FormattedOutput

January 2, 2025

[1]:

```
// Basic Formatting for a String
val name: String = "Navadeep"
val formattedName = String.format("Formatted Name: %s", name)
println(s"Using String Interpolator: $name") // Interpolator
printf("Using printf: %s\n", name) // printf
println(formattedName) // String.format
println("Using concatenation: " + name) // Concatenation
```

```
Using String Interpolator: Navadeep
Using printf: Navadeep
Formatted Name: Navadeep
Using concatenation: Navadeep
```

```
name = Navadeep
formattedName = Formatted Name: Navadeep
```

[1]:

```
Formatted Name: Navadeep
```

[2]:

```
// Basic Formatting for an Integer
val age: Int = 28
println(f"Age with Interpolator: $age%d") // Interpolator
printf("Age using printf: %d\n", age) // printf
printf("Age with leading zeros: %03d\n", age) // Leading zeros
```

```
Age with Interpolator: 28
Age using printf: 28
Age with leading zeros: 028
```

```
age = 28
```

[2]:

```
28
```

[4]:

```
// Formatting a Large Integer
val population = 7654321
println(f"Population with commas (Interpolator): $population%,d")
printf("Population with commas (printf): %,d\n", population)
println(f"Population in scientific format: ${population.toDouble}%.2e")
```

```
Population with commas (Interpolator): 7,654,321
Population with commas (printf): 7,654,321
Population in scientific format: 7.65e+06

lastException = null
population = 7654321
```

[4]: 7654321

[5]:

```
// Formatting a Double Variable
val price: Double = 456.789
println(f"Formatted price (2 decimals): $price%.2f") // Interpolator
printf("Price with printf: %.2f\n", price) // printf
printf("Price in scientific notation: %.3e\n", price) // Scientific notation
println(f"Price with leading zeros: $price%010.2f")
```

```
Formatted price (2 decimals): 456.79
Price with printf: 456.79
Price in scientific notation: 4.568e+02
Price with leading zeros: 0000456.79

price = 456.789
```

[5]: 456.789

[7]:

```
// Formatting Boolean Values
val isAvailable: Boolean = false
printf("Boolean value (printf): %b\n", isAvailable)
printf("Boolean value uppercase: %B\n", isAvailable)
println(f"Formatted Boolean using Interpolator: $isAvailable")
val boolAsText = if (isAvailable) "Available" else "Unavailable"
println(s"Boolean as Text: $boolAsText")
val boolFormatted = String.format("Boolean using format: %b", java.lang.Boolean.
  ↪valueOf(isAvailable))
println(boolFormatted)
```

```
Boolean value (printf): false
Boolean value uppercase: FALSE
Formatted Boolean using Interpolator: false
Boolean as Text: Unavailable
Boolean using format: false

isAvailable = false
boolAsText = Unavailable
boolFormatted = Boolean using format: false
```

[7]: Boolean using format: false

[8]:

```
// Additional Formatting Operations
val temperature: Double = 98.6
println(f"Temperature rounded to 1 decimal: $temperature%.1f")
println(f"Temperature with padding: $temperature%08.2f")
val percentage: Double = 85.3467
println(f"Percentage formatted: $percentage%.1f%%")
```

```
Temperature rounded to 1 decimal: 98.6
Temperature with padding: 00098.60
Percentage formatted: 85.3%
```

```
temperature = 98.6
percentage = 85.3467
```

[8]:

```
85.3467
```

[9]:

```
// Combining Multiple Values
val city = "Hyderabad"
val year = 2025
println(f"City: $city%s, Year: $year%d")
```

```
City: Hyderabad, Year: 2025
```

```
city = Hyderabad
year = 2025
```

[9]:

```
2025
```

[ ]:

```
```