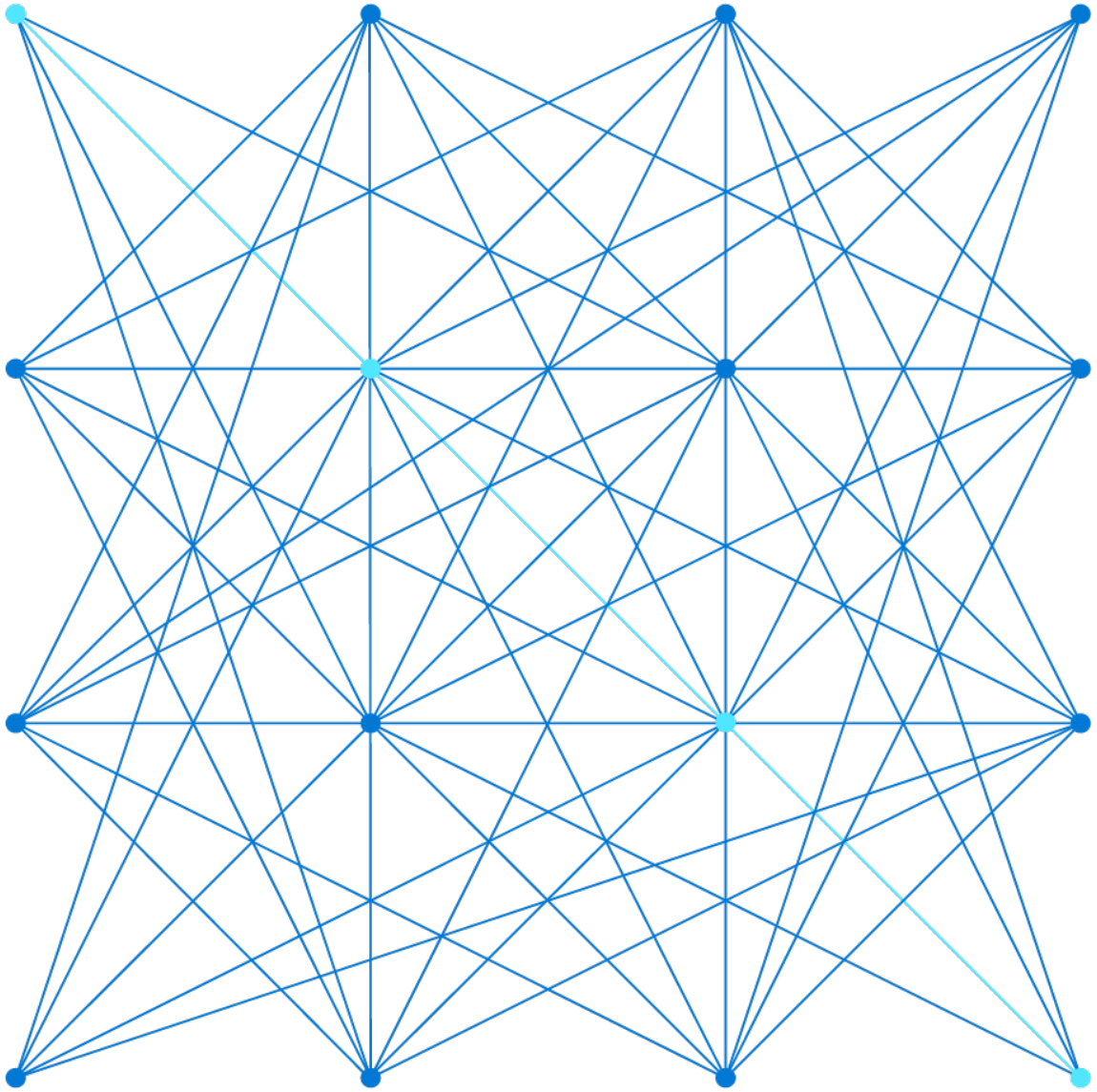
Microsoft Azure

SDK를 사용하여 Azure Cosmos DB for NoSQL에 연결



© Copyright Microsoft Corporation. All rights reserved.

안건

Azure Cosmos DB for NoSQL SDK 가져오기 및 사용

Azure Cosmos DB for NoSQL SDK 구성

© Copyright Microsoft Corporation. All rights reserved.

Azure Cosmos DB for NoSQL SDK 가져오기 및 사용

© Copyright Microsoft Corporation. All rights reserved.

Understand the SDK

Class	Description
Microsoft.Azure.Cosmos.**CosmosClient**	Client-side logical representation of an Azure Cosmos DB account and the primary class used for the SDK
Microsoft.Azure.Cosmos.**Database**	Logically represents a database client-side and includes common operations for database management
Microsoft.Azure.Cosmos.**Container**	Logically represents a container client-side and includes common operations for container management

The *Microsoft.Azure.Cosmos* library is the latest version of the .NET SDK for Azure Cosmos DB for NoSQL.

Import from a package manager

The *Microsoft.Azure.Cosmos* library, including all of its previous versions, are hosted on *nuget* to make it easier to import the library into a .NET application.

Importing a NuGet package

```
dotnet add package
Microsoft.Azure.Cosmos
```

```
dotnet add package
Microsoft.Azure.Cosmos \
    --version 3.22.1
```

.NET project file

```
<Project Sdk="Microsoft.NET.Sdk">
    <PropertyGroup>
        <OutputType>Exe</OutputType>
        <TargetFramework>net6.0</TargetFramework>
    </PropertyGroup>
    <ItemGroup>
        <PackageReference Include="Microsoft.Azure.Cosmos"
Version="3.22.1" />
    </ItemGroup>
</Project>
```

Connect to an online account

온라인 계정에 연결

Microsoft.Azure.Cosmos 라이브러리를 가져오고 나면 .NET 프로젝트 내에서 네임스페이스와 클래스를 사용하기 시작할 수 있습니다.

네임스페이스 가져오기

```
Using Microsoft.Azure.Cosmos;
```

CosmosClient 클래스 사용

```
// Use with a connection string
string connectionString = "AccountEndpoint=https-
://dp420.documents.azure.com:443/;AccountKey=fDR2
ci9QgkdkvERTQ==";

CosmosClient client = new (connectionString);
```

```
// Use with an endpoint and key
string endpoint = "https-
://dp420.documents.azure.com:443/";
string key = "fDR2ci9QgkdkvERTQ==";

CosmosClient client = new (endpoint, key);
```

계정의 속성 읽기

```
AccountProperties account = await client.ReadAccountAsync();
```

속성	설명
Id	계정의 고유한 이름을 가져옵니다.
ReadableRegions	계정의 판독 가능한 위치 목록을 가져옵니다.
WritableRegions	계정의 쓰기 가능한 위치 목록을 가져옵니다.
Consistency	계정의 기본 일관성 수준을 가져옵니다.

© Copyright Microsoft Corporation. All rights reserved.

Interact with a database and a container

Once you have a client instance, you can retrieve or create a database or container.

클라이언트 인스턴스가 있으면 데이터베이스 또는 컨테이너를 검색하거나 만들 수 있습니다.

데이터베이스와 상호 작용

```
// Retrieve an existing database
Database database =
client.GetDatabase("cosmicworks");
```

```
// Create a new database
Database database = await
client.CreateDatabaseAsync("cosmicworks");
```

```
// Create database if it doesn't already exist
Database database = await
client.CreateDatabaseIfNotExistsAsync("cosmicwork
s");
```

컨테이너와 상호 작용

```
// Retrieve an existing container
Container container = database.GetContainer("products");
```

```
// Create a new container
Container container = await database.CreateContainerAsync(
    "cosmicworks",
    "/categoryId",
    400
);
```

```
// Create container if it doesn't already exist
Container container = await database.CreateContainerIfNotExistsAsync(
    "cosmicworks",
    "/categoryId",
    400
);
```

© Copyright Microsoft Corporation. All rights reserved.

클라이언트 싱글톤 구현

사용자 대신 구현된 ***CosmosClient*** 클래스 기능:

- Instances are already thread-safe
- Instances efficiently manage connections
- Instances cache addresses when operating in **direct** mode

** The Azure Cosmos DB for NoSQL SDK team recommends that you use a single instance per **AppDomain** for the lifetime of the application.*

** Azure Cosmos DB for NoSQL SDK 팀에서는 애플리케이션의 수명 동안 `AppDomain`당 단일 인스턴스를 사용하도록 권장합니다.*

© Copyright Microsoft Corporation. All rights reserved.

연결 모드 구성

CosmosClientOptions 클래스는 클라이언트가 계정에 연결될 때 클라이언트에 대해 구성할 수 있는 다양한 옵션을 제공합니다.

기본 클라이언트 옵션 재정의

```
// Constructor that takes in an endpoint and key
CosmosClientOptions options = new ();
CosmosClient client = new (endpoint, key, options);
```

```
//  Constructor that takes the connection string
CosmosClientOptions options = new ();
CosmosClient client = new (connectionString, options);
```

연결 모드 변경

```
// Configures the client to use Direct connection mode.
CosmosClientOptions options = new ()
{ ConnectionMode = ConnectionMode.Direct };
```

```
// Configures the client to use Gateway connection mode.
CosmosClientOptions options = new ()
{ ConnectionMode = ConnectionMode.Gateway };
```

기본 애플리케이션 지역 설정

```
// Configs single preferred region for client to connect to.
CosmosClientOptions options = new ()
{ApplicationRegion = "westus" };
```

```
// Configs the client to use custom failover/priority list.
CosmosClientOptions options = new ()
{ApplicationPreferredRegions = new List<string> { "westus",
"eastus" } };
```

현재 일관성 수준 변경

```
// Configures the client to use eventual consistency.
CosmosClientOptions options = new ()
{ConsistencyLevel = ConsistencyLevel.Eventual };
```

Consistency Levels

- Bounded Staleness
- ConsistentPrefix
- Eventual
- Session
- Strong

© Copyright Microsoft Corporation. All rights reserved.

랩 - SDK를 사용하여 Azure Cosmos DB for NoSQL에 연결

개발 환경 준비

Azure Cosmos DB for NoSQL 계정 만들기

NuGet에서 Microsoft.Azure.Cosmos 라이브러리 보기

Microsoft.Azure.Cosmos 라이브러리를 . NET 프로젝트로 가져오기

Microsoft.Azure. Cosmos 라이브러리 사용

스크립트 테스트

© Copyright Microsoft Corporation. All rights reserved.

Azure Cosmos DB for NoSQL SDK 구성

© Copyright Microsoft Corporation. All rights reserved.

오프라인 개발 사용

Azure Cosmos DB 에뮬레이터

- 에뮬레이터는 Windows,, Linux에서 또는 Docker 컨테이너 이미지로 실행할 수 있습니다.
- Azure Cosmos DB 에뮬레이터 Docker 이미지를 끌어오려면 다음 CLI 명령을 실행합니다.

```
docker pull mcr.microsoft.com/cosmosdb/linux/azure-cosmos-emulator
```

에뮬레이터에 연결하도록 SDK 구성

```
// The emulator's endpoint is https://localhost:<port>/ using SSL with the default port set to 8081.
string endpoint = "https://localhost:8081/";

// The emulator's key is a static well-known authentication key. The default value for this key is:
string key = "C2y6yDjf5/R+ob0N8A7Cgv30VRDJIWEHLM+4QDU5DE2nQ9nDuVTqobD4b8mGGyPMbIZnqyMsEcaGQy67XIw/Jw==";

// Once those variables are set, create the CosmosClient like you would for a cloud-based account.
CosmosClient client = new (endpoint, key);
```

© Copyright Microsoft Corporation. All rights reserved.

연결 오류 처리

요청은 여러 가지 이유로 실패할 수 있으므로 오류 처리 논리가 있어야 합니다.

- . NET용 Azure Cosmos DB for NoSQL SDK에는 읽기 및 쿼리 요청에 대한 일반적인 일시적 오류를 처리하는 기본 제공 논리가 있습니다.
- SDK는 쓰기 요청을 자동으로 다시 시도하지 않습니다. 애플리케이션 코드는 실패한 쓰기에 대한 다시 시도 논리를 구현해야 합니다.

Common HTTP status codes where retrying your request makes sense.

- 429: Too many requests
- 449: Concurrency error
- 500: Unexpected service error
- 503: Service unavailable

Common HTTP status codes that also might need error handling.

- 400: Bad request
- 401: Not authorized
- 403: Forbidden
- 404: Not found

© Copyright Microsoft Corporation. All rights reserved.

스레딩 및 병렬 처리 구현

SDK는 스레드로부터 안전한 형식과 일부 수준의 병렬 처리를 구현합니다.

.NET에서 async/await 사용

```
// Use Task-based features to asynchronously invoke SDK client methods.
Database database = await
client.CreateDatabaseIfNotExistsAsync("cosmicworks");
```

```
//  Avoid blocking the async exec using Task.Wait or Task.Result like:
Database database =
client.CreateDatabaseIfNotExistsAsync("cosmicworks").Result;
```

최대 동시성, 병렬 처리 및 버퍼링된 항목 수 구성

```
// The query returns 500 items per page.
QueryRequestOptions options = new () { MaxItemCount = 500 };
```

```
// The query runs 5 concurrent operations on the client-side.
QueryRequestOptions options = new () { MaxConcurrency = 5 };
```

```
// The Query buffers 5000 items at the client-side.
QueryRequestOptions options = new () { MaxBufferedItemCount =
5000 };
```

LINQ 메서드 대신 기본 제공 반복기 사용

```
// Use SDK included methods such as ToFeedIterator<T> that
// asynchronously retrieves the results and don't block other calls.
container.GetItemLinqQueryable<T>()
    .Where(i => i.categoryId == 2)
    .ToFeedIterator<T>();
```

```
// Avoid LINQ methods such as ToList that block other calls.
container.GetItemLinqQueryable<T>()
    .Where(i => i.categoryId == 2)
    .ToList<T>();
```

클라이언트 쪽 리소스 관련 시간 제한을 피하세요.

요청 시간 제한을 발생시키는 클라이언트 컴퓨터의 문제

- 높은 CPU 사용률
- 높은 포트 사용률

© Copyright Microsoft Corporation. All rights reserved.

로깅 구성

The SDK includes a client builder class that simplifies the process of injecting custom handlers into the HTTP requests and responses.

클라이언트 작성기

```
// To use the builder, add the using directive Microsoft.Azure.Cosmos.Fluent.
using Microsoft.Azure.Cosmos.Fluent;

// Create an instance with either the connection string or endpoint/key.
CosmosClientBuilder builder = new (connectionString);
CosmosClientBuilder builder = new (endpoint, key);

// Add the fluent methods and then build the CosmosClient instance.
CosmosClient client = builder.Build();
```

사용자 지정 RequestHandler 구현 만들기

```
public class LogHandler : RequestHandler
{
    public override async Task<ResponseMessage> SendAsync(RequestMessage request, CancellationToken cancellationToken)
    {
        Console.WriteLine($"[{request.Method.Method}]\t{request.RequestUri}");
        ResponseMessage response = await base.SendAsync(request, cancellationToken);
        Console.WriteLine($"[{Convert.ToInt32(response.StatusCode)}]\t{response.StatusCode}");
        return response;
    }
}
```

© Copyright Microsoft Corporation. All rights reserved.

랩 - 오프라인 개발을 위한 Azure Cosmos DB for NoSQL SDK 구성

개발 환경 준비

Azure Cosmos DB Emulator 시작

SDK에서 에뮬레이터에 연결

에뮬레이터의 변경 내용 보기

새 컨테이너 만들기 및 보기

Azure Cosmos DB Emulator 중지

© Copyright Microsoft Corporation. All rights reserved.

검토

.NET용 Azure Cosmos DB SDK의 컨테이너와 상호 작용하려면 어떤 클래스를 사용해야 하나요?

- ❑ DocumentCollection.
- ❑ CosmosContainer.
- ☑ 컨테이너.

NuGet에서 .NET용 Azure Cosmos DB SDK의 이름은 무엇인가요?

- ☑ Microsoft. Azure. Cosmos.
- ❑ Microsoft. Azure. Documents.
- ❑ Microsoft. Azure. DocumentDB.

SDK 쪽 HTTP 요청을 가로채고 추가 논리를 삽입하는 클래스를 만들려면 어떤 클래스에서 상속해야 할까요?

- ☑ RequestHandler
- ❑ HttpRequest
- ❑ RequestMessage

© Copyright Microsoft Corporation. All rights reserved.

© Copyright Microsoft Corporation. All rights reserved.