C# Avançado
Aula 01

- Convenções de codificação em C#
- Collections
- Generics
- Delegates e eventos
- Trabalhando com datas
- Trabalhando com arquivos
- Exceções e tratamento de erros
- Programação funcional
- LINQ e expressões Lambda
- Programação assíncrona
- NuGet

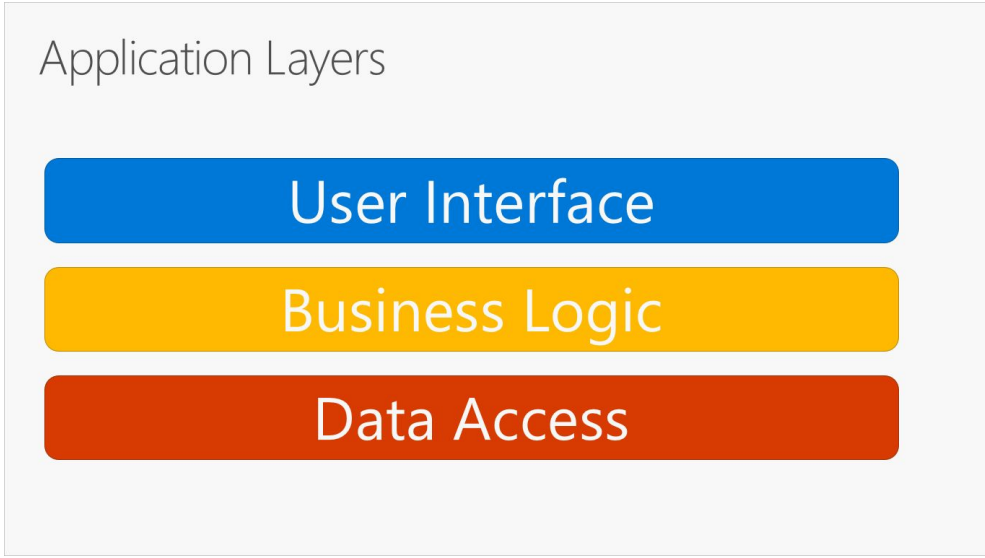
C# Avançado

O objetivo que temos agora e dar continuidade no que vimos até o momento, na aula de hoje o que iremos ver:

- **Rever como criar uma solução**
- **Maneira tradicional de dividir projetos**
- **Uma maneira de dividir aplicação mais atual.**
- **Como usar arquivo de configuração**

Camadas da solução

Existem diversas maneiras de arquitetar as nossas soluções, uma que muito usada é a n-layer, onde pode ser observado como a imagem abaixo:



Camadas da solução

- Vamo agora na prática criar um novo repositório para a nossa aula.
- Realizar o clone do novo repositório
  - git clone “https://github.com/drhamann/senai-avan-ado.git”
- Agora, abra o visual studio 2022, vá até “Arquivo -> Novo -> Projeto”
  - Na tela que irá aparecer, procurar por “Solução” e criar uma nova solução
  - Deve criar a solução na pasta que fez o clone

- Bom agora, vamos criar as nossas 3 pastas, “Apresentação”, “Lógica” e “AcessoAosDados”, após criar as pastas vamos criar nossos 3 projetos.
- Na primeira pasta que é Apresentação, criar um projeto do tipo console, nas outras duas pastas criar projetos do tipo Biblioteca.
- Criado os 3 projetos, vamos fazer com que eles se conversem e vamos adicionar um exemplo de código em cada camada.

Camadas da solução

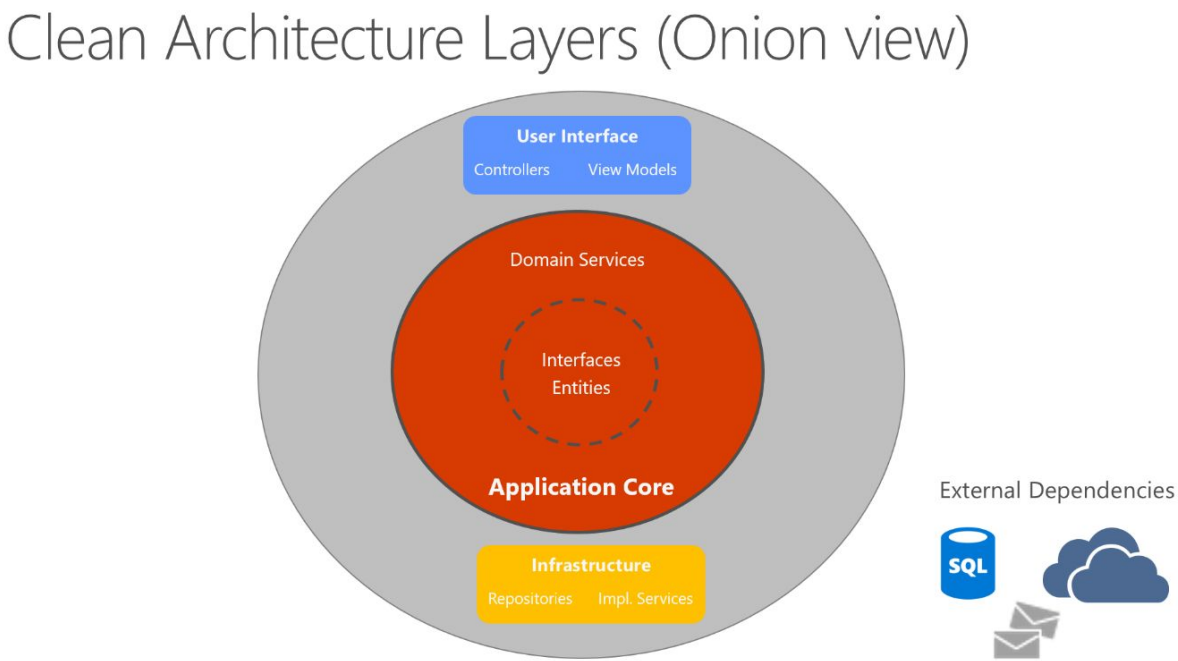
Uma desvantagem dessa abordagem tradicional de disposição em camadas é que as dependências em tempo de compilação são executadas de cima para baixo.

Ou seja, a camada de interface do usuário depende da camada de negócio, que depende da camada de dados.

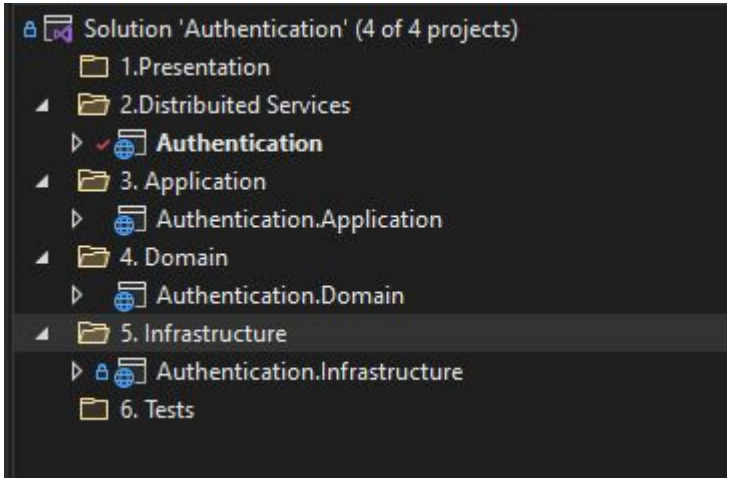
Isso significa que a camada de negócio, que normalmente contém a lógica mais importante no aplicativo, depende dos detalhes de implementação de acesso a dados (e geralmente da existência de um banco de dados).

O teste da lógica de negócios em uma arquitetura como essa costuma ser difícil, exigindo um banco de dados de teste e dependência das três camadas em conjunto.

Camadas da solução



Camadas da solução



Camadas da solução

1. Presentation/Apresentação
   a. Camada onde se recebe iteração do usuários. Projetos form, console, web mobile entre outros possíveis.
2. Distributed Services/ Serviços para distribuição
   a. Camada onde fica iteração com a camada de apresentação ou outros pontos de aplicações distintas é o meio de entrada para o back-end
3. Application/Aplicação
   a. Camada que contém implementação de regra de negócio e modelos/DTOS
4. Domain
   a. Camada que contém a definição e entidades do negócio
5. Infrastructure/Infraestrutura
   a. Camada que contém implementação referente ao sistema/IO, como leitura de arquivos, envio de e mails, acesso a banco de dados entre outros;

Camadas da solução

- Vamo agora na prática criar essa nova estrutura
- Criar outra solução
- Adicionar uma pasta cada camada, conforme o slide anterior
- Vamos fazer os projetos se conversarem
- Projeto existente das camadas
- https://github.com/drhamann/senac-back-end

Generics

- https://learn.microsoft.com/pt-br/dotnet/core/extensions/dependency-injection
- Vamos ver na prática

SENAI

Serviço Nacional de Aprendizagem Industrial

PELO FUTURO DO TRABALHO

0800 048 1212 **sc.senai.br**

Rodovia Admar Gonzaga, 2765 - Itacorubi - 88034-001 - Florianópolis, SC