# Programación estructurada

## Ejecución de un programa

Verónica E. Arriola-Rios

Facultad de Ciencias, UNAM

26 de octubre de 2020

# La pila de ejecución

# Temas

# Control de la ejecución

Al ejecutar un programa con varias funciones nos podemos preguntar:

- ¿Cómo registra la computadora qué línea de código debe ejecutar en qué momento?
  Es decir, ¿cómo sabe cuando brincar a una función, cúando y a dónde regresar cuando termina su ejecución?
- ¿Cómo determina qué variables son visibles dónde?
- Si se manda llamar la misma función varias veces ¿cómo sabe cuánto valen los parámetros actuales en cada ocasión?

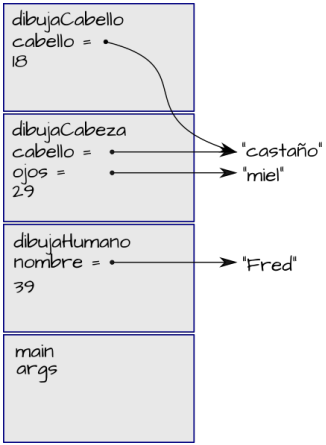
# Programa

```
package funciones;

public class DibujoHumano {

        private static void sop(String s) { System.out.println(s); }

        public static void dibujaCabello(String cabello) {
                sop("␣␣␣␣␣␣[dibujaCabello]␣Cabello␣" + cabello);
        }
        ...
        public static void dibujaHumano(String nombre) {
                sop("␣␣[dibujaHumano]␣Dibujando␣a␣" + nombre);
                dibujaCabeza("castaño", "miel");
                dibujaTorso();
                dibujaCadera();
                dibujaBrazos();
                dibujaPiernas();
        }
        /** Describe los pasos para dibujar un humano. */
        public static void main(String[] args) {
                sop("[main]␣Queremos␣dibujar␣un␣humano...");
                dibujaHumano("Fred");
                sop("[main]␣Humano␣dibujado");
        }
}
```



# Registros de llamadas a métodos

- En los *registros de llamadas a métodos* se guarda la información particular para cada vez que se invoca el código de una función. Esto incluye:
  - Los valores actuales de los parámetros formales.
  - Las variables declaradas y utilizadas localmente.
  - La dirección donde debe guardar el valor de regreso de la función, en caso de que devuelva algo.
  - La dirección de la instrucción de código que se deberá ejecutar cuando haya terminado la ejecución de la función.
- Si el parámetro es de tipo **primitivo** se pasa su valor.
- Si el parámetro es un **objeto** se pasa la dirección del objeto en el *montículo*.

```
1 public static double areaTriangulo(double base, double altura) {
2   double val = base * altura / 2;
3   return val;
4 }
```

# La pila de ejecución

- En la *pila de ejecución* se montan los registros de llamadas a métodos cada vez que un método invoca a otro.
- El registro de la llamada más reciente queda siempre sobre el registro de la llamada al método que invoca.
- El primer registro en ser insertado es el del método **main**, cuando inicia la ejecución del programa.

# Temas

# Variables locales

- Siguiendo el sistema anterior, las variables declaradas dentro de una función sólo son visibles dentro de la función misma.
- Abrir un bloque también crea un ambiente nuevo.
  - Por ejemplo en los de las instrucciones `if`, `for`, `while`, etc.
  - Las variables sólo existen dentro del bloque.
  - Las variables dentro de un bloque pueden ver a las de afuera.

```
1  public int unaFunción(boolean bParam) {
2    int todosMeVen = 0;
3    while(bParam) {
4      int siempreSoyCero = 0;
5      siempreSoyCero++;
6      todosMeVen++;
7      if (todosMeVen > 6) break;
8    }
9    // System.out.println(siempreSoyCero); // No compila
10   System.out.println(todosMeVen++);
11 }
```

# Bibliografía

# Bibliografía I


Mitchell, John C. (2003). *Concepts in Programming Languages*. Cambridge University Press.

# Licencia

Creative Commons
Atribución-No Comercial-Compartir Igual