Universidad Central del Ecuador

Facultad de Ingeniería y Ciencias Aplicadas.

Sistemas de Información.

Asignatura:
MARCOS DE DESARROLLO II

Nombre:

- Sebastián Granda.
- Juan Robalino.
- Alex Ortis.
- Johny Vilaña.
- Kevin Aguaisa.

DOCENTE:

Ing. Paulo Llaguno.

PERIODO: 2023 – 2024

¿Qué es ADO.NET?

ADO.NET es una biblioteca de acceso a datos para Microsoft .NET Framework. Proporciona un conjunto de clases que permiten a los desarrolladores interactuar con diversas fuentes de datos (como bases de datos relacionales, XML y servicios web) utilizando aplicaciones creadas en lenguajes compatibles con .NET (como C# o Visual Basic).

Historia

La historia de ADO.NET está asociada con el desarrollo de la tecnología de acceso a datos de Microsoft y el surgimiento de .NET Framework. ADO.NET es un desarrollo significativo con respecto a su predecesor ADO (ActiveX Data Objects) y juega un papel crucial en el desarrollo de aplicaciones basadas en la plataforma .NET. A principios de la década de 2000, Microsoft lanzó .NET Framework, un entorno de desarrollo que introdujo una nueva forma de crear aplicaciones web y de escritorio. En este contexto, ADO.NET se propone como la principal biblioteca de acceso a datos en esta plataforma. A diferencia de ADO, que se centra más en el modelo de objetos de datos y se basa en la tecnología COM (Modelo de objetos componentes), ADO.NET utiliza un enfoque más orientado a objetos y específico de conjuntos de datos para manejar conexiones y operaciones de datos.

La arquitectura ADO.NET se basa en tres componentes principales:

Conjunto de datos: es un conjunto de datos en memoria que puede contener tablas, relaciones y restricciones. La estructura es independiente de cualquier fuente de datos subyacente y se puede programar.

DataAdapter: actúa como puente entre DataSet y las fuentes de datos. Le permite completar un conjunto de datos con datos de la base de datos y actualizar los cambios del conjunto de datos en la base de datos.

DataReader: proporciona un método para leer datos secuencialmente desde una fuente de solo lectura (como una base de datos). A diferencia de un DataSet, un DataReader es más eficiente en términos de uso de memoria porque lee datos en un flujo secuencial.

El lanzamiento de ADO.NET marca un cambio importante en la forma en que los desarrolladores interactúan con los datos en las aplicaciones .NET. Proporciona un modelo orientado a objetos más potente para el acceso y la manipulación de datos, lo que proporciona un mejor rendimiento y una mayor flexibilidad en el desarrollo de aplicaciones.

¿Qué es ADO?

ADO, siglas de ActiveX Data Objects, es una tecnología de acceso a datos desarrollada por Microsoft. Fue una de las primeras soluciones de Microsoft para acceder y manipular bases de datos desde aplicaciones en entornos Windows.

ADO fue lanzado originalmente en la década de 1990 como una parte integral de la plataforma Microsoft Data Access Components (MDAC). Utilizaba el modelo de objetos de COM (Component Object Model) para proporcionar una interfaz de programación que permitía a los desarrolladores conectarse a diversas fuentes de datos, como bases de datos relacionales, sistemas de archivos y otros servicios de información.

La tecnología ADO ofrecía una forma consistente y eficiente de acceder a datos, utilizando una jerarquía de objetos como Connection, Command, Recordset y Field. Estos objetos permitían a los desarrolladores establecer conexiones con bases de datos, enviar consultas SQL, recuperar y manipular conjuntos de datos resultantes, así como acceder a campos individuales en esos conjuntos de datos.

El modelo de objetos de ADO proporcionaba una flexibilidad significativa para trabajar con diferentes tipos de bases de datos y proveedores de datos. Sin embargo, con la introducción del framework .NET, Microsoft desarrolló ADO.NET como una evolución de ADO, aprovechando las ventajas de la plataforma .NET y proporcionando una arquitectura más moderna y orientada a objetos para el acceso a datos. ADO.NET reemplazó gradualmente a ADO como la principal tecnología de acceso a datos en el entorno .NET.

ADO Y ADO.NET

Arquitectura:

ADO (ActiveX Data Objects) se basa en el modelo de objetos COM (Component Object Model) y fue diseñado para trabajar con lenguajes de programación compatibles con COM, como Visual Basic 6 y entornos previos a .NET.

ADO.NET, por otro lado, está diseñado específicamente para el framework .NET. Utiliza el modelo de objetos de .NET y se integra profundamente con el entorno de desarrollo y las características de .NET, como el lenguaje C#, Visual Basic .NET, etc.

Enfoque de Programación:

ADO se enfoca en un modelo de programación más procedural, donde se utilizan objetos como Recordset para manipular datos de forma secuencial.

ADO.NET introduce un enfoque más orientado a conjuntos de datos. Utiliza objetos como DataSet que almacenan datos de manera independiente de la fuente de datos original, permitiendo un mayor control y manipulación de datos en memoria.

Conectividad y Manejo de Datos:

ADO se centra en la conectividad con bases de datos y ofrece objetos como Connection, Command y Recordset para realizar operaciones de acceso y manipulación de datos.

ADO.NET proporciona una capa de acceso a datos más avanzada, con objetos como Connection, Command, DataAdapter y DataSet, que ofrecen funcionalidades más potentes para trabajar con datos en entornos desconectados y con múltiples fuentes de datos.

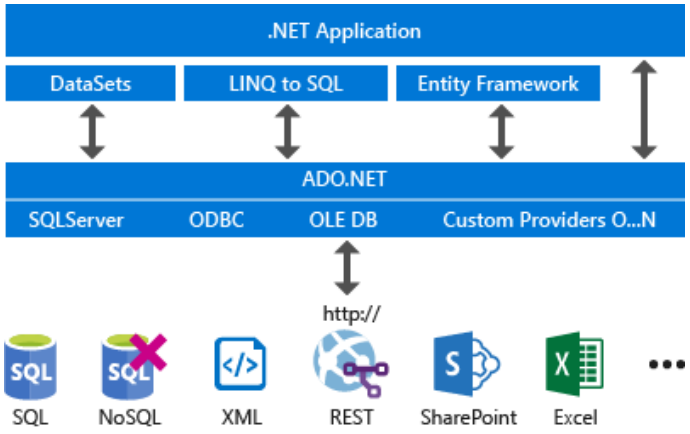
¿Cómo se da la arquitectura sin conexión con ADO?

La arquitectura autónoma en ADO.NET se basa en el uso de DataSets y DataAdapters para trabajar con datos independientemente de la conexión a la fuente de datos original en funcion Conjunto de datos: DataSet es el componente central de la arquitectura independiente ADO.NET.

Es una representación en memoria de datos que puede contener una o más tablas, relaciones entre estas tablas y restricciones. A diferencia de otros modelos de acceso a datos, los conjuntos de datos no están directamente vinculados a una fuente de datos específica (por ejemplo, una base de datos). Puede completarse con datos, modificarse, eliminarse o agregarse a la memoria sin estar conectado permanentemente a la fuente original

Utilizar objetos Command (como SqlCommand, OleDbCommand) para consultar la base de datos y completar el conjunto de datos con los resultados. Un flujo de trabajo típico en una arquitectura independiente con ADO.NET se vería así: Se establecen conexiones temporales con fuentes de datos (bases de datos, páginas web, archivos XML, etc.) mediante objetos de conexión. DataAdapter se crea y se asocia con comandos o consultas SQL para recuperar datos de la fuente. El DataAdapter completa el DataSet con los resultados de la consulta y luego cierra la conexión a la fuente de datos.

Los datos del conjunto de datos se pueden procesar completamente en la memoria sin requerir una conexión activa a la fuente original. Esta arquitectura fuera de línea es útil en situaciones en las que necesita trabajar con datos de forma independiente, lo que le permite realizar operaciones localmente antes de que los cambios se sincronicen con la fuente de datos original.



Orígenes de datos soportados:

Tiene diferentes orígenes entre ellos incluyen:

Bases de datos relacionales: Son estructuras de datos organizadas en tablas que están interrelacionadas mediante claves comunes. Algunos ejemplos son:

- SQL Server: Un sistema de gestión de bases de datos relacionales desarrollado por Microsoft.
- MySQL: Un sistema de gestión de bases de datos de código abierto y ampliamente utilizado.
- Oracle: Un sistema de gestión de bases de datos empresarial desarrollado por Oracle Corporation.

Archivos de datos: Permiten almacenar información en formatos específicos para su posterior manipulación. Ejemplos son:

- Archivos XML: Estructuras de datos legibles por máquinas y humanos, utilizadas para almacenar y transportar datos de forma jerárquica.
- Archivos de texto y CSV: Utilizados para almacenar datos en formato de texto plano, con valores separados por comas u otro delimitador.

Servicios web: Fuentes de datos que se exponen a través de servicios web que pueden ser consumidos por aplicaciones. Pueden proporcionar datos en formatos como JSON o XML.

Otros orígenes de datos específicos:

- Entity Framework: Permite trabajar con datos en forma de objetos y propiedades, proporcionando un mapeo entre el modelo de datos y la base de datos subyacente.
- DataSets: Pueden ser considerados también como un origen de datos dentro de la misma aplicación, almacenan datos en memoria sin estar necesariamente vinculados a una fuente externa.

¿Podemos tener más de un origen de datos?

Sí, ADO.NET permite trabajar con múltiples orígenes de datos simultáneamente en una misma aplicación. Esto significa que puedes conectarte, por ejemplo, a una base de datos SQL Server para obtener algunos datos y al mismo tiempo leer información de archivos XML o consumir datos de un servicio web. La capacidad de interactuar con varios orígenes de datos es una de las ventajas clave de ADO.NET, ya que proporciona flexibilidad para integrar y trabajar con datos provenientes de diversas fuentes en una aplicación.

Bibliografía:

Ado.net. (s/f). Microsoft.com. Recuperado el 11 de diciembre de 2023, de https://learn.microsoft.com/es-es/dotnet/framework/data/adonet/

Información general - Ado.net. (s/f). Microsoft.com. Recuperado el 11 de diciembre de 2023, de https://learn.microsoft.com/es-es/dotnet/framework/data/adonet/ado-net-overview

Informacion General Ado. Net. (s/f). Buenas Tareas. Recuperado el 11 de diciembre de 2023, de https://www.buenastareas.com/ensayos/Informacion-General-Ado-Net/63042619.html

(S/f). Recuperado el 11 de diciembre de 2023, de http://chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/http://sabia.tic.udc.es/docencia/is/old/2008-2009/docs/transparencias/2009-03-23-ADO.NET.pdf