Conversion de types en C#

Chaque type a une valeur associée, composée d'octets stockés en mémoire. Les valeurs sont lues à partir d'emplacements de mémoire qui sont également typés. Le type de l'emplacement détermine le type de la valeur. De nombreuses valeurs peuvent être exprimées sous forme de plusieurs types. Par exemple, la valeur 4 peut être exprimée sous forme d'un entier ou d'une valeur en virgule flottante. La conversion crée une valeur dans un nouveau type qui est équivalente à la valeur d'un ancien type, mais ne préserve pas nécessairement la valeur exacte des deux objets.

I. Les conversions implicites

Le tableau ci-dessous répertorie les conversions numériques implicites prédéfinies.

De	En
sbyte	short, int, long, float, double ou decimal
byte	short, ushort, int, uint, long, ulong, float, double ou decimal
short	int, long, float, double ou decimal
ushort	int, uint, long, ulong, float, double ou decimal
int	long, float, double ou decimal
uint	long, ulong, float, double ou decimal
long	float, double ou decimal
char	ushort, int, uint, long, ulong, float, double ou decimal
float	double
ulong	float, double ou decimal

II. Les conversions explicites

Le langage C# offre la possibilité de convertir le type d'une expression de manière explicite.

Une expression cast est utilisée pour convertir explicitement une expression en un type donné.

(type) expression

Une conversion numérique explicite permet de convertir, à l'aide d'une expression de conversion, n'importe quel type numérique en un autre type numérique lorsqu'il n'existe pas de conversion implicite. Le tableau ci-dessous répertorie ces conversions explicites.

De	En
sbyte	byte, ushort, uint, ulong ou char
byte	sbyte ou char
short	sbyte, byte, ushort, uint, ulong ou char