Type promotion in Java, also known as type conversion or type casting, refers to the automatic conversion of a smaller data type to a larger data type during operations. This is done to prevent data loss and ensure that operations are performed correctly. Java handles type promotion automatically in certain situations, such as arithmetic operations and method calls.

Key Concepts of Type Promotion
Widening Conversion: This is the automatic conversion of a smaller primitive type to a larger primitive type. It is safe because there is no risk of data loss. For example, converting an int to a long.

Narrowing Conversion: This involves converting a larger primitive type to a smaller primitive type. It is not done automatically because it can lead to data loss. Explicit casting is required.

Promotion in Expressions: When performing arithmetic operations, Java promotes smaller types to larger types to ensure the operation is performed correctly.

Type Promotion Rules
byte, short, and char are promoted to int when used in expressions.
If one of the operands is of type long, the whole expression is promoted to long.
If one of the operands is of type float, the whole expression is promoted to float.
If one of the operands is of type double, the whole expression is promoted to double.
Example of Type Promotion
Java

```
 Collapse
1public class TypePromotionExample {
2    public static void main(String[] args) {
3        byte b = 10;
4        short s = 20;
5        char c = 'A'; // ASCII value 65
6        int i = 30;
7        long l = 40L;
8        float f = 50.0f;
9        double d = 60.0;
10
11        // byte, short, and char are promoted to int
12        int result1 = b + s + c;
13        System.out.println("Result1 (int): " + result1); // Output: 95
14
15        // int is promoted to long
16        long result2 = i + l;
17        System.out.println("Result2 (long): " + result2); // Output: 70
18
19        // long is promoted to float
20        float result3 = l + f;
21        System.out.println("Result3 (float): " + result3); // Output: 90.0
22
23        // float is promoted to double
24        double result4 = f + d;
25        System.out.println("Result4 (double): " + result4); // Output: 110.0
26    }
27}
```

Key Points
Automatic Promotion: Java automatically promotes smaller data types to larger data types in expressions to prevent data loss and ensure precision.
Method Overloading: Type promotion can affect method overloading. If a method is overloaded with different parameter types, the method with the closest matching type is called.
Explicit Casting: When narrowing conversion is needed (e.g., from double to int), explicit casting is required to avoid data loss.
Method Overloading Example
Java

```
1public class OverloadingExample {
2    public static void main(String[] args) {
3        OverloadingExample example = new OverloadingExample();
4        example.print(5); // Calls the method with int parameter
5        example.print(5.0); // Calls the method with double parameter
```

```
6     }
7
8     void print(int num) {
9         System.out.println("Integer: " + num);
10     }
11
12     void print(double num) {
13         System.out.println("Double: " + num);
14     }
15}
```

In this example, the method print is overloaded with int and double parameters. When 5 is passed, it calls the int version, and when 5.0 is passed, it calls the double version.

Understanding type promotion is essential for writing efficient and error-free Java code, especially when dealing with arithmetic operations and method overloading.