Concepto de métodos de **static** y métodos de **instancia**. Diferencias. Método main de Java

MÉTODOS DE CLASE O STATIC FRENTE A MÉTODOS DE INSTANCIA.

Un método de instancia es el que se invoca **siempre** sobre una instancia (objeto) de una clase. Por ejemplo p1.getNombre(); siendo p1 un objeto de tipo Persona es un método de instancia: para invocarlo necesitamos una instancia de persona. **Un método de clase es aquel que puede ser invocado sin existir una instancia.**

Un método de clase se define agregando la palabra clave static antes del tipo en la signatura del método. Ejemplos:

```
//Ejemplo aprenderaprogramar.com
        public static String getNombre () { … }
        public static int getNumeroDeDiasDelMes () { …}
```

Los métodos de clase pueden ser invocados con la notación de punto de estas dos maneras:

```
//Ejemplo aprenderaprogramar.com
NombreDeLaClase.nombreDelMétodo (parámetros si los hay);
NombreDelObjeto.nombreDelMétodo (parámetros si los hay);
```

Por ejemplo si la clase Enero tuviera un método estático getNumeroDeDiasDelMes() podríamos escribir: int dias = Enero.getNumeroDeDiasDelMes();. La diferencia con un uso de método "normal" es que aquí no invocamos a un objeto, sino a una clase y por ello decimos que un método estático es un método de clase. Por ejemplo, los métodos de la clase Arrays del API de Java son métodos estáticos: no los invocamos sobre un objeto, sino sobre una clase. Otra clase que se basa en métodos estáticos es Math. Por ejemplo el método pow (double a, double b) es un método que devuelve la potencia a^b. Es un método estático porque no se invoca sobre un objeto. Nosotros podemos crear métodos estáticos en nuestro código. Los interpretaremos como paquetes de código asociados a la clase.

BlueJ es un entorno de desarrollo especial que permite trabajar con objetos y métodos de instancia directamente. En otros entornos esto no es posible y debe indicársele al IDE una clase "de arranque". Una vez se especifica esa clase, Java busca e invoca automáticamente el método main ubicado en dicha clase, y a partir de ese método comienza la creación de objetos y desarrollo de la aplicación. Los métodos de clase (estáticos) tienen estas limitaciones:

a) No pueden acceder a campos de instancia (lógico, pues los campos van asociados a objetos).

b) No pueden invocar a un método de instancia de la misma clase (lógico pues los métodos de instancia van asociados a objetos).

Ahora estamos en condiciones de reflexionar con un poco más de detenimiento sobre el método main. Recordemos su sintaxis: public static void main (String[ ] Args). El método main será siempre un método público, ya que por su papel de iniciador de la aplicación no tendría sentido que fuera privado y por tanto inaccesible desde el exterior. **El método main siempre es un método estático, ya que cuando se invoca no existen objetos** creados con anterioridad porque la ejecución del programa aún no ha comenzado. Si un programa no ha comenzado, existen clases pero no objetos (aunque BlueJ es un entorno educativo especial que sí permite crear objetos e interactuar con ellos antes de ejecutar el método main, esto debemos verlo como algo no habitual en el desarrollo de programas). El tipo de main siempre será void (nulo) ya que no es un tipo función que devuelva un valor: su misión es arrancar la ejecución, no devolver un valor.

En el método main se incluye como parámetro para su invocación un array de Strings. Este array permitiría iniciar el programa con argumentos adicionales: por ejemplo podríamos indicarle una gama de colores de presentación entre varias posibles, o si se trata de un juego, si se ejecuta en modo 1 ó 2 jugadores, etc. En este caso suponiendo que la clase que contiene el main se llama juego, para la ejecución por consola escribiríamos > java juego TwoPlayers Red, donde TwoPlayers y Red son parámetros que condicionan la ejecución del programa. Cada palabra después del nombre de la clase se introduce en un array que se pasa al array Args[ ] que va como parámetro de la clase main. No obstante, es muy frecuente que los programas se inicien sin parámetros, para lo cual simplemente en consola habríamos de escribir java y el nombre de la clase.

En teoría, el cuerpo de un método main puede contener todo lo que se quiera y ser tan largo como se quiera. No obstante, ya hemos indicado que un buen diseño pasa por hacer el método main lo más corto posible y evitar que contenga la lógica del programa. En esencia, debe limitarse a crear objetos e invocar sus métodos. El código que controla la lógica del programa no tiene por qué estar en la clase con el método main. En main nos limitaríamos a crear un objeto que llevara el control del programa.

EJERCICIO

Intenta compilar el siguiente código:

```
public class Test {
   int atributo1;
   Test (int atrib) {atributo1 = 0;}
   public static void main (String[ ] Args) {
      System.out.println ("Mensaje 1");
      System.out.println ("Atributo 1 vale" + this.getAtrib1());
   } //Cierre del main
   public int getAtrib1() {return atributo1;}
} //Cierre de la clase
```

¿Qué mensaje de error se obtiene? ¿Por qué se obtiene este mensaje de error? ¿Cómo podemos solucionarlo para que se ejecute lo que pretendemos?