

# Localization

## Formatting Values

# Formatting Numbers

- to format numbers we have to use `NumberFormat` interface
- we have already learned about to methods from this interface

```
public final String format(double number)
```

```
public final String format(long number)
```

- now we will introduce the concrete class `DecimalFormat`, which includes the constructor that takes a `String` pattern:

```
public DecimalFormat(String pattern)
```

- to create a pattern, we need to know to formatting characters:

# - omit position if no digit exists for it (e.g. \$2.2)

0 - put 0 in position if no digit exists for it (e.g. \$002.20)

```
import java.text.*;

public class MyClass {
    public static void main(String args[]) {
        double num = 12345.6789;

        NumberFormat f1 = new DecimalFormat("###,###,###.0");
        System.out.println(f1.format(num));

        NumberFormat f2 = new DecimalFormat("000,000.000000");
        System.out.println(f2.format(num));

        NumberFormat f3 = new DecimalFormat("My Balance: $#,###,###.##");
        System.out.println(f3.format(num));
    }
}
```

```
12,345.7
012,345.678900
My Balance: $12,345.68
```

# Formatting Dates and Times

- to display standard formats Java provides a class called `DateTimeFormatter`
- you can use predefined formats (e.g. `ISO_LOCAL_DATE`)
- to create your own `String` format use `ofPattern` method
  - used with common date/time symbols (given in the next table)

# Common date/time symbols

Symbol	Meaning	Example
y	Year	23, 2023
M	Month	2, 02, Feb, February
d	Day	7, 07
h	Hour	8, 08
m	Minute	25
s	Second	17
a	a.m. / p.m.	AM, PM
Z	Time zone name	Central European Time, CET
Z	Time zone offset	-600

```
LocalDate date = LocalDate.of(2023, Month.SEPTEMBER, 14);  
System.out.println(date.getDayOfWeek());  
=> THURSDAY  
System.out.println(date.getMonth());  
=> SEPTEMBER  
System.out.println(date.getYear());  
=> 2023  
System.out.println(date.getDayOfYear());  
=> 257
```

```
// to display standard formats
LocalDate date = LocalDate.of(2023, Month.SEPTEMBER, 14);
LocalTime time = LocalTime.of(9, 6, 24);
LocalDateTime dt = LocalDateTime.of(date, time);

System.out.println(date.format(DateTimeFormatter.ISO_LOCAL_DATE));
    => 2023-09-14
System.out.println(time.format(DateTimeFormatter.ISO_LOCAL_TIME));
    => 09:06:24
System.out.println(dt.format(DateTimeFormatter.ISO_LOCAL_DATE_TIME));
    => 2023-09-14T09:06:24
```

```
// creating custom formats
```

```
var dt = LocalDateTime.of(2022, Month.SEPTEMBER, 14, 19, 6, 14);
```

```
var f1 = DateTimeFormatter.ofPattern("dd.MM.yyyy. hh:mm:ss");
```

```
System.out.println(dt.format(f1));
```

```
var f2 = DateTimeFormatter.ofPattern("MMM-dd-yy HH:mm:ss");
```


```
System.out.println(dt.format(f2));
```

```
var f3 = DateTimeFormatter.ofPattern("MMMM-dd-yy hh:mm:ss a");
```

```
System.out.println(dt.format(f3));
```

```
// alternative
```

```
System.out.println(f3.format(dt));
```



```
14.09.2022. 07:06:14  
Sep-14-22 19:06:14  
September-14-22 07:06:14 PM  
September-14-22 07:06:14 PM
```



```
// to insert text values use single quotes  
var f4 = DateTimeFormatter.ofPattern("'Date:' dd.MM.yy. '| Time:' hh:mm:ss a");  
System.out.println(dt.format(f4));  
=> Date: 14.09.22. | Time: 19:06:14
```

```
var f5 = DateTimeFormatter.ofPattern("MMM-dd-yyyy 'at' HH'h'm'm'ss's");  
System.out.println(dt.format(f5));  
=> Sep-14-2022 at 19h6m14s
```

```
// NOTE: spaces can be added within or out of the single quotes
```