Working with Formatting Numbers

Jaya Bodkhey
Information Security & Automation Engineer

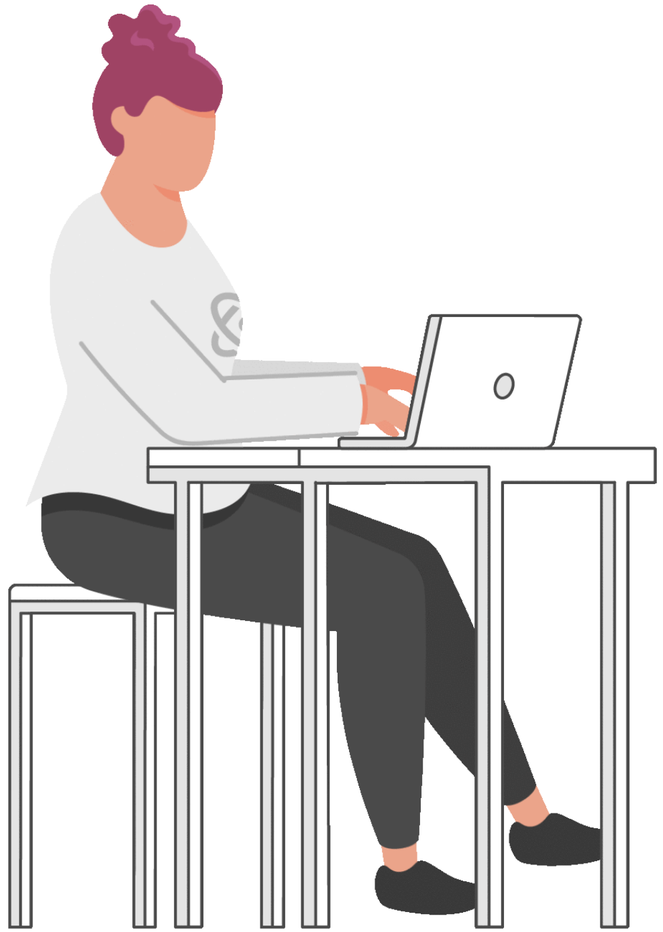
@jayabodkhey

Module Overview

Why to format numbers?

Java methods for formatting numbers

Java classes used for formatting numbers



Requirement to support different number formatting

Display numbers formatted as per locale preference

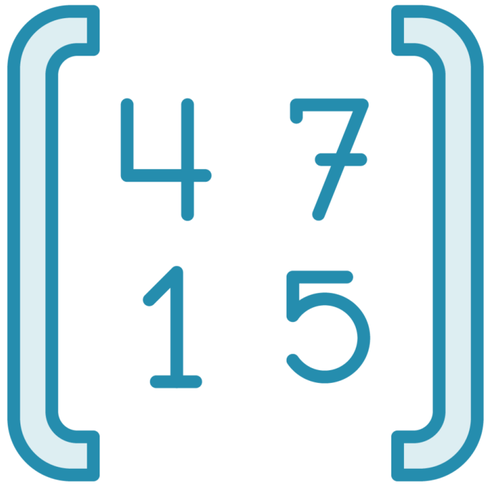
Why to Format Numbers?

Different locale specific standards to represent a number

Two thousand three hundred and sixty point eight five

- **2,360.85**
- **2’360.85**
- **2’360,85**
- **2360.85 (ISO)**

Formatting Numbers in Java



java.math.BigDecimal, java.lang.Integer, java.lang.Long

Expression language functions

Java classes for formatting numbers

Formatting number depends on

- **Way to display a number**
- **Maximum number of fraction digits**

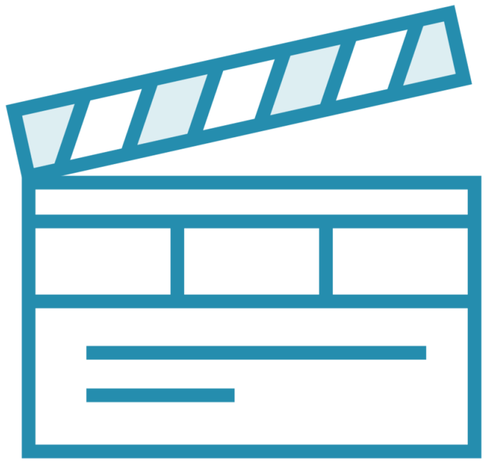
Expression Language Functions

Formatting Numbers

```
fnd:formatNumber(java.lang.Number decimalValueToBeFormatted)
```

```
fnd:formatNumber2(java.lang.Number decimalValueToBeFormatted, int
maximumNumberOfFractionDigits)
```

How Numbers Are Formatted at Run Time?



Application Core

Binding to the numberFormatPattern property in the applCorePrefs

Execution of bindings

Displaying numbers as per user preferences

What all Java classes can help format numbers in my application?

NumberFormat
DecimalFormat

NumberFormat Class

- Abstract class for all number formats
- Interface for formatting and parsing numbers
- Locale specific number formatting
- Locale independent code

NumberFormat Class Methods

format()	getCurrencyInstance()	getPercentInstance()
getInstance()	setMinimumFractionDigits()	setMaximumFractionDigits()

Formatting Numbers Using NumberFormat Class

```
public class FormatNumberDemo {
    public static void main(String args[]) {
        double num = 32360.856974;
        NumberFormat nf = NumberFormat.getInstance(new Locale("en", "US"));
        String value = nf.format(num);
        System.out.println("Number formatted for the locale - " + value);
    }
}
```

Number formatted for the locale - 32,360.857

Formatting Numbers Using NumberFormat Class

```
public class FormatNumberDemo {
    public static void main(String args[]) {
        double num = 32360.856974;
        NumberFormat nf = NumberFormat.getInstance(new Locale(“fr", “FR"));
        String value = nf.format(num);
        System.out.println("Number formatted for the locale - " + value);
    }
}
```

Number formatted for the locale -
32 360,857

Formatting Numbers Using NumberFormat Class

```
public class FormatNumberDemo {
    public static void main(String args[]) {
        double num = 32360.856974;
        NumberFormat nf = NumberFormat.getInstance(new Locale("en", "US"));
        nf.setMaximumFractionDigits(2);
        String value = nf.format(num);
        System.out.println("Number formatted for the locale - " + value);
    }
}
```

Number formatted for the locale - 32,360.86

DecimalFormat Class

- Subclass of NumberFormat class
- Formats decimal numbers
- Locale specific parsing and formatting
- Supports Western, Arabic and Indic digits

DecimalFormat Class Methods

format()

applyPattern()

setDecimalSeparatorAlwaysShown()

toLocalizedPattern()

setMinimumFractionDigits()

setMaximumFractionDigits()

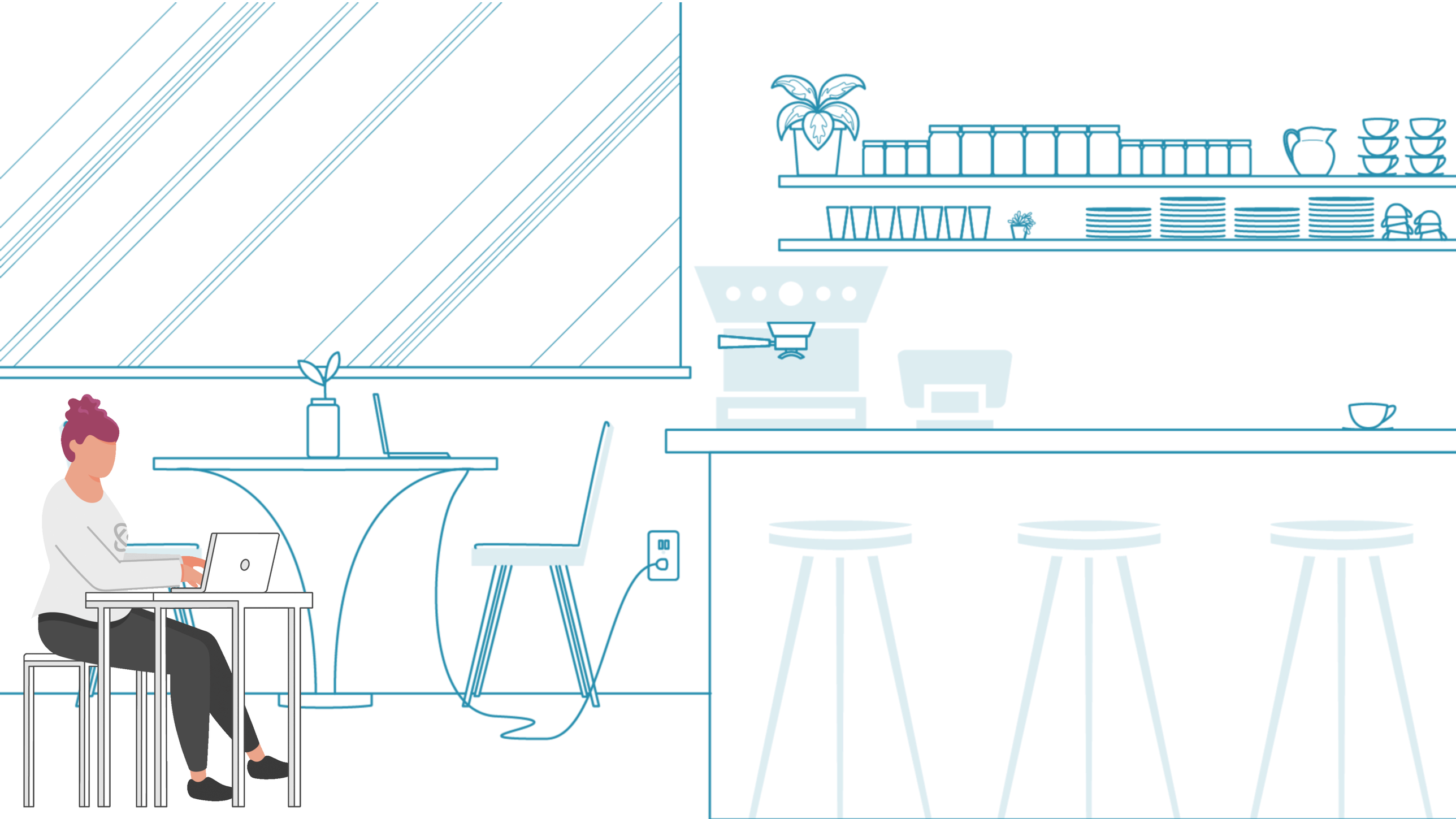
Formatting Numbers Using NumberFormat Class

```
public class FormatNumberDemo {
    public static void main(String args[]) {
        double num = 32360.856974;
        DecimalFormat df = new DecimalFormat("$#0.00");
        String valueDf = df.format(num);
        System.out.println("Number formatted using DecimalFormat - " + valueDf);
    }
}
```

Number formatted for the locale - 32,360.86

Demo

Using NumberFormat class and DecimalFormat class to format numbers



Module Summary

- Motivation behind formatting numbers
- Formatting using expression language functions
- NumberFormat class
- DecimalFormat class
- Formatting numbers using NumberFormat class and DecimalFormat class

Up Next:
Course Summary