# Applying Techniques to Combine and Format Strings

**Steve Gordon**

.NET Engineer and Microsoft MVP

@stevejgordon    www.stevejgordon.co.uk

## Overview

**Apply operators to concatenate strings**

**Format types such as numbers and dates**

**Apply composite formatting**

- Using the format component

**Using static methods to concatenate and join strings**

**Apply string interpolation**

- C# 10 interpolated string handlers

## Demo

Use operators to concatenate strings at runtime

## Requirements

- Produce a report summarising processed sales data.
- Show who exported the data.
- Include details for each sold product:
    - Date of sale
    - Product name
    - Product SKU

## Warning!

This approach for building strings, particularly inside loops, is NOT recommended for large strings.

Concatenation using operators is a compiler feature.

## Demo

**Produce strings representing data types**

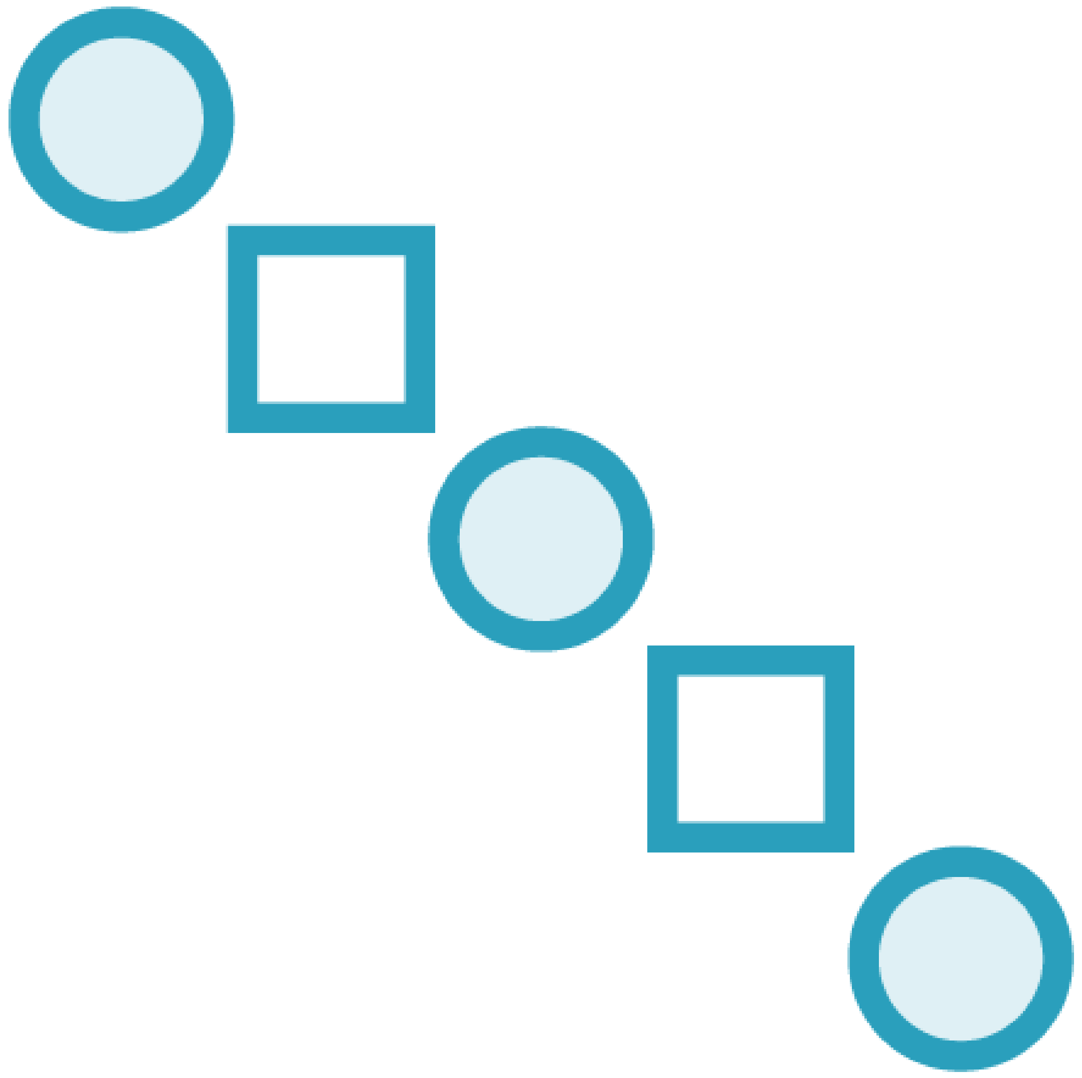
**Format DateTime(Offset) values into their string representation**

- Provide format specifiers to control the formatting

## Requirements

- Include each item's sales date as a row of data in the report.

## Formatting



Process of converting a value to a string

The object base class includes a virtual ToString method

By default, ToString returns the Type name

We may override ToString on our own types

Primitive types override ToString to display their value

Culture plays a large part in formatting

.NET provides mechanisms to control the format and culture during string formatting

```
"date.ToString("d")"
```

$$\texttt{"date.ToString}\underbrace{\texttt{("d")}}_{\text{Format Specifier}}\texttt{"}$$

```
"date.ToString("d")"
```

28 August 2022

Some format specifiers are case-sensitive.

# Standard Date and Time Format Strings

| Format specifier | Pattern | Example (en-GB) |
|---|---|---|
| “d” | Short date | 11/07/2022 |
| “D” | Long date | 11 July 2022 |
| “f” | Full date/time (short) | 11 July 2022 07:22 |
| “F” | Full date/time (long) | 11 July 2022 07:22:00 |
| “g” | General date/time pattern (short) | 11/07/2022 07:22 |
| “G” | General date/time pattern (long) | 11/07/2022 07:22:00 |
| “M” or “m” | Month/day | 11 July |
| “O” or “o” | Round trip date/time | 2022-07-11T07:22:00.0000000+01:00 |
| “R” or “r” | RFC1123 | Mon, 11 Jul 2022 08:22:00 GMT |
| “s” | Sortable date/time | 2022-07-11T08:22:00 |
| “t” | Short time | 08:22 |
| “T” | Long time | 08:22:00 |
| “u” | Universal sortable date/time | 2022-07-11 08:22:00Z |
| “U” | Universal full date/time | 11 July 2022 08:22:00 |
| “Y” or “y” | Year month | July 2022 |

*Source: https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings*

# Standard Date and Time Format Strings

| Format specifier | Pattern | Example (en-GB) |
|---|---|---|
| “d” | Short date | 11/07/2022 |
| “D” | Long date | 11 July 2022 |
| “f” | Full date/time (short) | 11 July 2022 07:22 |
| “F” | Full date/time (long) | 11 July 2022 07:22:00 |
| “g” | General date/time pattern (short) | 11/07/2022 07:22 |
| “G” | General date/time pattern (long) | 11/07/2022 07:22:00 |
| “M” or “m” | Month/day | 11 July |
| “O” or “o” | Round trip date/time | 2022-07-11T07:22:00.0000000+01:00 |
| “R” or “r” | RFC1123 | Mon, 11 Jul 2022 08:22:00 GMT |
| “s” | Sortable date/time | 2022-07-11T08:22:00 |
| “t” | Short time | 08:22 |
| “T” | Long time | 08:22:00 |
| “u” | Universal sortable date/time | 2022-07-11 08:22:00Z |
| “U” | Universal full date/time | 11 July 2022 08:22:00 |
| “Y” or “y” | Year month | July 2022 |

*Source: https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings*

# Standard Date and Time Format Strings

| Format specifier | Pattern | Example (en-GB) |
|---|---|---|
| “d” | Short date | 11/07/2022 |
| “D” | Long date | 11 July 2022 |
| “f” | Full date/time (short) | 11 July 2022 07:22 |
| “F” | Full date/time (long) | 11 July 2022 07:22:00 |
| “g” | General date/time pattern (short) | 11/07/2022 07:22 |
| “G” | General date/time pattern (long) | 11/07/2022 07:22:00 |
| “M” or “m” | Month/day | 11 July |
| “O” or “o” | Round trip date/time | 2022-07-11T07:22:00.0000000+01:00 |
| “R” or “r” | RFC1123 | Mon, 11 Jul 2022 08:22:00 GMT |
| “s” | Sortable date/time | 2022-07-11T08:22:00 |
| “t” | Short time | 08:22 |
| “T” | Long time | 08:22:00 |
| “u” | Universal sortable date/time | 2022-07-11 08:22:00Z |
| “U” | Universal full date/time | 11 July 2022 08:22:00 |
| “Y” or “y” | Year month | July 2022 |

*Source: https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings*

## Standard Date and Time Format Strings

| Format specifier | Pattern | Example (en-GB) |
|---|---|---|
| “d” | Short date | 11/07/2022 |
| “D” | Long date | 11 July 2022 |
| “f” | Full date/time (short) | 11 July 2022 07:22 |
| “F” | Full date/time (long) | 11 July 2022 07:22:00 |
| “g” | General date/time pattern (short) | 11/07/2022 07:22 |
| “G” | General date/time pattern (long) | 11/07/2022 07:22:00 |
| “M” or “m” | Month/day | 11 July |
| “O” or “o” | Round trip date/time | 2022-07-11T07:22:00.0000000+01:00 |
| “R” or “r” | RFC1123 | Mon, 11 Jul 2022 08:22:00 GMT |
| “s” | Sortable date/time | 2022-07-11T08:22:00 |
| “t” | Short time | 08:22 |
| “T” | Long time | 08:22:00 |
| “u” | Universal sortable date/time | 2022-07-11 08:22:00Z |
| “U” | Universal full date/time | 11 July 2022 08:22:00 |
| “Y” or “y” | Year month | July 2022 |

*Source: https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings*

# Standard Date and Time Format Strings

| Format specifier | Pattern | Example (en-GB) |
|---|---|---|
| “d” | Short date | 11/07/2022 |
| “D” | Long date | 11 July 2022 |
| “f” | Full date/time (short) | 11 July 2022 07:22 |
| “F” | Full date/time (long) | 11 July 2022 07:22:00 |
| “g” | General date/time pattern (short) | 11/07/2022 07:22 |
| “G” | General date/time pattern (long) | 11/07/2022 07:22:00 |
| “M” or “m” | Month/day | 11 July |
| “O” or “o” | Round trip date/time | 2022-07-11T07:22:00.0000000+01:00 |
| “R” or “r” | RFC1123 | Mon, 11 Jul 2022 08:22:00 GMT |
| “s” | Sortable date/time | 2022-07-11T08:22:00 |
| “t” | Short time | 08:22 |
| “T” | Long time | 08:22:00 |
| “u” | Universal sortable date/time | 2022-07-11 08:22:00Z |
| “U” | Universal full date/time | 11 July 2022 08:22:00 |
| “Y” or “y” | Year month | July 2022 |

*Source: https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings*

# Standard Date and Time Format Strings

| Format specifier | Pattern | Example (en-GB) |
|---|---|---|
| “d” | Short date | 11/07/2022 |
| “D” | Long date | 11 July 2022 |
| “f” | Full date/time (short) | 11 July 2022 07:22 |
| “F” | Full date/time (long) | 11 July 2022 07:22:00 |
| “g” | General date/time pattern (short) | 11/07/2022 07:22 |
| “G” | General date/time pattern (long) | 11/07/2022 07:22:00 |
| “M” or “m” | Month/day | 11 July |
| “O” or “o” | Round trip date/time | 2022-07-11T07:22:00.0000000+01:00 |
| “R” or “r” | RFC1123 | Mon, 11 Jul 2022 08:22:00 GMT |
| “s” | Sortable date/time | 2022-07-11T08:22:00 |
| “t” | Short time | 08:22 |
| “T” | Long time | 08:22:00 |
| “u” | Universal sortable date/time | 2022-07-11 08:22:00Z |
| “U” | Universal full date/time | 11 July 2022 08:22:00 |
| “Y” or “y” | Year month | July 2022 |

*Source: https://docs.microsoft.com/en-us/dotnet/standard/base-types/standard-date-and-time-format-strings*

```
"MMMM dd, yyyy"
```

August 28, 2022

Demo

**Manipulate strings using composite formatting**

- Apply composite formatting inside a report writer

## Composite Formatting

```
string.Format(“It’s {0}°C on {1}”, 15.4, DateTime.Now);
```

## Composite Formatting

```
string.Format(“It’s {0}°C on {1}”, 15.4, DateTime.Now);
```

Composite
format string

# Composite Formatting

$$\texttt{string.Format(}\underbrace{\texttt{“It’s \{0\}}^{\circ}\texttt{C on \{1\}”}}_{\text{Composite format string}}\texttt{, }\underbrace{\texttt{15.4, DateTime.Now}}_{\text{1 or more objects}}\texttt{);}$$

## Composite Formatting

```
string.Format(“It’s {0}°C on {1}”, 15.4, DateTime.Now);
```

Format item 0

Format item 1

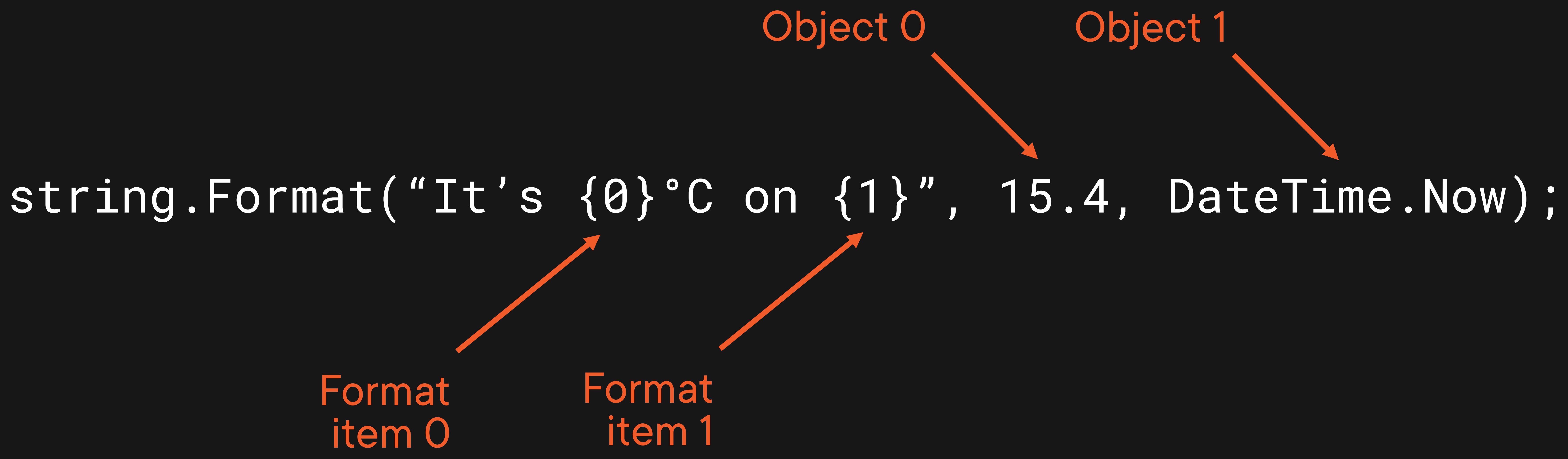
## Composite Formatting

Object 0 Object 1

```
string.Format(“It’s {0}°C on {1}”, 15.4, DateTime.Now);
```

Format item 0 Format item 1

# Composite Formatting



```
string.Format(“It’s {0}°C on {1}”, 15.4, DateTime.Now);
```

It’s 15.4 ° C on 5/1/2021 6:30:00 PM

Using string.Format will generally cause fewer allocations than concatenating using operators.

## Demo

**Control formatting by providing a format string component**

- Format a date using the long date format

## Format Item Syntax

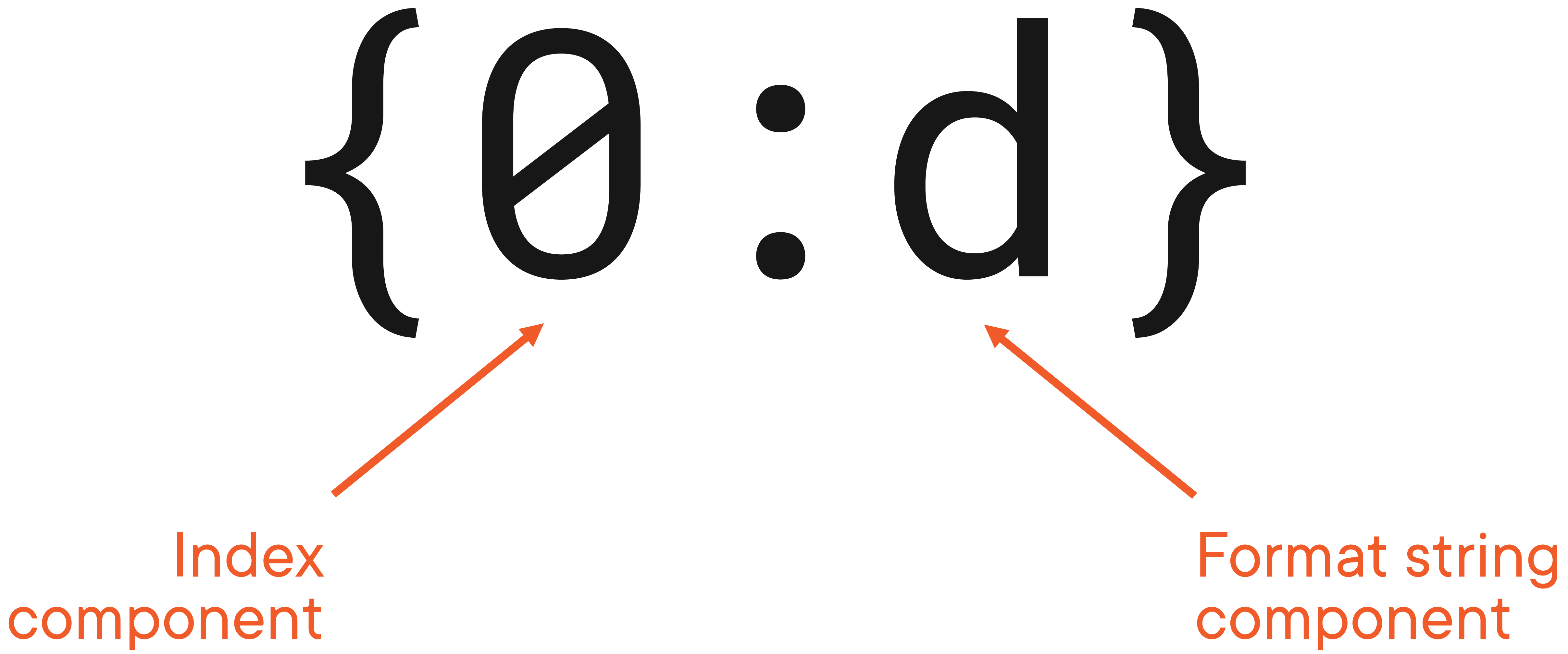
```
{0:d}
```

## Format Item Syntax

```
{0:d}
```

Index component

# Format Item Syntax



## Format String Component

When the format string component is not explicitly provided, the general format applies for the provided object.

Demo

**Concatenate string data using static methods**

- Use string.Join to create a comma separated sequence
- Use string.Concat to append data to an existing string

## Requirements

- Produce a CSV file of valid and normalized sales data.
- Each row should contain a sales data record with comma-separated values.

Demo

Apply interpolated strings

Learn about interpolated string handlers

## String Interpolation

Introduced in C# 6

Provides a more readable and convenient syntax to create formatted strings

Identified by the dollar character

A string literal with zero or more interpolation expressions

Expressions are replaced with their result

```
$"Today is
{DateTime.UtcNow}"
```

```
Today is
28/08/2022
09:30:00
```

```
$"Today is
{DateTime.UtcNow:d}"
```

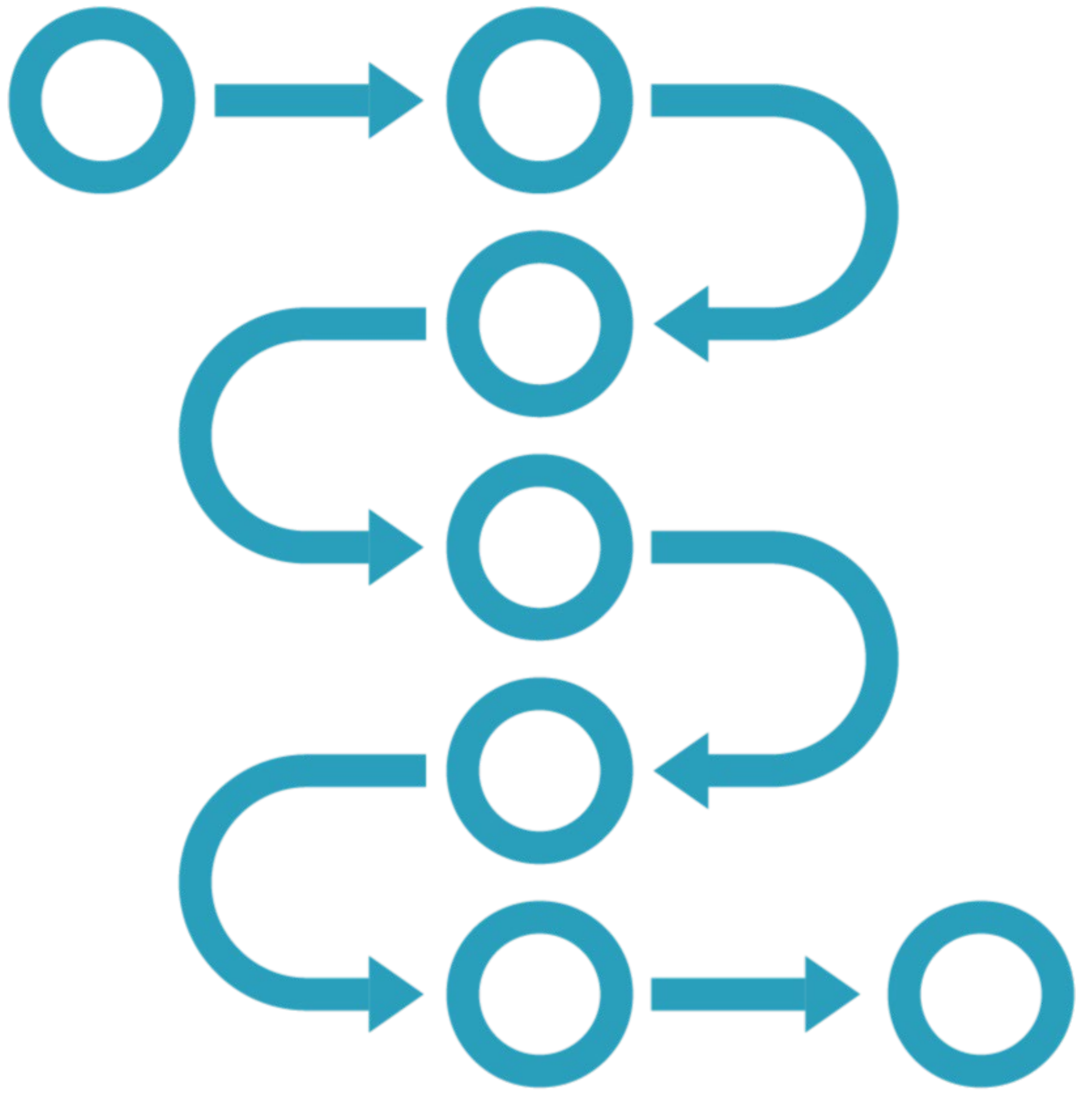
Today is
28/08/2022

We no longer need to provide object arguments, avoiding mistakes and making it easier to read the code.

## What about culture?

By default, the interpolated string will use the application's current culture.

## Interpolated Strings



**String interpolation is a language feature**

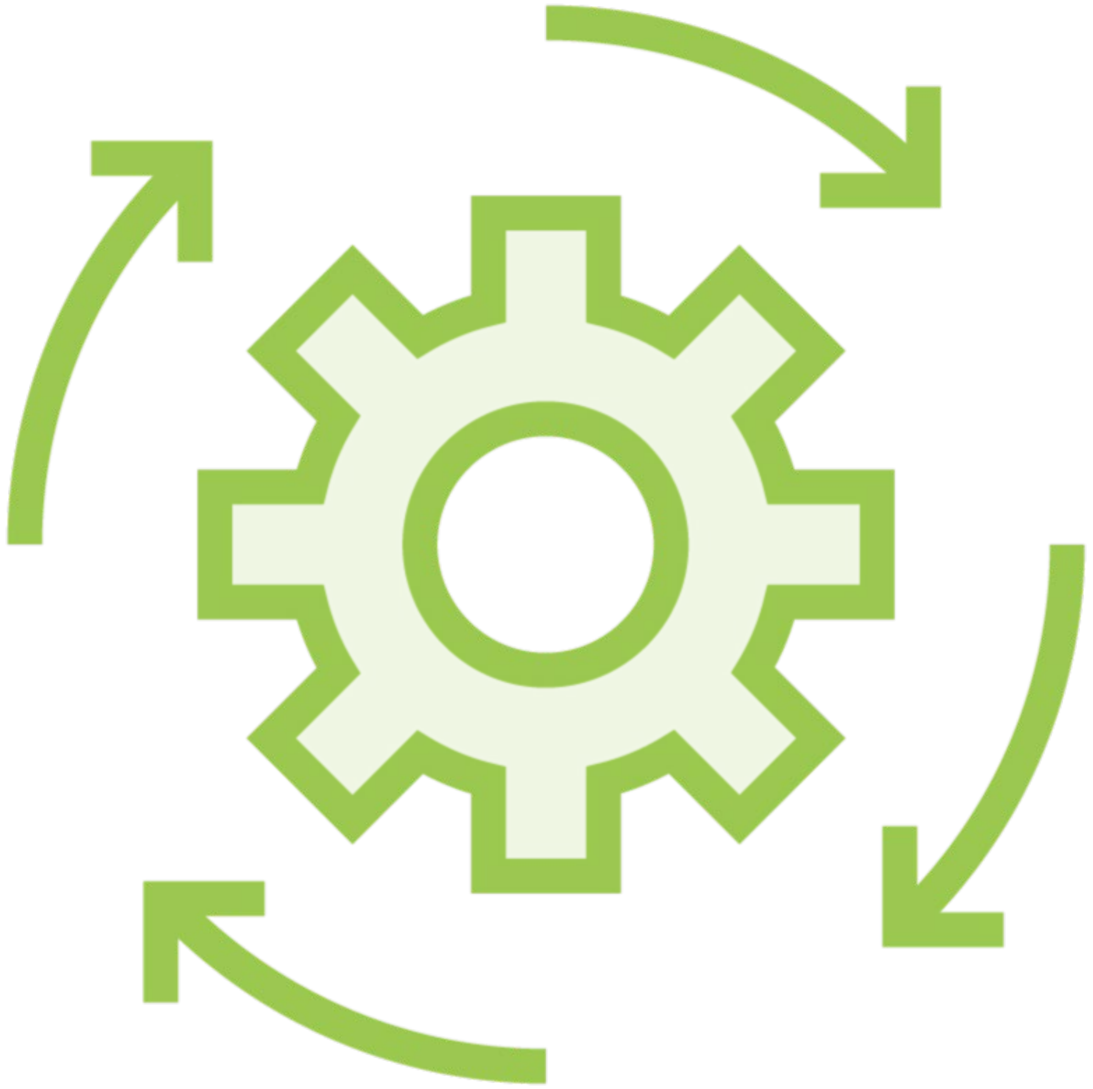
**The compiler is free to generate the most suitable code**

**Multiple mechanisms may be used**

- Such as string Concat or Format

**Considerations regarding allocations still apply**

# Interpolated String Handlers



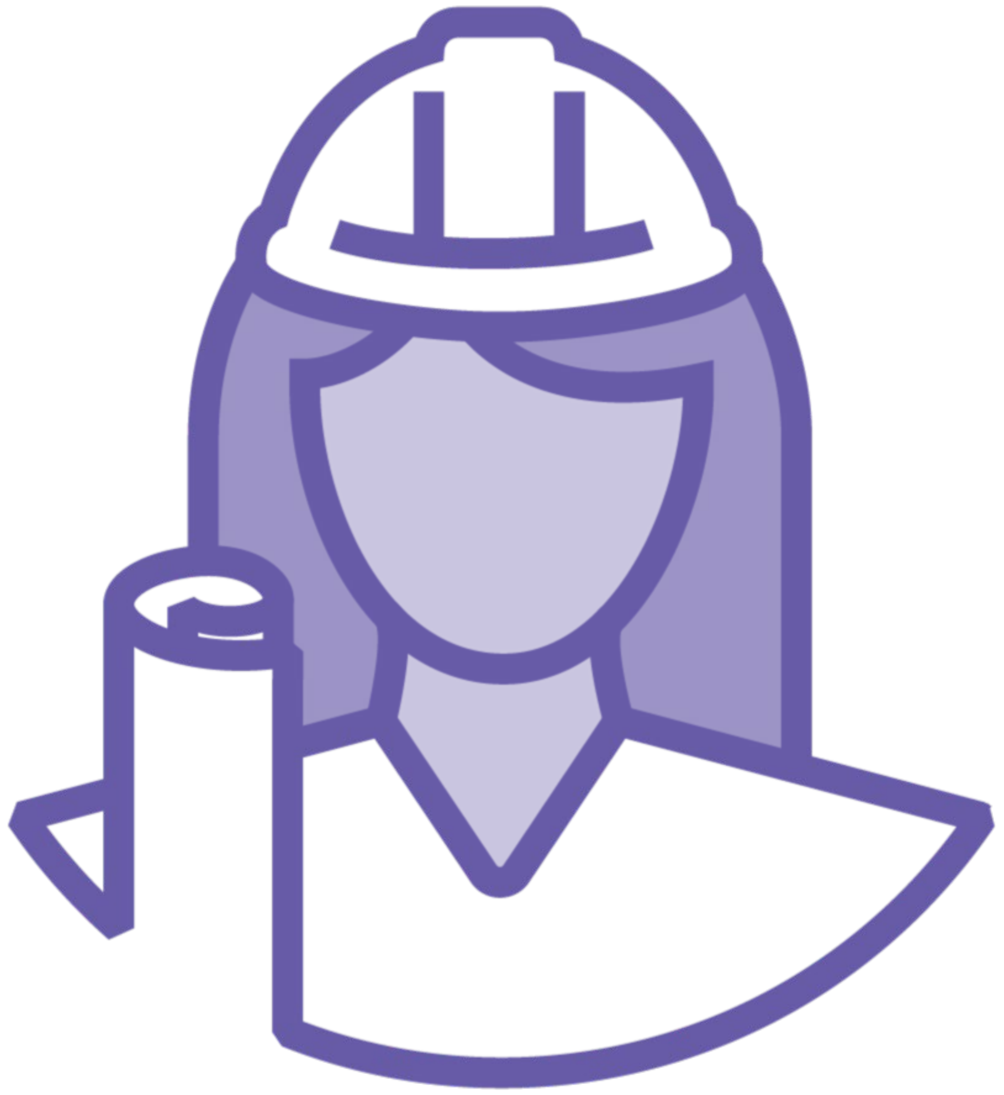
**C# 10 introduces a new mechanisms called interpolated string handlers**

**A value type that can efficiently process the placeholder expressions**

**Acts as a builder for the final string**

- Internal implementation avoids allocations

## Custom Interpolated String Handlers



**It's possible to define bespoke interpolated string handlers**

**For example:**

- A handler to build valid URLs with proper URL escaping
- A log message handler to avoid allocating strings for log messages that are not required based on their log level

**An advanced topic, not covered in this course**

Interpolated strings in C# 10 will generally cause the fewest allocations when compared to the other techniques covered in this module.

Up Next:

Efficient String Manipulation Using StringBuilder