OPTIMIZACIÓN Y DOCUMENTACIÓN

© ***Luis Pilo Aceituno*** ***lpilo@educarex.es***

1. REFACTORIZACIÓN

Definimos la refactorización como la transformación del software preservando su comportamiento, modificando su estructura interna para mejorarlo.

En ocasiones durante el transcurso de un proyecto nos vemos obligados a reevaluar y modificar el código creado anteriormente. La refactorización (o limpieza del código) ayuda a tener un código más sencillo de comprender, más compacto y más limpio y por supuesto más fácil de modificar.

1.1.Tabulación

Con la tabulación el código es más fácil de leer y resulta más claro.

1.2.Patrones de refactorización

Son métodos de refactorización.Plantean casos concretos y su resolución. Algunos métodos de factorización son.

1. ***Extraer método***

Si tenemos un fragmento de de código que puede agruparse lo convertimos en un método cuyo nombre explique el propósito del método.

Actividad 1. Muestra un ejemplo de esta técnica de refactorización.

- ***Primero debes mostrar el código como un bloque y después el método fruto de la refactorización.***

2. ***Eliminar asignaciones a parámetros.***

Un parámetro es usado para recibir una asignación. Usamos una variable en su lugar.

Ejemplo: El método descuento tiene el parámetro entradaValor al que se le asigna un valor. Se crea una variable con ese parámetro y la asignación se le hace a la variable.

```
Int descuento (int entradaValor, int cantidad, int año)
{
    if (entradaValor > 50) entradaValor =-2;
    /….
 {
```

Refactorizacion

Entradavalor se convertirá en variable.

```
int descuento (int entradaValor, int cantidad, int año)
{
   //convertir el parámetro en variable.
   int resultado  = entradaValor;

  if( entradaValor <50) resultado = -2;
   / …
{
```

Actividad 2. Muestra un ejemplo de esta técnica de refactorización.

- ***Primero debes mostrar el código como un bloque y después el método fruto de la refactorización.***

3. Consolidar Fragmentos Duplicados en Condicionales

El mismo fragmento de código está en todas las ramas de una expresión condicional. Sacamos dicho fragmento fuera de la expresión.

Ejemplo.

```
if (esAcuerdoEspecial () )
{
   total = precio * 0.9;
   enviar();
}else{
  total = precio * 0.8;
   enviar();
}
```

Refactorizacion

```
if (esAcuerdoEspecial () )
{
   total = precio * 0.9;
  }else{
  total = precio * 0.8;
}
 enviar();
```

Actividad 3. Muestra un ejemplo de esta técnica de refactorización.

- ***Primero debes mostrar el código como un bloque y después el método fruto de la refactorización.***

4. ***Consolidar Expresiones Condicionales***

Tenemos una secuencia de condicionales con el mismo resultado. Los combinamos en una sola expresión y lo extraemos.

Ejemplo

```
double cuantíaPorDiscapacidad()
{
  if (antiguedad < 2) return 0;
  if (mesesDiscapacitados > 12)  return 0;
 if (esTiempoParcial )  return 0;
  // ….
}
```

Refactorizamos

```
double cuantiaPorDiscapacidad()
{
   if (esNoElegibleParaDiscapacidad () )  return 0;
 // …
}
```

Actividad 4. Muestra un ejemplo de esta técnica de refactorización.

- ***Primero debes mostrar el código como un bloque y después el método fruto de la refactorización.***

Actividad 5. Busca información sobre métodos de refactorizacion de software y explica cada uno de ellos.

- ***Poner ejemplos de cada uno de ellos. No repitas los ejemplos expuestos aquí.***
- ***Uno de los patrones o catalogos de refactorización más populares es el de Martín Fowler.***

2. REFACTORIZACIÓN Y VISUAL STUDIO

La gran mayoría de los entornos de desarrollo traen una refactorizaciones básicas incluidas y unas automáticas fáciles de realizar.

2.1. IntelliSense

Es una aplicación de Microsoft integrada en el entorno de desarrollo Visual Studio destinada a autocompletar. Nada más empezar a escribir código detectara que es lo que queremos escribir, mostrando sugerencias alfabéticamente, que además tienen un icono identificador que nos indica si es una palabra reservada, una variable, un método o una clase.

También nos completa las propiedades y métodos inherentes a cualquier clase. funciona mostrando todas las propiedades o métodos accesibles mediante un “.”

También permite automatizar la estructura del as funciones

3. DOCUMENTACIÓN.

Es importante la documentación no solo para el usuario final sino para los desarrolladores.

El primer paso para una buena documentación es el uso de comentarios. Los comentarios en los IDEs suelen activarse con una combinación de teclas

***Actividad 6.*. Busca información sobre métodos de refactorizacion y documentación presentes en tu IDE preferido (eclipse Visual Sudio, etc). Prepara un trabajo en el que además de las explicaciones haya capturas de pantallas . Se valorarán positivamente los siguientes aspectos.**

- ***Demostraciones y ejemplos prácticos***
- ***Comparación entre varios IDEs***