void divide(int i, int j)=> return (float)i/j; //simple method definition

int count=10;

console.writeline($"Total counts:{count}");// called as string interpolation

optional parameters

public void getdata(string empName,string company="infosys", string location) //defining

{

}

getdata(empName:"chaitanya",location:"HYDSEZ"); //calling

Employee emp=null;

Null-Conditional operation ?

string name=emp?.Name //means if emp!=null then name=emp.Name else statement can be ignore

//filtering exception

catch(Exception ex) when (ex.GetType()!=typeof(NullReferenceException))

{

}

static using

public static class Helper

{

public static string getdate() { return datetime.now; }

}

using static Helper

strig dt=getdat();

Anonymous class

Var emp={Name=”sai”, Age=35}

emp is type of var i.e. it is anonymous class that has name and age properties.

Anonymous class has only public field and should be initialized, no static or private fields and any type of methods not allowed.

Difference between is and as operator

The is **operator** checks whether an object is compatible with a given type, and the result of the evaluation is a Boolean: true or false. ... The As **operator** is similar to a cast, but returns null instead of an exception if it fails. And the Is **operator** is used to check if one object is compatible with a certain type

Difference between int.Parse() and convert.toint()

Both **int**.**Parse and Convert**.**ToInt32** are used to **convert** string into the integer but Only **difference between** them is to **Convert**.**ToInt32** handle null and returns '0' as output and **int**.**parse** is not going to handle NULL and will give a Argument Null Exception. To check before conversion use int.tryparse method

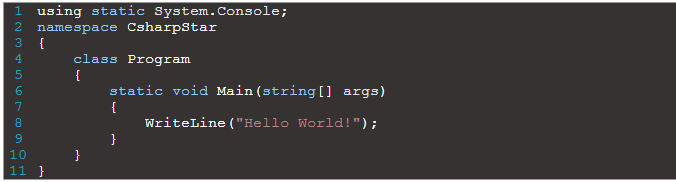
# Top 10 Features in C# 6.0

1. Static Using syntax
2. Auto property initializer
3. Await in Catch and finally block
4. Exception filters
5. Dictionary initializer
6. String interpolation
7. Name of expression
8. Null conditional operator
9. Getter-Only auto properties
10. Expression bodied functions and properties



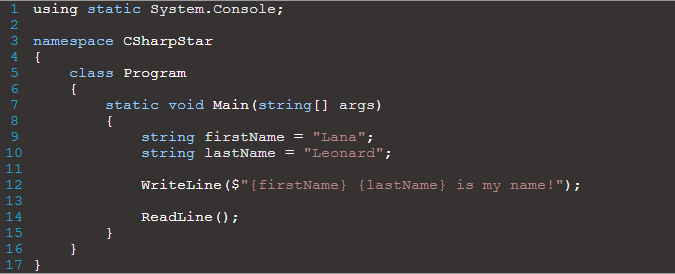
1. Static Using Syntax:

Instead of calling static classes methods with the syntax like <staticClass>.<staticMethod> whenever it is required, directly we can integrate the static class along with namespace declaration i.e. using blocks.



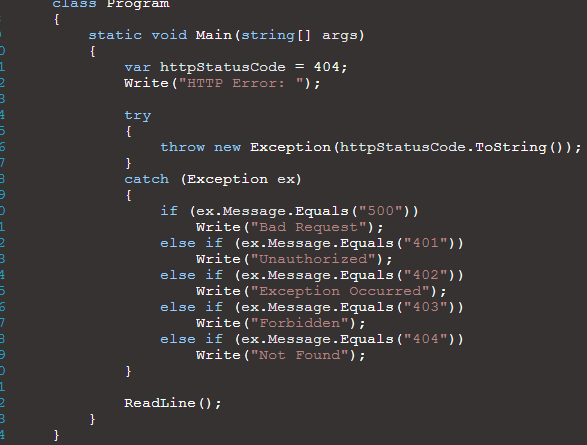
1. String interpolation

String Interpolation is a mechanism to concatenate two or more strings together. In older versions of C#, we were using “+” operator or String.Format method to concatenate strings but in C# 6.0, Microsoft has provided a feature named String Interpolation to concatenate strings

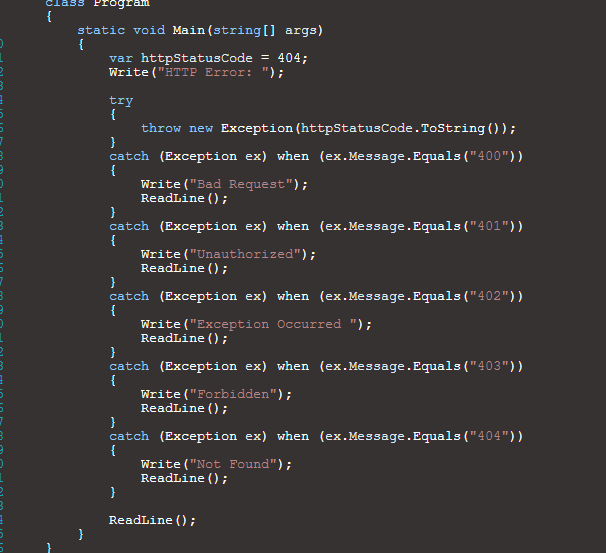


1. Exception Filters

See the below code sample which we are using currently while exception thrown.

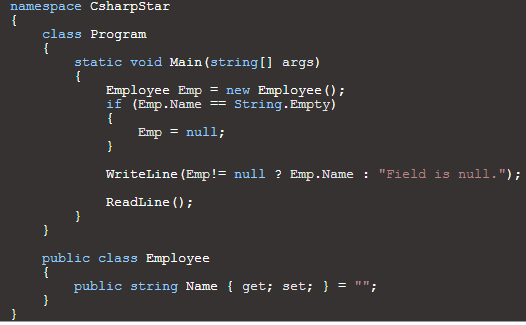
:

But in C# 6.0 we can filter the exceptions in the below way



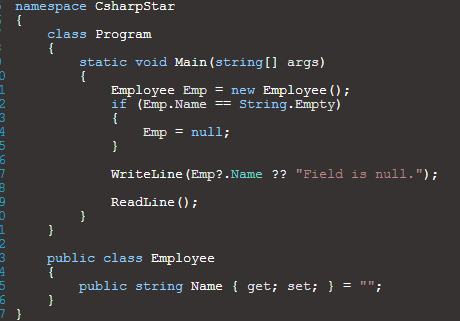
1. Null Conditional Operator

Most of us do not like NullReferenceException . Let’s look at below example, on how to handle NullReferenceException prior to C#6.0



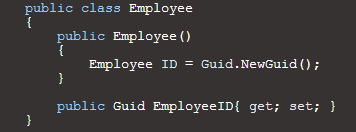
**New ways to handle NullReferenceException in C#6.0**:

In C# 6.0, you can use **?.** to check if an instance is null or not. Let’s look at below example to see how it works in C#6.0.

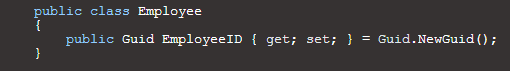


1. Auto Property initializer:

Prior to C#6.0, we were using constructor to initialize the auto properties to non-default value

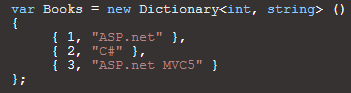


In C#6.0, you can modify the Employee class and populate the property called EmployeeID with inline initialization

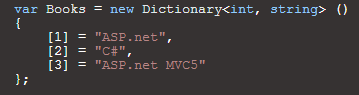


1. Dictionary Initializers

In C# 5, you would initialize the Dictionary with a {“Key”, “Value”} Pair. Let’s look at below example to understand it better

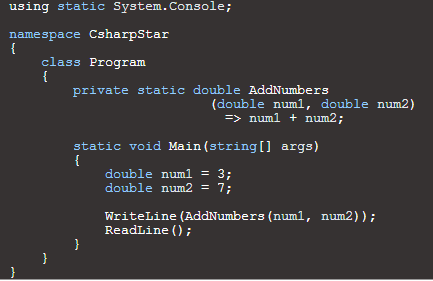


In C# 6, you can place the key between two square brackets [“Key”] and then set the value of the key [“Key”] = “value”.



1. Expression Bodied Function & Property

Expression Bodied Functions are functions with no statement body. Instead, you implement them with an expression following the function declaration.

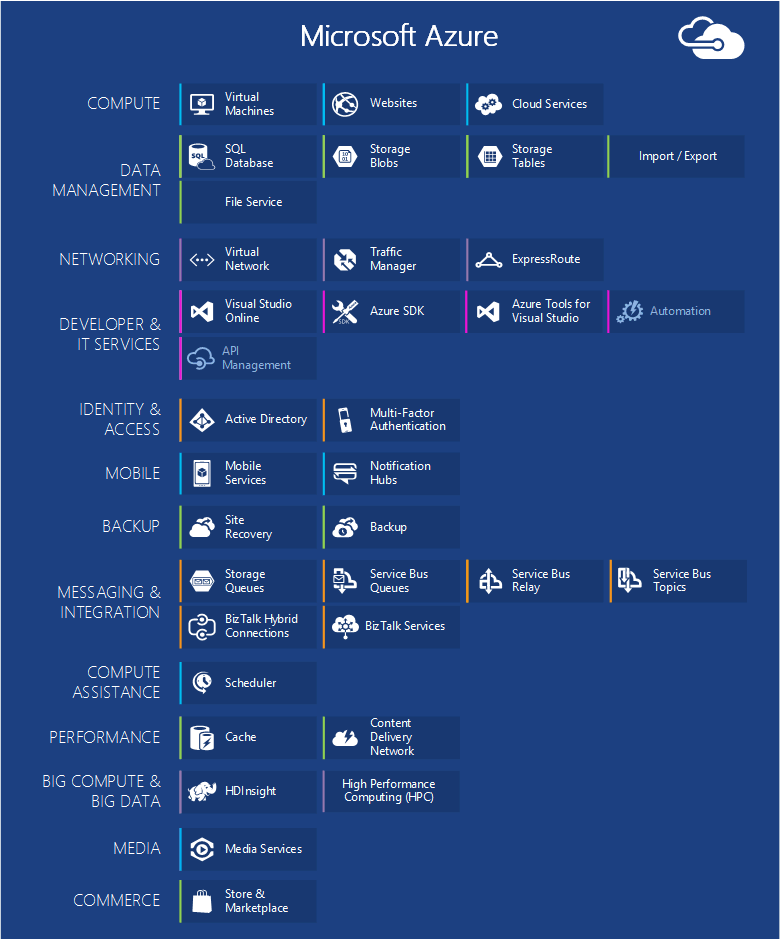


# ASP.Net Core

* ASP.Net core is an open source to build cross plat form solutions in cloud internet apps by including MVC features and Web API features in singled application.
* It supports built in Dependency Injection and it host the application on IIS or self-host on its own process.
* Dependency Injection is nothing but a Design Pattern that used to create loosely coupled code. Its way of injecting the dependencies of one class as objects.
* The great thing about ASP.Net core is, it has the best performance. Microsoft introduced new Kestrel web server that runs within your IIS host or behind another host process. Kestrel has been designed from the start to be the fastest .NET server available.
* In ASP.Net core, MVC Tag Helpers are used to generate the client-side code from .Net and make it easier to re-use in RAZOR markup.

# Introduction Microsoft Azure

Microsoft Azure is a Microsoft platform for the public cloud. The components of the azure are nothing but categorized the azure services. The below image has the services what Azure can provide.



### **Compute**

#### **Azure Virtual Machines**

The ability to create a virtual machine on demand, whether from a standard image or from one you supply, can be very useful. This approach, commonly known as Infrastructure as a Service (IaaS), is what Azure Virtual Machines provides

To create a VM, you specify which VHD to use and the VM's size. You then pay for the time that the VM is running. You pay by the minute and only while it's running, though there is a minimal storage charge for keeping the VHD available. Azure offers a gallery of stock VHDs (called "images") that contain a bootable operating system to start from. These include Microsoft and partner options, such as Windows Server and Linux, SQL Server, Oracle and many more. You're free to create VHDs and images, and then upload them yourself. You can even upload VHDs that contain only data and then access them from your running VMs.1

**Virtual Machine Scenarios:**

* Dev /Test Environment creation
* Move Application to Azure (Lift and Shift)
* Extending Data Senters

#### **Web Apps**

Web Role has the job to do the administrating the multiple VMs and underlying the OS, but deploying the Web applications/Web Sites and maintain them have still administration work. So, these responsibility take care by Web Apps.

When we need to setup a web site, it is best to start up with Azure Web Apps and proceed to Cloud Services once you need a feature that’s not available in Web Apps.

#### **Cloud Services**

Azure Cloud Services provides a place to run highly scalable custom code on a Platform as a Service (PaaS) environment

### **Data Management**

Azure provides many storage options, but all are designed for durable storage. With any of these options, there are always 3 copies of your data kept in sync across an Azure datacenter -- 6 if you allow Azure to use geo-redundancy to back up to another datacenter at least 300 miles away.

#### **In Virtual Machines**

In Virtual Machines, you can create any relational database like SQL Server or any other DBMS. You can also use NoSQL Technologies as MongoDB, Cassandra.

Again, the state of the Virtual Machine and any additional data disk you create or upload are backed by blob storage

#### **Azure SQL Database**

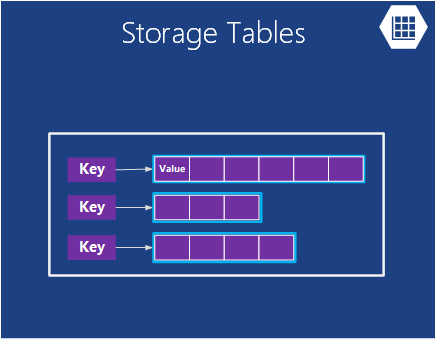
This is different than a typical SQL Database provided by SQL Server running on top of Windows Server.

Formerly called SQL Azure, Azure SQL Database provides all of the key features of a relational database management system, including atomic transactions, concurrent data access by multiple users with data integrity, ANSI SQL queries, and a familiar programming model. Like SQL Server, SQL Database can be accessed using Entity Framework, ADO.NET, JDBC, and other familiar data access technologies. It also supports most of the T-SQL language, along with SQL Server tools such as SQL Server Management Studio.

But SQL Database isn't just a DBMS in the cloud-it's a PaaS service. You still control your data and who can access it, but SQL Database takes care of the administrative grunt work, such as managing the hardware infrastructure and automatically keeping the database and operating system software up to date. SQL Database also provides high availability, automatic backups, point-in-time restore capabilities, and can replicate copies across geographical regions.

#### **Tables**

This tables can be called as Azure Tables or Storage Tables. All are given the same meaning. But its not a relational storage and example of NoSQL. It provides Key/Values pair but while complex operations like joins aren't supported, tables offer fast access to typed data.



*Azure Tables provides a flat NoSQL way to store data*

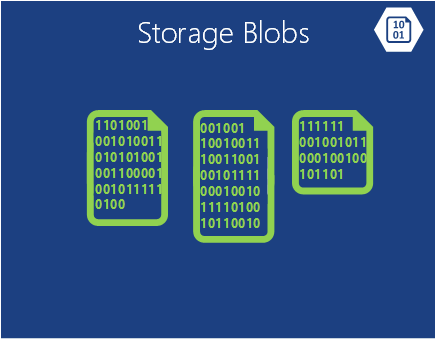
They're also very scalable, with a single table able to hold as much as a terabyte of data. And matching their simplicity, tables are usually less expensive to use than SQL Database's relational storage.

**Scenarios for Tables**

Suppose you want to create an Azure application that needs fast access to typed data, maybe lots of it, but doesn't need to perform complex SQL queries on this data.

#### **Blobs**

Azure Blobs (again "Blob Storage" and just "Storage Blobs" are the same thing) is designed to store unstructured binary data. Like Tables, Blobs provides inexpensive storage, and a single blob can be as large as 1TB (one terabyte). Azure applications can also use Azure drives, which let blobs provide persistent storage for a Windows file system mounted in an Azure instance. The application sees ordinary Windows files, but the contents are actually stored in a blob.



Azure Blobs provides unstructured binary data

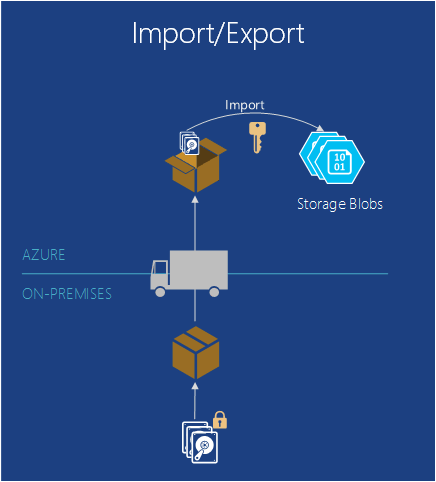
Blob storage is used by many other Azure features (including Virtual Machines), so it can certainly handle your workloads too.

**Scenarios for Blobs**

An application that stores video, massive files, or other binary information can use blobs for simple, cheap storage. Blobs are also commonly used in conjunction with other services like Content Delivery Network.

#### **Import / Export**

Sometimes you want to move a lot of data into Azure. That would take a long time, perhaps days, and use a lot of bandwidth. In these cases, you can use Azure Import/Export, which allows you to ship Bitlocker-encrypted 3.5" SATA hard drives directly to Azure data centers, where Microsoft will transfer the data into blob storage for you. After the upload is completed, Microsoft ships the drives back to you. You can also request that large amounts of data from Blob Storage be exported onto hard drives and sent back to you via mail.



Azure Import / Export provides the ability to ship a physical hard drive to or from Azure for faster and cheaper bulk data import or export

**Scenarios for Import / Export**

**Large Data Migration** - Anytime you have large amounts of data (Terabytes) that you want to upload to Azure, the Import/Export service is often much faster and perhaps cheaper than transferring it over the internet. Once the data is in blobs, you can process it into other forms such as Table storage or an SQL Database.

Archived Data Recovery - You can use Import/Export to have Microsoft transfer large amounts of data stored in Azure Blob Storage to a storage device that you send and then have that device delivered back to a location you desire. Because this will take some time, it's not a good option for disaster recovery. It's best for archived data that you don't need quick access to.

#### **File Service**

On-premises, it’s common to have large amounts of file storage accessible through the Server Message Block (SMB) protocol using a \\Server\share format. Azure now has a service that allows you to use this protocol in the cloud. Applications running in Azure can use it to share files between VMs using familiar file system APIs like ReadFile and WriteFile. In addition, the files can also be accessed at the same time via a REST interface, which allows you to access the shares from on-premises when you also set up a virtual network. Azure Files is built on top of the blob service, so it inherits the same availability, durability, scalability, and geo-redundancy built into Azure Storage

**Scenarios of Azure Files**