Java Type Casting

Before you learn about **Java Type Casting**, make sure you know about Java Data Types.

Type Casting

The process of converting the value of one data type (`int`, `float`, `double`, etc.) to another data type is known as typecasting.
In Java, there are 13 types of type conversion. However, in this tutorial, we will only focus on the major 2 types.

1. Widening Type Casting

2. Narrowing Type Casting

To learn about other types of type conversion, visit Java Type Conversion (official Java documentation).

Widening Type Casting

In **Widening Type Casting**, Java automatically converts one data type to another data type.

Example: Converting int to double

```
class Main {
  public static void main(String[] args) {
    // create int type variable
    int num = 10;
    System.out.println("The integer value: " + num);
```

```
    // convert into double type
    double data = num;
    System.out.println("The double value: " + data);
  }
}
```

Run Code

Output

```
The integer value: 10
The double value: 10.0
```

In the above example, we are assigning the `int` type variable named `num` to a `double` type variable named `data`.

Here, the Java first converts the `int` type data into the `double` type. And then assign it to the `double` variable.

In the case of **Widening Type Casting**, the lower data type (having smaller size) is converted into the higher data type (having larger size). Hence there is no loss in data. This is why this type of conversion happens automatically.

> **Note**: This is also known as **Implicit Type Casting**.

Narrowing Type Casting

In **Narrowing Type Casting**, we manually convert one data type into another using the parenthesis.

Example: Converting double into an int

```
class Main {
  public static void main(String[] args) {
    // create double type variable
    double num = 10.99;
    System.out.println("The double value: " + num);

    // convert into int type
    int data = (int)num;
```

```
    System.out.println("The integer value: " + data);
  }
}
```

Run Code

Output

```
The double value: 10.99
The integer value: 10
```

In the above example, we are assigning the `double` type variable named `num` to an `int` type variable named `data`.

Notice the line,

```
int data = (int)num;
```

Here, the `int` keyword inside the parenthesis indicates that that the `num` variable is converted into the `int` type.

In the case of **Narrowing Type Casting**, the higher data types (having larger size) are converted into lower data types (having smaller size). Hence there is the loss of data. This is why this type of conversion does not happen automatically.

> **Note**: This is also known as **Explicit Type Casting**.

Let's see some of the examples of other type conversions in Java.

Example 1: Type conversion from int to String

```
class Main {
  public static void main(String[] args) {
    // create int type variable
    int num = 10;
    System.out.println("The integer value is: " + num);

    // converts int to string type
```

```
    String data = String.valueOf(num);
    System.out.println("The string value is: " + data);
  }
}
```

Run Code

Output

```
The integer value is: 10
The string value is: 10
```

In the above program, notice the line

```
String data = String.valueOf(num);
```

Here, we have used the `valueOf()` method of the Java String class to convert the int type variable into a string.

Example 2: Type conversion from String to int

```
class Main {
  public static void main(String[] args) {
    // create string type variable
    String data = "10";
    System.out.println("The string value is: " + data);

    // convert string variable to int
    int num = Integer.parseInt(data);
    System.out.println("The integer value is: " + num);
  }
}
```

Run Code

Output

```
The string value is: 10
The integer value is: 10
```

In the above example, notice the line

```
int num = Integer.parseInt(data);
```

Here, we have used the `parseInt()` method of the Java `Integer` class to convert a string type variable into an `int` variable.