**C# vs Java**

**1.- SISTEMA DE TIPOS**

**Tipos**

En C# existen los tipos básicos, objetos, estructuras (inmutables), y a partir de la versión 2.0, los ***Nullable***

que permiten tener valores nulos o desconocidos (frecuente en entornos de bases de datos); sin embargo en

Java solo existen los tipos básicos y objetos.

**Declaración de arrays**

Mientras que en Java se puede declarar los ***arrays*** de dos formas, una con la notación de C/C++, y otra con

la notación comúnmente aceptada, C# usa solo este último tipo de declaración.

**String**

Tanto en Java como en C# son inmutables, pero en este último no es necesario llamar a un metodo ***equals***

para compararlas pudiendo usar ***==*** y ***!=*** sin que compare las referencias.

**Declaración de constantes**

En Java se utiliza ***final*** para declarar constantes y el valor puede ser declarado en tiempo de ejecución o

compilación mientras que en C# ***const*** se usa para asignar valores constantes en tiempo de compilación y

***readonly*** para declaraciones en tiempo de ejecución.

**2.- INSTRUCCIONES DE CONTROL**

**Bucle for-each**

En C# se usa la palabra reservada ***foreach*** e ***in*** para iterar sobre los elementos de una colección de objetos

sin necesidad de crear un índice de forma explícita. No existía ninguna instrucción similar en Java hasta la

versión 1.5, donde se incluye una mejora del ***for*** con una funcionalidad muy similar.

**Switch**

En Java sólo se admiten enteros, caracteres y booleanos como condiciones de cada ***case***; en C#, pueden ser

además cadenas y enumerados. Además, a diferencia de Java, cada ***case*** ha de terminar con un ***break***, y, si

queremos saltar de un ***case*** a otro, podemos usar ***goto***.

**3.- CLASES DE OBJETOS**

**Propiedades**

Una propiedad es un campo especial que permite al programador controlar el acceso al mismo tanto en

lectura como en escritura. C# implementa las propiedades como parte de la sintaxis del lenguaje, a diferencia de Java que para conseguir la misma funcionalidad se recurre a patrones o convenciones de nombres.

**Parámetros**

En C# existen los parámetros ***out*** y ***ref*** que permiten modificar los valores de los argumentos proporcionados en una llamada a un método. También se puede hacer uso del parámetro ***params***, que recibe un número de argumentos variables.

**Objetos**

En C# los tipos pueden pertenecer a alguna de las siguientes categorías: estructuras, clases de objetos (ambas con ancestro común a la clase ***object***), estructuras enumeradas y tipos delegados. En Java, por otro lado, hay una clase padre llamada ***Object*** de la que heredan el resto, pero también existen los tipos básicos, que no son objetos y por tanto no cumplen esta propiedad.

**Indizadores**

En C# se pueden crear indizadores, los cuales nos permiten construir fácilmente clases a cuyos objetos se

puede acceder mediante un índice; en Java esto no es posible.

**Eventos y delegados**

Un evento es cualquier acontecimiento en la vida de un objeto que puede ser de interés para otros y un delegado es el tipo de un evento. No hay un mecanismo general para manejar eventos en Java, y en su lugar existen unos patrones de diseño que son usados por ciertas clases y de los cuales el programador se puede beneficiar. Para que un objeto se pueda suscribir a un evento debe implementar cierta interfaz y acabar con la palabra ***Listener***(p.e. ***MouseListener***, ***KeyListener***...). El objeto notificador normalmente tiene un método que empieza por ***add*** o ***remove*** y termina con ***Listener***(p.e. ***addMouseListener***, ***addKeyListener***...) y que son usados para registrarse o darse de baja para un evento. En C# se usa ***event*** para especificar automáticamente que un campo dentro de un suscriptor es un delegado que será usado durante un evento. Los operadores ***+=*** y ***-=*** son los análogos a los usados en Java para suscribirse o desuscribirse.

**Destructores**

En C# los destructores son el método que se ejecuta justo antes de destruir un objeto. Sin embargo en Java no existe este concepto, siendo lo más parecido el método ***finalize()*** que es llamado cuando se ejecuta el recolector de basura, y usado generalmente para liberar memoria y otros recursos usados por los objetos. En C# tenemos using { }.

**4.- POLIMORFISMO Y EXTENSIÓN DE CLASES**

**Sintáxis de la herencia**

En C# se usa ***":"*** tanto para herencia en clases como para implementar interfaces, a diferencia de Java que

usa las palabras reservadas ***extend*** se ***implements***.

**Herencia múltiple**

En Java existe la herencia múltiple mientras que en C# no.

**Polimorfismo**

En C# un método no se declara como ***virtualy*** es necesario poner ***override*** para sobrescribir un método abstracto.

**Modificadores de interfaz**

En Java existe ***public*** y sin modificador (por defecto) mientras que en C# se introducen ***new***, ***protected***,

***Internal*** y ***private***.

**4.- BIBLIOTECA**

**Clases y archivos**

En Java un archivo sólo contiene a una clase, mientras que en C# un archivo puede contener varias clases.

**Espacios de nombre**

Un espacio de nombre es una manera de agrupar una colección de clases. En Java, un espacio de nombres determina la disposición física de los ficheros fuente en la jerarquía de directorios, mientras en C# solo representa su estructura lógica.