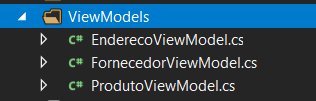
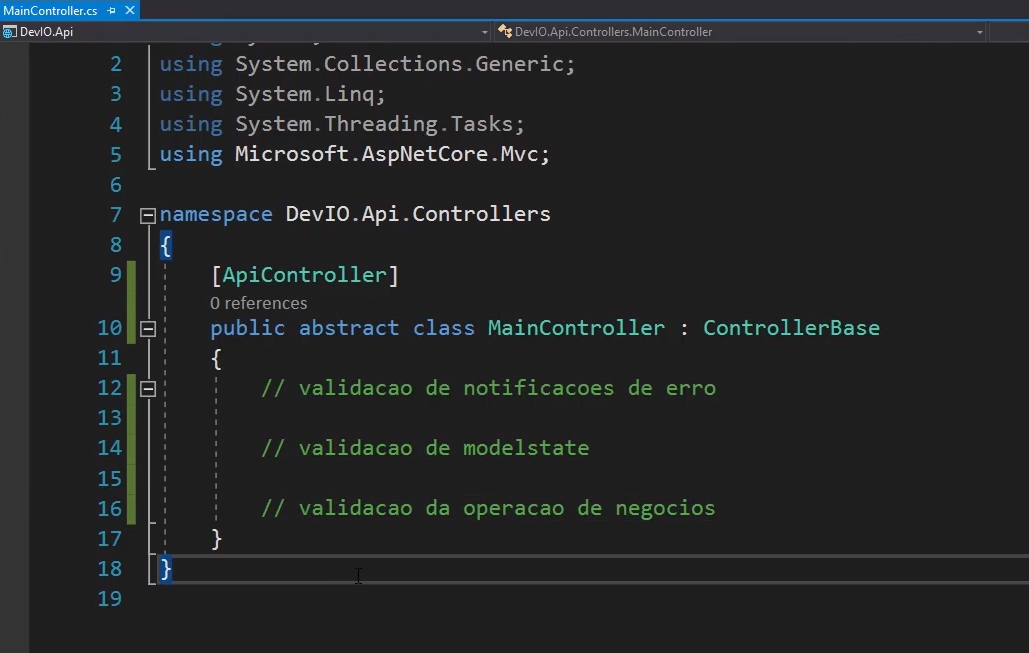


## Implementando DTOs (ViewModels)

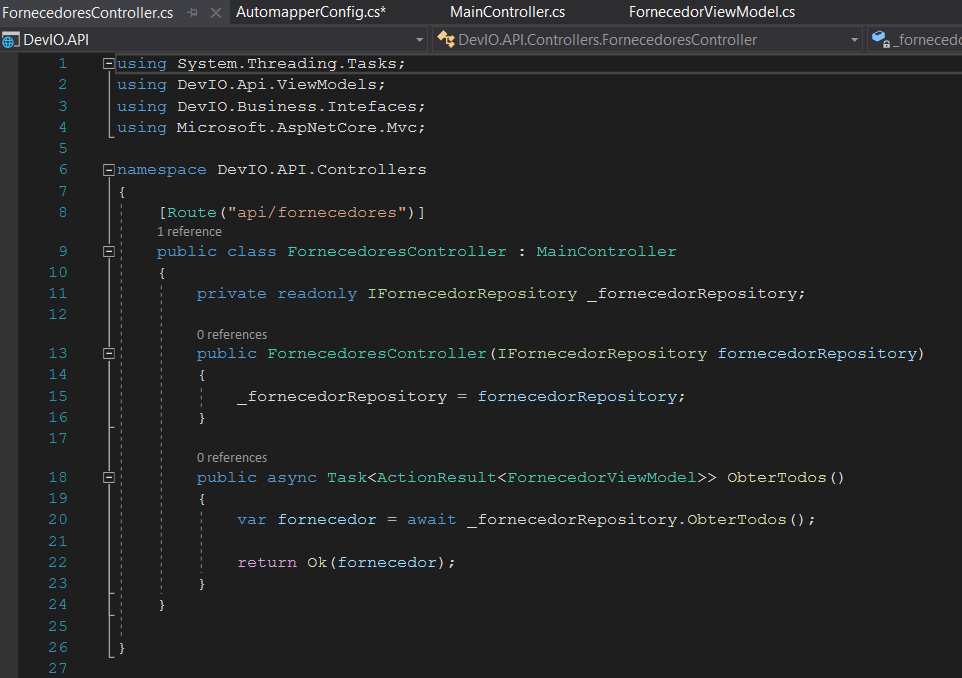
Criar as ViewModels:



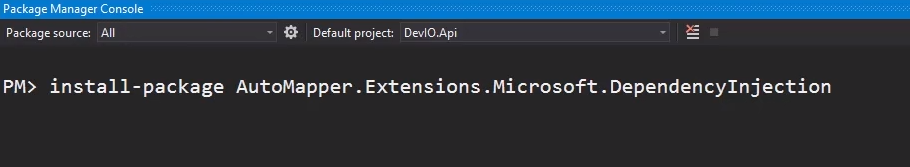
Criar a MainController



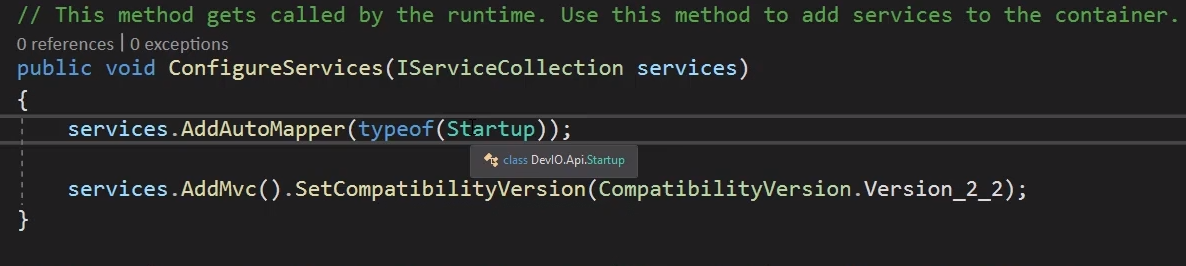
- Criar a classe FornecedorController herdando da Main Controller:



- Instalar o automapper para conversão das entidades em ViewModels.

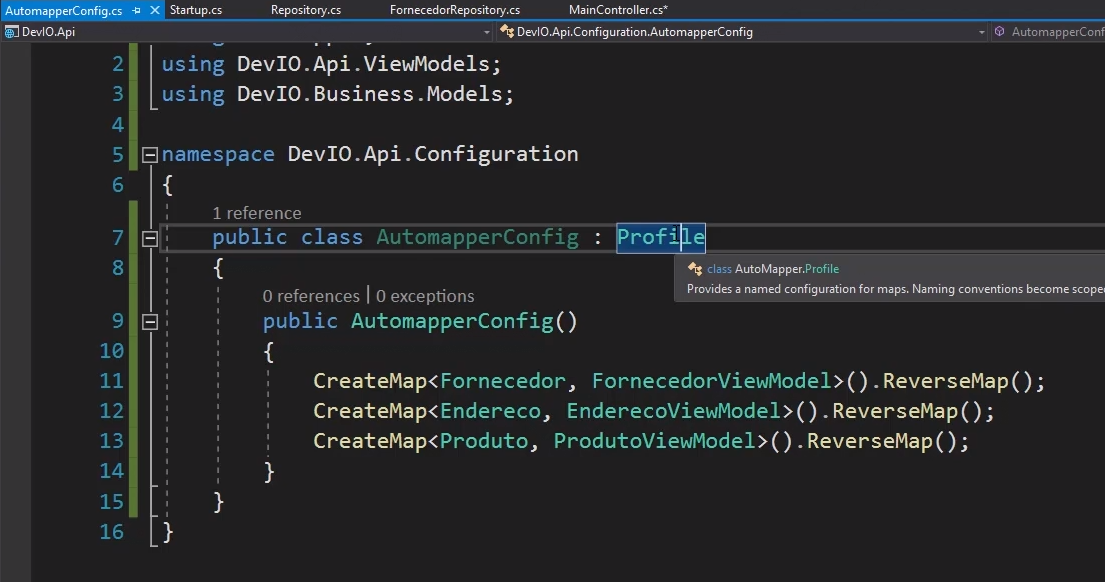


Para utilizar configurar na startup.cs



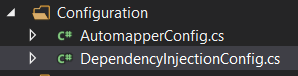
Configurar a Lista de Para, criando a classe AutomapperConfig:

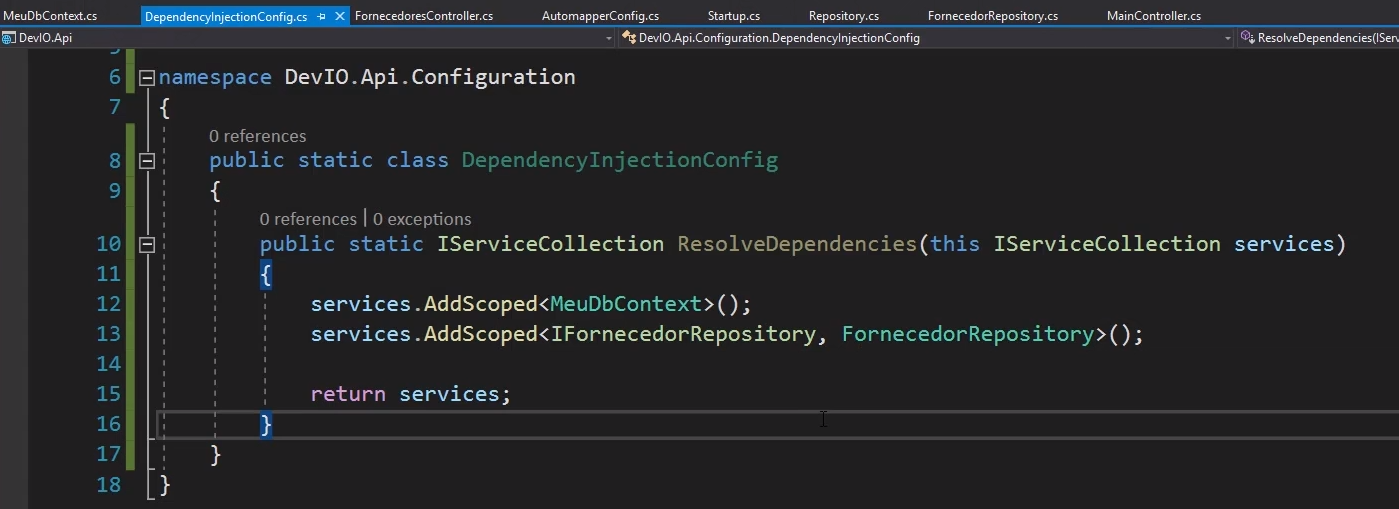




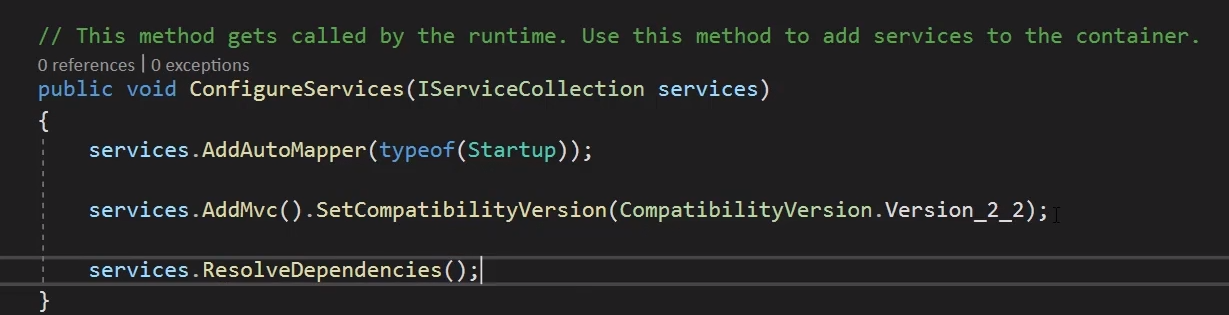
Injetando o automapper como dependencia:

Criar a classe DependencyInjectioConfig:

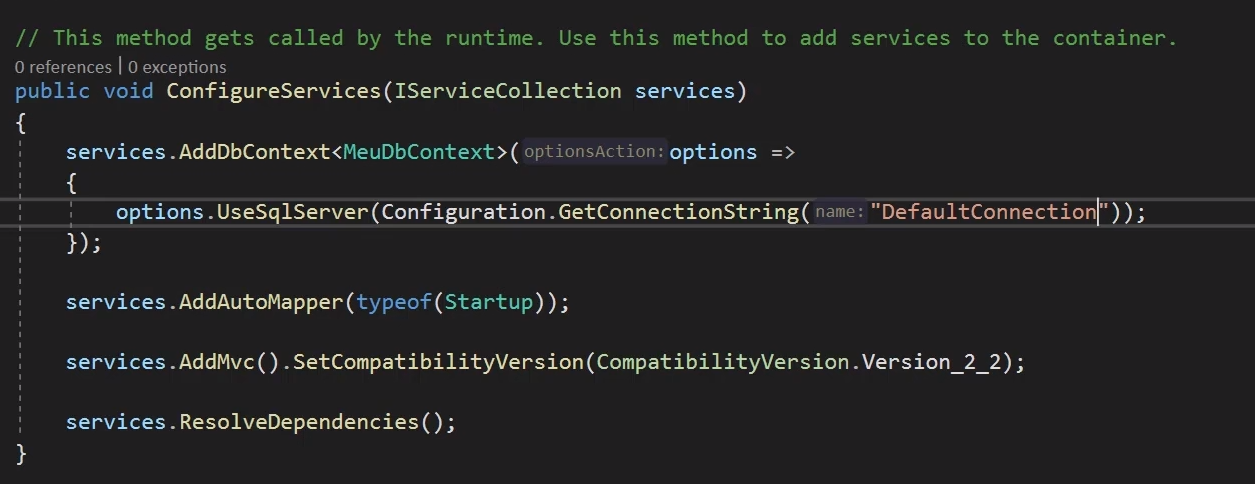




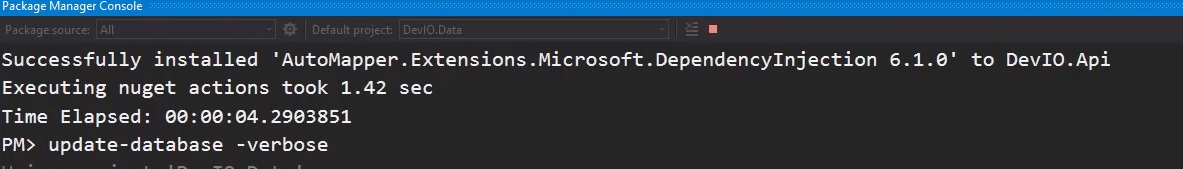
Chamar na Startup:



Configurar o DBContext - EntityFramework:



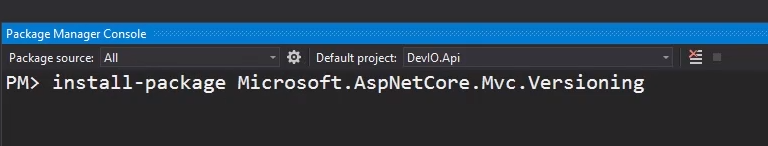
Criar o banco de dados, apontando para o projeto data:



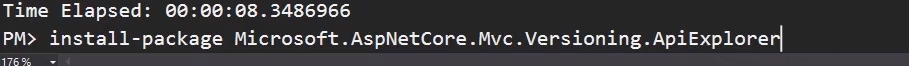
Verionamento de API.

Instalar os seguintes pacotes:

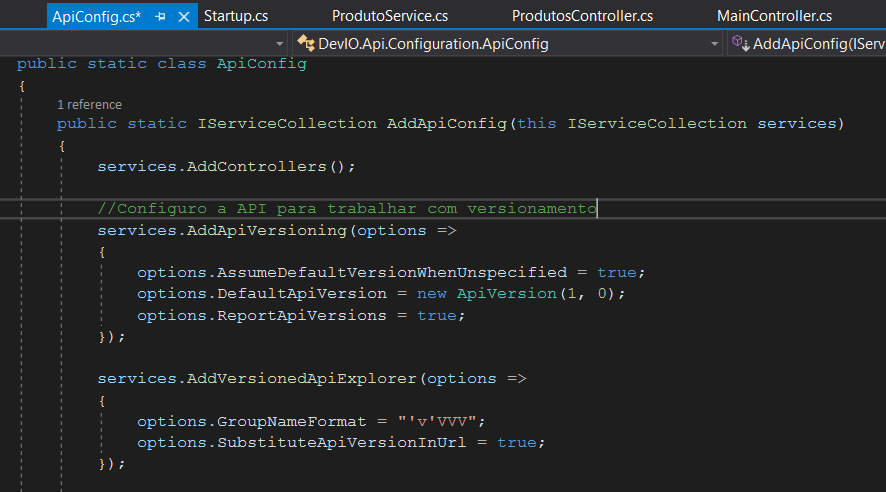
install-package Microsoft.AspNetCore.Mvc.Versioning



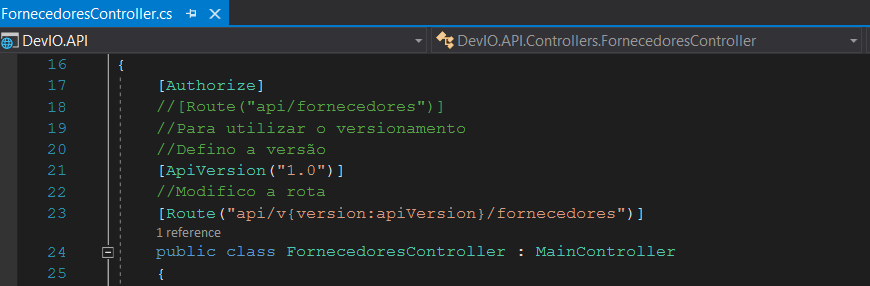
install-package Microsoft.AspNetCore.Mvc.Versioning.ApiExplorer



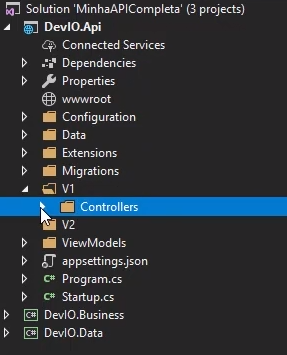
Configurar nossa API para Trabalhar com versionamento:



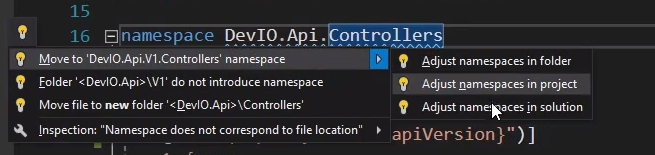
Para utilizar o versionamento, nas controllers, adicionar o atributo ApiVersion e modificar a o rota para utilização da versão da API.



Criar pastas de versionamentos para as controllers, ex. V1, V2... e copiar a pasta controller para dentro dela.

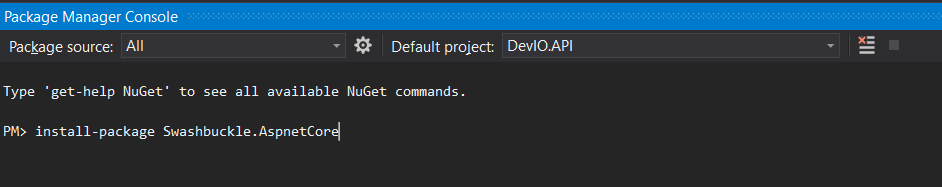


Ajustar o Namespace, adicionando V1, em todos os arquivos do projeto, desta forma não terei problema de ter a classe com o mesmo nome.

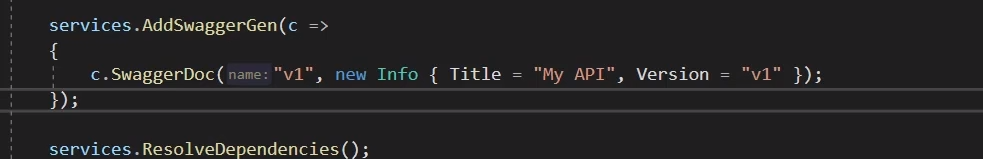


## Documentando a API com Swagger

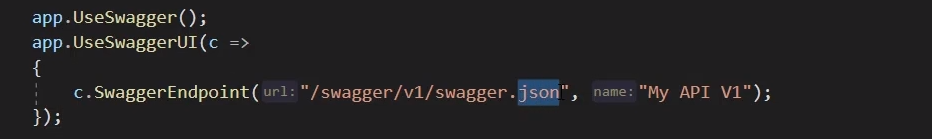
Instalar o Swagger:



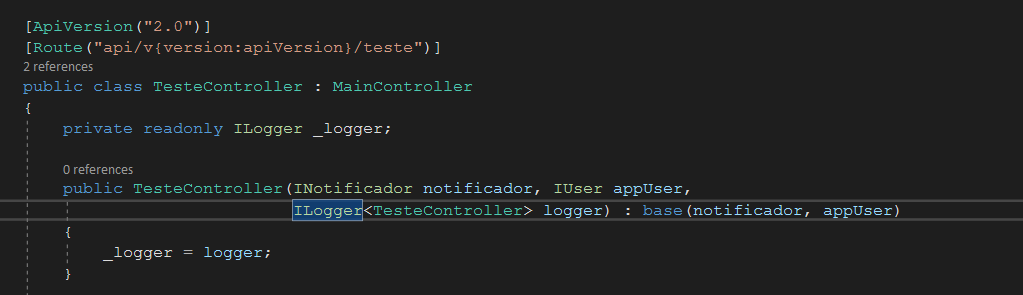
No Startup.cs, ConfigureService(), implementar o código de geração da documentação:

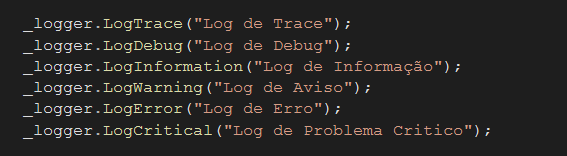


E no configure:



## Implementando Logging e monitoramento na API

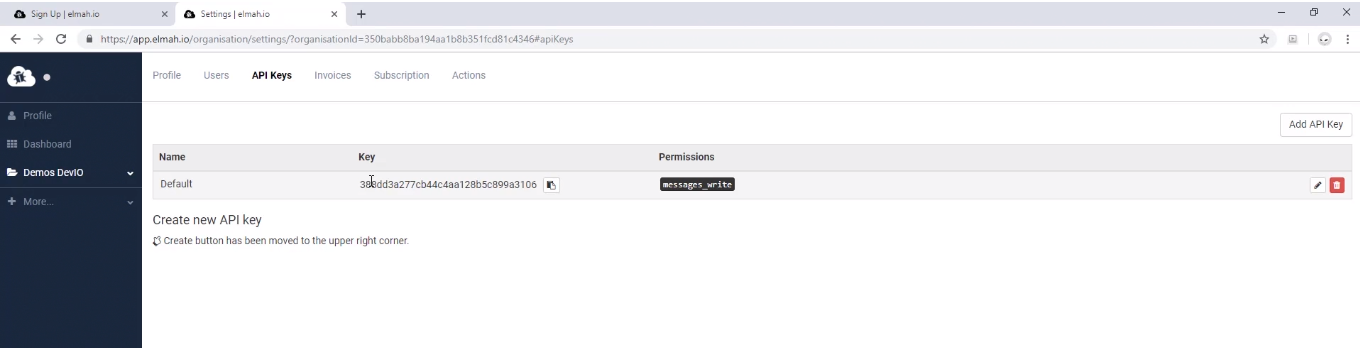


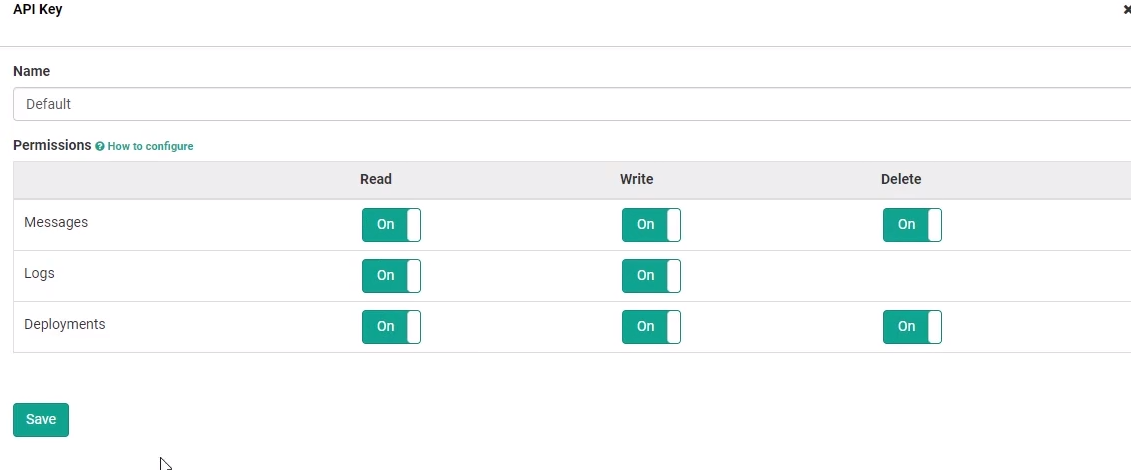


**Utilizando log de terceiros - Elmah**

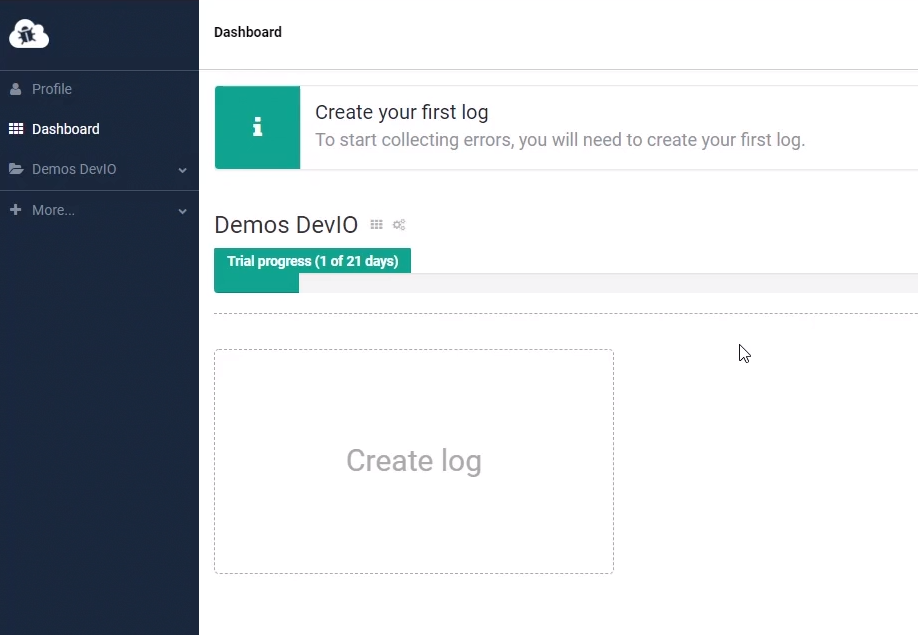
- Criar a conta no site Elmah.io.

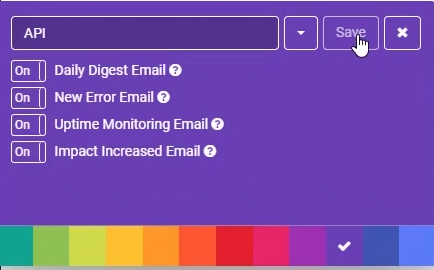
- Em configurações, dar todas as permissões de escrever, ler e excluir a chave default.



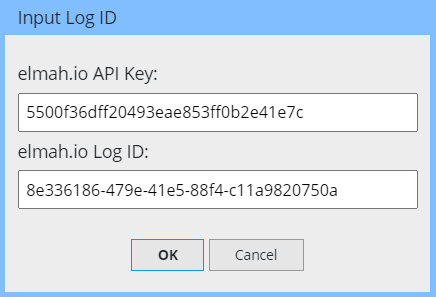


- Criar o log:

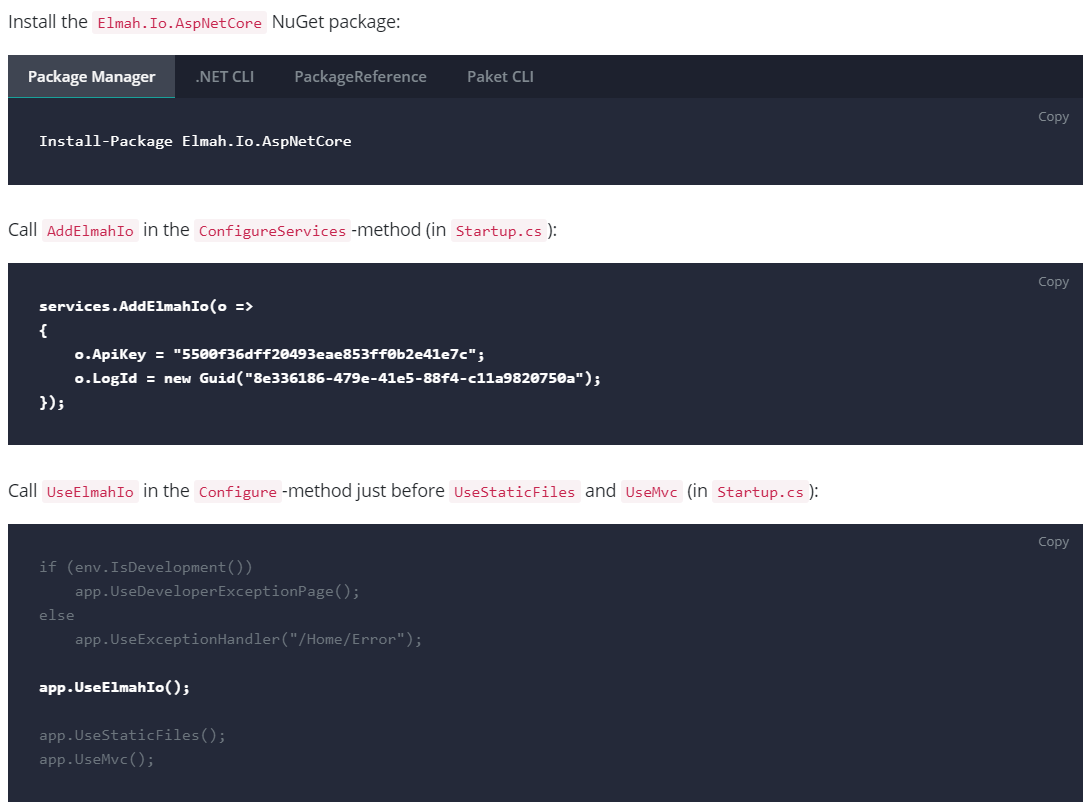




- Anotar as chaves de organização e do log:

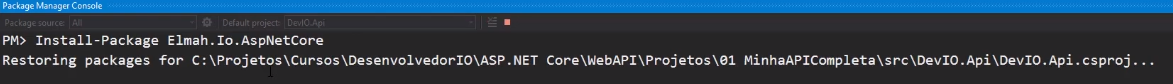


- Selecionar o Framework para o Core:



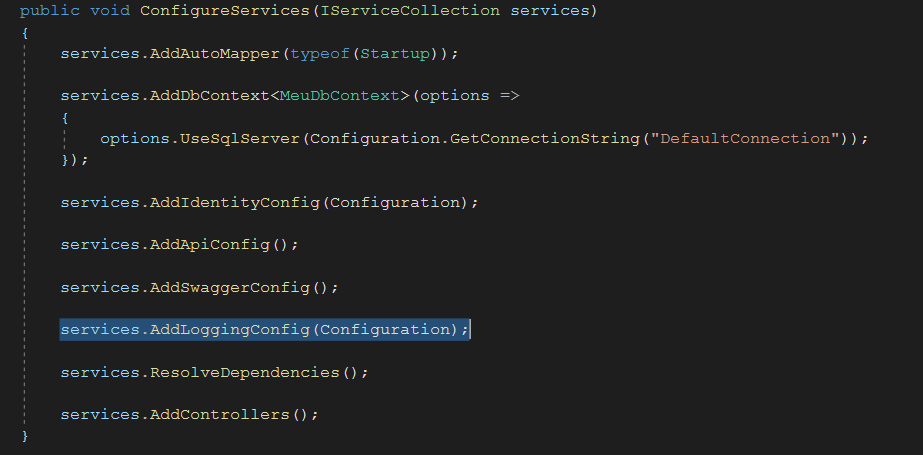


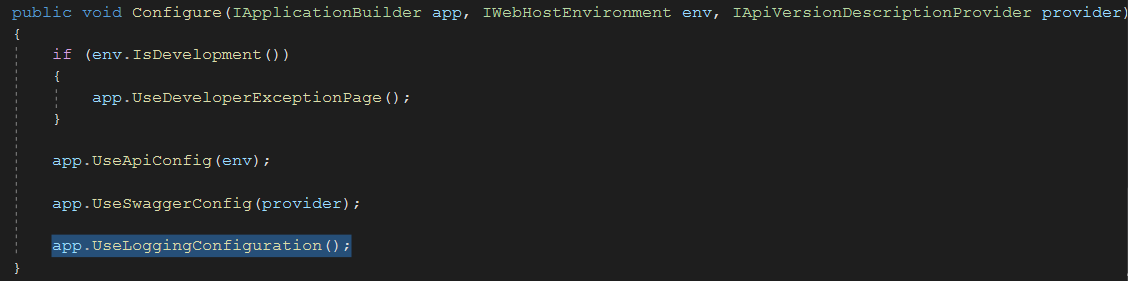
Instalar no projeto DevIO.App:



- Criar a Classe padrão LoggerConfig, igual a do projeto.

- Implementar chamadas na Startup:

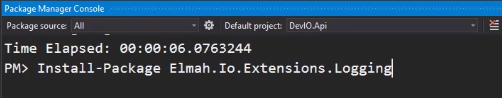




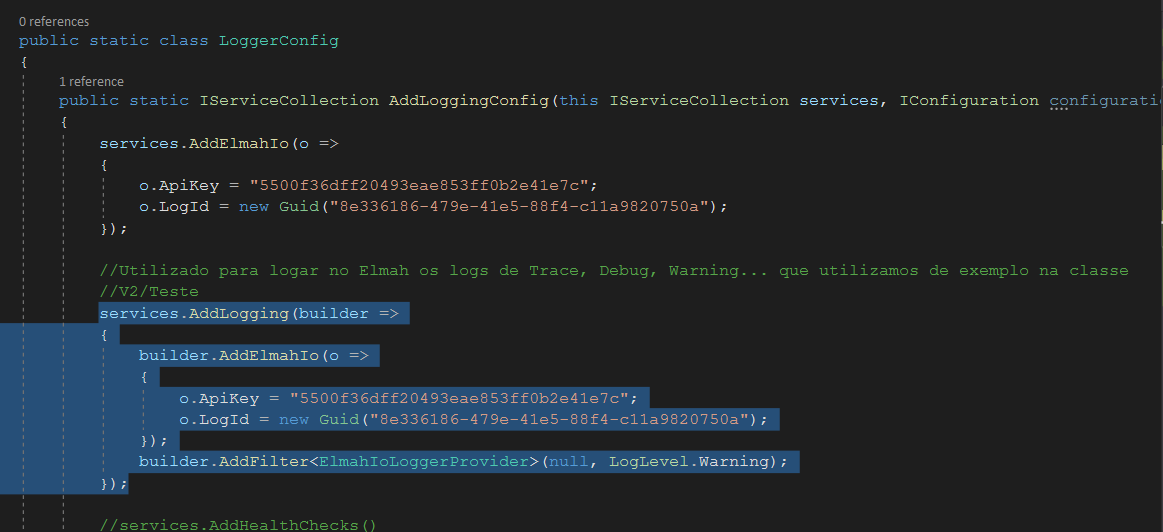
Pronto desta forma já está configurado e salvando no serviço do elmah.

Para conectar nossos logs (Debug, Information, Warning....) com o elmah, devemos fazer a seguinte configuração:

- Instalar o pacote: Install-Package Elmah.Io.Extensions.Logging



Na classe LoggerConfig, implementar a seguinte configuração:



Para capturar qualquer erro mesmo um que não foi tratado via Emah, devemos fazer o seguinte:

- Criar a classe de extensão padrão ExceptionMiddleware.cs, ver projeto.

- Imprementar a chamada no Startup:

