



Azure Data Storage Solutions



November 17, 2011
Keith Franklin
Director of Cloud and .NET Services

www.mpspartners.com

© 2009 MPS Companies. All rights reserved.



Who We Are

- MPS Partners is a Microsoft Gold Certified Managed Partner with deep expertise in defining and deploying solutions based on Microsoft technology
- We focus in a few key areas:
 - ◆ Collaboration and enterprise content management
 - ◆ Business Intelligence
 - ◆ Integration of the Microsoft toolset with diverse technology landscapes and the cloud
 - ◆ We are especially known for having this expertise within accounts that run SAP



- Founded in 2006, MPS Partners is **Microsoft's 2010 Central Region Partner of the Year**. We have earned the distinction of being a Microsoft Gold Managed Partner – an elite designation within the Microsoft partner community.
- Our experienced staff and leadership team are business people first that focus on bringing valuable **business solutions** to market.
- We are part of **SPR Family of Companies**, a 35-year-old professional services firm headquartered in the Willis Tower in Chicago, IL.



SOA and Business Process
Information Worker Solutions
Business Intelligence
Custom Development Solutions



Windows phone



www.mpspartners.com

Application Platform and Integration



- BizTalk Integration Solutions
- Enterprise Service Bus
- Cloud Integration
- Supply Chain Solutions

.NET and Cloud Solutions



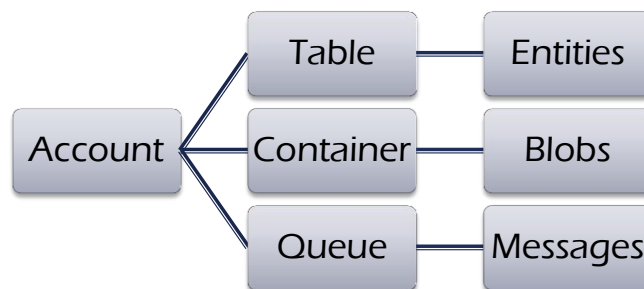
- Advanced .NET Solutions
- Windows Azure Development
- Mobile Solutions

Providing Technology solutions that magnify the value of previous IT investments and enhance communication with business partners.

- What is available?
- Making the connection
- Creating Data
- Reading Data
- Updating Data
- Deleting Data
- Thoughts on what, when and why

- SQL Azure
 - ▶ Tried and true Relational Database based on SQL Server 2008
- Table Storage
 - ▶ NoSQL (Not only SQL) approach to storing information
- Blob Storage
 - ▶ For storage of non-structured information ideally suited for documents, media, etc.
- Queue Service
 - ▶ Messaging system to use between applications or between sub-systems
- Azure Drives
 - ▶ Mount a VHD stored in Blob Storage as a Hard drive and use it as if it was a real drive

- SQL Azure based off of SQL Server 2008
- Every table must have a clustered index
- No physical database server manage or configure
- SQL cannot use USE command to hop across databases
- Not available
 - ▶ Agent
 - ▶ Encryption
 - ▶ Free-Text Search
 - ▶ SQLCLR
- Use client libraries you are already familiar with



- Table : <http://<account>.table.core.windows.net/<table>>
- Blob: <http://<account>.blob.core.windows.net/<container>>
- Queue: <http://<account>.queue.core.windows.net/<queue>>

- Schemaless – Simple two level approach to storage
 - ▶ Tables with Entities with 1 required piece of information and that is simply the *Name*
- Whereas Relational implement ACID semantics Table Storage implements BASE semantics
 - ▶ Basically available
 - ▶ Soft state
 - ▶ Eventually consistent
- Partitionable across geographic boundaries based on Azure Data Centers for performance and scalability
- Windows Azure Storage Client Library use WCF Data Services

- Easy created by inheriting from **TableServiceEntity**
- Entities in a Table do not all have to be the same structure
- Each entity has 1 distinct Key made up of 2 parts
 - ▶ RowKey
 - ▶ PartitionKey
- Entities are sets of Name/Value pairs
- Each entity is limited to 1MB in size
- Each entity is limited to 252 user defined properties
- Limited to following types:
 - ▶ Byte[] – Max 64KB
 - ▶ Boolean
 - ▶ DateTime – Automatically converted to local time when retrieved
 - ▶ Double
 - ▶ Guid
 - ▶ Int32
 - ▶ Int64
 - ▶ String – Max 64KB

- 2 types
 - ▶ Page Blob – Larger, optimized for random access. Use by Azure for VHDs
 - ▶ Block Blob – Smaller optimized for streaming
- What you would use for unstructured data like
 - ▶ Documents
 - ▶ Images
 - ▶ Video
 - ▶ Audio
- Really a compliment to other storage in most business systems

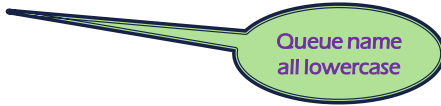
- Simple API for sending messages to other parts of application or other applications
- Queues
 - ▶ Queue names must be all lowercase
 - ▶ Queues can have Metadata associated with them
 - ▶ No limit to number of messages
- Messages
 - ▶ Last at most 1 week. Garbage collected after that.
 - ▶ Max 8KB in size.
 - ▶ Returned as XML. Binary messages Base64 encoded.
 - ▶ Reads are non destructive, but hide message
 - Use VisibilityTimeout to control how long message is hidden
 - Default 30 seconds
 - Maximum 2 hours
 - Denoted in seconds
 - Delete message before VisibilityTimeout expires to stop duplicate processing

- Connect to an Azure Storage Account
 - ▶ Use a connection string to establish a CloudStorageAccount
- Table Storage
 - ▶ Use the CloudStorageAccount to create a CloudTableClient
 - ▶ Example
- Blob Storage
 - ▶ Use the CloudStorageAccount to create a CloudBlobClient
 - ▶ Example
- Queue Services
 - ▶ Use the CloudStorageAccount to create a CloudQueueClient
 - ▶ Example
- SQL Azure
 - ▶ Use any client library you like just like SQL Server

- Table Storage
 - ▶ Establish Connection to Azure Storage
 - ▶ Use DataServiceKey Attribute to declare PartitionKey property and RowKey property
 - Or
 - ▶ Inherit from TableServiceEntity which includes PartitionKey and RowKey properties
 - ▶ Public properties will be serialized into Table Storage
 - ▶ Example

- Blob Storage
 - ▶ Establish connection Azure Storage
 - ▶ Reference Blob Container
 - ▶ Create Blob (Empty)
 - ▶ Fill Metadata
 - ▶ Set Content Type
 - ▶ Upload ByteArray
 - ▶ Example

- Queues
 - ▶ Establish Connection
 - ▶ Reference Queue
 - ▶ AddMessage
 - ▶ Example



Queue name
all lowercase

- Table Storage
 - ▶ Establish connection to Azure Storage
 - ▶ Use DataServiceQuery and Linq to retrieve and filter items in Table
 - ▶ Example

- Blob Storage
 - ▶ Establish connection to Azure Storage
 - ▶ Establish connection to Container
 - ▶ Once connection to Container is established
 - List Blobs in Container
 - Get reference to Blob with GetBlobReference or GetPageBlobReference or GetBlockBlobReference
 - Can check Metadata before you pull down Blob
 - ▶ Examples
 - ▶ Also Blobs become reference-able URIs

- Queues
 - ▶ Establish Connection
 - ▶ Reference Queue
 - ▶ GetMessage to retrieve and hide message temporarily
 - ▶ PeekMessage to retrieve message and leave visible
 - ▶ Example

- Table Storage
 - ▶ Establish connection to Azure Storage
 - ▶ Use DataServiceQuery and Linq to retrieve and filter entities in Table
 - ▶ Set new values in entity
 - ▶ Invoke UpdateObject to update the entity
 - ▶ Invoke one of the Save methods
 - ▶ Example
- Blob Storage
 - ▶ No functionality to update an existing Blob
- Queue
 - ▶ No functionality to update an existing Message

- Table Storage
 - ▶ Establish connection to Azure Storage
 - ▶ Use DataServiceQuery and Linq to retrieve and filter items in Table
 - ▶ Invoke DeleteObject to Delete the entity
 - ▶ Invoke one of the Save methods
 - ▶ Example

- Blob Storage
 - ▶ Establish connection to Azure Storage
 - ▶ Establish connection to Container
 - ▶ Once connection to Container is established
 - Get reference to Blob
 - Invoke one of the Delete Methods
 - ▶ Example

- Queue
 - ▶ Unlike MSMQ Azure Queues Reads aren't destructive
 - GetMessage hides message temporarily

 - ▶ Establish Connection
 - ▶ Reference Queue
 - ▶ GetMessage to retrieve and hide message temporarily
 - ▶ Process Message
 - ▶ If success then Delete Message
 - ▶ Example

- What?
 - ▶ Partition
 - Vertical – Some in SQL, some in Table, some in Blob
 - Horizontal – Spread data out across locations(nodes)
 - Easier to do with Table Storage
 - Harder to do with SQL Azure
 - Hybrid – Best of all worlds
 - ▶ Strive for Hybrid Solution between SQL Azure and Table Storage for structured data
 - ▶ Blob Storage is a given for Web assets and CDN
 - ▶ Move any data that you would have stored as Blobs in SQL to Blob Storage
 - ▶ Use Queues as you would a queuing solution like MSMQ or a database polling solution

- Azure Storage benefits
 - ▶ Hardware/OS management built in
 - ▶ Fault tolerance built in
- Change
 - ▶ All SQL
 - Most have done this
 - ▶ All SQL with Blobs for media
 - Very common
 - ▶ NoSQL
 - New for most

- Cost
 - ▶ A big reason to move to Azure is potential cost savings
 - ▶ Cost of Storage
 - SQL on premise - \$\$\$\$\$
 - SQL Azure - \$\$\$\$ - 10GB ~\$100 per month
 - NoSQL - \$ - 10GB \$1.40 per month plus transactions
 - Hybrid - \$\$
 - ▶ Cost of Change
 - SQL is easy (relative)
 - Developer Tools, Knowledge base, Support Tools
 - NoSQL
 - Lack of Tools, Minimal knowledge base, support tools?

- Additional Information
 - ▶ Azure SDK
<http://www.microsoft.com/windowsazure/sdk/>
 - ▶ Azure Platform Training Kit
<http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=8396>
Pay special attention to slide deck called Storage Strategies
 - ▶ Microsoft Windows Azure Development Cookbook by Neil Mackenzie
 - ▶ Wade Wegner talk on Cloud Storage and Windows 8
<http://channel9.msdn.com/Events/BUILD/BUILD2011/SAC-861T>
- Need assistance with Azure contact MPS Partners, LLC
 - ▶ Keith.Franklin@MPSPartners.Com