INSTITUTO TECNOLÓGICO DE OAXACA

INGENIERÍA EN SISTEMAS COMPUTACIONALES

TÓPICOS AVANZADOS DE PROGRAMACIÓN

SU

LIC. ADELINA CRUZ NIETO

UNIDAD IV

INVESTIGACIÓN I (MÉTODO JOIN())

ANAYA BAUTISTA GUADALUPE BERENICE

25/07/18

Si un Thread necesita esperar a que otro termine (por ejemplo el Thread padre espera a que termine el hijo) puede usar el método join(). ¿Por que se llama así? Crear un proceso es como una bifurcación, se abren 2 caminos, que uno espere a otro es lo contrario, una unificación.

A continuación se presenta un ejemplo más complejo: una reunión de alumnos. El siguiente ejemplo usa Threads para activar simultáneamente tres objetos de la misma clase, que comparten los recursos del procesador peleándose para escribir a la pantalla.

```
public static void main(String args[]) throws InterruptedException{

        Thread juan = new Thread (new Alumno("Juan"));
        Thread luis = new Thread (new Alumno("Luis"));
        Thread nora = new Thread (new Alumno("Nora"));

        juan.start();
        juan.join();

        pepe.start();
pepe.join();

        jorge.start();
jorge.join();
}
```

El metodo join() que llamamos al final hace que el programa principal espere hasta que este Thread este "muerto"(finalize su ejecucion). Este método puede disparar la excepción InterruptedException, por lo que lo hemos tenido en cuenta en el encabezamiento del método.

En nuestro ejemplo, simplemente a cada instancia de Alumno(…) que creamos la hemos ligado a un Thread y puesto a andar. Corren todas en paralelo hasta que mueren de muerte natural, y también el programa principal termina.

De acuerdo al API de Java:

boolean	**isInterrupted()** Tests whether this thread has been interrupted.
void	**join()** Waits for this thread to die.
void	**join(long millis)** Waits at most millis milliseconds for this thread to die.
void	**join(long millis, int nanos)** Waits at most millis milliseconds plus nanos nanoseconds for this thread to die.
void	**resume()** Deprecated. *This method exists solely for use with **suspend()**, which has been deprecated because it is deadlock-prone. For more information, see **Why are Thread.stop, Thread.suspend and Thread.resume Deprecated?**.*

BIBLIOGRAFIA:

http://labojava.blogspot.com/2012/10/manejo-de-threads-metodos.html

https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.html