Epoca Especial 2020/2021

Pergunta 1

Um delegate em C# tem várias vantagens uma delas é o mesmo ser um ponteiro de função e ao mesmo tempo ser possível alterar o seu nome ou até mesmo usa-lo como método anónimo. Ou seja, podemos ter uma função e usamos um delegate para apontar para aquela função e depois podemos simplesmente usar o delegate com um nome mais simples.

Uma vez que um delegate é um ponteiro de função podemos tambem usar o mesmo como argumento de outra função, tendo assim um código mais limpo.

Pergunta 2

No ASP.Net MVC 5.0 não é diretamente produzido todo o código HTML que é depois mostrado nas páginas de um site aquando da sua apresentação no browser porque podemos por exemplo utilizar os HTML Helpers, Tag Helpers e mesmo o Razor. Os HTML Helpers e os Tag Helpers ajudam a compactar algum código HTML tendo apenas o necessário e posteriormente gera as tags necessárias para ser um HTML válido. Como o Razor é uma linguagem de marcação, abre-nos a parte a termos um website dinâmico que consegue receber dados dos controlladores para posteriormente as views mostrarem os mesmos.

Pergunta 3

Não é sempre necessário ser feita a validação no client-side e no server-side no entanto é uma boa prática o mesmo ser feito uma vez que é possível fazer algumas validações client-side a niveis de inputs em formulários e voltar a verificar outras validações em server-side. Ou seja, no fundo as validações client-side só servem para UI/UX ou seja a parte visual, sempre que queremos fazer validações lógicas convem usarmos o server-side.

Pergunta 4

Os modelos representam os nossos modelos de dominio ou seja é aqui feita as definições das nossas tabelas por exemplo numa abordagem em Code First em que definimos os nosso modelos para depois mapearmos para uma base de dados. Usamos estes modelos para definir os objetos com que vamos trabalhar.

Depois existe os controllers que são o cerebro uma vez que são eles que recebem os dados das views e consoante o requesito utilizam os modelos para fazerem querys à base de dados. Os controllers posteriormente recebem o retorno da query e retonam para a view o resultado final pedido pelo utilizador.

As views como dito anteriormente servem para receber dados do cliente, fazer pedidos ao controller e receber os dados do controller para novamente mostrar

dados ao cliente sendo que aqui não se deve de usar qualquer tipo de lógica e no fim apenas os controllers falam com os modelos de domínio.

Pergunta 5

a)

LINQ representa Language Integrated Query e é muito importante uma vez que em aplicações ASP.Net Core MVC normalmente usamos SQL Server e precisamos de alguma ferramente que traduza os nossos modelos de dominio e as nossas querys à base de dados em Queries SQL uma vez que o SQL Server apenas reconhece a sintaxe de SQL. Para isso é usado o LINQ, para traduzir querys feitas em C# para SQL e vice versa tornando possível a comunicação em ambos os sentidos.

b)

É indiferente a utilização de Query Syntax ou Extension Methods uma vez que o intuito de ambos é o mesmo, ou seja comunicar com a base de dados, no entanto os Extension Methods são tendencialmente mais simples de serem usados e são mais recentes.

Pergunta 6

a)

O operador => chama-se de operador lambda e o mesmo é bastante usado porque o podemos utilizar para definir funções anónimas e no LINQ é bastante usado porque podemos usar esse operador para aceder aos atributos de um objeto sem ter de criar funções para tal.

b)

Na instrução, o operador lambda é usado para conseguirmos aceder ao atributo do objeto que neste caso é o "Nome" sem termos de criar uma função para tal. A intrução pesquisa numa suposta lista de alunos (utilizando o Where), qual é o aluno que tem o ultimo nome começado com a letra "J".

Pergunta 7

Existem dois workflows para a construção de websites em ASP.Net MVC.O primeiro é o Code First em que partimos do principio que não temos nenhuma base de dados criada e vamos primeiro criar os nossos modelos de domínio para depois conseguirmos aplicar as migrações fornecidas pelo Entity Framework para conseguirmos então criar uma base de dados em que as nossas tabelas são o espelho dos nossos modelos. Normalmente esta é a maneira mais utilizada quando não temos nenhuma base de dados já construida.

O outro workflow é chamado de Database First em que partimos com uma base de dados já construida que pode ou não ter alguns dados e não os queremos apagar nem perder. Então para isso utilizamos o Database First juntamente com o Entity Framework que nos permite utilizar as tabelas já existentes e fazer um scaffolding das mesmas para os nossos modelos de domino serem criados. Ou seja, neste caso não começamos por criar modelos, começamos com uma base de dados e fazendo um Scaffolding da mesma e utilizando o Entity Framework os dominios vão ser criados. Isto é muito bom porque é uma maneira simples de não apagarmos dados de uma base de dados já existente e continuarmos a conseguir utilizar a mesma sem ter de fazer nenhuma alteração fisica à mesma.

Pergunta 8

A arquitetura mostrada na imagem mostra uma arquitetura em que usamos o LINQ como intermediário para a comunicação entre os nossos modelos de dominio e o nosso servidor de base de dados. Ou seja, aqui vemos que para conseguirmos mapear entre modelos e base de dados utilizamos o linq para fazer essa conversão onde o mesmo traduz as queries feitas aos modelos para queries sql sendo que o SQL Server apenas entende queries sql. Depois o LINQ transforma o retorno das queries de SQL em formato de objeto para ser possível utilizar este objeto em C# utilizando um paradigma de programação orientada a objetos.