Python Type Conversion

In programming, type conversion is the process of converting data of one type to another. For example: converting `int`Â data to `str`.

There are two types of type conversion in Python.

- Implicit Conversion - automatic type conversion
- Explicit Conversion - manual type conversion

Python Implicit Type Conversion

In certain situations, Python automatically converts one data type to another. This is known as implicit type conversion.

Example 1: Converting integer to float

Let's see an example where Python promotes the conversion of the lower data type (integer) to the higher data type (float) to avoid data loss.

```
integer_number = 123
float_number = 1.23

new_number = integer_number + float_number

# display new value and resulting data type
print("Value:",new_number)
print("Data Type:",type(new_number))
```

Output

```
Value: 124.23
Data Type: <class 'float'>
```

In the above example, we have created two variables: *integer_number* and *float_number* of `int`Â and `float` type respectively.

Then we added these two variables and stored the result in *new_number*.

As we can see *new_number* has value **124.23** and is of the `float` data type.

It is because Python always converts smaller data types to larger data types to avoid the loss of data.

Note:

- We get `TypeError`, if we try to add `str`Â and `int`. For example, `'12' + 23`. Python is not able to use Implicit Conversion in such conditions.
- Python has a solution for these types of situations which is known as Explicit Conversion.

Explicit Type Conversion

In Explicit Type Conversion, users convert the data type of an object to required data type.

We use the built-in functions like int(), float(), str(), etc to perform explicit type conversion.

This type of conversion is also called typecasting because the user casts (changes) the data type of the objects.

Example 2: Addition of string and integer Using Explicit Conversion

```
num_string = '12'
num_integer = 23

print("Data type of num_string before Type Casting:",type(num_string))

# explicit type conversion
num_string = int(num_string)

print("Data type of num_string after Type Casting:",type(num_string))

num_sum = num_integer + num_string

print("Sum:",num_sum)
print("Data type of num_sum:",type(num_sum))
```

Output

```
Data type of num_string before Type Casting: <class 'str'>
Data type of num_string after Type Casting: <class 'int'>
Sum: 35
Data type of num_sum: <class 'int'>
```

In the above example, we have created two variables: *num_string* and *num_integer* with `str`Â and `int`Â type values respectively. Notice the code,

```
num_string = int(num_string)
```

Here, we have used `int()` to perform explicit type conversion of *num_string* to integer type.

After converting *num_string* to an integer value, Python is able to add these two variables.

Finally, we got the *num_sum* value i.e **35** and data type to be `int`.

Key Points to Remember

1. Type Conversion is the conversion of an object from one data type to another data type.
2. Implicit Type Conversion is automatically performed by the Python interpreter.
3. Python avoids the loss of data in Implicit Type Conversion.
4. Explicit Type Conversion is also called Type Casting, the data types of objects are converted using predefined functions by the user.
5. In Type Casting, loss of data may occur as we enforce the object to a specific data type.

Also Read:

- Python Numbers, Type Conversion and Mathematics

Table of Contents