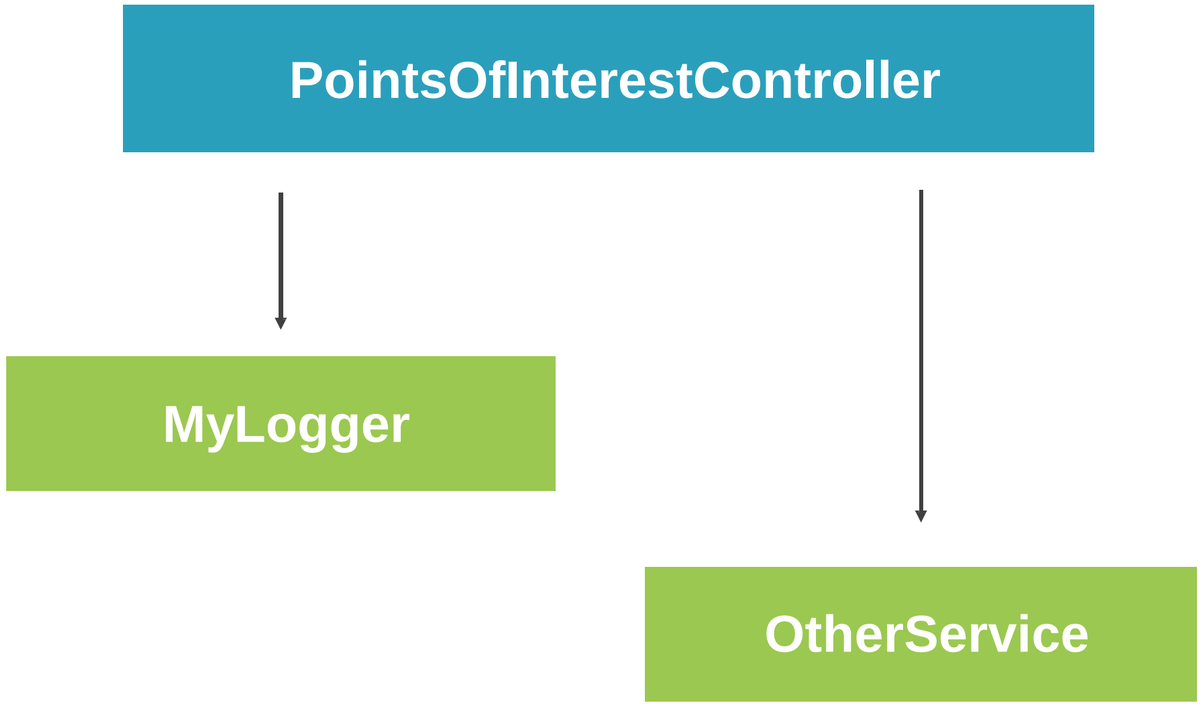
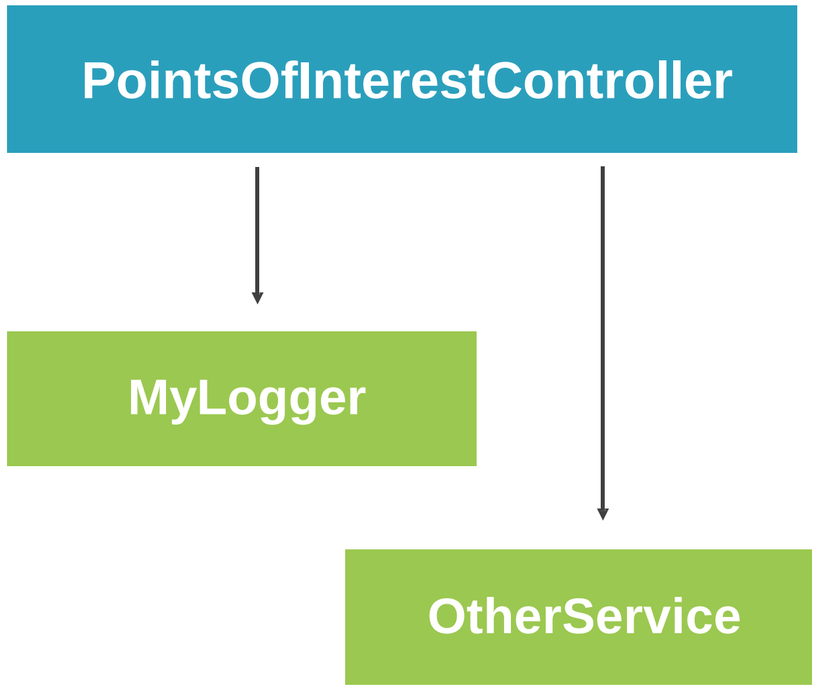
Trabajando con servicios e inyección de dependencias

Resumen

- **Inversión de control e inyección de dependencias**
- **Logging en Asp.Net Core**
- **Crear y utilizar servicios personalizados**
- **Trabajo con archivos de configuración ajustados a entornos**

Inversión de control e inyección de dependencias





- La implementación de la clase ha de cambiar cuando alguna dependencia cambie
- Es muy difícil de testar
- La clase es la que administra el ciclo de vida de la dependencia
- Estan fuertemente acoplados

Inversion de Control

La inversión de control delega la función de seleccionar un tipo a implementar concreto para las dependencias de una clase a un componente externo

Inyección de dependencias

La inyección de dependencias es una especialización del patrón Inversión de Control.
El patrón de inyección de dependencias utiliza un objeto (el contenedor) para inicializar objetos y proporcionar las dependencias necesarias al objeto.

Inversión de control e inyección de dependencias

Los servicios se han de registrar en el contenedor

- El contenedor es el responsable de proveer instancias cuando sea necesario. Gestiona el ciclo de vida

```
public class PointsOfInterestController :
Controller
{

    private
    ILogger<PointsOfInterestController>
    _logger;

    public PointsOfInterestController(
    ILogger<PointsOfInterestController>
    logger)
    {
        _logger = logger;
    }

...

}
```

- **Interfaz, no implementación concreta**
- **Inyección de constructor**

Inversión de control e inyección de dependencias

La clase está desacoplada del tipo concreto

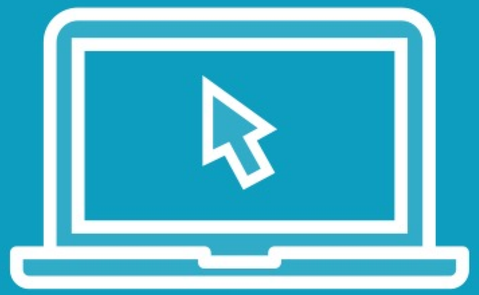
- Es posible cambiar las dependencias de forma sencilla
- Es mas sencillo probar las clases

Inversión de control e inyección de dependencias

La inyección de dependencias está incorporada en Asp.Net Core

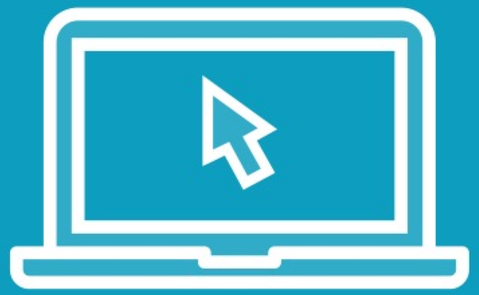
Registra los servicios en el contenedor en la clase Program

Demo



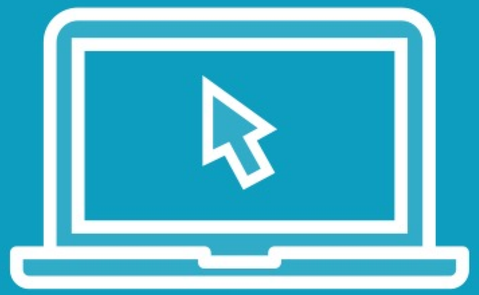
Inyectar y utilizar un logger

Demo



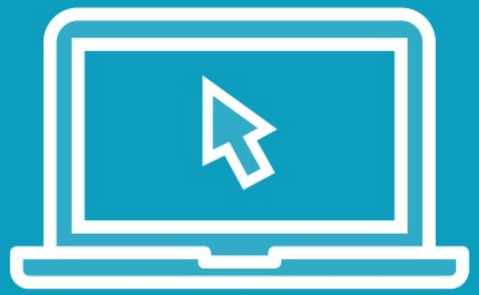
Logging de excepciones

Demo



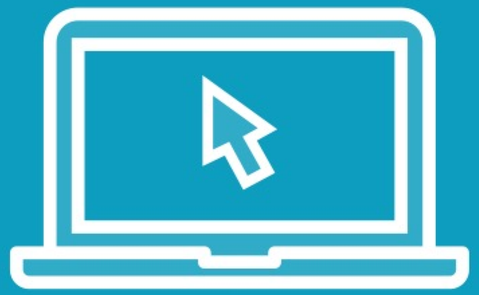
Reemplazar el logger predeterminado para hacer un log a un archivo

Demo



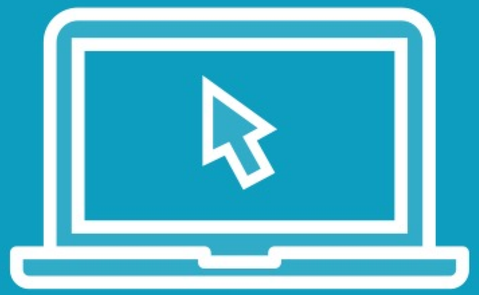
Implementar y utilizar un servicio personalizado

Demo



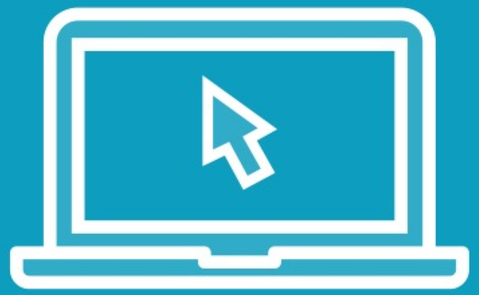
Registrar un servicio por interfaz

Demo



Trabajo con archivos de configuración

Demo



Ajustar configuración a entornos

Resumen

Inyección de dependencias

- Especialización de inversión de control
  - Acoplamiento bajo
  - Menos cambios en el código
  - Mejor testabilidad

Resumen

Los servicios personalizados se registran en el contenedor integrado

- Transient
- Scoped
- Singleton

Usa archivos de configuración para guardar datos de configuración restringidos al ámbito de un entorno

A continuacion
Introducción a Entity Framework Core