Sérialisation (C# et Visual Basic)

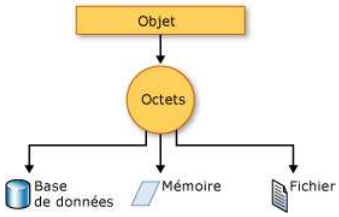
Visual Studio 2013

La sérialisation est le processus de conversion d'un objet en un flux des octets pour stocker l'objet ou le transmettre à la mémoire, une base de données, ou dans un fichier. Son objectif principal est d'enregistrer l'état d'un objet pour pouvoir le recréer si nécessaire. Le processus inverse est appelé désérialisation.

Pour obtenir des exemples sur la sérialisation, consultez Rubriques connexes et exemples plus loin dans cette rubrique.

Fonctionnement de la sérialisation

Cette illustration affiche le processus global de la sérialisation.



L'objet est sérialisé à un flux qui contient non seulement les données, mais également des informations sur le type d'objet, notamment sa version, sa culture et son nom d'assembly. À partir de ce flux, il peut être stocké dans une base de données, dans un fichier ou en mémoire.

Utilisations de la sérialisation

La sérialisation permet au développeur d'enregistrer l'état d'un objet et de le recréer si nécessaire, en fournissant un stockage d'objets ainsi qu'un échange de données. Par le biais de la sérialisation, un développeur peut effectuer des actions comme l'envoi de l'objet à une application distante via un service Web, le passage d'un objet d'un domaine à un autre, via un pare-feu sous forme de chaîne XML ou la conservation des informations de sécurité ou spécifiques à l'utilisateur sur les applications.

Rendre un objet sérialisable

Pour sérialiser un objet, vous avez besoin de l'objet à sérialiser, d'un flux de données pour stocker l'objet sérialisé et d'un Formatter. System.Runtime.Serialization contient les classes nécessaires à la sérialisation et la désérialisation des objets.

Appliquez l'attribut SerializableAttribute à un type pour indiquer que ses instances peuvent être sérialisées. Une exception SerializationException est levée si vous essayez d'effectuer la sérialisation et que le type n'a pas l'attribut SerializableAttribute.

Si vous ne souhaitez pas qu'un champ de votre classe soit sérialisable, appliquez l'attribut NonSerializedAttribute. Si un champ d'un type sérialisable contient un pointeur, un handle ou une autre structure de données spécifique à un environnement particulier et ne peut pas être reconstitué de manière significative dans un autre environnement, il est possible de le rendre non sérialisable.

Si une classe sérialisée contient des références aux objets d'autres classes qui sont marquées SerializableAttribute, ces objets seront également sérialisés.

Sérialisation binaire et XML

La sérialisation binaire ou XML peut être utilisée. Dans une sérialisation binaire, tous les membres, même ceux qui sont en lecture seule, sont sérialisés et les performances sont améliorées. La sérialisation XML fournit un code plus lisible, ainsi qu'une plus grande souplesse du partage et de l'utilisation des objets pour des raisons d'interopérabilité.

Sérialisation binaire

La sérialisation binaire utilise l'encodage binaire afin de produire une sérialisation compacte destinée notamment au stockage ou au flux réseau socket.

Sérialisation XML

La sérialisation XML sérialise les champs et les propriétés publics d'un objet, ou les paramètres et valeurs de retour des méthodes, en un flux XML conforme à un document de langage XSD (XML Schema Definition) spécifique. La sérialisation XML génère des classes fortement typées avec des propriétés publiques et des champs convertis au format XML. System.Xml.Serialization contient les classes nécessaires à la sérialisation et la désérialisation XML.

Vous pouvez appliquer les attributs à des classes et à des membres de classe pour contrôler la manière dont XmlSerializer sérialise ou désérialise une instance de la classe.

Sérialisation SOAP

La sérialisation XML peut également être utilisée pour sérialiser des objets en flux XML se conformant à la spécification SOAP. SOAP est un protocole basé sur XML, conçu spécifiquement pour transporter des appels de procédure à l'aide de XML. Comme avec la sérialisation XML, les attributs peuvent également être utilisés pour contrôler les messages SOAP de style littéral générés par un service Web XML.

Sérialisation de base et personnalisée

La sérialisation peut être effectuée de deux manières, de base ou personnalisée. La sérialisation de base utilise le .NET Framework pour sérialiser automatiquement l'objet.

Sérialisation de base

La seule condition de la sérialisation de base est que l'attribut SerializableAttribute soit appliqué à l'objet. L'NonSerializedAttribute peut être utilisé pour empêcher que des champs spécifiques ne soient sérialisés.

Lorsque vous utilisez la sérialisation de base, le versioning des objets peut causer des problèmes, auquel cas la sérialisation personnalisée peut être préférable. La sérialisation de base constitue le moyen le plus facile pour effectuer la sérialisation, mais ne permet pas un contrôle efficace sur le processus.

Sérialisation personnalisée

Dans la sérialisation personnalisée, vous pouvez spécifier exactement les objets à sérialiser et le mode de déroulement du processus. La classe doit être marquée SerializableAttribute implémenter l'interface ISerializable.

Si vous souhaitez que votre objet soit également désérialisé de manière personnalisée, vous devez utiliser un constructeur personnalisé.

Sérialisation du concepteur

La sérialisation du concepteur est une forme de sérialisation spéciale qui implique le type de persistance des objets généralement associé aux outils de développement. La sérialisatic du concepteur est le processus de conversion d'un graphique d'objets en un fichier source qui peut être utilisé ultérieurement pour récupérer le graphique d'objets. Un fichier source peut contenir du code, un balisage ou même des informations sur les tables SQL. Pour plus d'informations, consultez Vue d'ensemble de la sérialisation du concepteur.

Rubriques connexes et exemples

Procédure pas à pas : persistance d'un objet (C# et Visual Basic)
: Montre comment utiliser la sérialisation pour rendre les données d'un objet persistantes entre les instances, ce qui vous permet de stocker des valeurs et de les récupérer lors la prochaine instanciation de l'objet.

Comment : lire des données d'objet à partir d'un fichier XML (C# et Visual Basic)
: Indique comment lire des données d'objet écrites précédemment dans un fichier XML à l'aide de la classe XmlSerializer.

Comment : écrire des données d'objet dans un fichier XML (C# et Visual Basic)
: Indique comment écrire l'objet d'une classe dans un fichier XML à l'aide de la classe XmlSerializer.

Ajouts de la communauté