  
        System.out.println("Hello " + who);  
    }  
}
```


Start three thread

```
Thread thread1 = new HelloThread("John");
```

```
Thread thread2 = new HelloThread("Mary");
```

```
Thread thread3 = new HelloThread("Kyle");
```

```
// 猜猜執行結果？
```

```
thread1.start();
```

```
thread2.start();
```

```
thread3.start();
```



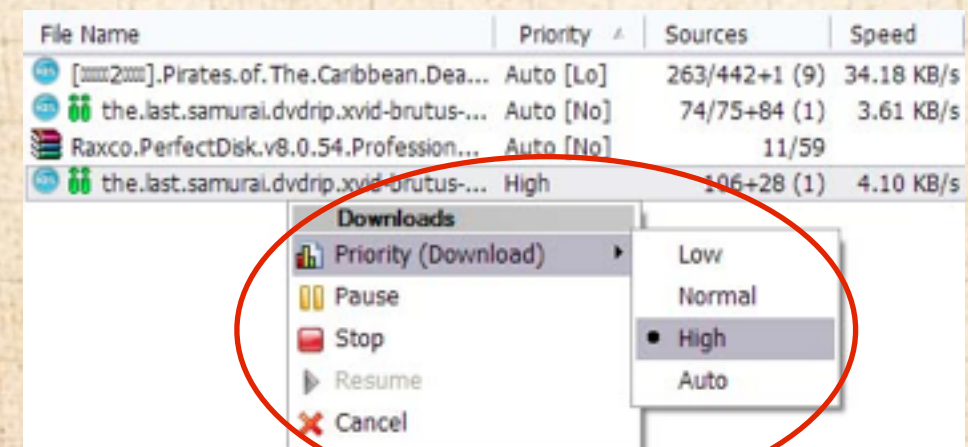
Thread Priority

Thread.MAX_PRIORITY //10

Thread.MIN_PRIORITY //1

Thread.NORM_PRIORITY //5

//優先權高，得到處理器時間的機會較高，但是“不保證”

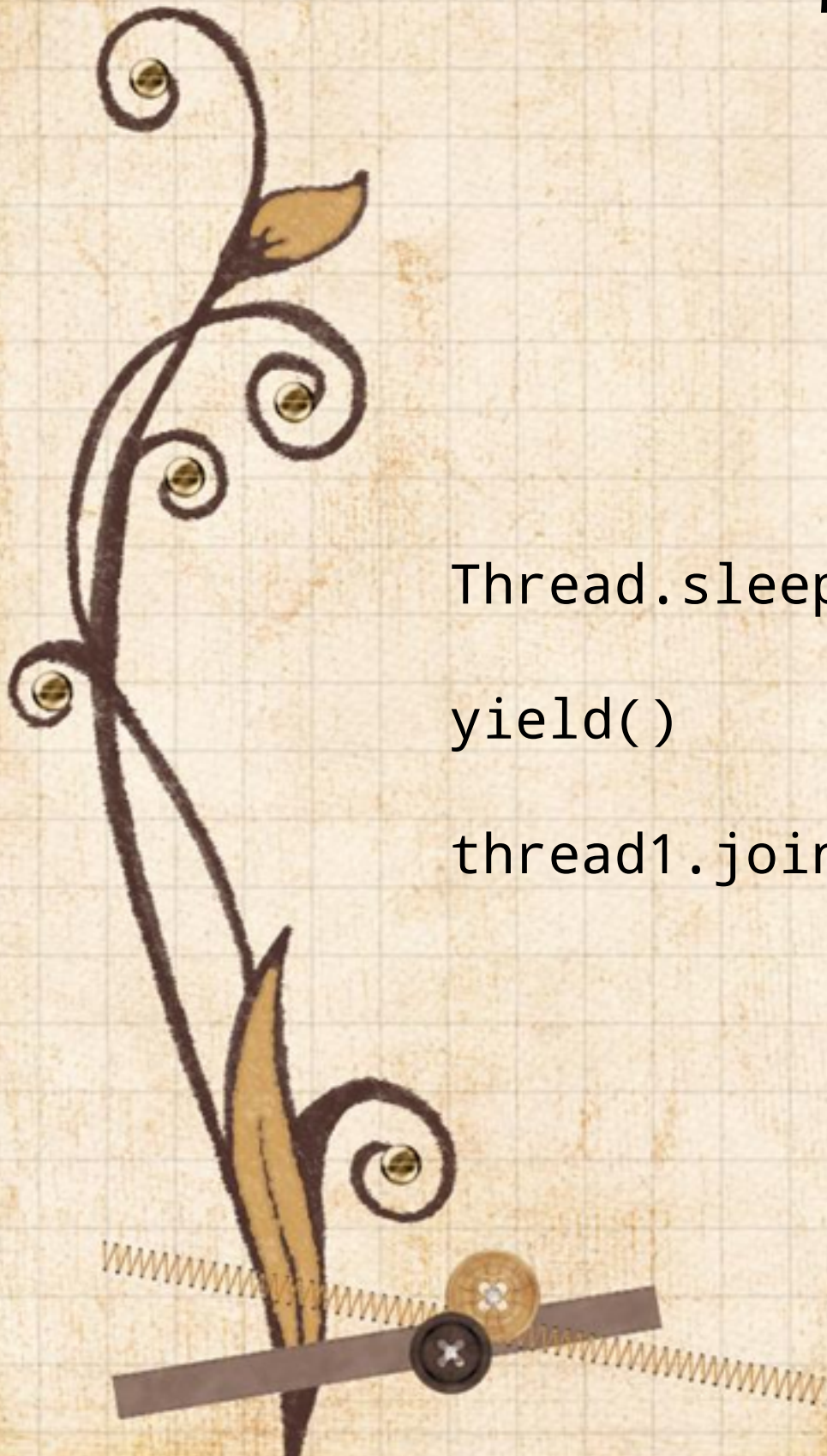


執行緒控制


`Thread.sleep(int time)` //讓目前的執行緒休息

`yield()` //讓給其他執行緒先處理

`thread1.join()` //等thread1結束再處理



CountDownLatch

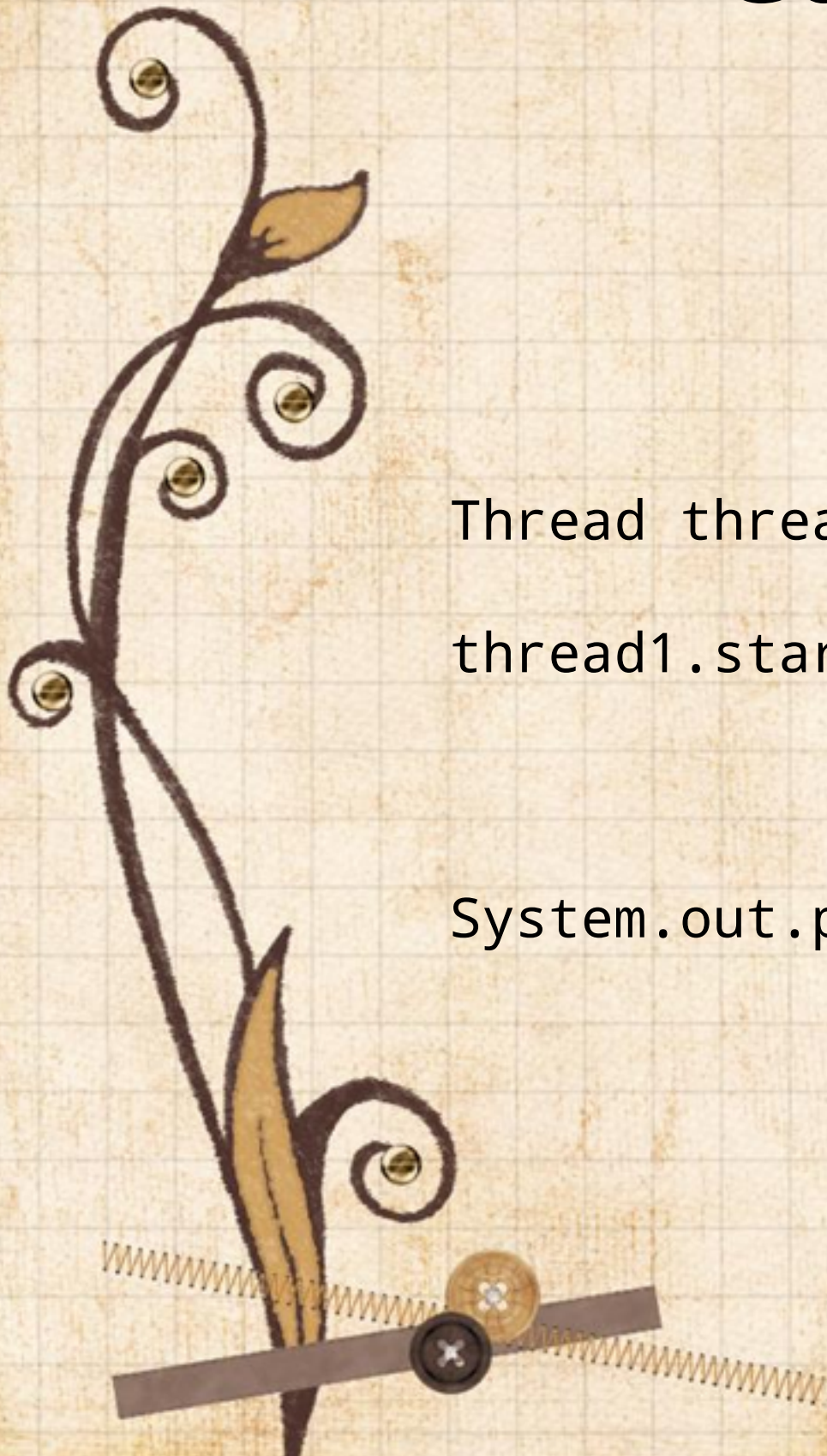


```
class CountDownLatch extends Thread {  
    private final int num;  
    public CountDownLatch(int num) {  
        this.num = num;  
    }  
    @Override  
    public void run() {  
        for (int i = num; i >= 0; i--) {  
            System.out.println(i);  
        }  
    }  
}
```


Count Down !!!

```
Thread thread1 = new CountDownLatch(100);  
thread1.start();
```

```
System.out.println("END");
```



Count Down with join()

```
Thread thread1 = new CountDownLatch(100);  
thread1.start();
```

```
thread1.join();
```

```
System.out.println("END");
```

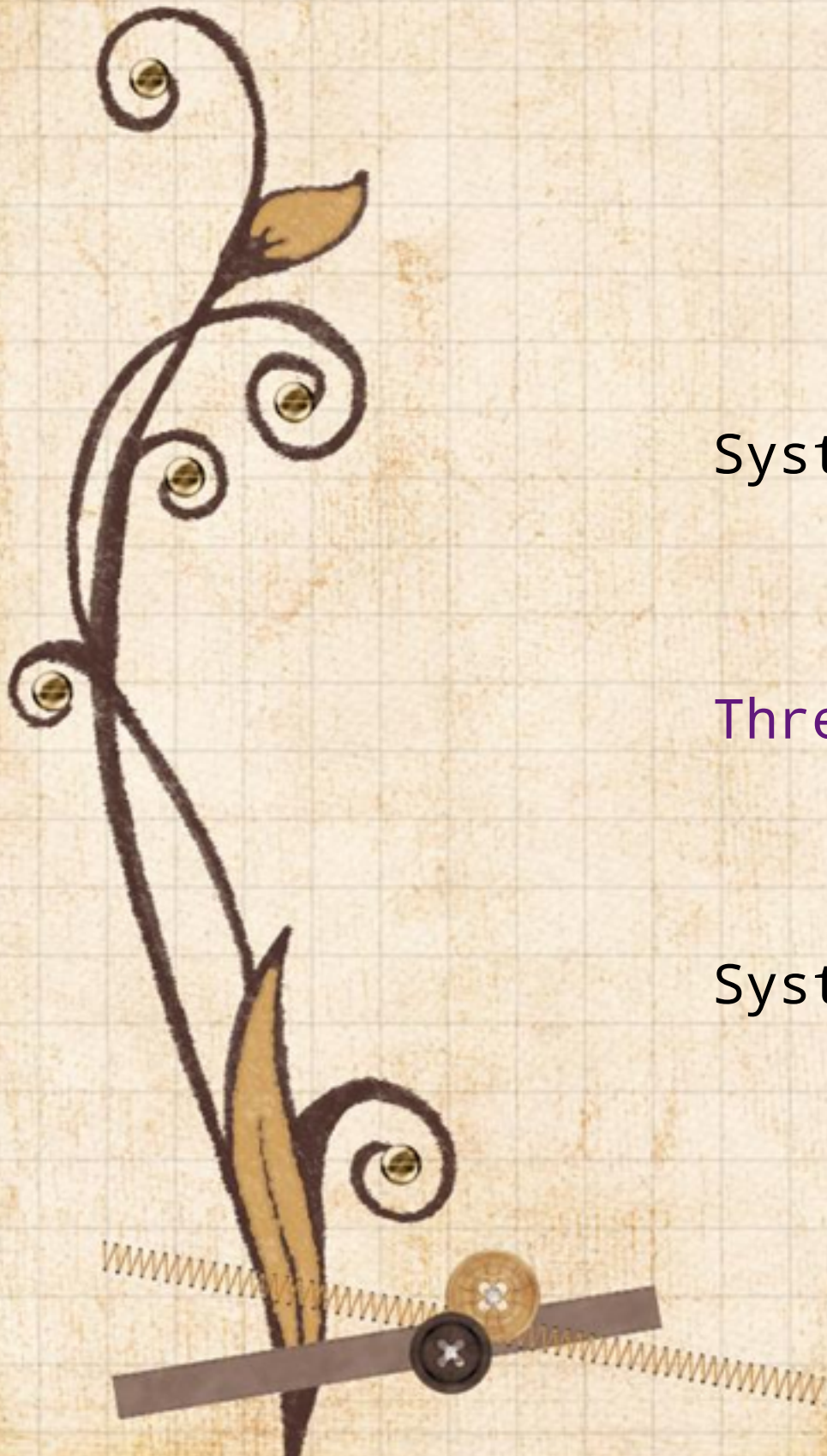


Sleep ...


```
System.out.println(new Date());
```

```
Thread.sleep(5000);
```

```
System.out.println(new Date());
```




Counter



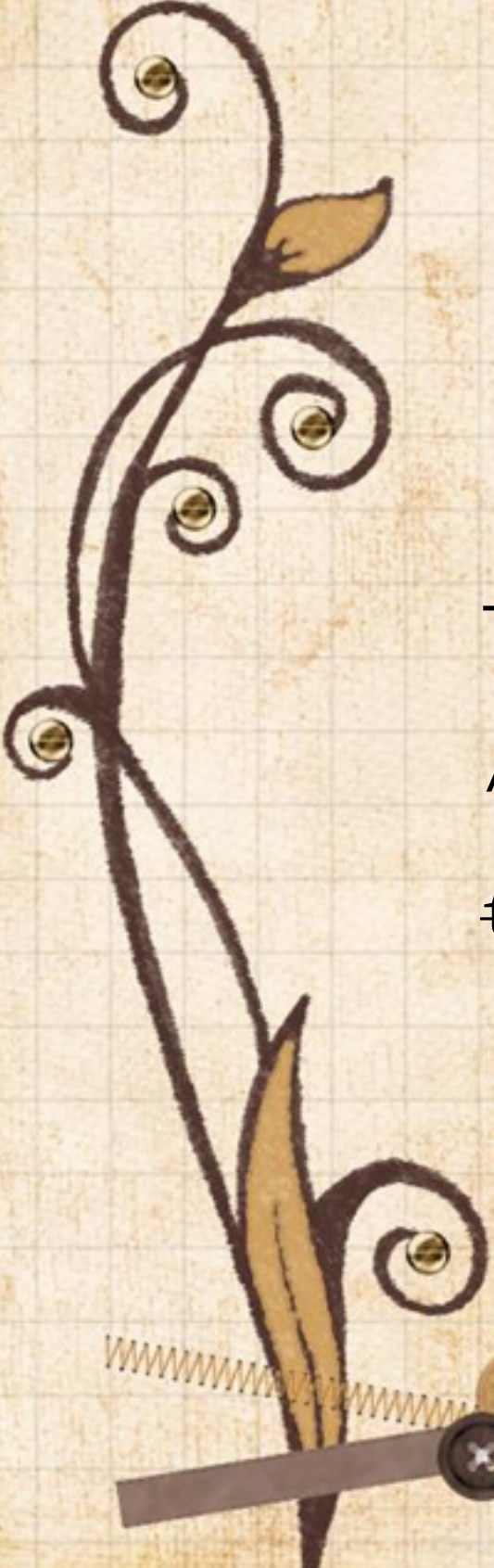
```
class Counter {  
    public int count = 0;  
    private Object obj = new Object();  
    public void plus() { count++; }  
    public synchronized void syncPlus() { count++; }  
    public void syncPlus2() {  
        synchronized(obj) {  
            count++;  
        }  
    }  
}
```


CounterThread



```
class CounterThread extends Thread {  
    private final Counter counter;  
    public CounterThread(Counter counter) {  
        this.counter = counter;  
    }  
    @Override  
    public void run() {  
        counter.plus();  
        //counter.syncPlus();  
        //counter.syncPlus2();  
    }  
}
```



停止執行緒



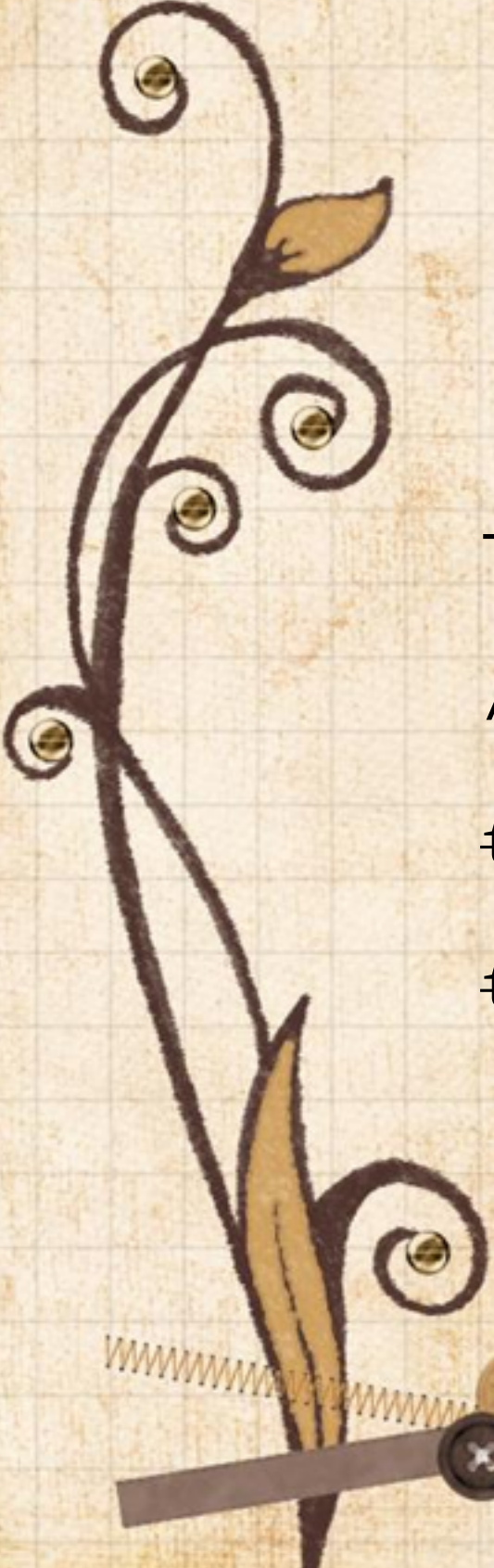
```
Thread thread1 = new SomeThread();
```

```
//deprecated
```

```
thread1.stop();
```



暫停、繼續執行緒




```
Thread thread1 = new SomeThread();
```

```
//deprecated
```

```
thread1.suspend();
```


```
thread1.resume();
```



Game

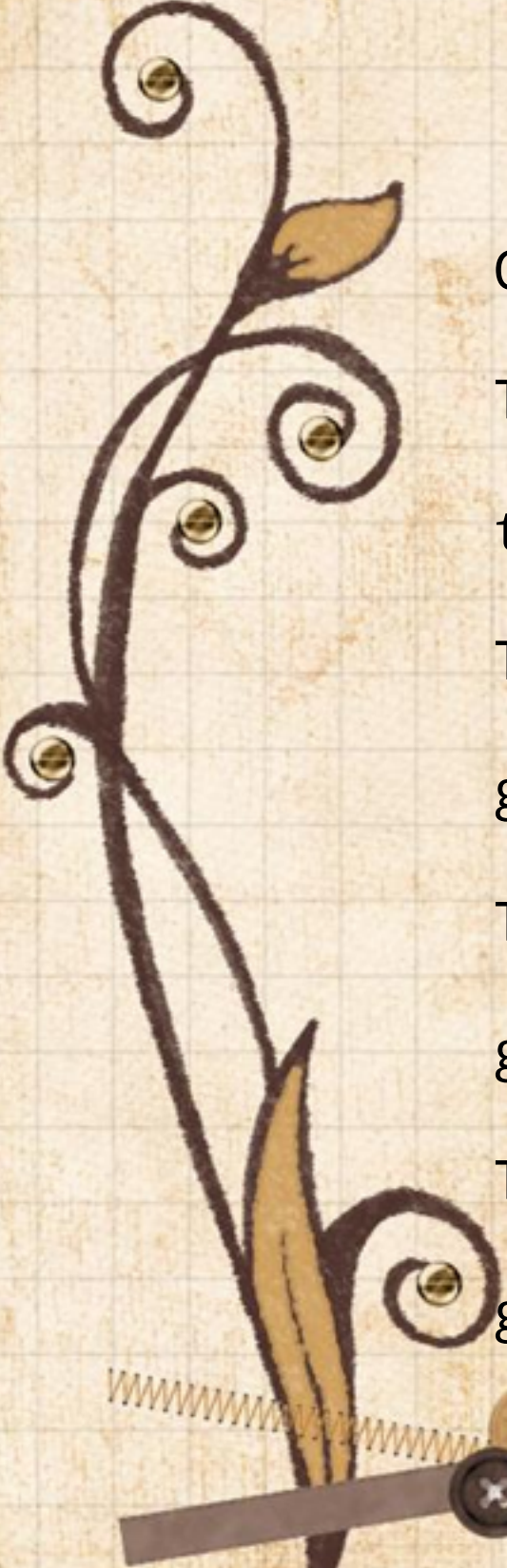
```
class Game implements Runnable {  
    private boolean isContinue = true;  
    private boolean isSuspend = false;  
    private int round = 0;  
  
    @Override  
    public void run() {  
        //...  
    }  
  
    public void stop() { isContinue = false; }  
    public void suspend() { isSuspend = true; }  
    public void resume() { isSuspend = false; }  
}
```


Game run()



```
@Override
public void run() {
    while (isContinue) {
        while (isContinue && !isSuspend) {
            try {
                Thread.sleep(1000);
                System.out.println("round " + ++round);
            } catch (InterruptedException ex) {
                ex.printStackTrace();
            }
        }
    }
}
```


Control the Game

A decorative vertical scrollwork element on the left side of the slide. It features a dark brown, swirling vine-like structure with several small, golden-yellow circular accents. At the bottom, there are two stylized leaves, one light brown and one dark brown, and a small, dark brown rectangular bar with a golden-yellow circular button and a dark brown cross-shaped button.

```
Game game = new Game();  
Thread thread = new Thread(game);  
thread.start();  
Thread.sleep(2000);  
game.suspend();  
Thread.sleep(2000);  
game.resume();  
Thread.sleep(3000);  
game.stop();
```


Swing Counter Practice



Start

1

2

3

4

5

6

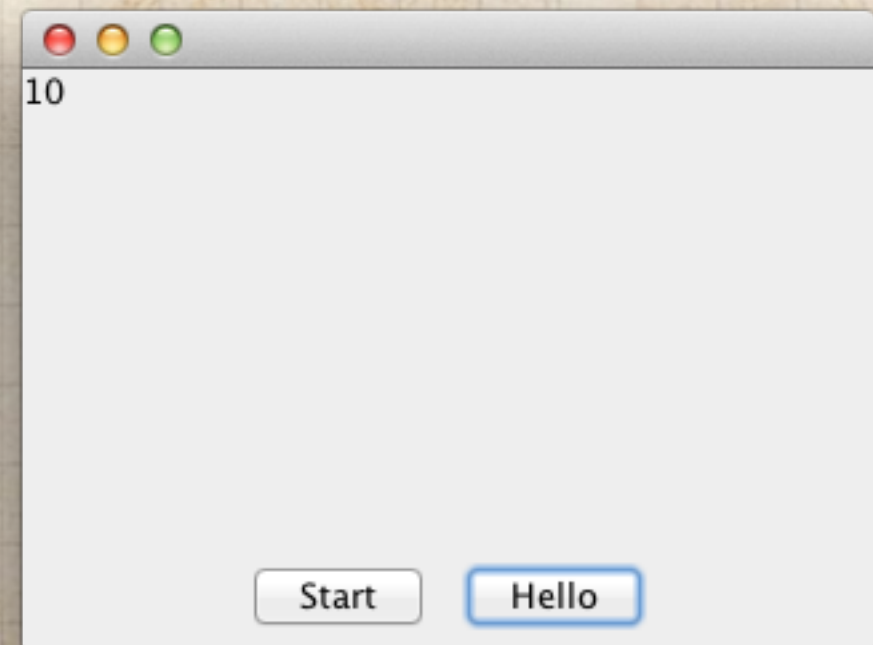
7

8

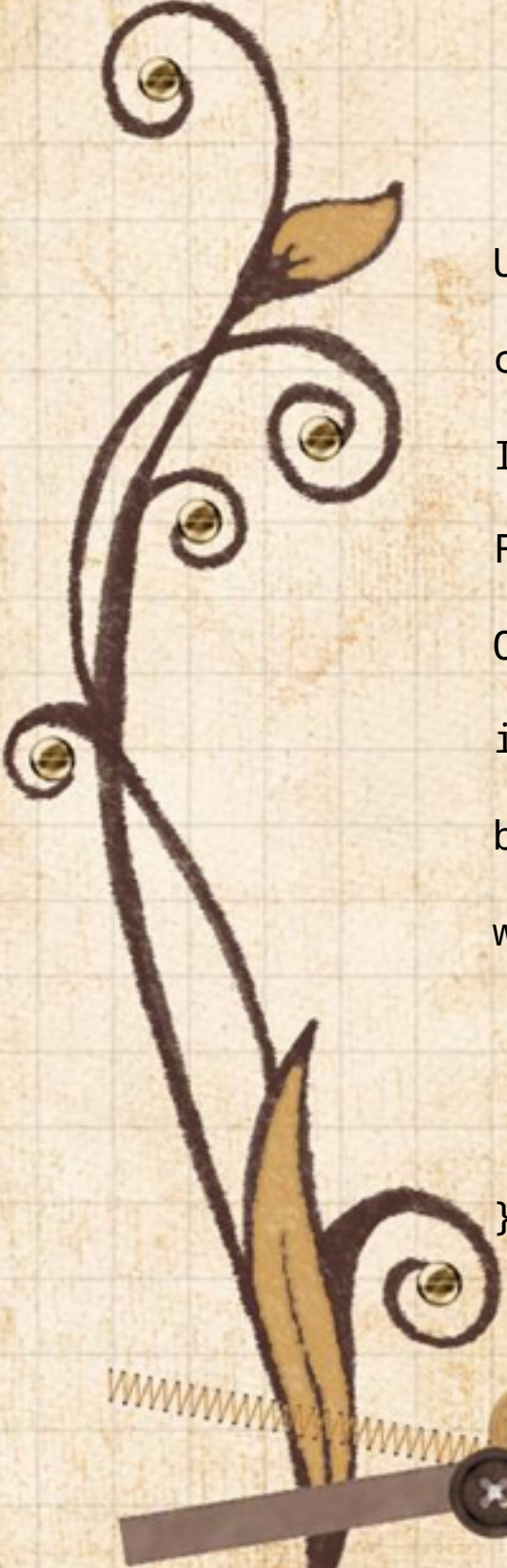
Hello

9

10



File Downloader Practice



```
URLConnection conn = url.openConnection();  
conn.connect();  
InputStream stream = conn.getInputStream();  
File saveToFile = new File(destPath, url.getFile());  
OutputStream outputStream = new FileOutputStream(saveToFile);  
int len = 0;  
byte[] bytes = new byte[1024];  
while ((len = stream.read(bytes)) >= 0) {  
    System.out.printf("Get %d bytes from %s\n", len, url);  
    outputStream.write(bytes, 0, len);  
}
```