La classe java.math.BigDecimal

Temps de lecture : 2 minutes

La classe `BigDecimal` permet de réaliser des calculs en virgule flottante avec une précision dans les résultats similaire à celle de l'arithmétique scolaire.

Elle permet ainsi une représentation exacte des valeurs ce que ne peuvent garantir les données primitives de type numérique flottant (float ou double). Les calculs en virgule flottante privilégient en effet la vitesse de calcul plutôt que la précision.

Utilisation de `BigDecimal`

```
BigDecimal valeur1 = new BigDecimal("10");
BigDecimal valeur2 = new BigDecimal("0.09");

// Affiche 0.90
System.out.println(valeur1.multiply(valeur2));
```

Le calcul ci-dessus affiche comme résultat `0.90`. On peut voir que la précision a été respectée et c'est un avantage par rapport à l'utilisation des types primitifs :

```
double valeur1=10.0;
double valeur2=0.09;

//Affiche 0.8999999999999999
System.out.println(valeur1 * valeur2);
```

On voit bien ici que contrairement aux types primitifs flottants, les résultats des opérations de BigDecimal garantit les résultats, à la précision voulue.

Attention aux constructeurs

Il est très préférable d'utiliser le constructeur qui accepte une `String` plutôt qu'un `double`, même si les deux compilent :

```
BigDecimal valeur1 = new BigDecimal(2.8);
BigDecimal valeur2 = new BigDecimal("2.8");

System.out.println("valeur1="+valeur1);
System.out.println("valeur2="+valeur2);

// Affiche valeur1=2.799999999999999822364316059974953532218933105468
75
// valeur2=2.8
```

On voit qu'avec le constructeur recevant un `double`, la précision est perdue.

Personnaliser l'arrondis

Pour modifier la précision d'un `BigDecimal`, il est possible de le faire ainsi :

```
BigDecimal bigDec = new BigDecimal("2.8823931").setScale
(4, RoundingMode.DOWN);
System.out.println(bigDec); // Affiche 2.8823
```

Ici, on voit bien que l'on demande une précision de 4 chiffres après la virgule, et que le mode d'arrondis se fait vers le bas. Voici les différentes valeurs d'arrondis que l'on peut utiliser avec l'énumération `RoundingMode` :

- `ROUND_CEILING`
- `ROUND_DOWN`
- `ROUND_FLOOR`
- `ROUND_HALF_UP`
- `ROUND_HALF_DOWN`
- `ROUND_HALF_EVEN`
- `ROUND_UP`

La précision et le mode d'arrondi doivent être choisis avec attention parce que leur choix peut avoir de grandes conséquences sur les résultats de calculs notamment si le résultat final est constitué de multiples opérations. Dans ce cas, il est préférable de garder la plus grande précision durant les calculs et de n'effectuer l'arrondi qu'à la fin.

Méthodes de calculs

Tout comme `BigInteger`, de nombreuses méthodes de calculs applicables aux objets sont déclarées dans `BigDecimal` :

```
var x = new BigDecimal("10.90");
var y = new BigDecimal("20.90");
x.add(y);
x.subtract(y);
x.multiply(y);
x.divide(y);
x.max(y);
x.abs();
//...
```

Note : Il n'y a pas de méthodes de calculs trigonométriques dans la classe `BigDecimal`.