C#

TÓPICOS ESPECIAIS

Value Types

Value Types:

Uma variável deste tipo contém o valor, e não um endereço de referência para o valor;

Derivam de System.ValueTypes;

Variáveis de escopo local precisam ser inicializadas antes de serem utilizadas;

Atribuir o valor de variável a outra, implicitamente, é feita uma cópia do conteúdo da variável. Sendo assim, qualquer alteração no conteúdo de uma delas, não afetará a outra. Quanto maior for um objeto deste tipo mais custosa será sua cópia.

Reference Types

Reference Types:

Uma variável contém a referência (ou endereço) para o objeto que está na Heap;

Atribuir o valor de uma variável para outra faz uma cópia da referência, e não do próprio objeto. Ou seja, não é feita a cópia do objeto, e sim do endereço de memória do objeto, o que não gera muito custo para objetos grandes;

São alocados na Heap e seus objetos são coletados pelo Garbage Collector;

São passados por referência, enquanto que Value Types são passados por valor. Ou seja, a alteração de um objeto afetará todas as instâncias que apontam para ele.

Boxing e Unboxing

Boxing é o processo de converter um *“Value Type”* para o tipo de object ou a qualquer tipo de interface implementada por este tipo de valor *“Reference Type”*.

Unboxing extrai o *“Value Type”* do objeto. Transforma um objeto em um tipo simples.

Expressões Lambda

Entendemos que a expressão lambda é uma espécie de função, porém sem nome, elas realizam cálculos, filtros, e retornam um valor (ou uma coleção de valores).

Para criar uma expressão lambda é necessário fazer uso do operador lambda “=>” que podemos entender como “Vai para”.

Uma expressão lambda sempre consiste de duas partes (esquerda e direita) separadas pelo “=>”. A parte à esquerda do “=>” contém uma lista de argumentos (de tipo não necessariamente definido, pois os tipos podem ser automaticamente indicados pelo compilador). O lado direito contém as instruções.