Sezione 09

Servizi e dependency injection

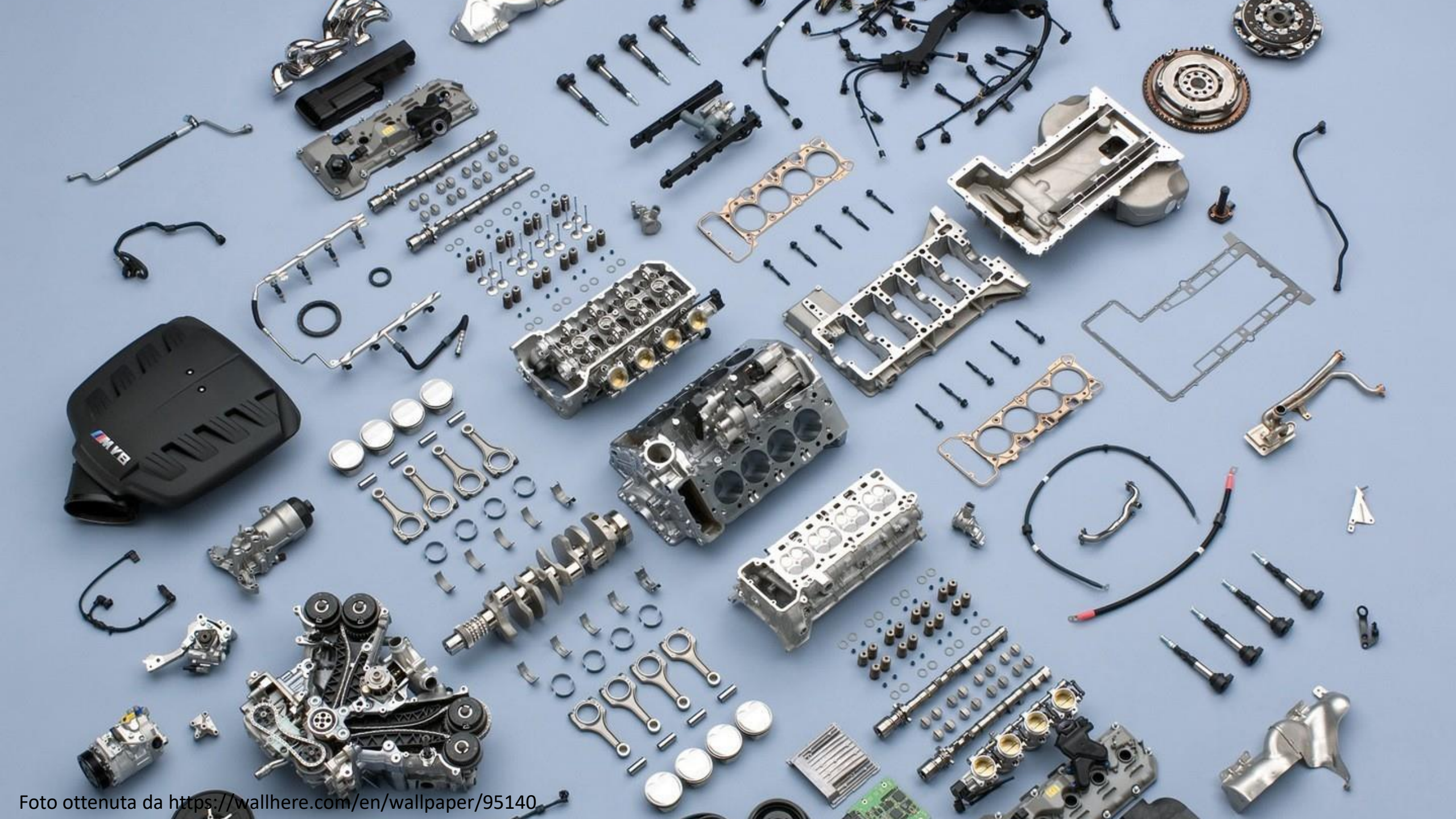


Foto ottenuta da https://wallhere.com/en/wallpaper/95140

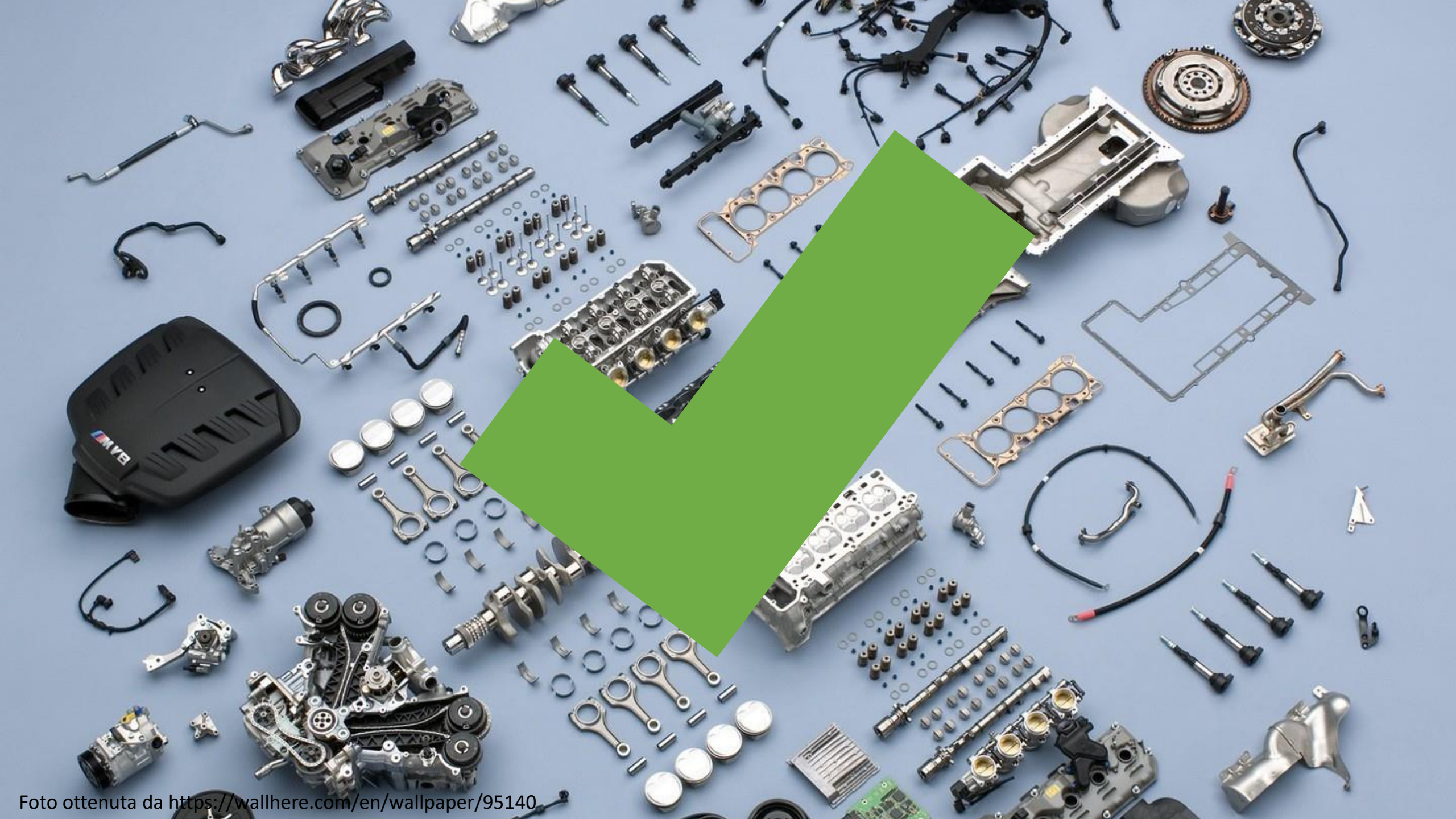


Foto ottenuta da https://wallhere.com/en/wallpaper/95140

Foto ottenuta da https://wallhere.com/en/wallpaper/95140

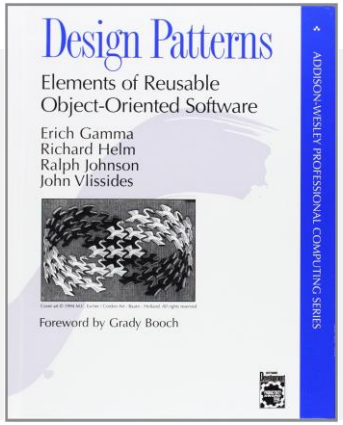
Dependency injection

- Uno dei modi corretti di esprimere la dipendenza di un componente da un altro componente è quello di definire un parametro nel costruttore: `public class CoursesController(CourseService courseService)`

- ASP.NET Core "risolve le dipendenze" cioè crea le istanze dei servizi e le passa al costruttore;
  - Affinché ciò avvenga, dobbiamo registrare i nostri servizi all'interno del metodo `ConfigureServices` della classe `Startup`.

Foto ottenuta da http://uniting4climate.net/tag/00-honda-civic-front-bumper/

Accoppiamento debole



"Program to an interface, not an implementation."

~~`public class CoursesController(CourseService courseService)`~~
```
public class CoursesController(ICourseService courseService)
```

- Migliora la testabilità dei componenti

Video di Patrick Mikolajczak da https://www.youtube.com/watch?v=xbvC20obbbI

Ciclo di vita dei servizi

Abbiamo 3 metodi per registrare servizi in `ConfigureServices`.
Ciascuno di essi influenza il ciclo di vita del servizio in modo diverso.

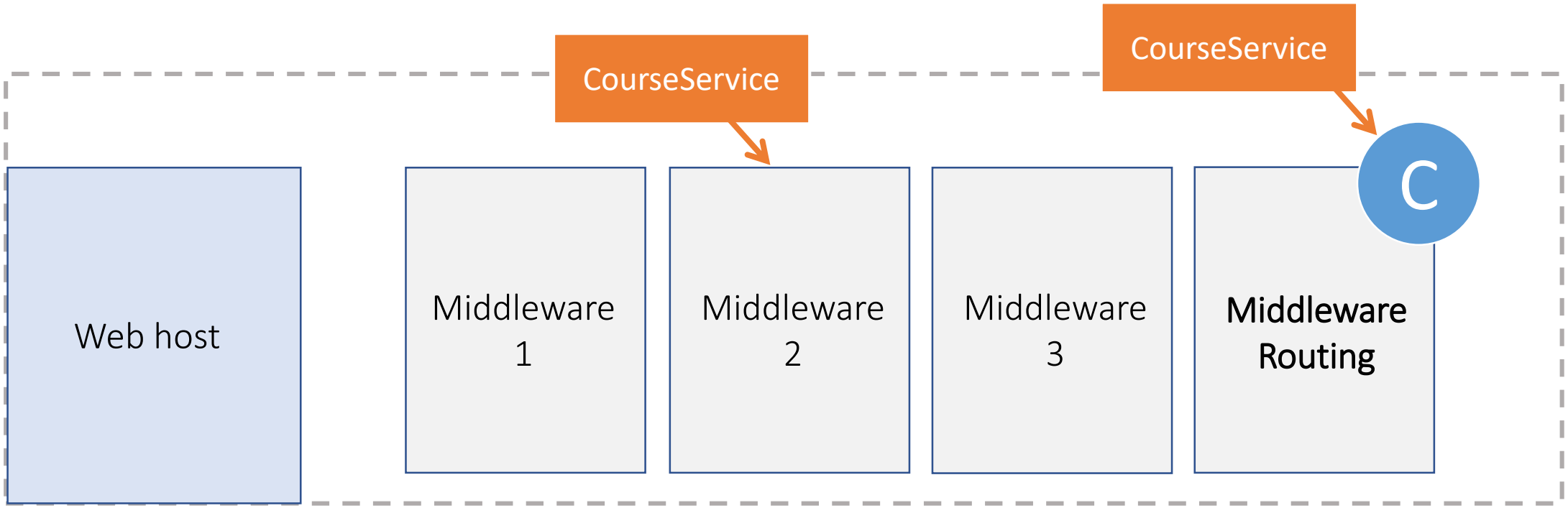
```
services.AddTransient<ICourseService, CourseService>
```

```
services.AddScoped<ICourseService, CourseService>
```

```
services.AddSingleton<ICourseService, CourseService>
```

AddTransient

- ASP.NET Core crea una nuova istanza del servizio ogni volta che un componente ne ha bisogno, e poi la distrugge dopo che è stata usata.

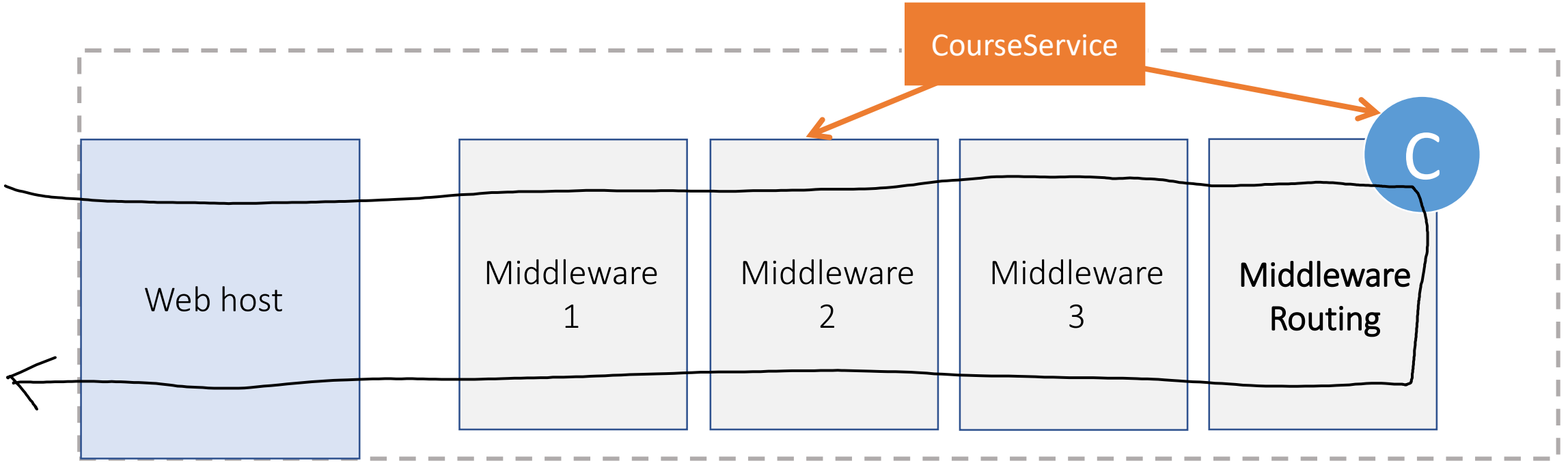


Usiamo AddTransient quando…

- I nostri servizi sono "veloci da costruire" e quindi non ci sono problemi prestazionali se ne vengono costruite più istanze.

AddScoped

- ASP.NET Core crea una nuova istanza e la inietta in tutti i componenti che ne hanno bisogno, e poi la distrugge al termine della richiesta HTTP.

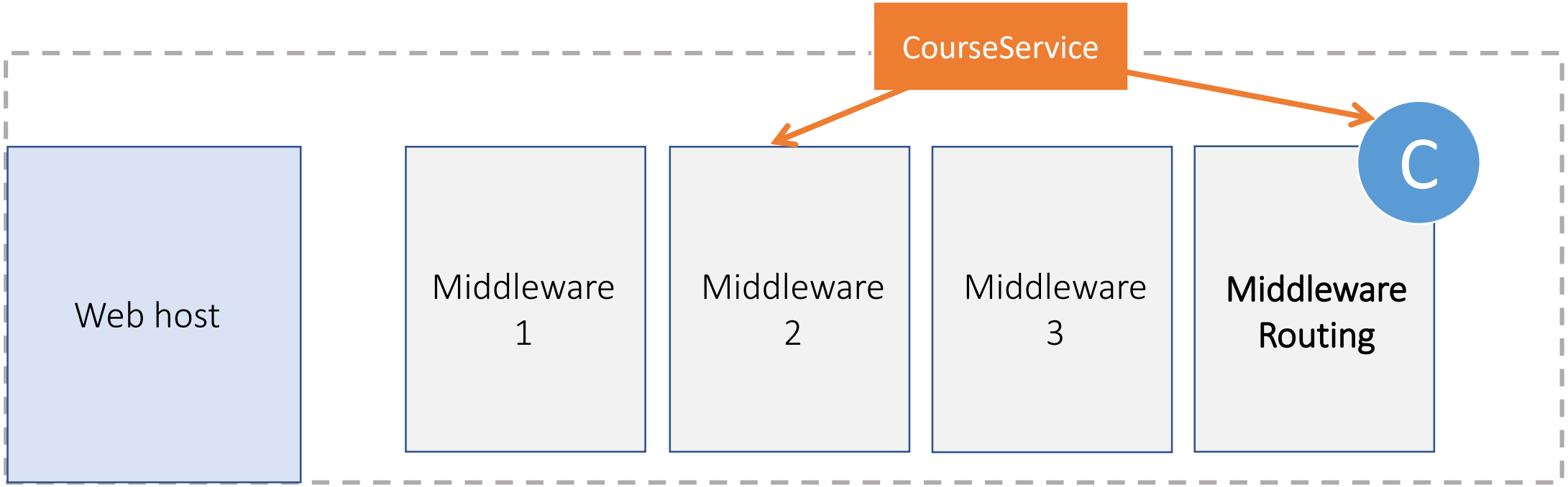


Usiamo AddScoped quando…

- Il servizio è "costoso da costruire", come il DbContext di Entity Framework Core che vedremo prossimamente, e perciò vogliamo ridurre i tempi di costruzione;
- Il servizio si collega a un database e vogliamo aprire la connessione la prima volta che il servizio viene usato e chiuderla quando il servizio viene distrutto (cioè al termine della richiesta HTTP);
- ATTENZIONE! Dato che l'istanza è la stessa, se il middleware scrive una proprietà sul servizio, poi il controller può rileggere quel valore.

AddSingleton

- ASP.NET Core crea un'istanza e la inietta in tutti i componenti che ne hanno bisogno, anche in richieste HTTP diverse e concorrenti.



Usiamo AddSingleton quando…

- Abbiamo servizi che devono agire in base alle richieste che arrivano.
  - Un servizio che spedisce e-mail, ma le deve spedire una alla volta;
  - Un servizio che conteggia il numero di richieste.

- **ATTENZIONE!** Un servizio singleton deve essere **thread-safe** perché può essere usato da più thread contemporaneamente!

  https://docs.microsoft.com/it-it/dotnet/standard/threading/synchronizing-data-for-multithreading