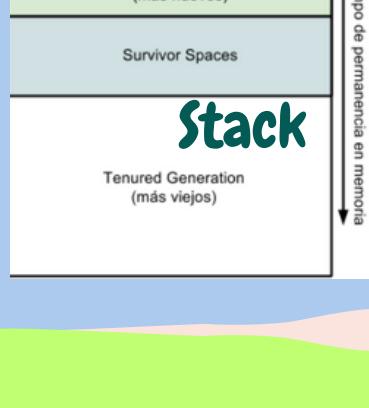


# Java Virtual Machine

## TIPOS DE MEMORIA QUE UTILIZA

### Heap

Representa la cantidad de memoria disponible para todos los hilos de ejecución del proceso. La heap memory es dividida en dos grupos, memoria de nueva generación y memoria de vieja generación (young & old generations). Los nuevos objetos son creados en la young heap memory y son movidos a la old heap memory cuando han sobrevivido al recolector de basura, lo que indicaría que son objetos aún referenciados por otros.



## QUE SE ALMACENA EN CADA MEMORIA

se almacenan las instancias de clases (Objetos) y arreglos. El Garbage Collector (GC) es el encargado de reclamar el espacio que los objetos van liberando.

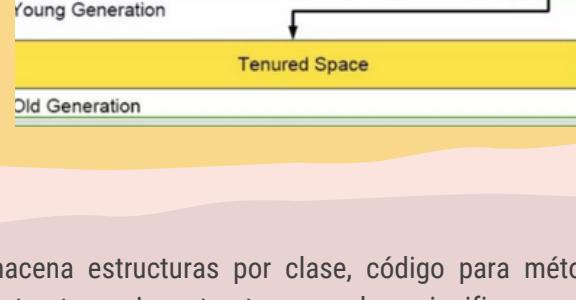
### Young Generation

Eden Space: Esta es el área inicial donde se inicializan la mayoría de los objetos.

Survivor Space: En esta área se almacenan los objetos que han sobrevivido a la recolección de basura en el Eden. En general esta área está dividida en dos partes From y To.

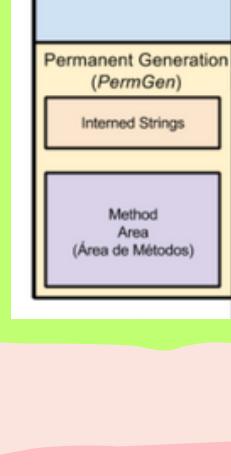
### - Old Generation

Tenured Space: Contiene los objetos que han existido por un tiempo largo y que han pasado por el Survivor space.



### Non-Heap

Esta área incluye los objetos que son considerados parte de la JVM. Al igual que el heap esta puede crecer o ser reducida automáticamente bajo demanda. Se compone de 'Área de método' y otra memoria necesaria para el procesamiento interno. Así que aquí el jugador principal es 'Área de método'.



Almacena estructuras por clase, código para métodos y constructores. La estructura por clase significa constantes de tiempo de ejecución y campos estáticos .

### Permanent Generation

Este espacio contiene todos los datos reflectivos de la JVM como por ejemplo clases y métodos. Además contiene la estructura por clase (Propiedades, Firma de métodos e implementación de métodos). Esta área además contiene dos espacios llamados Shared-RO and Shared-RW

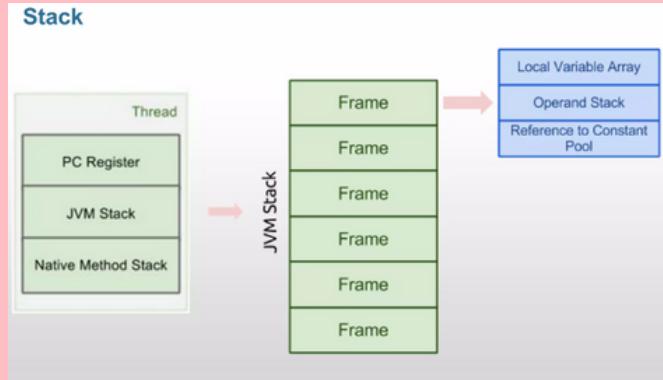
### \* Code Cache

Contiene la memoria usada para el código compilado por el JIT compiler y almacenado de código nativo



### Stack

Las pilas de Java se crean privadas para un subproceso. Cada subproceso tendrá un contador de programa (PC) y una pila de Java.



almacena parámetros, variables locales y direcciones de retorno durante las llamadas a métodos.

Se utilizará la pila de Java para almacenar los valores intermedios, la vinculación dinámica, los valores de retorno para los métodos y las excepciones de envío. Esto se usa en lugar de registros.

