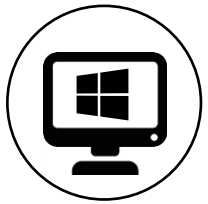
Sezione 19

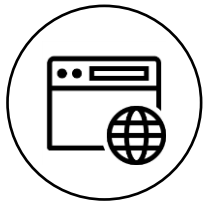
Novità di .NET 5

Unificazione in corso...



Desktop Windows
(Winforms o WPF)

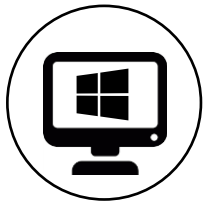
Server Web
(ASP.NET Core)



Browser
(Blazor WebAssembly)

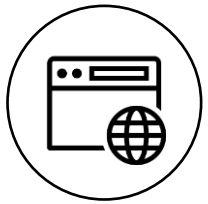
Mobile
(Xamarin.Android)

Unificazione in corso...



Desktop Windows
(Winforms o WPF)

Server Web
(ASP.NET Core)



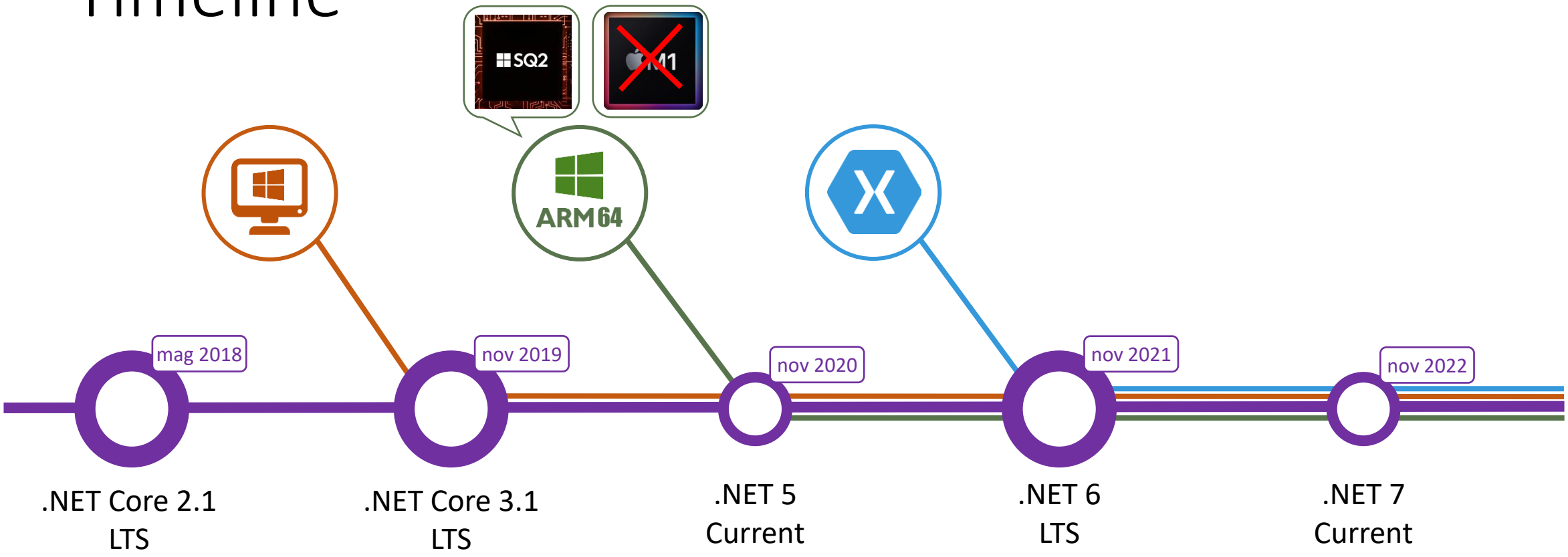
Browser
(Blazor WebAssembly)

Mobile
(Xamarin.Android)

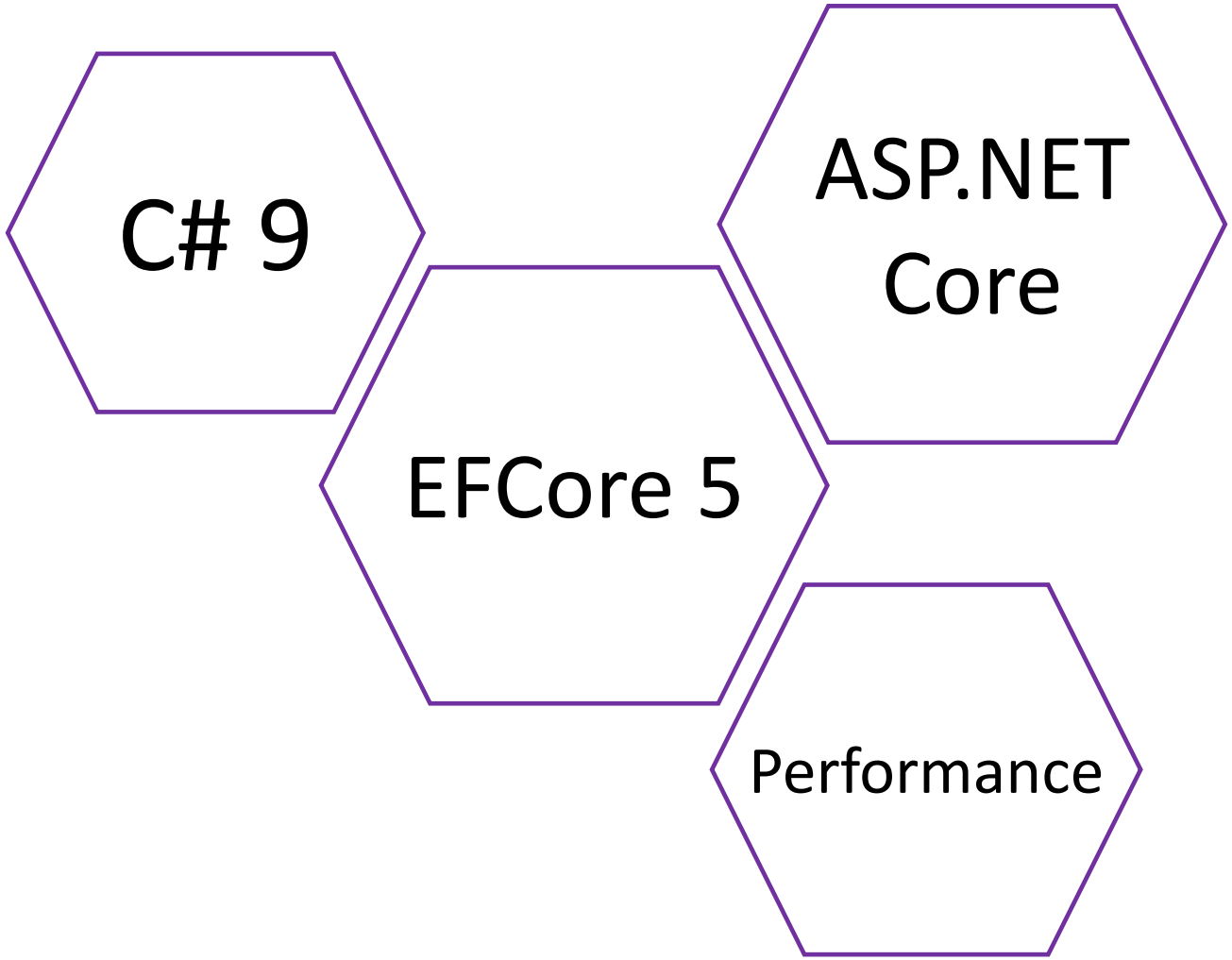
.NET 6

CoreCLR Runtime

Timeline


SQ2
M1
ARM64
mag 2018
nov 2019
nov 2020
nov 2021
nov 2022
.NET Core 2.1
LTS
.NET Core 3.1
LTS
.NET 5
Current
.NET 6
LTS
.NET 7
Current

Novità introdotte con .NET 5



C# 9: record

```
var price1 = new Money { Currency = Currency.EUR, Amount = 15.00m };
var price2 = new Money { Currency = Currency.EUR, Amount = 15.00m };
Console.WriteLine(price1 == price2);
```

false

C# 8

```
public class Money {
    public Currency Currency { get; set; }
    public decimal Amount { get; set; }
}
```

```
var price1 = new Money(Currency.EUR, 15.00m);
var price2 = new Money(Currency.EUR, 15.00m);
Console.WriteLine(price1 == price2);
```

true

C# 9

```
public record Money (Currency Currency, decimal Amount);
```

C# 9: mutare un record

Le proprietà autogenerate di un record sono init-only.

```
Money imponibile = new Money(Currency.EUR, 15.00m);
imponibile.Amount = imponibile.Amount * 1.22m;
```

Errore di compilazione!

```
Money ivato = imponibile with { Amount = imponibile.Amount * 1.22m };
```

Grazie a `with` creiamo una nuova istanza del record emendando i valori delle proprietà di un altro record.

C# 9: init-only setters

```
public record Money (Currency Currency, decimal Amount);
```

Per definire un record, possiamo usare la forma compatta…

```
public record Money
{
    public Currency Currency { get; init; }
    public decimal Amount { get; init; }
}
```

…oppure definire esplicitamente le sue proprietà. Usiamo `init` al posto di `set` per rendere il record immutabile.

```
var imponibile = new Money { Currency = Currency.EUR, Amount = 15.00m };

imponibile.Amount = imponibile.Amount * 1.22m;
```

Errore di compilazione!

C# 9: nuove keyword **and**, **or** e **not**

C# 8

```
if (string.IsNullOrEmpty(title))
{
  throw new ArgumentException("Title cannot be empty");
}
```

C# 9

```
if (title is null or "")
{
  throw new ArgumentException("Title cannot be empty");
}

if (title is "MyCourse" or not ({ Length: >= 10 } and { Length: <= 100 }))
{
  throw new ArgumentException("Title is not valid");
}
```

C# 9: target-typed **new**

C# 8

```
List<string> lista = new List<string>();

var lista = new List<string>();
```

C# 9

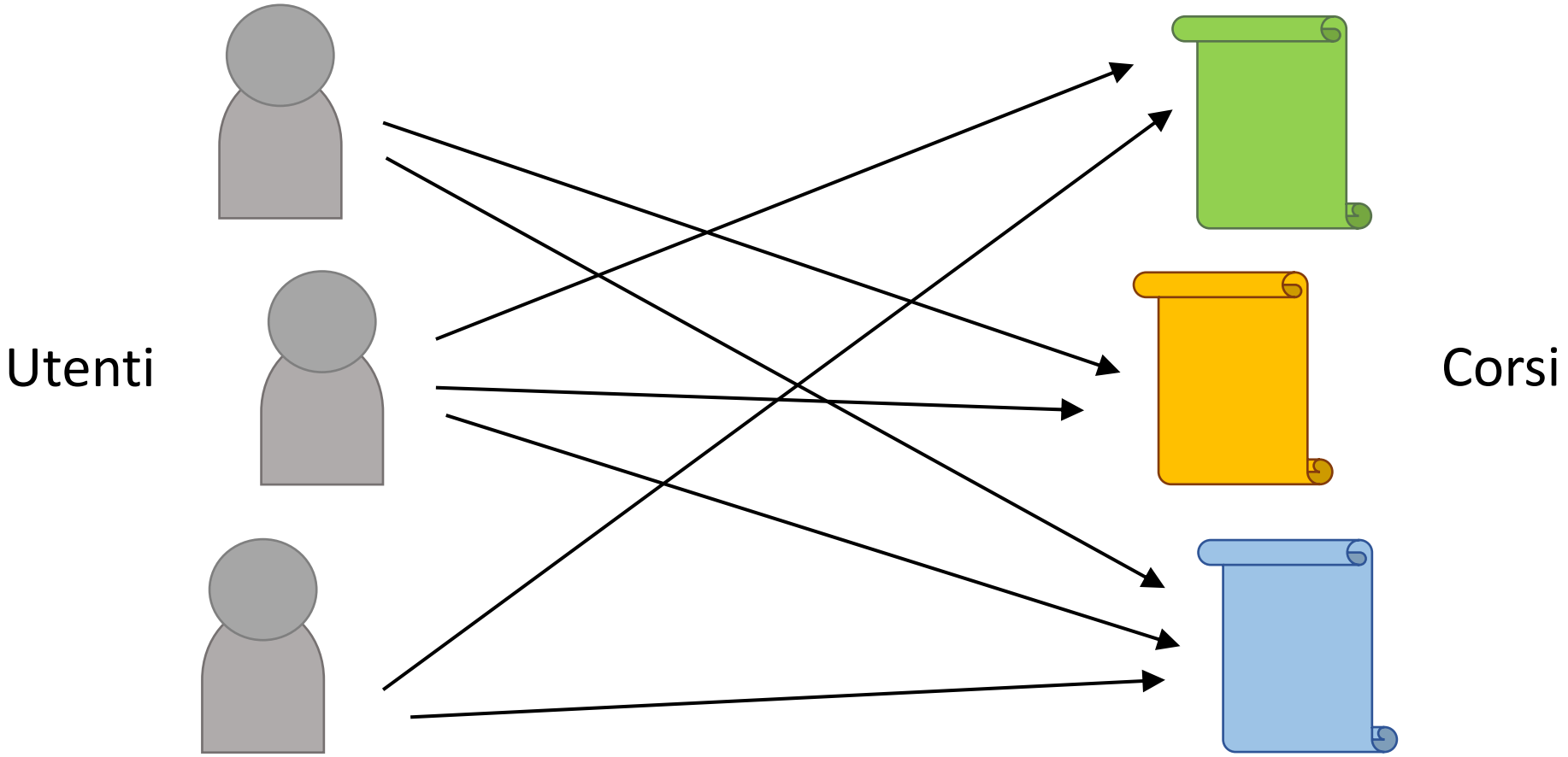
```
List<string> lista = new();

public class SantaClausHelper
{
    private readonly List<string> good = new();
    private readonly List<string> naughty = new();
}
```

C# NOTEBOOKS

- Per sperimentare con il linguaggio C# senza dover creare un progetto;
- Sono un ottimo strumento di apprendimento;
- Si installano come estensione di VSCode: https://marketplace.visualstudio.com/items?itemName=ms-dotnettools.dotnet-interactive-vscode

EFCore 5: relazioni many-to-many



EFCore 5: relazioni many-to-many

```
public class ApplicationUser
{
  public string FullName { get; set; }
  public ICollection<Course> SubscribedCourses { get; set; }
}
```

```
public class Course
{
  public string Title { get; set; }
  public ICollection<ApplicationUser> Students { get; set; }
}
```

EFCore 5: relazioni many-to-many

```
modelBuilder.Entity<Course>(entity =>
{
  entity.HasMany(course => course.Students)
        .WithMany(student => student.SubscribedCourses)
        .UsingEntity(join => join.ToTable("CourseStudents"));
}
```

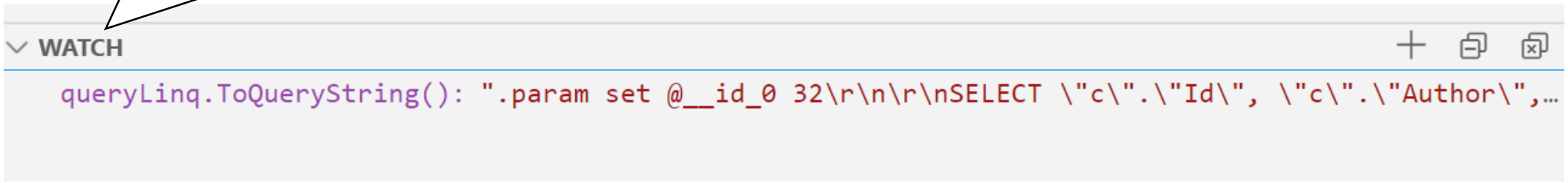
EFCore 5: ispezione e log delle query SQL

```
IQueryable<Course> queryLinq = dbContext.Courses.Where(course => course.Rating > 4.5);

string querySql = queryLinq.ToQueryString();

List<Course> courses = await queryLinq.ToListAsync();
```

Oppure nel pannello watch durante il debug



EFCore 5: ispezione e log delle query SQL

```
services.AddDbContextPool<MyCourseDbContext>(optionsBuilder => {
    string connectionString = Configuration.GetSection("ConnectionStrings").GetValue<string>("Default");
    optionsBuilder.UseSqlite(connectionString);
    optionsBuilder.LogTo()
```

```
DbContextOptionsBuilder
DbContextOptionsBuilder.LogTo(Action<string> action,
LogLevel minimumLevel = LogLevel.Debug,
DbContextLoggerOptions? options = null)
```

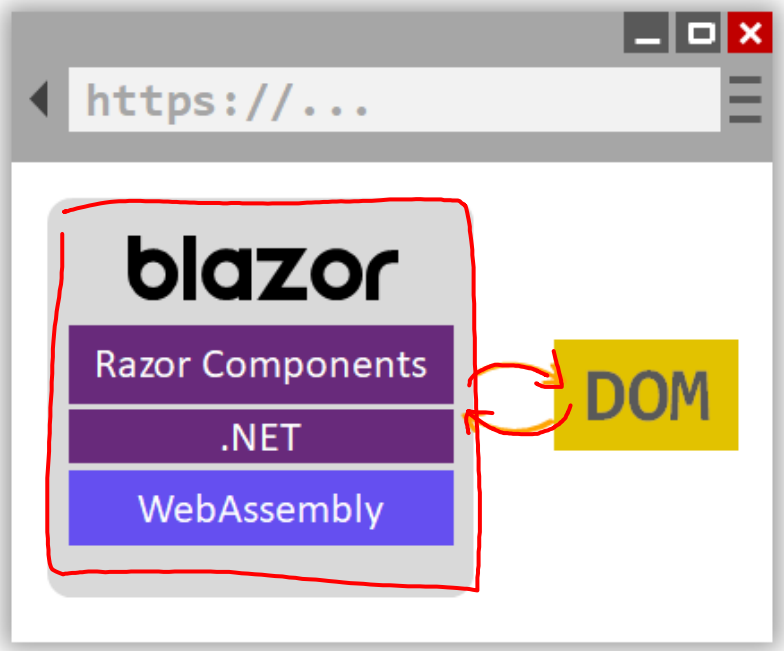
action: Delegate called when there is a message to log.

Logs using the supplied action. For example, use optionsBuilder.LogTo(Console.WriteLine) to log to the console.

1/5

This overload allows the minimum level of logging and the log

ASP.NET Core su .NET 5: Blazor WebAssembly



- Supporto all'upload di file
- Pre-rendering server side
- Lazy loading delle dipendenze
- 3x performance

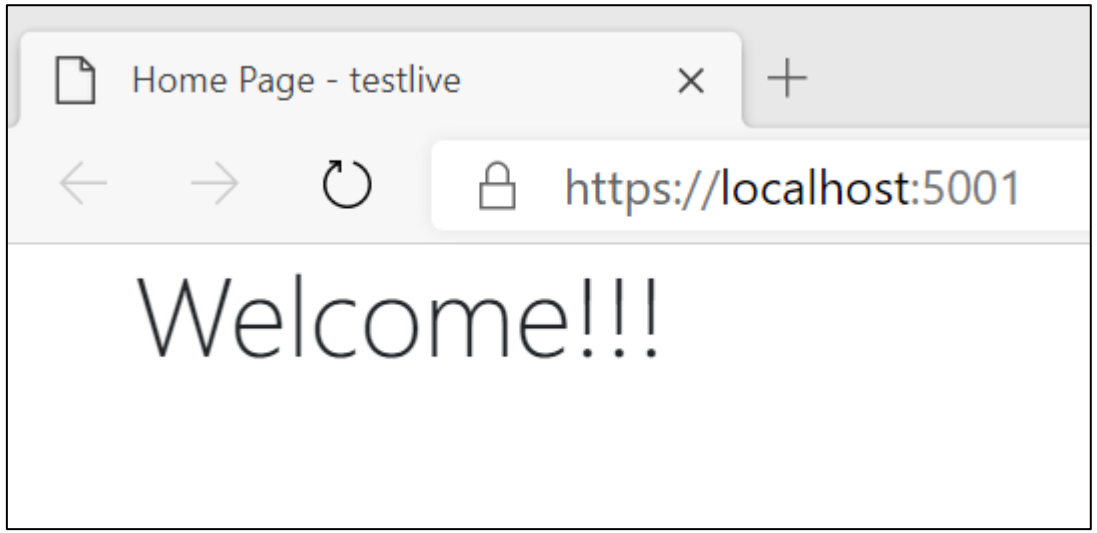
ASP.NET Core su .NET 5: live reload integrato

```
dotnet watch run
```

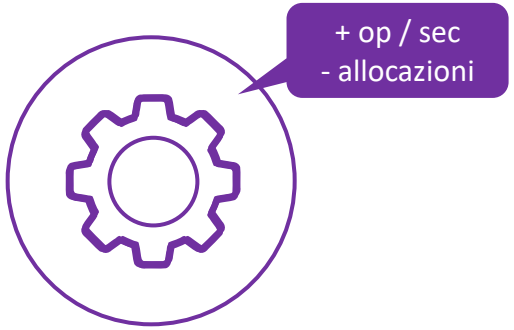
Index.cshtml

Views > Home > Index.cshtml

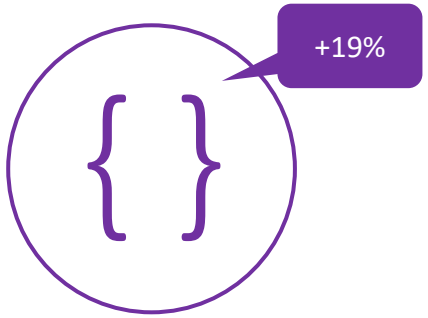
```
1   @{
2       ViewData["Title"] = "Home Page";
3   }
4
5   <div class="text-center">
6       <h1 class="display-4">Welcome!!!</h1>
7   </div>
8
```



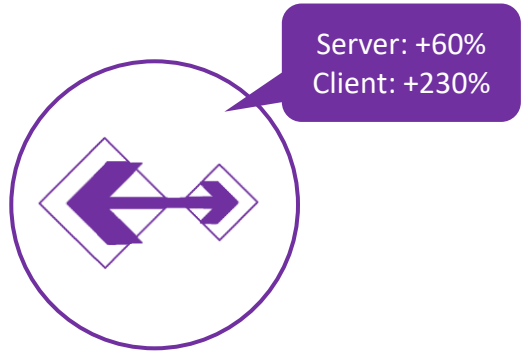
Migliori prestazioni rispetto a .NET Core 3.1



Base Class Library &
Garbage Collector

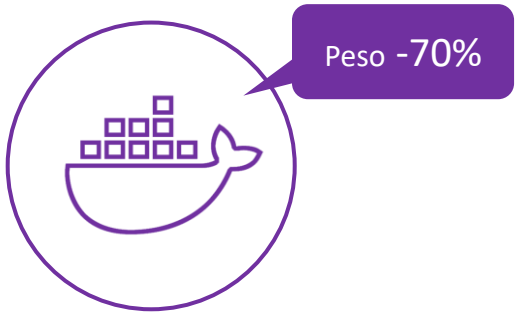


Serializzazione
JSON



gRPC

Kestrel
HTTP/2



Immagini
Docker

Breaking change

Prima di aggiornare il progetto a .NET 5, verificare se ci sono breaking change che possano costringerci a modificare il codice dell'applicazione.

https://docs.microsoft.com/en-us/dotnet/core/compatibility/5.0

Aggiornare il progetto da .NET Core 3 a .NET 5

- Aggiornare gli strumenti di sviluppo / download .NET 5 SDK;
- Aggiornare la versione della SDK nel file `global.json`;
- Aggiornare il `TargetFramework` nel file `.csproj`;
  - Da `netcoreapp3.1` a `net5.0`
- Aggiornare le versioni dei pacchetti NuGet;
  - `dotnet tool update --global dotnet-outdated-tool`
  - `dotnet outdated -u:Prompt`
- Aggiornare il file `.vscode/launch.json`.

Semplificazioni al codice

- Rimuovere `SetCompatibilityVersion` dalla classe `Startup`;
- `ASPNETCORE_HOSTINGSTARTUPASSEMBLIES` nel `launchSettings.json` al posto di `AddRazorRuntimeCompilation`;
- In C# 9.0, i value type possono diventare dei record;
- Usare Target-typed `new` per rendere più concise e leggibili le assegnazioni di variabili e campi privati.