

Programming with the .NET SDK



Leonard Lobel

CTO, SLEEK TECHNOLOGIES

lennilobel.wordpress.com



Client Development

**Build web-scale
applications**

Uses REST/HTTP

Platform SDKs

.NET / .NET Core

Java

Node.js

Python



Introducing the .NET SDK for the SQL API

Create a DocumentClient instance

Supply connection information
(endpoint and key)

Invoke methods to access resources

Create, modify, and delete resources
Use POCOs or dynamics for document objects

Task Parallel Library (TPL)

Simplified asynchronous programming
Use async/await keywords with Task objects



Introducing the .NET SDK for the SQL API

Synchronous code

```
private void Main()
{
    DoSomething();
}

private void DoSomething()
{
    // do some work
}
```



Asynchronous code

```
private async void Main()
{
    await DoSomething();
}

private async Task DoSomething()
{
    // do some asynchronous work
}
```

access resources

delete resources

for document objects

Task Parallel Library (TPL)

Simplified asynchronous programming

Use async/await keywords with Task objects



Introducing the .NET SDK for the SQL API

Synchronous code

```
private void Main()
{
    var result = GetSomething();
}

private string GetSomething()
{
    // do some work
    return "Hello";
}
```



Asynchronous code

```
private async void Main()
{
    var result = await GetSomething();
}

private async Task<string> GetSomething()
{
    // do some asynchronous work
    return "Hello";
}
```

access resources

delete resources

for document objects

Task Parallel Library (TPL)

Simplified asynchronous programming

Use async/await keywords with Task objects



Introducing the .NET SDK for the SQL API

Create a DocumentClient instance

Supply connection information
(endpoint and key)

Invoke methods to access resources

Create, modify, and delete resources
Use POCOs or dynamics for document objects

Task Parallel Library (TPL)

Simplified asynchronous programming
Use async/await keywords with Task objects

LINQ provider

Automatically translates LINQ queries to SQL



Demo



Getting started with the .NET SDK



Demo



Working with databases



Demo



Working with collections



Demo



Creating documents



Demo



Querying for documents



Demo



Replacing and deleting documents



Indexing Policies

Hash index

Equality queries
Strings and numbers

Range index

Equality, range, ORDER BY
Strings and numbers

Spatial index

Distance and intersection
Points, polygons,
line strings



Indexing Policies

Collection-wide policy

Established when creating a collection

Can be changed after collection is created



Microsoft Azure cdbwus - Data Explorer

cdbwus - Data Explorer
Azure Cosmos DB account

Search (Ctrl+/)

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Quick start

Data Explorer

SETTINGS

Replicate data globally

Default consistency

Firewall

Keys

Add Azure Search

Add Azure Function

Locks

Automation script

New Collection

New SQL Query

New Store

COLLECTIONS

mydb

mystore

Families

Families

Documents

Scale & Settings

Stored Procedures

User Defined Functions

Triggers

Scale & Sett

Save

Throughpu

1000

Estimated s

Contact sup

Settings

Time to Liv

Off

Partition ke

/location.

Indexing Policy

```
1 {
2   "indexingMode": "consistent",
3   "automatic": true,
4   "includedPaths": [
5     {
6       "path": "/*",
7       "indexes": [
8         {
9           "kind": "Range".
```

Target Information

Welcome

Source Information

Target Information

Advanced

Summary

Results

Specify target information

Export to:

DocumentDB - Sequential record import (partitioned collection)

Persist Date and Time as

String

☒ Enter Indexing Policy ☐ Select Policy File

```
{
  "indexingMode": "consistent",
  "automatic": true,
  "includedPaths": [
    {
      "path": "/*",
      "indexes": [
        {
```

Number of Retries on Failure

30

Retry Interval

00:00:01

Connection Mode ?

DirectTcp

Previous

Next

Indexing Policies

Collection-wide policy

Established when creating a collection
Can be changed after collection is created

Automatic indexing

Can switch to manual
Can override on a per-document basis

Selective indexing

Include/exclude selected property paths

Individual properties (?)

/name/?
/address/state/?

Recursive properties (*)

/address/*
/item/colors[]/*



Indexing Policies

Collection-wide policy

Established when creating a collection
Can be changed after collection is created

Automatic indexing

Can switch to manual
Can override on a per-document basis

Selective indexing

Include/exclude selected property paths

Indexing modes

Consistent (synchronous)
Lazy (asynchronous)



Demo



Custom indexing



Users, Permissions, and Resource Tokens

Resource tokens vs. master key

Provides granular control over security

Create database users

Then create permissions for each user

Get resource token from permission

Read or **All** access to a single resource
(collection, document, stored procedures,
triggers, user-defined functions)

Connect using resource tokens

Access will be granted based on
all supplied resource tokens



Demo



Working with users and permissions



Summary



.NET SDK DocumentClient

- Databases
- Collections
- Documents
- Indexing policies
- Users and permissions

