INTITUTO TECNOLOGICO DE LAS AMERICAS

NOMBRE: GLAVINTON

APELLIDOS: PARRA HIDALGO

MATRICULA: 2019-8463

PROFESOR: KELYN TEJADA BELLIARD

PRACTICA: FINAL DEL CUATRIMETRE

TEMA DEL PROYECTO FINAL

A que le llamamos CRUD en programación

Para que nos puede servir entity framework

Para que nos puede servir scaffolding en Programación

Desarrollo una descripción sobre ASP.NET

Desarrolla una descripción sobre MVC

Desarrollo una descripción la importancia de la programación orientada a objeto

CRUD en programación

El concepto CRUD está estrechamente vinculado a la gestión de datos digitales. CRUD hace referencia a un acrónimo en el que se reúnen las primeras letras de las cuatro operaciones fundamentales de aplicaciones persistentes. En pocas palabras, CRUD resume las funciones requeridas por un usuario para crear y gestionar datos. Varios procesos de gestión de datos están basados en CRUD, en los que dichas operaciones están específicamente adaptadas a los requisitos del sistema y de usuario, ya sea para la gestión de bases de datos o para el uso de aplicaciones. Para los expertos, las operaciones son las herramientas de acceso típicas e indispensables para comprobar, por ejemplo, los problemas de la base de datos, mientras que, para los usuarios, CRUD significa crear una cuenta (créate) y utilizarla (read), actualizarla (update) o borrarla (delete) en cualquier momento. Dependiendo de la configuración regional, las operaciones CRUD pueden implementarse de diferentes maneras, como lo muestra la siguiente tabla:

sistemas de bases de datos:

- Create (Crear registros)
- Read bzw. Retrieve (Leer registros)
- Update (Actualizar registros)
- Delete bzw. Destroy (Borrar registros)

Frameworks CRUD: capa de acceso a las bases de datos

Si los objetos individuales son visualizados por medio de una interfaz gráfica y modificados con las llamadas operaciones CRUD, entonces se habla de un Frameworks CRUD o de un CRUD grid. Por lo general, se trata de interfaces HTML. Un Frameworks CRUD demanda varios pasos de transacción, de forma que los datos no se recogen una vez se han introducido, sino que es necesario pulsar la opción “Guardar” o “Continuar”. Las operaciones de un Frameworks CRUD pueden aplazarse para ser ejecutadas en diferentes plazos, sin que los datos de dichos periodos de tiempo se bloqueen para otros usuarios. Este hecho resulta de gran importancia para sistemas multiusuario, pues permite que varias personas lean los mismos datos al mismo tiempo. Los frameworks CRUD facilitan el acceso al sistema de bases de datos y son utilizados tanto en el desarrollo como en el uso de aplicaciones. Existen numerosos frameworks con un concepto CRUD basados en diferentes lenguajes y plataformas.

Para que nos puede servir entity Frameworks

De manera simplificada podemos decir que Entity Framework, como cualquier ORM, permite acceder a una base de datos utilizando clases que representan cada una de las entidades de ésta, pudiendo realizar cualquier operación sobre los datos simplemente llamando a métodos de estas clases. La Entity Framework es un conjunto de tecnologías de ADO.NET que admiten el desarrollo de aplicaciones de software orientadas a datos. Los arquitectos y programadores de aplicaciones orientadas a datos se han enfrentado a la necesidad de lograr dos objetivos muy diferentes. Deben modelar las entidades, las relaciones y la lógica de los problemas empresariales que resuelven, y también deben trabajar con los motores de datos que se usan para almacenar y recuperar los datos.

Los datos pueden abarcar varios sistemas de almacenamiento, cada uno con sus propios protocolos; incluso las aplicaciones que funcionan con un único sistema de almacenamiento deben equilibrar los requisitos del sistema de almacenamiento con respecto a los requisitos de escribir un código de aplicación eficaz y fácil de mantener. También da un enfoque de diseño habitual para crear una aplicación o un servicio consiste en dividir la aplicación o el servicio en tres partes: un modelo de dominio, un modelo lógico y un modelo físico. El modelo de dominio define las entidades y relaciones del sistema que se está modelando. El modelo lógico de una base de datos relacional normaliza las entidades y relaciones en tablas con restricciones de claves externas. El modelo físico abarca las capacidades de un motor de datos determinado especificando los detalles del almacenamiento en forma de particiones e índices.

El Entity Framework proporciona una vida a los modelos al permitir a los programadores consultar las entidades y relaciones en el modelo de dominio, denominado modelo *conceptual* en el Entity Framework, mientras se confía en el Entity Framework para traducir esas operaciones en comandos específicos del origen de datos. Esto libera a las aplicaciones de las dependencias codificadas de forma rígida en un origen de datos determinado. Al trabajar con Code First, el modelo conceptual se asigna al modelo de almacenamiento en código. El Entity Framework puede deducir el modelo conceptual en función de los tipos de objeto y las configuraciones adicionales que defina. Los metadatos de asignación se generan durante el tiempo de ejecución basándose en una combinación de cómo se definen los tipos de dominio e información de configuración adicional que se proporciona en código. Entity Framework genera la base de datos según sea necesario en función de los metadatos. Para obtener más información, vea crear un modelo.

El Entity Framework utiliza estos archivos de asignación y modelo para crear, leer, actualizar y eliminar operaciones en las entidades y relaciones del modelo conceptual en las operaciones equivalentes en el origen de datos. La Entity Framework incluso admite la asignación de entidades en el modelo conceptual a procedimientos almacenados en el origen de datos. Para obtener más información, vea las Especificaciones de CSDL, SSDL y MSL.

Para que nos puede servir scaffolding en Programación

En programación el scaffolding es un método o técnica que permite construir aplicaciones basadas en bases de datos, esta técnica está soportada por algunos frameworks del tipo MVC en el cuál podemos escribir una especificación que describe cómo debe ser usada la base de datos. Luego el compilador utiliza esa especificación para generar el código que la aplicación usará para crear, leer, actualizar y eliminar registros de la base de datos, esto es conocido como CRUD (create, read, update, delete). Una parte integral de scaffolding es un modelo de datos que representa las entidades de base de datos como tipos de Common Language Runtime que es tiempo de ejecución de lenguaje común (CLR). Los datos dinámicos utilizan los metadatos del modelo de datos para crear la interfaz de usuario de la aplicación web y para administrar los cambios que realiza el usuario.

Scaffolding hace referencia a los elementos de datos dinámicos que generan automáticamente páginas web para cada tabla de una base de datos. Estas páginas web generadas automáticamente proporcionan operaciones de creación, lectura, actualización y eliminación (CRUD) para cada tabla. El scaffolding está compuesto de plantillas de página, plantillas de página de entidad, plantillas de página de campo y plantillas de filtro. Estas plantillas le permiten compilar rápidamente un sitio web controlado por datos funcional.

ASP.NET scaffolding es un marco de generación de código para aplicaciones Web de ASP.NET. Visual Studio 2013 incluye generadores de código preinstalados para proyectos de MVC y API Web. Agregue scaffolding al proyecto si desea agregar rápidamente código que interactúe con los modelos de datos. El uso de la técnica scaffolding puede reducir la cantidad de tiempo que se tarda en desarrollar operaciones de datos estándar en el proyecto.

Visual Studio 2013 Update 2 (actualmente RC) proporciona la capacidad de ampliar el scaffolding de ASP.NET para satisfacer los requisitos de su escenario. Con esta funcionalidad, puede crear una plantilla de scaffolding personalizada y agregarla para agregar nuevo cuadro de diálogo de scaffolding. Dentro de la plantilla personalizada, especifique el código que se genera al agregar un elemento con scaffolding. Para obtener más información, vea crear un scaffolding personalizado para Visual Studio.

De forma predeterminada, Visual Studio 2013 no admite la generación de código para un proyecto de formularios Web Forms, pero puede usar la técnica scaffolding con formularios Web Forms agregando dependencias de MVC al proyecto o instalando una extensión. A continuación, se muestran ambos enfoques. La ventana Agregar controlador le ofrece la oportunidad de seleccionar opciones para generar el controlador, incluso si desea usar las nuevas características asincrónicas de Entity Framework

Desarrollo una descripción sobre ASP.NET

ASP.NET ofrece tres marcos para crear aplicaciones web: formularios Web Forms, ASP.NET MVC y ASP.NET Web Pages. Los tres marcos son estables y están maduros, y puede crear excelentes aplicaciones web con cualquiera de ellos. Independientemente del marco que elija, obtendrá todas las ventajas y características de ASP.NET Everywhere. Cada marco de trabajo tiene como destino un estilo de desarrollo diferente. El que elija depende de una combinación de los recursos de programación (conocimiento, conocimientos y experiencia de desarrollo), el tipo de aplicación que está creando y el enfoque de desarrollo con el que está familiarizado. A continuación, se muestra información general de cada una de las plataformas y algunas ideas sobre cómo elegir entre ellas. Si prefiere un vídeo de introducción, consulte creación de sitios web

Con los formularios Web Forms ASP.NET, puede crear sitios web dinámicos con un modelo familiar basado en eventos de arrastrar y colocar. Una superficie diseño y cientos de controles y componentes le permiten crear rápidamente sitios potentes y sofisticados sitios controlados por IU con datos. También el ASP.NET MVC ofrece una eficaz forma de compilar sitios web dinámicos basada en modelos, lo que permite una separación clara de intereses y aporta control total sobre el marcado para lograr un desarrollo ameno y rápido. ASP.NET MVC incluye muchas características que permiten el desarrollo para TDD rápido para crear aplicaciones sofisticadas que usan los estándares web más recientes.

Los tres marcos de ASP.NET se basan en el .NET Framework y comparten la funcionalidad básica de .NET y de ASP.NET. Por ejemplo, los tres marcos ofrecen un modelo de seguridad de inicio de sesión basado en la pertenencia, y los tres comparten las mismas funciones para administrar solicitudes, controlar sesiones, etc., que forman parte de la funcionalidad básica de ASP.NET. Además, los tres marcos no son totalmente independientes y elegir uno no impide el uso de otro. Dado que los marcos de trabajo pueden coexistir en la misma aplicación Web, no es raro ver componentes individuales de aplicaciones escritas con diferentes marcos. Por ejemplo, las partes orientadas al cliente de una aplicación pueden desarrollarse en MVC para optimizar el marcado, mientras que el acceso a datos y las partes administrativas se desarrollan en formularios Web para aprovechar los controles de datos y el acceso a datos simple.

Desarrolla una descripción sobre MVC

MVC era inicialmente un patrón Arquitectura, un modelo o guía que expresa cómo organizar y estructurar los componentes de un sistema software, sus responsabilidades y las relaciones existentes entre cada uno de ellos. Su nombre, MVC, parte de las iniciales de Modelo-Vista-Controlador (Model View Controller, en inglés), que son las capas o grupos de componentes en los que organizaremos nuestras aplicaciones bajo este paradigma.

MVC se usa inicialmente en sistemas donde se requiere el uso de interfaces de usuario, aunque en la práctica el mismo patrón de arquitectura se puede utilizar para distintos tipos de aplicaciones. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos. MVC es un "invento" que ya tiene varias décadas y fue presentado incluso antes de la aparición de la Web. No obstante, en los últimos años ha ganado mucha fuerza y seguidores gracias a la aparición de numerosos Frameworks de desarrollo web que utilizan el patrón MVC como modelo para la arquitectura de las aplicaciones web.

Su fundamento es la separación del código en tres capas diferentes, acotadas por su responsabilidad, en lo que se llaman Modelos, Vistas y Controladores, o lo que es lo mismo, Model, Views & Controllers, si lo prefieres en inglés. En este artículo estudiaremos con detalle estos conceptos, así como las ventajas de ponerlos en marcha cuando desarrollamos. La rama de la ingeniería del software se preocupa por crear procesos que aseguren calidad en los programas que se realizan y esa calidad atiende a diversos parámetros que son deseables para todo desarrollo, como la estructuración de los programas o reutilización del código, lo que debe influir positivamente en la facilidad de desarrollo y el mantenimiento. Los ingenieros del software se dedican a estudiar de qué manera se pueden mejorar los procesos de creación de software y una de las soluciones a las que han llegado es la arquitectura basada en capas que separan el código en función de sus responsabilidades o conceptos. Por tanto, cuando estudiamos MVC lo primero que tenemos que saber es que está ahí para ayudarnos a crear aplicaciones con mayor calidad.

Durante la manipulación de datos en una aplicación es posible que estemos accediendo a los mismos datos en lugares distintos. Por ejemplo, podemos acceder a los datos de un artículo desde la página donde se muestra éste, la página donde se listan los artículos de un manual o la página de backend donde se administran los artículos de un sitio web. Si un día cambiamos los datos de los artículos (alteramos la tabla para añadir nuevos campos o cambiar los existentes porque las necesidades de nuestros artículos varían), estamos obligados a cambiar, página a página, todos los lugares donde se consumían datos de los artículos.

Desarrollo una descripción la importancia de la programación orientada a objeto

el POO es muy potente porque nos permite modelar de manera sencilla datos y comportamientos complejos del mundo real. Al poder manejar los datos y los comportamientos de cada objeto de manera independiente nos evita tener que mantener datos globales y coordinar todo eso. En su momento fue una verdadera revolución. hoy en día la tecnología orientada a objetos ya no se aplica solamente a los lenguajes de programación, además se viene aplicando en el análisis y diseño con mucho éxito, al igual que en las bases de datos. Es que para hacer una buena programación orientada a objetos hay que desarrollar todo el sistema aplicando esta tecnología, de ahí la importancia del análisis y el diseño orientado a objetos. La programación orientada a objetos es una de las formas más, populares de programar y viene teniendo gran acogida en el desarrollo de proyectos de software desde los últimos años. Esta acogida se debe a sus grandes capacidades y ventajas frente a las antiguas formas de programar. Lo anterior por si solo es estupendo. Sin embargo, no es suficiente. Para poder manejar de manera eficiente las clases y los objetos que se generan con la Programación Orientada a Objetos son necesarios algunos principios que nos ayudarán a reducir la complejidad, ser más eficientes y evitar problemas. Son los 4 pilares de la POO. Todos los lenguajes orientados a objetos los implementan de una u otra manera, y es indispensable conocerlos bien. Tradicionalmente, la programación fue hecha en una manera secuencial o lineal, es decir una serie de pasos consecutivos con estructuras consecutivas y bifurcaciones. os lenguajes basados en esta forma de programación ofrecían ventajas al principio, pero el problema ocurre cuando los sistemas se vuelven complejos. Estos programas escritos al estilo “espaguetti” no ofrecen flexibilidad y el mantener una gran cantidad de líneas de código en sólo bloque se vuelve una tarea complicada. Frente a esta dificultad aparecieron los lenguajes basados en la programación estructurada. La idea principal de esta forma de programación es separar las partes complejas del programa en módulos o segmentos que sean ejecutados conforme se requieran. De esta manera tenemos un diseño modular, compuesto por módulos independientes que puedan comunicarse entre sí. Poco a poco este estilo de programación fue reemplazando al estilo “espaguetti” impuesto por la programación lineal. Pues la creciente tendencia de crear programas cada vez más grandes y complejos llevó a los desarrolladores a crear una nueva forma de programar que les permita crear sistemas de niveles empresariales y con reglas de negocios muy complejas. Para estas necesidades ya no bastaba la programación estructurada ni mucho menos la programación lineal. Es así como aparece la programación orientada a objetos POO.