APRENDERAPROGRAMAR.COM

EJEMPLOS CONVERSIÓN DE TIPOS EN JAVA. CASTING, CLASSCASTEXCEPTIONS. INSTANCEOF. EJEMPLOS. (CU00689B)

Sección: Cursos
Categoría: Curso “Aprender programación Java desde cero”

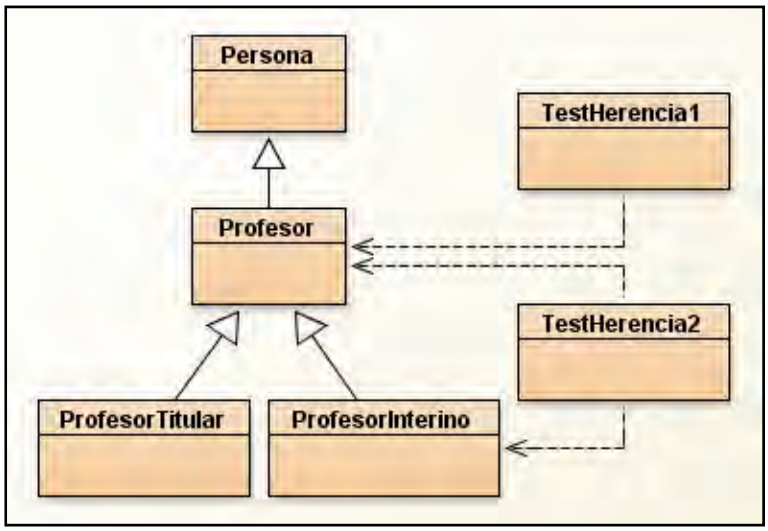
Fecha revisión: 2029

Resumen: Entrega nº89 curso Aprender programación Java desde cero.

Autor: Alex Rodríguez

CONVERSIÓN DE TIPOS. CASTING. CLASSCASTEXCEPTIONS.

Java admite la conversión de tipos con ciertas limitaciones. Consideremos una jerarquía de herencia como la que vemos en el siguiente diagrama de clases, sobre el que vamos a analizar las posibilidades de conversión de tipos de distintas formas.



1. **Conversión hacia arriba**

Se trataría por ejemplo de poner lo que está a un nivel inferior en un nivel superior, por ejemplo poner un profesor interino como profesor. Posible código: *profesor43 = profesorinterino67;*

Ponemos lo que está abajo (el profesor interino) arriba (como profesor). Esto es posible, pero dado que lo que está abajo generalmente contiene más campos y métodos que lo que está arriba, perderemos parte de la información. Sobre el objeto profesor43 ya no podremos invocar los métodos propios de los profesores interinos.

2. **Conversión hacia abajo**

Se trataría de poner lo que está arriba abajo, por ejemplo poner un Profesor como ProfesorInterino. Esto no siempre es posible. El supertipo admite cualquier forma (es polimórfico) de los subtipos: si el supertipo almacena el subtipo al que queremos realizar la conversión, será posible usando lo que se

denomina “Enmascaramiento de tipos” o “Hacer Casting” (cast significa “moldear”). Si el supertipo no almacena el subtipo al que queremos convertirlo, la operación no es posible y saltará un error. Ejemplo:

Profesor p1; //p1 es tipo Profesor. Admite ser Profesor, ProfesorTitular o ProfesorInterino.

ProfesorInterino p44 = new ProfesorInterino(); //p44 es ProfesorInterino.

p1 = p44; // Conversión hacia arriba: sin problema. Ahora p1 que es tipo profesor, almacena un profesor interino

p44 = p1 // ERROR en la conversión hacia abajo. El compilador no llega tan lejos como para saber si p1 almacena un profesor interino u otro tipo de profesor y ante la incertidumbre salta un error. La forma de forzar al compilador a que “comprenda” que p1 está almacenando un profesor interino y por tanto puede asignarse a una variable que apunta a un profesor interino se llama “hacer casting”. Escribiríamos lo siguiente: *p44 = (ProfesorInterino) p1;*

El nombre de un tipo entre paréntesis se llama “operador de enmascaramiento de tipos” y a la operación en sí la llamamos enmascaramiento o hacer casting. El casting es posible si el objeto padre contiene a un objeto hijo, **pero si no es así aunque la compilación sea correcta en tiempo de ejecución saltará un error “ClassCastException”**, o error al hacer cast con las clases. No siempre es fácil determinar a qué tipo de objeto apunta una variable. Por ello el casting o enmascaramiento debiera evitarse en la medida de lo posible ya que el que un programa compile pero contenga potenciales errores en tiempo de ejecución es algo no deseable. Un programa bien estructurado normalmente no requerirá hacer casting reiteradamente, o lo hará de forma muy controlada. Si durante la escritura de un programa nos vemos en la necesidad de realizar casting, debemos plantearnos la posibilidad de reestructurar código para evitarlo.

Cuando incluyamos conversiones de tipos usando casting en nuestro código, para evitar errores conviene filtrar las operaciones de enmascaramiento asegurándonos antes de hacerlas de que el objeto padre contiene un hijo del subtipo al que queremos hacer la conversión. Esta verificación se hace usando la palabra clave *instanceof* como explicaremos a continuación.

3. Conversión de lado a lado

Se trataría de poner lo que está a un lado al otro lado, por ejemplo convertir un ProfesorInterino en ProfesorTitular o al revés. Esto no es posible en ningún caso.

DETERMINACIÓN DEL TIPO DE VARIABLES CON INSTANCE OF

La palabra clave instanceof, todo en minúsculas, sirve para verificar el tipo de una variable. La sintaxis que emplearemos para instanceof y sus normas de uso serán las siguientes:

```
if (profesor43 instanceof ProfesorInterino) {…} else { …}
```

a) Sólo se pueden comparar instancias que relacionen dentro de la jerarquía de tipos (en cualquier dirección) pero no objetos que no relacionen en una jerarquía. Es decir, no podemos comparar profesores con taxis por ejemplo, porque no relacionarán dentro de una jerarquía.

b) Solo se puede usar instanceof asociado a un condicional. No se puede usar por ejemplo directamente en una impresión por pantalla con System.out.println(…).

Ejemplos de sintaxis y ejemplo de código. Escribe y compila el código y comprueba el resultado:

```
if (profesor73 instanceof ProfesorInterino) --> la sintaxis es válida.
if (interino1 instanceof ProfesorInterino) --> la sintaxis es válida.
if (interino1 instanceof Profesor) --> la sintaxis es válida.
if (fecha1 instanceof Profesor) --> Error: las instancias no están en una jerarquía de tipos
```

```
import java.util.Calendar;
//Test conversión de tipos. Ejemplo de código aprenderaprogramar.com
public class TestHerencia3 {
   public static void main (String [ ] Args) {
      Profesor profesor1 = new Profesor ("Juan", "Hernández García", 33);
      Calendar fecha1 = Calendar.getInstance();
      fecha1.set(2019,10,22); //Los meses van de 0 a 11, luego 10 representa noviembre
      ProfesorInterino interino1 = new ProfesorInterino("José Luis", "Morales Pérez", 54, fecha1);

      Profesor profesor73 = interino1; //Ahora el supertipo contiene un subtipo, en principio con pérdida de información
      if (profesor73 instanceof ProfesorInterino) {
         System.out.println ("***profesor73 es un objeto de tipo ProfesorInterino" );}
      if (profesor73 instanceof Profesor) { System.out.println ("profesor73 es un objeto de tipo Profesor ¡ES POLIMÓRFICO!" ); }

      if (interino1 instanceof Profesor) { System.out.println ("interino1 es un objeto de tipo Profesor ¡ES POLIMÓRFICO TAMBIÉN!" );
      } else { System.out.println ("interino1 no apunta a un objeto de tipo Profesor" ); }

      if (profesor1 instanceof ProfesorInterino) {
         System.out.println ("profesor1 es un objeto de tipo ProfesorInterino" );
      } else { System.out.println ("profesor1 no es un objeto de tipo ProfesorInterino. Nunca ha sido un interino." ); }
   } //Cierre del main
} //Cierre de la clase ejemplo aprenderaprogramar.com
```

```
***profesor73 es un objeto de tipo ProfesorInterino
profesor73 es un objeto de tipo Profesor ¡ES POLIMÓRFICO!
interino1 es un objeto de tipo Profesor ¡ES POLIMÓRFICO TAMBIÉN!
profesor1 no es un objeto de tipo ProfesorInterino. Nunca ha sido un interino.
```

Próxima entrega: CU00690B

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:

http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=68&Itemid=188