

# REFERENCIAS

CESSI #ArgentinaPrograma #YoProgramo

---

Leonardo Blautzik - Federico Gasior - Lucas Videla  
Agosto -Diciembre de 2021

## Concepto:

- Una **referencia** es un signo que remite a otro signo.
- Una **referencia** es la dirección de memoria en la que se encuentra almacenado un **objeto**.

## Instancias

- Todos los objetos se manipulan a través de referencias
- Una referencia **apunta** a un objeto que se encuentra en la memoria **HEAP**

## Variables

- Las variables de tipo primitivo (int, double, boolean, char) almacenan el valor propiamente dicho.

Las variables que no son de tipo primitivo almacenan una referencia a una instancia de ese tipo de objeto.

```
Fecha miNacimiento = new Fecha(15,7,1989);
```

(1) Fecha miNacimiento

??????

**Figure 1:** Declaración de la variable de referencia miNacimiento de tipo Fecha

```
Fecha miNacimiento = new Fecha(15,7,1989);
```

(2) new Fecha

dia	0
mes	0
anio	0

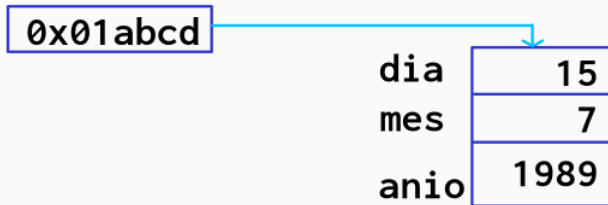
**Figure 2:** new Fecha reserva el espacio de memoria en el Heap para un objeto de tipo Fecha

```
Fecha miNacimiento = new Fecha(15,7,1989);  
    (3) new Fecha(15,7,1989)
```

dia	15
mes	7
anio	1989

**Figure 3:** La ejecución del constructor parametrizado

(4)  
Fecha miNacimiento = new Fecha(15,7,1989);



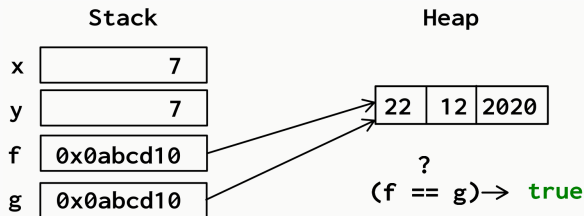
**Figure 4:** Asignación de la variable de referencia



# La Asignación de referencias

Sea el siguiente fragmento de código:

```
int x = 7;  
int y = x;  
Fecha f = new Fecha(22, 12, 2020);  
Fecha g = f;
```

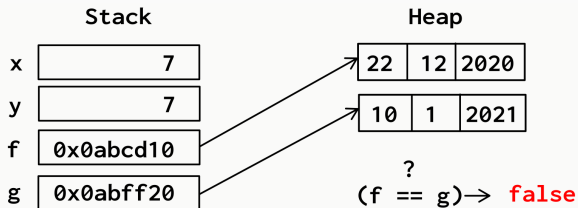


**Figure 5:** Dos variables se refieren a un mismo objeto Fecha

# La Asignación de referencias

Hacemos una reasignación de g a un nuevo objeto Fecha para el 10 de enero de 2021:

```
int x = 7;  
int y = x;  
Fecha f = new Fecha(22,12,2020);  
Fecha g = new Fecha(10,1,2021);
```

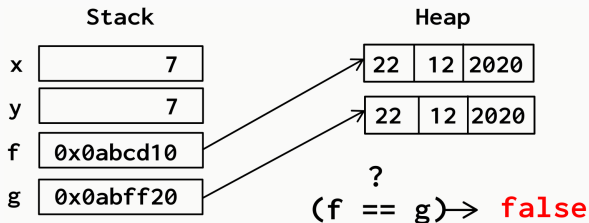


**Figure 6:** Ahora g referencia al nuevo objeto Fecha

# La Asignación de referencias

Ahora asignamos g a un nuevo objeto Fecha pero para el 22 de diciembre de 2020:

```
int x = 7;  
int y = x;  
Fecha f = new Fecha(22,12,2020);  
Fecha g = new Fecha(22,12,2020);
```



**Figure 7:** Ahora g referencia al nuevo objeto Fecha pero con el mismo contenido que f

## El operador ==

**El operador == realiza una comparación de equivalencia.**

Dadas dos referencias **x** e **y**:

**x == y** devuelve **true** si y solo si **x** e **y** refieren al mismo objeto.

# El método equals

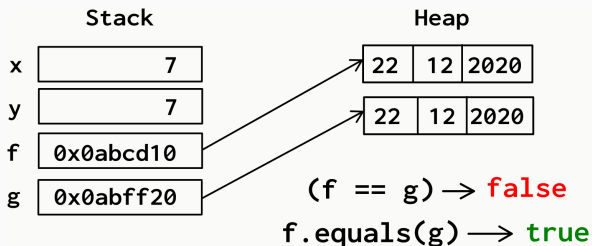
Para poder comparar dos objetos a fin de saber si son iguales, debemos proveer (sobrescribir) el método **equals**.

```
@Override
```

```
public boolean equals(Object obj){  
    if(this == obj) //pregunta si las referencias son iguales  
        return true;  
    if(obj == null) //pregunta si el parámetro es null  
        return false;  
    if(this.getClass() != obj.getClass() //pregunta si los objetos  
        //son de distinta clase  
        return false;  
    Fecha f = (Fecha) obj; // Casteo de obj a tipo Fecha  
    //finalmente compara uno a uno los atributos  
    return (this.dia == f.dia && this.mes == f.mes && this.anio == f.anio)  
}
```

# El método equals

```
int x = 7;  
int y = x;  
Fecha f = new Fecha(22,12,2020);  
Fecha g = new Fecha(22,12,2020);
```



**Figure 8:** g referencia al objeto Fecha pero con el mismo contenido que f. Comparamos con equals.

# Garbage Collector

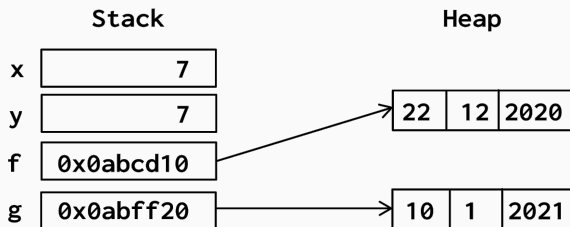
## Objetos que quedan desreferenciados.

```
int x = 7;
```

```
int y = x;
```

```
Fecha f = new Fecha(22,12,2020);
```

```
Fecha g = new Fecha(10,01,2021);
```



**Figure 9:** `f` y `g` referencian a dos objetos `Fecha` diferentes.

# Garbage Collector

## Objetos que quedan desreferenciados.

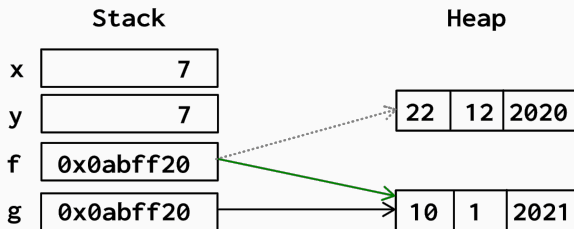
```
int x = 7;
```

```
int y = x;
```

```
Fecha f = new Fecha(22,12,2020);
```

```
Fecha g = new Fecha(10,01,2021);
```

```
f = g;
```



**Figure 10:** En `f` se asigna el valor de `g`, ahora `f` y `g` referencian al mismo objeto.



# Garbage Collector

## Objetos que quedan desreferenciados.

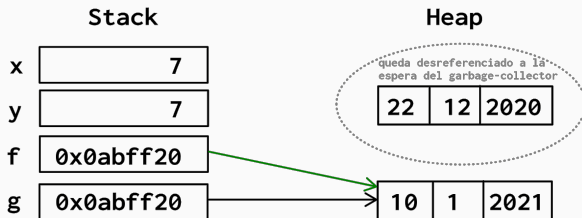
```
int x = 7;
```

```
int y = x;
```

```
Fecha f = new Fecha(22,12,2020);
```

```
Fecha g = new Fecha(10,01,2021);
```

```
f = g;
```



**Figure 11:** La Fecha [22/12/2020] queda desreferenciada e inaccesible.

### **La palabra reservada `this` se usa para:**

- La resolución de ambigüedades entre parámetros y atributos.
- El pasaje del objeto actual como parámetro a otro método.
- La invocación explícita de métodos de la propia clase.
- La invocación del constructor de la clase actual.

## Veamos el uso de `this` en la implementación de la class `Circulo`

```
class Circulo {  
  
    private double radio;  
  
    public Circulo(double radio) {  
        this.setRadio(radio);  
    }  
  
    public Circulo() {  
        this(1.0);  
    }  
}
```

## La interfaz pública - Los Setters

```
public void setRadio(double radio) {  
    if (radio <= 0)  
        throw new Error("Radio Inválido");  
    this.radio = radio;  
}
```

```
public void setDiametro(double diametro) {  
    setRadio(diametro/2);  
}
```

## La interfaz pública - Los Getters

```
public double getRadio() {  
    return this.radio;  
}  
  
public double getDiametro(){  
    return this.getRadio() * 2;  
}  
  
public double getPerimetro() {  
    return this.getDiametro() * Math.PI;  
}  
  
public double getArea() {  
    return Math.PI * Math.pow(this.getRadio(),2);  
}  
}
```